

Parallel quantum signal processing via polynomial factorization

John M. Martyn¹, Zane M. Rossi², Kevin Z. Cheng², Yuan Liu^{3,4,5}, and Isaac L. Chuang^{2,6}

¹Center for Theoretical Physics, Massachusetts Institute of Technology, Cambridge, MA 02139, USA

²Department of Physics, Massachusetts Institute of Technology, Cambridge, MA 02139, USA

³Department of Electrical and Computer Engineering, North Carolina State University, Raleigh, NC 27606, USA

⁴Department of Computer Science, North Carolina State University, Raleigh, NC 27606, USA

⁵Department of Physics, North Carolina State University, Raleigh, NC 27606, USA

⁶Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139, USA

Quantum signal processing (QSP) is a methodology for constructing polynomial transformations of a linear operator encoded in a unitary. Applied to an encoding of a state ρ , QSP enables the evaluation of nonlinear functions of the form $\text{tr}(P(\rho))$ for a polynomial $P(x)$, which encompasses relevant properties like entropies and fidelity. However, QSP is a sequential algorithm: implementing a degree- d polynomial necessitates d queries to the encoding, equating to a query depth d .

Here, we reduce the depth of these property estimation algorithms by developing *Parallel Quantum Signal Processing*. Our algorithm parallelizes the computation of $\text{tr}(P(\rho))$ over k systems and reduces the query depth to d/k , thus enabling a family of time-space tradeoffs for QSP. This furnishes a property estimation algorithm suitable for distributed quantum computers, and is realized at the expense of increasing the number of measurements by a factor $O(\text{poly}(d)2^{O(k)})$. We achieve this result by factorizing $P(x)$ into a product of k smaller polynomials of degree $O(d/k)$, which are each implemented in parallel with QSP, and subsequently multiplied together with a swap test to reconstruct $P(x)$. We characterize the achievable class of polynomials by appealing to the fundamental theorem of algebra, and demonstrate application to canonical problems including entropy estimation and partition function evaluation.

1 Introduction

The increasing sophistication of quantum computers has pressured researchers to clearly demarcate problems for which quantum algorithms provide provable speedups over their classical counterparts. Toward resolving this tension, recent work has proposed a qual-

ified ‘unification of quantum algorithms’, based on the related frameworks of quantum signal processing (QSP) [1, 2, 3, 4] and the quantum singular value transformation (QSVT) [5]. These algorithms enable the application of tunable polynomial functions to the singular values of large linear operators, in turn unifying and simplifying the presentation of most known quantum algorithms [6], while simultaneously exhibiting good numerical properties [7, 8, 9, 10], near-optimal query complexity [11], and fruitful connections to well-studied techniques in matrix decomposition and functional analysis [12, 10, 13, 14].

Despite this broad success, we are not yet in a world of fault tolerant quantum devices and are thus unable to leverage the full apparatus of QSP/QSVT. Experimental implementations of QSP/QSVT on existing hardware have largely been limited to small examples on noisy devices [15, 16]. Consequently it is an interesting question whether the theoretical success of QSP and QSVT can be extended to experimental systems with mild resource constraints, such as limited coherence times. Specifically, QSP and QSVT are sequential algorithms without intermediate measurement, and indeed derive their pleasing properties from their circuit depths. Nonetheless, it seems reasonable that even with a limited circuit depth some of these properties could be recovered at the cost of increased circuit width and/or sample complexity.

As suggested by the title of this work, we are interested in realizing this by parallelizing QSP, motivated by the ubiquitous use of parallel processing in classical computation [17]. At a high-level, such processes take advantage of the fact that certain problems permit division into simpler sub-problems. Showing that such divide-and-conquer strategies can prove advantageous requires that (1) the original problem can be subdivided efficiently, (2) the sub-problems can be solved faster than the original problem, and (3) solutions to sub-problems can be efficiently reconciled to produce the full solution. We translate this notion to the cir-

cuit model of quantum computation in a straightforward way, where sub-processes (analogously to ‘threads’ in the classical world) are considered in parallel when they act on disjoint subsets of qubits, and where problem subdivision takes place entirely *classically*, before the execution of the quantum circuit.

This work proposes one scheme for parallelizing QSP problems into multiple independent sub-problems, each of which requires shallower circuits. This technique, termed *Parallel Quantum Signal Processing* (Parallel QSP) is applicable to the task of computing *nonlinear functions* of quantum states, with wide application in the estimation of common properties like entropies and entanglement measures [18, 19, 20], and the efficient realization of disparate multi-state tests [21]. As QSP generates polynomial transformations, subdivision of a problem will correspond to polynomial factorization, and reconciliation to multiplication of factor polynomials.

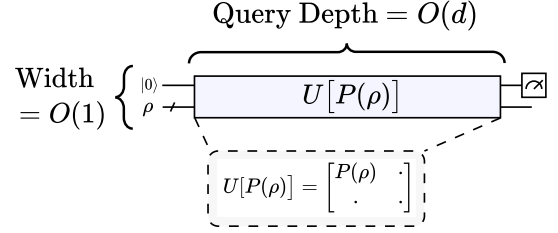
Our construction sits at the intersection of two lines of work for estimating properties of quantum states: quantum signal processing [22, 5], and multivariate trace estimation [18, 23]. The benefit of combining these techniques is rooted in jointly leveraging their individual strengths. While QSP can prepare nonlinear functions of a quantum state, the circuits required are often quite deep [5, 24, 25]. Conversely, while recently proposed methods for multivariate trace estimation can make do with shallow circuits (e.g., constant depth [23]), the achievable class of nonlinear functions is limited. This work provides candidate problems for which these two toolkits can be applied jointly, with the benefit of analytic simplicity and asymptotic savings in circuit depth.

1.1 Results and Paper Outline

In standard QSP, generation of a degree- d polynomial requires d successive queries to the encoding unitary, corresponding to a *query depth* d and circuit depth $O(d)$. Given that large circuit depths are prohibitive on near-term devices limited by short coherence times, we seek valid methods to parallelize QSP into many shorter QSP circuits and reduce the corresponding query depth.

Here we parallelize computation over k threads and reduce query depth by an $O(k)$ factor. In practical situations, $k = O(1) \ll d$, such as when parallelizing a large degree polynomial over a few quantum devices. This parallelization establishes parallel QSP as a suitable algorithm for distributed quantum computation [26], where multiple devices can be run concurrently. However, the reduction in depth afforded by parallelization comes with an increased number of measurements $O(\text{poly}(d)2^{O(k)})$. Crucially, this scales polynomially in d rather than exponentially, in contrast to techniques

a) Standard QSP:



b) Parallel QSP:

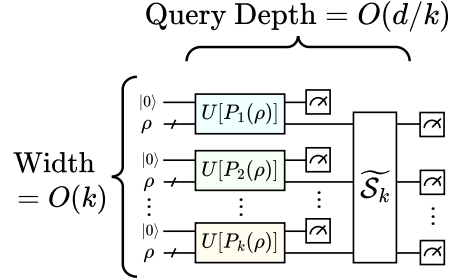


Figure 1: Illustration of standard QSP (a) vs. parallel QSP (b). The operators $U[P(\rho)]$ denote block-encodings of $P(\rho)$ (as depicted in the inset), realized through QSP. For a degree- d polynomial, standard QSP generally requires query depth $2d = O(d)$. In contrast, parallel QSP distributes the computation over k threads by implementing factor polynomials $P_k(x)$ in parallel, and achieves a reduced query depth $O(d/k)$. This is accomplished using a *swap test*, schematically denoted here by an operation $\tilde{S}_k$ and subsequent measurements; see Sec. 3.1 for explanation.

like error mitigation that feature super-polynomial scaling in depth [27]. On the other hand, this comes with an exponential cost in the number of threads, similar to that seen in quantum circuit cutting [28].

An informal statement of our main result is given in the following Theorem 1.1. For intuition, we also provide a comparison of standard QSP and parallel QSP in Fig. 1.

Theorem 1.1 (Informal statement of Theorem 4.3). *Let $P(x)$ be a real-valued polynomial of degree d , that is bounded as $\max_{x \in [-1,1]} |P(x)| \leq 1$. Given access to an input state ρ and a block encoding thereof, we can invoke parallel QSP across k threads to estimate the property*

$$w = \text{tr}(P(\rho)) \quad (1)$$

with a circuit of width $O(k)$ and query depth at most $\approx d/2k$. The number of measurements required to resolve

w to additive error ϵ is

$$O\left(\frac{\text{poly}(d)2^{O(k)}}{\epsilon^2}\right), \quad (2)$$

where the terms $\text{poly}(d)2^{O(k)}$ depend on the chosen factorization of $P(x)$.

We provide an explicit polynomial decomposition and factorization that ensures the measurement overhead scales as $O(\|P\| \cdot d^4(1 + \sqrt{2})^{4k}/k^2)$ for a function norm $\|P\|$ defined later. This norm depends on the polynomial P , and in general scales as a polynomial in d , and in some cases even remains constant. In the worst possible case (corresponding to the most ill-behaved polynomial), we show that the function norm scales as $O(d^{2k+1}/(k!)^2)$, i.e., as a degree- $2k+1$ polynomial in d , that decays super-factorially with k and dominates the $2^{O(k)}$ factor in the overhead.

Toward an exposition of the main theorem, we review QSP and its application to density matrices and trace estimation in Sec. 2. Following this, in Sec. 3 we present parallel QSP, including a characterization of the achievable class of polynomials. We then adapt parallel QSP to arbitrary property estimation problems in Sec. 4, and exemplify this construction in Sec. 5 for the estimation of Rényi entropies, partition functions, and the von Neumann entropy. Discussion and comparison with alternative methods are included in Sec. 6, with detailed proofs of results confined to the appendices.

2 Preliminaries

In this section, we review the preliminaries for parallel QSP: standard QSP (Sec. 2.1), its application to density matrices (Sec. 2.2), and its use in estimating the trace of matrix functions (Sec. 2.3).

2.1 Quantum Signal Processing

Quantum signal processing (QSP) is a method for realizing a polynomial transformation of a quantum subsystem [2, 3, 4]. The QSP algorithm works by interleaving a *signal operator* U , and a *signal processing operator* S . Conventionally, U is taken to be an x -rotation through a fixed angle and S a z -rotation through variable angle ϕ :

$$U(x) = \begin{bmatrix} x & i\sqrt{1-x^2} \\ i\sqrt{1-x^2} & x \end{bmatrix}, \quad S(\phi) = e^{i\phi Z}. \quad (3)$$

Introducing a set of $d+1$ QSP phases $\vec{\phi} = (\phi_0, \phi_1, \dots, \phi_d) \in \mathbb{R}^{d+1}$, the following QSP sequence is

defined as an interleaved product of U and S :

$$U_{\vec{\phi}}(x) = S(\phi_0) \prod_{i=1}^d U(x) S(\phi_i). \quad (4)$$

The matrix elements of the QSP sequence are manifestly polynomials of x :

$$U_{\vec{\phi}} = \begin{bmatrix} P(x) & iQ(x)\sqrt{1-x^2} \\ iQ^*(x)\sqrt{1-x^2} & P^*(x) \end{bmatrix}, \quad (5)$$

where $P(x)$ and $Q(x)$ are polynomials parameterized by $\vec{\phi}$ that obey

1. $\deg(P) \leq d$, $\deg(Q) \leq d-1$;
 2. $P(x)$ has parity $d \bmod 2$, and $Q(x)$ has parity $(d-1) \bmod 2$;
 3. $|P(x)|^2 + (1-x^2)|Q(x)|^2 = 1$, $\forall x \in [-1, 1]$.
- (6)

This result implies that one can construct polynomials in x by projecting into a block of $U_{\vec{\phi}}$, e.g. $\langle 0|U_{\vec{\phi}}|0\rangle = P(x)$. Importantly, realizing a degree- d polynomial necessitates d sequential calls to the signal operator, translating to a *query depth* d , or a circuit depth $O(d)$.

While the conditions of Eq. (6) restrict the class of realizable polynomials, a broader class of polynomials can be implemented by projecting into other bases and using extensions of QSP. For instance, the $|+\rangle\langle+|$ matrix element $\langle+|U_{\vec{\phi}}|+\rangle = \text{Re}(P(x)) + i\text{Re}(Q(x))\sqrt{1-x^2}$ can realize any real polynomial of definite parity, obviating condition 3 above. Even more powerful is *generalized QSP*, introduced in Ref. [29] as an extension of the QSP sequence in Eq. (4). As we review in Appendix A, generalized QSP enables one to design an arbitrary polynomial $P(x)$ restricted only by the condition $|P(x)| \leq 1$ over $x \in [-1, 1]$. This encompasses complex polynomials and those of indefinite parity. To realize a degree- d polynomial, generalized QSP requires a query depth $2d$.

By this reasoning, QSP can encode polynomials that need only be bounded as $\|P\|_{[-1,1]} \leq 1$, where $\|\cdot\|_{[-1,1]}$ is the function norm

$$\|f\|_{[-1,1]} := \max_{x \in [-1,1]} |f(x)|. \quad (7)$$

For an arbitrary degree- d polynomial, the requisite query depth is $2d$; however, the query depth reduces to d for a polynomial of definite parity. In addition, the converse of this result holds: for any polynomial $\|P\|_{[-1,1]} \leq 1$, there exist corresponding QSP phases that can be efficiently computed with a classical algorithm [7, 8, 30, 31, 32], thus amounting to a classical pre-computation step.

Remarkably, the methodology of QSP can be extended to prepare a polynomial transformation of a Hermitian operator through its extension to the quantum eigenvalue transformation (QET) [3, 4, 5]. This is achieved analogous to QSP: provided access to a unitary that block-encodes an operator A in a matrix element, we can design a sequence that encodes a polynomial transformation $P(A)$:

$$U[A] = \begin{bmatrix} A & \cdot \\ \cdot & \cdot \end{bmatrix} \mapsto U_{\tilde{\phi}}[A] = \begin{bmatrix} P(A) & \cdot \\ \cdot & \cdot \end{bmatrix}, \quad (8)$$

where the unspecified entries ensure unitarity. Unitarity also requires $\|A\| \leq 1$ and $\|P(A)\| \leq 1$; otherwise, these entries must be rescaled by a constant to meet these conditions. Mirroring Eq. (4), $U_{\tilde{\phi}}[A]$ is an interleaved sequence of $U[A]$ and parameterized rotations. Essentially, this applies QSP within each eigenspace of A and outputs a degree- d polynomial transformation $P(A)$. As above, the cost of realizing an arbitrary degree- d polynomial is $2d$ sequential queries to the block-encoding of A , translating to a query depth $2d$, although this reduces to d for a polynomial of definite parity. Lastly, while Eq. (8) specializes to a block-encoding in the $|0\rangle\langle 0|$ matrix element, one can more generally take A to be accessed by orthogonal projectors Π, Π' as $A = \Pi U[A] \Pi'$.

2.2 QSP On Density Matrices

To exemplify QSP, let us consider its application to a density matrix ρ . This requires a block encoding of ρ , which is directly achievable (sans rescaling) because the norm $\|\rho\| \leq 1$ for any state.

While in principle there exist various methods to block encode a density matrix ρ , a sufficient oracle is a unitary that prepares a purification of ρ [4]. This oracle model is known as the *quantum purified query access model*, and has been used in recent works on quantum entropy estimation and property testing [33, 34]. To see how this model works, let $\rho = \sum_j p_j |\chi_j\rangle\langle\chi_j|$ be an n -qubit density matrix, and V_ρ be a unitary that prepares a purification of ρ as

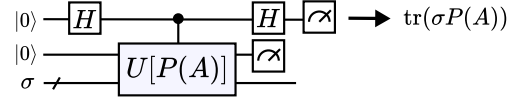
$$V_\rho|0\rangle^{\otimes 2n} = |\psi_\rho\rangle_{AB} = \sum_j \sqrt{p_j} |j\rangle_A |\chi_j\rangle_B \quad (9)$$

on n -qubit subsystems A and B , such that $\text{tr}_A(|\psi_\rho\rangle\langle\psi_\rho|_{AB}) = \rho_B$. Then, introduce an additional n -qubit system C , and let SWAP_{BC} be an operator that swaps subsystems B and C . One can then show that ρ is block encoded in the operator $U[\rho] := (V_\rho^\dagger)_{AB} \cdot \text{SWAP}_{BC} \cdot (V_\rho)_{AB}$ as [4]

$$(|0\rangle_{AB}^{\otimes 2n} \otimes I_C) \cdot U[\rho] \cdot (|0\rangle_{AB}^{\otimes 2n} \otimes I_C) = \rho_C. \quad (10)$$

Therefore, access to V_ρ enables one to block encode ρ , and thus apply QSP to generate polynomials $P(\rho)$.

a) Hadamard Test



b) QSP Test

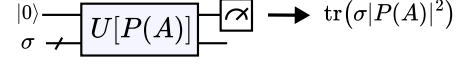


Figure 2: **a)**: The Hadamard test for trace estimation of QSP polynomials, specialized to a block-encoding in the $|0\rangle\langle 0|$ block. Here, the second register corresponds to the qubit that block-encodes $P(A)$; it is measured and post-selected on the outcome $|0\rangle$, thereby projecting out $P(A)$ and enabling the extraction of the trace $\text{tr}(\sigma P(A))$. **b)**: The QSP test for trace estimation of QSP polynomials, also specialized to a block-encoding in the $|0\rangle\langle 0|$ block. Here again, the top register block-encodes $P(A)$ and is measured to extract the trace $\text{tr}(\sigma |P(A)|^2)$.

2.3 Trace Estimation with QSP

Among the applications of QSP, a notable use case is to estimate the trace of a matrix function, i.e. $\text{tr}(f(A))$. There are two methods that we will explore here for estimating such a trace with QSP, both of which approximate $f(A)$ with a QSP polynomial $P(A)$. However, the first method estimates $\text{tr}(P(A))$ with a *Hadamard test* [35], while the second method, which we will refer to as the *QSP test*, estimates the squared trace $\text{tr}(|P(A)|^2)$ by measuring the block-encoding qubit(s). Here we discuss both methods and specialize to the block-encoding convention $\Pi = |0\rangle\langle 0|$ for ease of presentation.

2.3.1 Hadamard Test

In the first method, the Hadamard test [35] is applied to an input state σ , with the target unitary set to a QSP sequence that block encodes $P(A)$. We illustrate this circuit in Fig. 2a. As explained in Lemma 9 of Ref. [36], upon post-selecting the block-encoding qubit(s) to project out the target polynomial $P(A)$, the probability of measuring the ancilla qubit in the state $|0\rangle$ is

$$p_1 = \frac{1}{2} + \frac{1}{2} \text{Re}[\text{tr}(\sigma P(A))]. \quad (11)$$

By instead applying a conjugated phase gate to the ancilla qubit after the Hadamard gate, the probability of measuring $|0\rangle$ becomes

$$p_2 = \frac{1}{2} + \frac{1}{2} \text{Im}[\text{tr}(\sigma P(A))], \quad (12)$$

such that the full trace can be reconstructed as $\text{tr}(\sigma P(A)) = 2p_1 - 1 + i(2p_2 - 1)$. By estimating p_1

and p_2 each to error at most $\epsilon/4$, one obtains an approximation to the trace $\text{tr}(\sigma P(A))$ with error at most ϵ . By the central limit theorem, this requires $O(1/\epsilon^2)$ measurements.

2.3.2 QSP Test

In the second method, one applies the QSP sequence to an input state $|0\rangle\langle 0| \otimes \sigma$, and then estimates the probability that the block-encoding qubit is measured in the state $|0\rangle$. This is equivalently the probability that the correct block of the QSP sequence is applied to the input state, which is

$$p = \text{tr}(\sigma |P(A)|^2). \quad (13)$$

Therefore, an estimation of this to error ϵ furnishes an approximation of the trace $\text{tr}(\sigma |P(A)|^2)$ with error ϵ , and requires $O(1/\epsilon^2)$ measurements. We will refer to this method as the *QSP test*, and depict its circuit in Fig. 2b.

The QSP test is distinct from the Hadamard test in that the QSP polynomial $P(A)$ is squared in the trace. While the Hadamard test can evaluate traces involving the QSP polynomial $P(A)$ directly (i.e., $\text{tr}(\sigma P(A))$), the QSP test is limited to traces involving its magnitude squared $|P(A)|^2$ (i.e., $\text{tr}(\sigma |P(A)|^2)$). This leads to a tradeoff in the capabilities of these approaches, where the polynomial $|P(A)|^2$ is restricted to be real and non-negative, yet is of twice the degree of $P(A)$.

2.3.3 Utility in Trace Estimation

Both the Hadamard test and the QSP test can be used to estimate the trace of a matrix function $\text{tr}(f(A))$ by setting the input state to the maximally mixed state $\sigma = I/2^n$, where n is the number of qubits. By then selecting a polynomial that approximates $f(x)$ as $P(x) \approx f(x)$ or $|P(x)|^2 \approx f(x)$, respectively, both methods output an approximation to $\text{tr}(f(A))/2^n$. Due to this rescaling by 2^n , resolving $\text{tr}(f(A))$ to error ϵ requires a number of measurements $O(2^{2n}/\epsilon^2)$. That this cost scales exponentially with the number of qubits is a generic feature, because arbitrary traces $\text{tr}(f(A))$ can be exponentially large in the dimension of A . Ref. [33] uses this approach to develop an algorithm for estimating the α -Rényi entropy of a density matrix, and notes the same cost scaling.

However, for estimating a trace $\text{tr}(f(\rho))$ of a density matrix ρ , it is advantageous to set both the input state and block encoding to be ρ , i.e., $\sigma = \rho$ and $A = \rho$. In this case, the Hadamard test and QSP test output the traces $\text{tr}(\rho P(\rho))$ and $\text{tr}(\rho |P(\rho)|^2)$, respectively. From these traces, one can estimate $\text{tr}(f(\rho))$ by selecting polynomials that satisfy $xP(x) \approx f(x)$ or $x|P(x)|^2 \approx f(x)$,

respectively. Importantly, this approach circumvents the rescaling by 2^n , such that estimating $\text{tr}(f(\rho))$ to error ϵ requires $O(1/\epsilon^2)$ measurements. This streamlined approach is used in Ref. [37] to design new algorithms for estimating the α -Rényi entropy and von Neumann entropy, while avoiding an exponentially large number of measurements.

Lastly, while both the Hadamard and QSP tests achieve complexity $O(1/\epsilon^2)$, we note that one could alternatively use techniques like amplitude/phase estimation to reduce the complexity to $O(1/\epsilon)$. However, this complexity corresponds to a large $O(1/\epsilon)$ depth [38, 39]. As the central focus of this work is reducing depth, we forgo these techniques in favor of the Hadamard and QSP tests, which achieve shallower depths at the expense of an increased measurement overhead.

3 Parallel Quantum Signal Processing

With the preliminaries laid out, we now present our algorithm for parallel QSP. In its simplest incarnation, parallel QSP enables the estimation of a trace of the form $\text{tr}(\rho^k R(\rho))$, where $R(x)$ is a degree- d polynomial, and k is the number of systems over which the computation is parallelized, i.e., the *number of threads*. While standard QSP can compute this trace with query depth $\sim d + k$, parallel QSP achieves this computation with a query depth $\approx d/k$. This is achieved by factorizing $R(x)$ into k polynomials of degree $O(d/k)$, which are implemented in parallel with QSP, and subsequently multiplied together with a *generalized swap test* (Sec. 3.1). However, this depth reduction of parallel QSP is realized at the expense of increasing the circuit width to $O(k)$, and the number of measurements by a factor that depends on the chosen factorization of $R(x)$. Moreover, while $\text{tr}(\rho^k R(\rho))$ encompasses a limited class of properties, later in Sec. 4 we expand this class to arbitrary polynomial functions.

In this section, we first review the generalized swap test (Sec. 3.1), which will underpin parallel QSP. We then present the parallel QSP algorithm (Sec. 3.2), including a characterization of the achievable polynomials and a discussion of its resource requirements, and conclude by commenting on the implications of our algorithm (Sec. 3.3).

3.1 The Generalized Swap Test

An essential ingredient of parallel QSP is a tool that we will refer to as *the generalized swap test*. As its name suggests this is an extension of the usual swap test, introduced in Ref. [20] to measure the overlap between two quantum states, i.e. $\text{tr}(\rho\sigma)$. Explicitly, the generalized swap test uses the identity that the expectation

value of a cyclic shift applied to a product state $\rho^{\otimes k}$ (for an integer $k \geq 1$), is equal to the trace of the multiplicative product [40]:

$$\text{tr}(\mathcal{S}_k \cdot \rho^{\otimes k}) = \text{tr}(\rho^k), \quad (14)$$

where $\mathcal{S}_k$ is a cyclic shift on the k systems comprising $\rho^{\otimes k}$, and acts as

$$\mathcal{S}_k[|\psi_1\rangle|\psi_2\rangle|\psi_3\rangle\cdots|\psi_k\rangle] = |\psi_k\rangle|\psi_1\rangle|\psi_2\rangle\cdots|\psi_{k-1}\rangle. \quad (15)$$

Notably, this identity converts a tensor product $\rho^{\otimes k}$ to a multiplicative product ρ^k , and reduces to the usual swap test for $k = 2$. In addition, this identity holds for a tensor product of distinct states ρ_j :

$$\text{tr}\left(\mathcal{S}_k \cdot \bigotimes_{j=1}^k \rho_j\right) = \text{tr}\left(\prod_{j=1}^k \rho_j\right). \quad (16)$$

Using the generalized swap test, one can estimate the trace $\text{tr}(\rho^k)$ by measuring the expectation value of $\mathcal{S}_k$ on k copies of ρ . Because the states comprising $\rho^{\otimes k}$ can be arranged in parallel in a quantum circuit, this effectively parallelizes the computation of the multiplicative product ρ^k , without ever having to explicitly multiply ρ sequentially. Accordingly, the generalized swap test has been employed to compute Rényi entropies in quantum Monte Carlo [41], estimate nonlinear functions of state on a quantum computer [40, 42, 23], and perform entanglement spectroscopy [43, 18, 44].

In practice, the expectation value of $\mathcal{S}_k$ can be estimated with various techniques. While an elementary implementation as a Hadamard test applied to the cyclic shift operator translates to a depth $O(k)$ [18, 45], recent works have put forth novel constructions of the generalized swap test that achieve $O(1)$ quantum depth [44, 23]. Ref. [44] achieves this using $2k$ copies of a purification of ρ and additional classical post-processing, leading to an $O(1)$ depth independent of both n and k . Alternatively, Ref. [23] prepares an ancilla system in a special GHZ state, from which the cyclic shift $\mathcal{S}_k$ can be measured in depth $O(1)$. Ultimately, these results demonstrate that the generalized swap test can estimate the trace $\text{tr}(\rho^k)$ with a circuit of width $O(k)$ and depth $O(1)$, thus fully parallelizing the computation of the multiplicative product.

3.2 Parallel QSP

Parallel QSP is a synthesis of the QSP test (Sec. 2.3) and the generalized swap test (Sec. 3.1). At a high level, parallel QSP works by first using QSP to implement block encodings of k polynomials $\{P_j(\rho)\}_{j=1}^k$ across k threads, and separately applying each to an input state ρ . Then applying the generalized swap test

to the resulting state, we can extract the trace of the corresponding multiplicative product:

$$z := \text{tr}\left(\prod_{j=1}^k P_j(\rho)\rho P_j(\rho)^\dagger\right) = \text{tr}\left(\rho^k \prod_{j=1}^k |P_j(\rho)|^2\right). \quad (17)$$

By appealing to the fundamental theorem of algebra, the product $\prod_{j=1}^k |P_j(\rho)|^2$ can represent an arbitrary real, non-negative polynomial. If this target polynomial is of degree d , then each polynomial factor $P_j(\rho)$ can be guaranteed to have degree at most $\approx \frac{d}{2k}$, thus dividing the query depth by $O(k)$. While the trace z encompasses only a limited class of functions, in Sec. 4 we show that an arbitrary polynomial can be decomposed into this form and made amenable to parallel QSP, enabling general property estimation algorithms at reduced query depth.

As a hybrid of the QSP test and the generalized swap test, parallel QSP requires access to both ρ and a block encoding of ρ . As shown in Sec. 2.2, the purified query access model provides an oracle that prepares a purification of ρ and thus furnishes a block encoding of ρ . This oracle also provides access to ρ by disregarding the ancilla system, and thus is sufficient for parallel QSP. Nonetheless, this is not the only possibility, as other oracles can also provide access to both ρ and a block encoding thereof.

3.2.1 The Parallel QSP Algorithm

To sharpen our analysis, we first present the parallel QSP circuit in Fig. 3. The initial state of the algorithm is a product state $\rho^{\otimes k}$ across the k threads, as well as ancilla qubits used to access block encodings. The circuit then consists of (1) the QSP stage, and (2) the generalized swap test stage. The QSP stage comprises k unitaries $\{U[P_j(\rho)]\}_{j=1}^k$ that block encode polynomials $P_j(\rho)$, realized by QSP. We apply each unitary to the input state in parallel and post-select on the successful application of $P_j(\rho)$. Collectively, this succeeds with probability

$$\text{Pr}(\text{QSP Success}) = \prod_{j=1}^k \text{tr}[P_j(\rho)\rho P_j(\rho)^\dagger], \quad (18)$$

and outputs the product state

$$\bigotimes_{j=1}^k \frac{P_j(\rho)\rho P_j(\rho)^\dagger}{\text{tr}[P_j(\rho)\rho P_j(\rho)^\dagger]}. \quad (19)$$

Next, we apply the generalized swap test to this product state to compute the trace of the corresponding mul-

multiplicative product, which we denote by $\tilde{z}$:

$$\begin{aligned}\tilde{z} &:= \text{tr} \left(\prod_{j=1}^k \frac{P_j(\rho) \rho P_j(\rho)^\dagger}{\text{tr} [P_j(\rho) \rho P_j(\rho)^\dagger]} \right) \\ &= \frac{\text{tr} \left(\rho^k \prod_{j=1}^k |P_j(\rho)|^2 \right)}{\prod_{j=1}^k \text{tr} [\rho |P_j(\rho)|^2]} = \frac{z}{\text{Pr}(\text{QSP Success})}.\end{aligned}\quad (20)$$

We can then estimate z by resolving $\tilde{z}$ and $\text{Pr}(\text{QSP Success})$ to sufficient accuracy. This is the essence of parallel QSP: the computation of a trace of a product of polynomials, by executing these polynomials in parallel rather than sequentially.

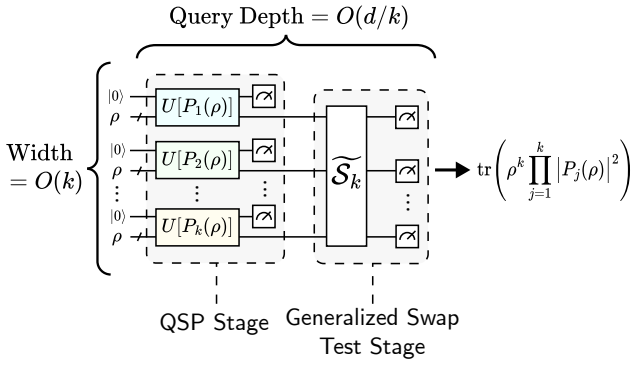


Figure 3: The quantum circuit for the parallel QSP algorithm. The operations $U[P_j(\rho)]$ are unitary block-encodings of polynomials $P_j(\rho)$, realized with QSP. For illustrative simplicity, we specialize to block encodings in the $|0\rangle\langle 0|$ block. Upon application of these polynomials to initial states ρ , one enacts a generalized swap test, denoted schematically as $\widetilde{S}_k$ and subsequent measurements; this serves as a symbolic proxy for the various implementations of the generalized swap test [18, 23, 44]. Observe that the parallel QSP circuit has a reduced query depth $O(d/k)$, at the expense of increasing its width to $O(k)$.

With this understanding, we formalize the parallel QSP algorithm with the following theorem:

Theorem 3.1 (Parallel QSP). *Provided access to a density matrix ρ and a block encoding thereof, the parallel QSP circuit executed across k threads enables the estimation of the quantity*

$$z = \text{tr} \left(\rho^k \prod_{j=1}^k |P_j(\rho)|^2 \right), \quad (21)$$

where each $P_j(\rho)$ is a block-encoded polynomial implemented with QSP. More specifically, z can be estimated to additive error ϵ by running the parallel QSP circuit $O(\frac{1}{\epsilon^2})$ times, where the requisite query depth is $2 \max_j \{\deg(P_j)\}$ and the circuit width is $O(k)$.

Proof. Using the parallel QSP circuit of Fig. 3, consider the measurement cost of resolving $z =$

$\text{Pr}(\text{QSP Success}) \times \tilde{z}$ to additive error ϵ . Obviously, $\text{Pr}(\text{QSP Success})$ is a probability and can naturally be estimated by repeatedly running the circuit. On the other hand, the expression for $\tilde{z}$ depends on the chosen implementation of the generalized swap test, but in general can be expressed as an expectation value.

For instance, a simple implementation of the generalized swap test is provided by applying a Hadamard test to $\mathcal{S}_k$. In this case, the probability of measuring the ancilla qubit (of the Hadamard test) in the state $|0\rangle$ upon successful application of each QSP sequence is

$$\text{Pr}(\text{Ancilla} = |0\rangle \mid \text{QSP Success}) = \frac{1}{2}(1 + \tilde{z}). \quad (22)$$

Then we can express z as

$$\begin{aligned}z &= \text{Pr}(\text{QSP Success}) \\ &\quad \times \left(2 \cdot \text{Pr}(\text{Ancilla} = |0\rangle \mid \text{QSP Success}) - 1 \right) \\ &= 2 \cdot \text{Pr}(\text{QSP Success, and Ancilla} = |0\rangle) \\ &\quad - \text{Pr}(\text{QSP Success}).\end{aligned}\quad (23)$$

Therefore, estimating both of these probabilities to additive error $\epsilon/3$ provides an approximation to z with additive error ϵ , and the central limit theorem implies that this requires $O(1/\epsilon^2)$ runs of the parallel QSP circuit. While this specializes to a specific implementation of the generalized swap test, this result is in fact general, because other implementations also approximate $\tilde{z}$ as a combination of expectation values.

Next, consider the query depth of the parallel QSP circuit. In the QSP stage, each unitary $U[P_j(\rho)]$ requires query depth at most $2 \deg(P_j)$ for an arbitrary polynomial as we discussed in Sec. 2.1, corresponding to a total query depth $2 \max_j (\deg(P_j))$. On the other hand, the generalized swap test makes no queries to the block encoding and does not contribute to the query depth. It however can contribute to the circuit depth depending on its implementation, but this can be reduced to $O(1)$ using the constructions of Refs. [44, 23]

Lastly, as the QSP stage consists of k unitaries enacted in parallel across k systems, its width is $O(k)$. Likewise, while the precise width of the generalized swap test stage depends on its implementation, the constructions of Refs. [18, 45, 44, 23] use $O(k)$ system copies arranged in parallel, equating to a width $O(k)$. $\square$

3.2.2 Characterization of Parallel QSP Polynomials

According to Theorem 3.1, the parallel QSP algorithm estimates the trace $z = \text{tr} \left(\rho^k \prod_{j=1}^k |P_j(\rho)|^2 \right)$, thus parallelizing the computation of polynomials that take the form $x^k \prod_{j=1}^k |P_j(x)|^2$. We can characterize this class of polynomials by appealing to the fundamental theorem of algebra:

Lemma 3.1 (Factorization of Real, Non-negative Polynomials). *Consider a polynomial $R(x)$ of even degree d that is real and non-negative over the real axis $x \in \mathbb{R}$. For real inputs x , this polynomial can be expressed as the product of the squared magnitudes of $k \leq d$ factor polynomials $\mathcal{R}_j(x)$:*

$$R(x) = \prod_{j=1}^k |\mathcal{R}_j(x)|^2. \quad (24)$$

While the factor polynomials $\mathcal{R}_j(x)$ are not unique, there exists a factorization in which every factor polynomial is guaranteed to have degree at most $\deg(\mathcal{R}_j) \leq \lceil d/2k \rceil$.

Proof. Applying the fundamental theorem of algebra to a polynomial $R(x)$ of even degree¹ d that is real and non-negative over $x \in \mathbb{R}$, implies that its real roots have even multiplicity, and its complex roots (with non-zero imaginary part) come in complex conjugate pairs. Accordingly, $R(x)$ can be written as

$$R(x) = C \prod_{i=1} (x - r_i)^{2\alpha_{r_i}} \prod_{l=1} (x - c_l)^{\beta_{c_l}} (x - c_l^*)^{\beta_{c_l}}, \quad (25)$$

for distinct real roots r_i of even multiplicity $2\alpha_{r_i}$, and distinct complex roots c_l of multiplicity β_{c_l} , and a coefficient $C \in \mathbb{R}$. For real $x \in \mathbb{R}$, $R(x)$ can therefore be expressed as

$$R(x) = \left| \sqrt{C} \prod_{i=1} (x - r_i)^{\alpha_{r_i}} \prod_{l=1} (x - c_l)^{\beta_{c_l}} \right|^2 = |\mathcal{R}(x)|^2, \quad (26)$$

where $\mathcal{R}(x)$ is a degree $d/2$ polynomial.

To show the decomposition stated in this theorem, we need to factorize $\mathcal{R}(x)$ into a product of k factor polynomials: $\mathcal{R}(x) = \prod_{j=1}^k \mathcal{R}_j(x)$. This can be achieved by partitioning the $d/2$ terms in Eq. (26) into k groups, and defining $\mathcal{R}_j(x)$ as the product over terms in the j th group, times $C^{1/2k}$. Many such groupings exist, so a factorization of $\mathcal{R}(x)$ is not unique. Nonetheless, one can partition the roots such that the first $d/2 \bmod k$ groups are of size $\lfloor d/2k \rfloor + 1$, and the remaining groups are of size $\lfloor d/2k \rfloor$. If $d/2$ divides k , then the maximal size is $\lfloor d/2k \rfloor = d/2k$; if $d/2$ does not divide k , then the maximal size is $\lfloor d/2k \rfloor + 1 = \lceil d/2k \rceil$. In either case, this guarantees that each factor polynomial has degree at most $\deg(\mathcal{R}_j) \leq \lceil d/2k \rceil$. $\square$

By Lemma 3.1, an arbitrary real, non-negative polynomial of even degree d can be decomposed into a product of k factor polynomials squared: $R(x) =$

¹Note that the degree is necessarily even; otherwise, the condition of non-negativity for all $x \in \mathbb{R}$ cannot be obeyed.

$\prod_{j=1}^k |\mathcal{R}_j(x)|^2$, where the factor polynomials are of degree at most $\lceil d/2k \rceil = O(d/k)$. This decomposition makes the polynomial $R(x)$ amenable to parallel QSP according to Theorem 3.1, given that we implement the factor polynomials $\mathcal{R}_j(x)$ with QSP. With this insight, we can characterize the class of polynomials achievable with parallel QSP:

Theorem 3.2 (Parallel QSP Polynomial Characterization). *Let $R(x)$ be a polynomial of even degree d , that is real and non-negative over the real axis $x \in \mathbb{R}$. By Lemma 3.1, let $R(x)$ factorize into k factor polynomials as $R(x) = \prod_{j=1}^k |\mathcal{R}_j(x)|^2$, where $\deg(\mathcal{R}_j) \leq \lceil d/2k \rceil$. Invoking parallel QSP across k threads with block-encoded polynomials $\mathcal{R}_j(x)$, we can estimate the trace*

$$z = \text{tr}(\rho^k R(\rho)). \quad (27)$$

The requisite query depth is at most $2\lceil d/2k \rceil \approx d/k$ and the circuit width is $O(k)$. The number of measurements required to estimate z to additive error ϵ is

$$O\left(\frac{\mathcal{K}(R)^4}{\epsilon^2}\right), \quad (28)$$

where $\mathcal{K}(R)$ is a quantity we call the “factorization constant”, whose value depends on the chosen factorization of $R(x)$ as²

$$\mathcal{K}(R) = \prod_{j=1}^k \|\mathcal{R}_j\|_{[-1,1]}, \quad (29)$$

where the norm $\|\cdot\|_{[-1,1]}$ was defined in Eq. (7).

Proof. The density matrix ρ is Hermitian and its eigenvalues real. Therefore, as per Lemma 3.1, the action of $R(x)$ on a ρ factorizes as a product of factor polynomials:

$$R(\rho) = \prod_{j=1}^k |\mathcal{R}_j(\rho)|^2, \quad (30)$$

where $\deg(\mathcal{R}_j) \leq \lceil d/2k \rceil$. Provided block encodings of the factor polynomials, we can apply the results of Theorem 3.1 to extract the trace $\text{tr}(\rho^k \prod_{j=1}^k |\mathcal{R}_j(\rho)|^2) = \text{tr}(\rho^k R(\rho))$, as desired. Therefore, all that remains is to construct block encodings of the factor polynomials with QSP.

However, the factor polynomials do not necessarily obey the conditions of QSP polynomials. For instance, even if $\|R\|_{[-1,1]} \leq 1$, it is not necessarily true that the factor polynomials also obey this condition. In addition,

²Note that the factorization constant is fundamentally a function of the factor polynomials $\{\mathcal{R}_j(x)\}$. However, for brevity of notation, we denote it as a function of $R(x)$.

the factor polynomials are in general not of fixed parity. Hence, in full generality, these factor polynomials must be implemented by rescaling by a constant and using a tool like generalized QSP.

In more detail, we can implement an arbitrary factor polynomial $\mathcal{R}_j(x)$ by rescaling as

$$\frac{\mathcal{R}_j(x)}{\|\mathcal{R}_j\|_{[-1,1]}}, \quad (31)$$

which guarantees that this is bounded in magnitude by 1. We can then block-encode this rescaled polynomial with generalized QSP. As we discussed in Sec. 2.1, the requisite query depth of this procedure is $\leq 2\lceil d/2k \rceil$. Note however that this simplifies if $\mathcal{R}_j(x)$ is of fixed parity and can be implemented with standard QSP; in this simpler case, the query depth is at most $\lceil d/2k \rceil$.

Then, to estimate $z = \text{tr}(\rho^k R(\rho))$, we block encode the k rescaled factor polynomials $\mathcal{R}_j(x)/\|\mathcal{R}_j\|_{[-1,1]}$ in parallel and execute the parallel QSP circuit of Fig. 3, which produces an estimate of

$$\text{tr} \left(\rho^k \prod_{j=1}^k \left| \frac{\mathcal{R}_j(\rho)}{\|\mathcal{R}_j\|_{[-1,1]}} \right|^2 \right) = \frac{\text{tr}(\rho^k R(\rho))}{\prod_{j=1}^k \|\mathcal{R}_j\|_{[-1,1]}^2} = \frac{z}{\mathcal{K}(R)^2}, \quad (32)$$

where $\mathcal{K}(R) := \prod_{j=1}^k \|\mathcal{R}_j\|_{[-1,1]}$ is the *factorization constant*, which depends on the chosen factorization of $R(x)$. In order to resolve z to additive error ϵ , it suffices to resolve Eq. (32) to additive error $\epsilon/\mathcal{K}(R)^2$. According to Theorem 3.1, this requires a number of measurements $O(\mathcal{K}(R)^4/\epsilon^2)$. $\square$

Theorem 3.2 furnishes the following algorithm for estimating the trace $z = \text{tr}(\rho^k R(\rho))$, whose pseudocode we present in Algorithm 1. The first step is to factorize $R(x)$, either analytically or numerically. This can be achieved numerically by determining the roots of $R(x)$ by computing the eigenvalues of its companion matrix [46]; for a degree- d polynomial, this requires $O(d^3)$ time and can be performed as a classical pre-computation step. The next step is to implement the factor polynomials with QSP, and then finally run the parallel QSP circuit as per Theorem 3.1 to obtain an estimate of z .

We will refer to the number of measurements required by parallel QSP as its *measurement cost*. From Theorem 3.2, this is $O(\mathcal{K}(R)^4/\epsilon^2)$, which crucially depends on the chosen factorization of $R(x)$ through the factorization constant $\mathcal{K}(R)$. The factorization constant measures the cost of implementing the factor polynomials, which in general requires rescaling. A poor choice of factorization can result in this constant scaling exponentially in the degree d , dashing any quantum advantage provided by this protocol. For instance, by factorizing

Algorithm 1: Parallel Quantum Signal Processing

Input: (1) Access to a state ρ and a block encoding of ρ ; (2) a polynomial $R(x)$ of even degree d , that is real and non-negative over $x \in \mathbb{R}$.

Output: An estimate of $z = \text{tr}(\rho^k R(\rho))$ to additive error ϵ

Cost : $O(\mathcal{K}(R)^4/\epsilon^2)$ executions of a circuit of width $O(k)$ and query depth $O(d/k)$, where $\mathcal{K}(R) = \prod_{j=1}^k \|\mathcal{R}_j\|_{[-1,1]}$ is a constant that depends on the chosen factorization of $R(x)$.

Procedure:

- 1 Classically determine a factorization $R(x) = \prod_{j=1}^k |\mathcal{R}_j(x)|^2$, such that $\deg(\mathcal{R}_j) \leq \lceil d/2k \rceil$ for all j ;
 - 2 Using QSP, construct block encodings of $\mathcal{R}_j(\rho)$ (possibly rescaled as in Eq. (31));
 - 3 Run the parallel QSP circuit of Fig. 3 a number of times $O(\mathcal{K}(R)^4/\epsilon^2)$.
-

the order d Chebyshev polynomial (of the first kind) into two factor polynomials composed of its positive-valued and negative-valued roots respectively, the resulting factorization constant scales as $2^{O(d)}$.

Therefore, to minimize the measurement cost in practice, it is best to select a *low-norm factorization* of $R(x)$, whose factorization constant scales at worst as $\text{poly}(d)2^{O(k)}$, rather than $2^{O(d)}$. This can be achieved by selecting factor polynomials of modest norm, which generally requires some analytic knowledge of the polynomial's structure.³

In addition, as we remarked in the proof of Theorem 3.2, while in general the query depth is upper bounded by $2\lceil d/2k \rceil$, if the factor polynomials can all be chosen to be of definite parity, then they can be implemented through standard QSP with a query depth at most $\lceil d/2k \rceil$. Nonetheless, any realization of parallel QSP will attain a query depth scaling as $O(d/k)$. This is because any circuit that prepares a degree- d polynomial $R(\rho)$ requires $O(d)$ instances of ρ , whether arranged in parallel or in series. If these instances are parallelized over k threads, the query depth across the threads must be at least $O(d/k)$.

³That said, low-norm factorizations are not necessarily rare. For instance, it can be shown that any infinite family of real polynomials whose roots partition the interval $[-1, 1]$ into segments of size $O(1/d)$ admit factorizations with norms $\|\mathcal{R}_j\|_{[-1,1]} = \text{poly}(d)$. This can generically be achieved by interleaving the roots of the factor polynomials.

3.3 Remarks

As presented, the parallel QSP algorithm reduces the query depth needed to compute the trace $z = \text{tr}(\rho^k R(\rho))$. While a standard QSP implementation of the polynomial $\rho^k R(\rho)$ (which has definite parity because $R(x)$ is even) requires a query depth $k + d$, parallel QSP requires a query depth at most $2\lceil d/2k \rceil \approx d/k$. This shrinks the query depth by a factor $\approx k$, yet requires increasing the width to $O(k)$, and the number of measurements by a factor $\mathcal{K}(R)^4$. As $\mathcal{K}(R)$ is a product of k function norms, it generally scales as $2^{O(k)}$. Thus, parallel QSP enables a tradeoff between quantum and classical resources, and is most suitable for platforms limited by short coherence times.

Interestingly, this tradeoff is reminiscent of that encountered in quantum circuit cutting [28, 47]. In that context, a quantum circuit is cut across K wires to decompose it into circuits of smaller depth and/or width, which are repeatedly executed to simulate the original quantum circuit. The corresponding measurement overhead scales as $2^{O(K)}$, resembling that of parallel QSP. Likewise, parallel QSP shares similarities with the randomized QSP algorithms presented in Refs. [48, 49, 50]. These algorithms randomly sample over QSP polynomials of different degrees and reduce the average degree/query depth. However, while parallel QSP reduces the maximal query depth, these randomized algorithms do not, as high-degree polynomials are still sampled. This makes parallel QSP better suited for quantum hardware constrained by coherence times, where deep circuits are out of reach.

Moreover, the depth-width tradeoff afforded by parallel QSP merits careful explanation. While coherence times constrain circuit depths, they also tend to decrease with the number of qubits, making a tradeoff between depth and width nontrivial in general. Crucially however, in parallel QSP this tradeoff is not arbitrary, but rather is carefully structured: the computation is distributed across k independent threads that do not interact until a final generalized swap test. Because this swap test can be executed in constant depth [23], the entangled state across all k threads only needs to be maintained briefly. Consequently, the effective coherence requirement for parallel QSP is governed primarily by the coherence time of an individual thread, rather than that of all k threads. Further, the independence of the threads provides a natural form of error resilience: because errors cannot propagate across threads until the swap test, one can simply use error detection to detect and discard erroneous threads, preventing the accumulation of error across the whole computation.

Furthermore, as currently presented, the scope of parallel QSP is limited to computing functions of a density matrix. This follows from the use of the generalized

swap test, which enables the multiplication of density matrices arranged in parallel. This limited scope is unsurprising: if an arbitrary QSP algorithm could be parallelized, then one could parallelize Hamiltonian simulation and violate the no-fast forwarding theorem [51] which forbids circuit depths sub-linear in the simulation time [52]. Nonetheless, parallel QSP can still be applied to a general operator if it is encoded in a density matrix, such as a Hamiltonian encoded in a thermal state $\rho \propto e^{-\beta H}$ or the state $\rho \propto H + cI$ considered in sample-based Hamiltonian simulation [53].

4 Parallel QSP for Property Estimation

A noteworthy application of QSP is estimating properties of a quantum state, expressed as $\text{tr}(f(\rho))$ for a function $f(\rho)$. For instance, the von Neumann entropy is captured by the function $f(\rho) = -\rho \ln \rho$. In practice, such a property can be estimated with QSP by implementing a polynomial $P(\rho) \approx f(\rho)$, and approximating the trace using the techniques of Sec. 2.3. This approach has established algorithms for evaluating the von Neumann entropy [37], Rényi entropies [33], fidelities [36], and other related properties.

However, as currently framed, parallel QSP suffers from two limitations that render it inapplicable to general property estimation: parallel QSP (1) applies to a limited class of polynomials $x^k R(x)$ where $R(x)$ is real and non-negative over $x \in \mathbb{R}$, and (2) requires knowledge of a low-norm factorization of $R(x)$ to achieve a reasonable measurement cost (e.g., $\text{poly}(d)2^{O(k)}$ rather than $2^{O(d)}$).

In this section we show how to overcome both of these challenges by developing a method that enables parallel QSP to accommodate arbitrary polynomials, while maintaining a query depth $O(d/k)$ and guaranteeing a reasonable measurement cost. This dramatically expands the class of polynomials amenable to parallel QSP, and furnishes property estimation algorithms with reduced query depth.

4.1 Prelude

To formalize our problem of interest, suppose we wish to estimate a property by the trace of a real, degree- d polynomial $P(x)$:

$$w = \text{tr}(P(\rho)), \quad (33)$$

where

$$P(x) = \sum_{n=0}^d a_n x^n, \quad \|P\|_{[-1,1]} \leq 1. \quad (34)$$

For a general $P(x)$, estimating w with standard QSP requires a query depth $2d = O(d)$, and also a number of

measurements $O(1/\epsilon^2)$ to resolve w with additive error ϵ . In contrast, here we will use parallel QSP to parallelize this computation over k threads, and achieve a query depth $O(d/k + k)$. The resulting measurement cost will depend on the chosen decomposition and factorization of $P(x)$.

As we remarked above, parallel QSP cannot be directly applied to an arbitrary polynomial $P(x)$. Instead, in order to parallelize the computation over k threads, we split $P(x)$ into a sum of two *constituent polynomials*:

$$\begin{aligned} P(x) &= \sum_{n=0}^{k-1} a_n x^n + x^k \sum_{n=k}^d a_n x^{n-k} \\ &=: P_{<k}(x) + x^k P_{\geq k}(x). \end{aligned} \quad (35)$$

where,

$$P_{<k}(x) := \sum_{n=0}^{k-1} a_n x^n, \quad P_{\geq k}(x) := \sum_{n=0}^{d-k} a_{n+k} x^n, \quad (36)$$

are the constituent polynomials of $P(x)$. $P_{<k}(x)$ and $P_{\geq k}(x)$ are real polynomials of degree $k-1$ and $d-k$, respectively. With this decomposition, the desired property can be written as a sum of two *constituent traces*

$$\begin{aligned} w &= w_{<k} + w_{\geq k}, \\ w_{<k} &= \text{tr}(P_{<k}(\rho)), \quad w_{\geq k} = \text{tr}(\rho^k P_{\geq k}(\rho)). \end{aligned} \quad (37)$$

Therefore, to estimate w , it will equivalently suffice to estimate $w_{<k}$ and $w_{\geq k}$.

Importantly, $w_{<k}$ is the trace of a polynomial of degree $k-1$, which can be easily estimated with standard QSP at query depth $2(k-1)$ and width $O(1)$. On the other hand, $w_{\geq k}$ is the trace of ρ^k times a polynomial of degree $d-k$, which nearly fits into the framework of parallel QSP. By incorporating appropriate algebraic manipulations to ensure that $P_{\geq k}(x)$ is non-negative, we will estimate $w_{\geq k}$ with parallel QSP at a query depth $\approx (d-k)/k < d/k$ and circuit width $O(k)$. Therefore, the overall requisite query depth to estimate w is guaranteed to never exceed $\approx \max\{2k, d/k\}$. In practice, $k \ll d$ (e.g., a large degree polynomial parallelized over a few threads), in which case the query depth reduces to $\approx d/k$.

In this manner, property estimation with parallel QSP can be viewed as a hybrid of standard and parallel QSP, where the low degree terms are estimated with standard QSP, and the higher degree terms with parallel QSP. Below, we will investigate this procedure more closely. We first consider the case in which $P_{\geq k}(x)$ is non-negative and hence directly amenable to parallel QSP. We next show that even if $P_{\geq k}(x)$ is not non-negative, it can be decomposed into a basis of non-negative polynomials and thus made amenable to parallel QSP. In both situations, we include bounds on the

requisite query depth and measurement costs to estimate w to a desired level of error. In the first situation, the cost crucially depends on the factorization of $P_{\geq k}(x)$ as per Theorem 3.2. In the second case, the measurement cost depends on the factorization of our purported decomposition; we prove the existence of a low-norm factorization such that this contribution scales as $O(d^{42O(k)}/k^2)$.

Lastly, as we will see in the following, although $P(x)$ is bounded as $\|P\|_{[-1,1]} \leq 1$, the constituent polynomials are not necessarily bounded the same: $\|P_{<k}\|_{[-1,1]}, \|P_{\geq k}\|_{[-1,1]} \not\leq 1$. As a result, the measurement cost of estimating w will also depend on the norms $\|P_{<k}\|_{[-1,1]}$ and $\|P_{\geq k}\|_{[-1,1]}$. We prove in Appendix C that for any bounded polynomial $P(x)$, its constituent polynomials are upper bounded as

$$\begin{aligned} \sup_{P(x), \|P\|_{[-1,1]} \leq 1} \|P_{<k}\|_{[-1,1]} &\leq O\left(\frac{d^{k-1}}{(k-1)!}\right), \\ \sup_{P(x), \|P\|_{[-1,1]} \leq 1} \|P_{\geq k}\|_{[-1,1]} &\leq O\left(\frac{d^k}{k!} \sqrt{\frac{d}{k}}\right). \end{aligned} \quad (38)$$

Therefore, for $k = O(1)$, it is necessarily the case that $\|P_{<k}\|_{[-1,1]}, \|P_{\geq k}\|_{[-1,1]} = O(\text{poly}(d))$ scales at worst as a polynomial in d . Moreover, we emphasize that these are worst case bounds, and that many polynomials of interest have constituent polynomials with much smaller norms. For example, the polynomial approximation to the exponential function $e^{-\beta(x+1)}$ (e.g., for thermal state preparation) has constituent polynomials both upper bounded in magnitude by $O(1)$, independent of d .

4.2 Estimation by Direct Application of Parallel QSP

If $P_{\geq k}(x)$ is non-negative, then we can use the above intuition to estimate the property $w = \text{tr}(P(\rho))$ by a direct application of parallel QSP, and achieve a query depth $\approx d/k$:

Theorem 4.1 (Parallel QSP for Property Estimation: Direct Application). *Consider a real polynomial $P(x)$ of degree d , that is bounded as $\|P\|_{[-1,1]} \leq 1$. Let $P(x)$ decompose according to Eq. (35) as*

$$P(x) = P_{<k}(x) + x^k P_{\geq k}(x), \quad (39)$$

and suppose that $P_{\geq k}(x)$ is non-negative over $x \in \mathbb{R}$. By invoking parallel QSP across k threads, we can estimate $w = \text{tr}(P(\rho))$ with query depth at most $\max\{2(k-1), 2\lceil \frac{d-k}{2k} \rceil\} \lesssim \max\{2k, d/k\} = O(d/k+k)$. The number of measurements required to resolve w to additive error

ϵ is

$$O\left(\frac{\|P_{<k}\|_{[-1,1]}^2 + \mathcal{K}(P_{\geq k})^4}{\epsilon^2}\right), \quad (40)$$

where $\mathcal{K}(P_{\geq k})$ is the factorization constant defined in Eq. (29) and depends on the chosen factorization of $P_{\geq k}(x)$.

Proof. Decompose $w = w_{<k} + w_{\geq k}$ as per Eq. (37). We then want to estimate $w_{<k}$ and $w_{\geq k}$ each to error $\leq \epsilon/2$, such that their sum estimates w with error at most ϵ .

First, one can estimate $w_{<k}$ to error $\epsilon/2$ with standard QSP, using for instance a Hadamard test. Because $P_{<k}(x)$ is a real polynomial of degree $k-1$ and indefinite parity, this can be achieved with query depth $2(k-1)$, and a number of measurements $O(\|P_{<k}\|_{[-1,1]}^2/\epsilon^2)$.

Next, if $P_{\geq k}(x)$ is non-negative over $x \in \mathbb{R}$, then the constituent trace $w_{\geq k}$ obeys the conditions of Theorem 3.2 and can be directly estimated with parallel QSP. Because $P_{\geq k}(x)$ is a polynomial of degree $d-k$, this can be achieved at a query depth $2\lceil \frac{d-k}{2k} \rceil$, and a number of measurements $O(\mathcal{K}(P_{\geq k})^4/\epsilon^2)$.

In aggregate, the maximal query depth is $\max\{2(k-1), 2\lceil \frac{d-k}{2k} \rceil\}$, and the total number of measurements is

$$O\left(\frac{\|P_{<k}\|_{[-1,1]}^2}{\epsilon^2}\right) + O\left(\frac{\mathcal{K}(P_{\geq k})^4}{\epsilon^2}\right). \quad (41)$$

□

Evidently, the measurement cost of Theorem 4.1 depends on the quantities $\|P_{<k}\|_{[-1,1]}$ and $\mathcal{K}(P_{\geq k})$, which necessarily depend on the polynomial $P(x)$ under consideration. According to the bounds of Eq. (38), we can guarantee that for any such polynomial, $\|P_{<k}\|_{[-1,1]} \leq \text{poly}(d)$, implying that this term grows polynomially in d , even in the worst case. On the other hand, as we mentioned in Sec. 3.2.1, minimizing $\mathcal{K}(P_{\geq k})$ requires determining a low-norm factorization of $P_{\geq k}(x)$, whose factorization constant is not prohibitively large.

For numerical intuition, we have developed code to implement parallel QSP according to Theorem 4.1. Provided a number of threads k and a polynomial $P(x)$, this code decomposes $P(x)$ into constituent polynomials $P_{<k}(x)$ and $P_{\geq k}(x)$, factors $P_{\geq k}(x)$ across k threads, and then determines the corresponding QSP phases of the factor polynomials. Our code can be found on GitHub at Ref. [54]; see Appendix B for more details.

4.3 Estimation by Parallel QSP and Decomposition into a Well-Behaved Basis

For an arbitrary polynomial $P(x)$, the constituent polynomial $P_{\geq k}(x)$ is not necessarily non-negative, which

renders Theorem 4.1 inapplicable. Likewise, while simply factorizing $P_{\geq k}(x)$ can be done efficiently, guaranteeing that this corresponds to a modest factorization constant is more difficult, and in general requires knowledge about the structure of $P_{\geq k}(x)$. If either of these criteria are unsatisfied, then as we show here, parallel QSP can still be applied by taking special pre-computation steps to reduce the problem back to Theorem 4.1.

We achieve this by decomposing $P_{\geq k}(x)$ into a basis of polynomials that are each amenable to parallel QSP and known to admit a low-norm factorization. We additionally prove that the measurement cost incurred by this factorization scales as $O(d^4 2^{O(k)}/k^2)$, which crucially maintains polynomial scaling in d . This enables parallelization of a large class of property estimation problems, and furnishes the main result of this paper:

Theorem 4.2 (Parallel QSP for Arbitrary Property Estimation: Definite Parity). *Let $P(x)$ be a real polynomial of degree d and definite parity, that is bounded as $\|P\|_{[-1,1]} \leq 1$. By invoking parallel QSP across k threads, where k has the same parity as d , we can estimate*

$$w = \text{tr}(P(\rho)) \quad (42)$$

with a circuit of width $O(k)$ and query depth at most $\lceil \frac{d-k}{2k} \rceil + k - 1 = O(d/k + k)$. The number of measurements required to resolve w to additive error ϵ is

$$O\left(\frac{\|P_{<k}\|_{[-1,1]}^2}{\epsilon^2} + \frac{\|P_{\geq k}\|_{[-1,1]}^2 d^4 (1 + \sqrt{2})^{4k}}{k^2 \epsilon^2}\right) \\ = O\left(\frac{\|P_{<k}\|_{[-1,1]}^2 + \|P_{\geq k}\|_{[-1,1]}^2 d^4 2^{O(k)}}{\epsilon^2}\right). \quad (43)$$

For brevity, we defer the full proof of this theorem to Appendix E. As an overview, the proof works by first decomposing $P_{\geq k}(x)$ into the basis of Chebyshev polynomials. By then using properties of the Chebyshev polynomials (specifically, their composition and product relations), we can re-express $P_{\geq k}(x)$ as a linear combination of products of squared Chebyshev polynomials. These products are each amenable to parallel QSP and clearly exhibit a low-norm factorization, contributing a factor of $O(d^4 2^{O(k)}/k^2)$ to the measurement cost. In addition, as the Chebyshev polynomials are real and of definite parity, they can be implemented directly with QSP and correspond to a query depth $\approx d/2k$, in contrast to the query depth $\approx d/k$ for general polynomials according to Theorem 4.1.

We can also extend Theorem 4.2 to polynomials of indefinite parity, which yields an analogous result:

Theorem 4.3 (Parallel QSP for Arbitrary Property Estimation: Indefinite Parity). *Let $P(x)$ be a real polynomial of degree d , that is bounded as $\|P\|_{[-1,1]} \leq 1$. By invoking parallel QSP across k threads, we can estimate*

$$w = \text{tr}(P(\rho)) \quad (44)$$

with a circuit of width $O(k)$ and query depth at most $\lfloor \frac{d-k}{2(k-1)} \rfloor + k - 2 \approx d/2k + k = O(d/k + k)$. The number of measurements required to resolve w to additive error ϵ is

$$\begin{aligned} & O\left(\frac{\|P_{<k}\|_{[-1,1]}^2}{\epsilon^2} + \frac{\|P_{\geq k}\|_{[-1,1]}^2 d^4 (1 + \sqrt{2})^{4k}}{k^2 \epsilon^2}\right) \\ &= O\left(\frac{\|P_{<k}\|_{[-1,1]}^2 + \|P_{\geq k}\|_{[-1,1]}^2 d^4 2^{O(k)}}{\epsilon^2}\right). \end{aligned} \quad (45)$$

Proof. Decompose $P(x)$ into its even and odd components: $P(x) = P_{\text{even}}(x) + P_{\text{odd}}(x)$. These are both bounded as $\|P_{\text{even}}\|_{[-1,1]}, \|P_{\text{odd}}\|_{[-1,1]} \leq \|P\|_{[-1,1]} \leq 1$ by the triangle inequality. Therefore, we can apply Theorem 4.2 to estimate the traces $\text{tr}(P_{\text{even}}(\rho))$ and $\text{tr}(P_{\text{odd}}(\rho))$ each to additive error $\epsilon/2$, such that their sum approximates $\text{tr}(P(\rho))$ to error ϵ .

However, Theorem 4.2 requires that k have the same parity as the polynomial whose trace is being estimated. This is not possible for both $P_{\text{even}}(x)$ and $P_{\text{odd}}(x)$ given only a single value of k . Thus, if k is even (odd), then estimate $\text{tr}(P_{\text{even}}(\rho))$ over k ($k-1$) threads and $\text{tr}(P_{\text{odd}}(\rho))$ over $k-1$ (k) threads. This amounts to replacing k with $k-1$ in the requisite query depth and number of measurements. As per Theorem 4.2, this corresponds to a query depth at most $\lfloor \frac{d-k+1}{2(k-1)} \rfloor + k - 2 = O(d/k + k)$, and a total number of measurements

$$O\left(\frac{\|P_{<k}\|_{[-1,1]}^2 + \|P_{\geq k}\|_{[-1,1]}^2 d^4 2^{O(k)}}{\epsilon^2}\right). \quad (46)$$

□

Ultimately, Theorem 4.3 is applicable to any real, bounded polynomial $P(x)$, with a dependence only on the norms of the constituent polynomials. As a result, this theorem captures most properties of interest and renders parallel QSP applicable to a broad class of problems. In the following section, we will see that in problems of interest, this norm scales as a polynomial in d or even remains constant. For completeness, we can assess the worst-case overhead (corresponding to the most ill-behaved polynomial) by inserting the general bounds Eq. (38), which equate to a worst-case overhead $\|P_{\geq k}\|_{[-1,1]}^2 = O(\frac{d^{2k+1}}{k \cdot (k!)^2})$. While this expression grows as a degree- $2k+1$ polynomial in d , it also

decays quadratic-factorially in k , ultimately dominating the otherwise $2^{O(k)}$ factor in the measurement overhead. We now turn to applications of parallel QSP in property estimation.

5 Applications

Here we use parallel QSP to develop parallelized algorithms for various problems in property estimation. We highlight the estimation of Rényi entropy, general polynomials, partition functions, and the von Neumann entropy.

5.1 Rényi Entropy: Integer Order

One of the most straightforward demonstrations of parallel QSP is the computation of the Rényi entropy. For a state ρ , the Rényi entropy of order α is defined as $S_\alpha(\rho) = \frac{1}{1-\alpha} \log(\text{tr}(\rho^\alpha))$ for $\alpha > 0$, $\alpha \neq 1$. The Rényi entropy provides a probe of entanglement in both quantum and classical simulation [55, 19, 56, 41, 57], and can be used to approximate the spectrum of ρ through entanglement spectroscopy [18].

Let us first consider estimating $S_\alpha(\rho)$ for integer orders $\alpha \geq 2$, deferring non-integer orders to Sec. 5.4. In this case, prior work has introduced QSP-based algorithms that implement ρ^α as a QSP polynomial, enabling the estimation of $S_\alpha(\rho)$ with query depth α and width $O(1)$ [33, 37, 58]. On the other hand, Refs. [41, 18, 23] invoke the generalized swap test across α systems to evaluate $S_\alpha(\rho)$, corresponding to a query depth $O(1)$ and width $O(\alpha)$. Here, we will illustrate how parallel QSP interpolates between these two regimes, achieving query depth $O(\alpha/k)$ and width $O(k)$.

To demonstrate this, note that the polynomial of interest here is $P(\rho) = \rho^\alpha$. This is a monomial that trivially factorizes into a product of smaller monomials. Following this logic, we arrive at the following theorem:

Theorem 5.1 (Parallel QSP for Estimating Rényi Entropy: Integer Order). *For positive integers $\alpha \geq 2$, one can invoke parallel QSP across k to estimate the Rényi entropy $S_\alpha(\rho) = \frac{1}{1-\alpha} \log(\text{tr}(\rho^\alpha))$ with a circuit of query depth $\lfloor \frac{1}{k} \lfloor \frac{\alpha-k}{2} \rfloor \rfloor + 1 = O(\alpha/k)$ and width $O(k)$. The number of measurements required to achieve additive error ϵ is $O\left(\frac{1}{s_\alpha^2 \alpha^2 \epsilon^2}\right)$ where $s_\alpha = \text{tr}(\rho^\alpha)$.*

Proof. Decompose $P(\rho)$ according to Eq. (35) as

$$P(\rho) = \rho^\alpha = \rho^k \rho^{(\alpha-k) \bmod 2} \left| \rho^{\lfloor \frac{\alpha-k}{2} \rfloor} \right|^2, \quad (47)$$

where we assume $\alpha > k$ (otherwise there is no need to parallelize). This nearly fits into the family of polynomials amenable to parallel QSP as per Theorem 3.1, the

only difference being the additional factor $\rho^{(\alpha-k) \bmod 2}$. This factor is only relevant if $(\alpha - k) \bmod 2 = 1$, in which case it can be easily incorporated by including an additional thread in the initial state ρ into the generalized swap test stage, which increases the threads to $k + 1$ and still equates to a width $O(k)$.

This decomposition corresponds to constituent polynomials $P_{<k}(\rho) = 0$ and $P_{\geq k}(\rho) = \rho^{\lfloor \frac{\alpha-k}{2} \rfloor}$. While $P_{<k}(\rho)$ is non-existent, $P_{\geq k}(\rho)$ is bounded as $\|P_{\geq k}\|_{[-1,1]} \leq 1$ and factorizes into a product of monomials:

$$\rho^{\lfloor \frac{\alpha-k}{2} \rfloor} = \prod_{j=1}^{\lfloor \frac{\alpha-k}{2} \rfloor \bmod k} \rho^{\lfloor \frac{1}{k} \lfloor \frac{\alpha-k}{2} \rfloor \rfloor + 1} \times \prod_{j'=\lfloor \frac{\alpha-k}{2} \rfloor \bmod k+1}^k \rho^{\lfloor \frac{1}{k} \lfloor \frac{\alpha-k}{2} \rfloor \rfloor}. \quad (48)$$

These factor polynomials are all real and of definite parity, and thus can be implemented directly via QSP with a query depth at most $\lfloor \frac{1}{k} \lfloor \frac{\alpha-k}{2} \rfloor \rfloor + 1 = O(\alpha/k)$. Because these monomials are bounded in magnitude by 1, the corresponding factorization constant is simply $\mathcal{K}(P_{\geq k}) = 1$. Plugging these values into Theorem 4.1, the measurement cost of estimating $s_\alpha = \text{tr}(\rho^\alpha)$ to additive error ϵ' is $O(1/\epsilon'^2)$.

However, our quantity of interest is $S_\alpha(\rho) = \frac{1}{1-\alpha} \log(s_\alpha)$. Ref. [33] shows that an estimate $\tilde{s}_\alpha$ of s_α to within multiplicative error ε (i.e., $|\tilde{s}_\alpha/s_\alpha - 1| \leq \varepsilon$) provides an approximation to $S_\alpha(\rho)$ with additive error $\varepsilon/(\alpha - 1)$:

$$\left| \frac{1}{1-\alpha} \log(\tilde{s}_\alpha) - S_\alpha(\rho) \right| \leq \frac{\varepsilon}{\alpha - 1}. \quad (49)$$

Therefore, to achieve additive error ϵ in the estimate of the Rényi entropy, select $\epsilon' = s_\alpha \varepsilon = s_\alpha \epsilon (\alpha - 1)$, which translates to a total number of measurements

$$O\left(\frac{1}{s_\alpha^2 \alpha^2 \epsilon^2}\right). \quad (50)$$

□

As a hybrid of QSP and the generalized swap test, parallel QSP combines the strengths of both algorithms in estimating the Rényi entropy. Its query depth $O(\alpha/k)$ and width $O(k)$ provide a smooth interpolation between the circuit requirements of these approaches, allowing for full use of one's available quantum resources.

5.2 General Polynomials in the Monomial Basis

Above, we used parallel QSP to parallelize the computation of the trace of a monomial $\text{tr}(\rho^\alpha)$. This naturally extends to more general polynomials $P(\rho) = \sum_{n=0}^d c_n \rho^n$

by parallelizing each monomial. Following this line of thought, we can prove the following:

Theorem 5.2 (Parallel QSP for Estimation of General Polynomial Traces). *For a degree- d polynomial $P(x) = \sum_{n=0}^d c_n x^n$, one can invoke parallel QSP across k threads to estimate $\text{tr}(P(\rho))$ with a circuit of query depth $\lfloor \frac{1}{k} \lfloor \frac{d-k}{2} \rfloor \rfloor + 1 = O(d/k)$ and width $O(k)$. The number of measurements required to attain additive error ϵ is*

$$O\left(\frac{\|c\|_1^2}{\epsilon^2}\right), \quad (51)$$

where $\|c\|_1 = \sum_{n=0}^d |c_n|$ is the 1-norm of the polynomial coefficients.

Proof. Our aim is to parallelize the computation of $\text{tr}(P(\rho)) = \sum_{n=0}^d c_n \text{tr}(\rho^n)$ by using the method of Theorem 5.1 for parallelizing the monomial trace $\text{tr}(\rho^n)$. While one could achieve this by estimating each trace $\text{tr}(\rho^n)$ in sequence (from $n = 0$ up to $n = d$), a more efficient approach is furnished by importance sampling. That is, sample an integer n from the distribution $p(n) = |c_n|/\|c\|_1$ where $\|c\|_1 = \sum_{n=0}^d |c_n|$, and evaluate the estimator of $\text{tr}(\rho^n)$ (i.e. the measurement result of the parallel QSP circuit). As we show in Appendix D, this procedure provides an estimator for $\text{tr}(P(\rho))/\|c\|_1$. Then, estimating $\text{tr}(P(\rho))$ to additive error ϵ is equivalent to estimating $\text{tr}(P(\rho))/\|c\|_1$ to error $\epsilon/\|c\|_1$, which requires a measurement cost $O(\|c\|_1^2/\epsilon^2)$.

In measuring the estimator of each monomial trace $\text{tr}(\rho^n)$, employ parallel QSP according to Theorem 5.1. The corresponding query depth is at most $\lfloor \frac{1}{k} \lfloor \frac{d-k}{2} \rfloor \rfloor + 1 = O(d/k)$, and the circuit width is $O(k)$. □

As the measurement cost grows with $\|c\|_1$, this approach is best suited for polynomials where this 1-norm is not prohibitively large. For many polynomials of interest, $\|c\|_1$ is poly(d) or even just $O(1)$. For instance, a truncation of $e^{-\beta x}$ achieves $\|c\|_1 = O(e^\beta)$, independent of d . However, $\|c\|_1$ can grow exponentially with d for certain polynomials, in which case this approach is prohibitively costly. For example, the order d Chebyshev polynomial $T_d(x)$ has 1-norm $\|c\|_1 = 2^{O(d)}$. In this case, it is better to invoke Theorem 4.3 to estimate $\text{tr}(P(\rho))$, which applies to all polynomials agnostic to their 1-norm, and achieves a cost that necessarily scales polynomially in d .

While estimation of functions in the monomial basis serves as a pedagogical introduction to the mechanics of parallel QSP (after all, factoring monomials is trivial!), the true strength of our approach lies in handling polynomials with large 1-norm $\|c\|_1$ in the monomial basis, such as the Chebyshev polynomials $T_d(x)$. In such cases, prior techniques [23, 45], including Theorem 5.2, incur a cost that scales with $\|c\|_1$ and can be

exponentially large in d . In contrast, parallel QSP, as explained in Theorem 4.3, applies a polynomial transformation coherently, with a complexity determined by a function norm that is often significantly smaller than $\|c\|_1$. This makes parallel QSP especially powerful for property estimation tasks where the target polynomial is not sparse or has large coefficients, such as those used in estimating the non-integer Rényi entropy or the von Neumann entropy. We explore these polynomials later in Secs. 5.4 and 5.5.

5.3 Partition Function

A problem to which Theorem 5.2 applies nicely is the estimation of $\text{tr}(e^{-\beta\rho})$. This can be used to evaluate a partition function $Z = \text{tr}(e^{-\beta H})$ when the Hamiltonian H is encoded in a density matrix as $\rho \propto H + cI$, such as in sample-based Hamiltonian simulation [53] or density matrix exponentiation [59]. Here we can prove the following:

Theorem 5.3 (Estimation of $\text{tr}(e^{-\beta\rho})$ with Parallel QSP). *One can invoke parallel QSP across k threads to estimate $\text{tr}(e^{-\beta\rho})$ with a circuit of query depth $O\left(\frac{\beta}{k} \log(\beta D/\epsilon)\right)$ and width $O(k)$, where $D = \dim(\rho)$. To achieve additive error ϵ , the requisite number of measurements is $O\left(\frac{e^{2\beta}}{\epsilon^2}\right)$.*

Proof. Consider the polynomial $P(x) = \sum_{n=0}^d \frac{(-\beta)^n}{n!} x^n$. This approximates the exponential $e^{-\beta x}$ with an additive error at most ϵ' over $x \in [0, 1]$ by choosing a degree $d = O(\beta \log(\beta/\epsilon'))$ [60]. As our goal is to evaluate $\text{tr}(P(\rho))$ to error ϵ , select a polynomial error $\epsilon' = \epsilon/2D$, where $D = \dim(\rho)$. This guarantees that $|\text{tr}(P(\rho)) - \text{tr}(e^{-\beta\rho})| \leq |\text{tr}(\epsilon')| = \epsilon/2$. Therefore, if we can evaluate $\text{tr}(P(\rho))$ to error $\epsilon/2$, we will approximate $\text{tr}(e^{-\beta\rho})$ to error ϵ .

To perform this evaluation, we first note the 1-norm bound:

$$\|c\|_1 = \sum_{n=0}^d \frac{\beta^n}{n!} < e^\beta. \quad (52)$$

According to Theorem 5.2, we can estimate $\text{tr}(P(\rho))$ with a circuit of query depth $O(d/k) = O\left(\frac{\beta}{k} \log(\beta D/\epsilon)\right)$ and width $O(k)$, and the requisite number of measurements is

$$O\left(\frac{\|c\|_1^2}{(\epsilon/2)^2}\right) = O\left(\frac{e^{2\beta}}{\epsilon^2}\right). \quad (53)$$

□

Relative to standard QSP, parallel QSP reduces the query depth by a factor $O(k)$, without worsening the scaling of the measurement cost. This is because $e^{-\beta x}$ admits a Taylor series whose coefficients have a 1-norm bounded by $O(1)$, independent of the truncation degree.

5.4 Rényi Entropy: Non-Integer Order

It is also of interest to compute the Rényi entropy for non-integer α . Rather straightforwardly, this may be achieved by implementing ρ^α as a QSP polynomial and estimating its trace [33, 37]. This however is more costly to compute than the case of integer order because the function x^α is non-analytic at $x = 0$, and hence can only be well approximated by a polynomial away from this singular point. Due to this singularity, polynomial approximations to x^α have coefficients whose 1-norm scales exponentially as $\|c\|_1 = 2^{O(d)}$ [23], thus rendering Theorem 5.2 impractical. In this case, it is advantageous to instead use Theorem 4.3, whose measurement cost is guaranteed to scale polynomially in d .

We can use this reasoning to prove the following theorem for estimating the Rényi entropy. For generality, we state this theorem for a degree- d polynomial approximation of x^α . As we will discuss after the proof, the necessary degree depends on properties of ρ (condition number, rank, etc.).

Theorem 5.4 (Parallel QSP for Estimating Rényi Entropy: Non-Integer Order). *Suppose $\alpha > 0$ is a non-integer, and let $P(x)$ be a degree- d polynomial approximation to x^α . Then, one can invoke parallel QSP across k threads to estimate the Rényi entropy $S_\alpha(\rho) = \frac{1}{1-\alpha} \log(\text{tr}(\rho^\alpha))$ as $\tilde{S}_\alpha(\rho) = \frac{1}{1-\alpha} \log(\text{tr}(P(\rho)))$, with a circuit of query depth at most $\lfloor \frac{d-k}{2(k-1)} \rfloor + k - 2 = O(d/k + k)$ and width $O(k)$. The number of measurements necessary to estimate $\tilde{S}_\alpha(\rho)$ to additive error ϵ is*

$$O\left(\left(\frac{d^{k+2}}{(k+1)!}\right)^2 \frac{d}{k} \frac{1}{\tilde{s}_\alpha^2 \epsilon^2 (\alpha - 1)^2}\right), \quad (54)$$

where $\tilde{s}_\alpha = \text{tr}(P(\rho)) \approx \text{tr}(\rho^\alpha)$.

Proof. Without a specific polynomial approximation to x^α (which we will address after this proof) nor an analytic factorization thereof, we can invoke the generic construction of Theorem 4.3, which is applicable to all polynomials. This reduces the query depth to $\lfloor \frac{d-k}{2(k-1)} \rfloor + k - 2 = O(k + d/k)$. Inserting the generic bounds of Eq. (38), we find that the measurement cost to estimate $\tilde{s}_\alpha = \text{tr}(P(\rho))$ to additive error ϵ' is

$$\begin{aligned} O\left(\left(\frac{d^k}{k!} \sqrt{\frac{d}{k}}\right)^2 \frac{d^4 (1+\sqrt{2})^{4k}}{k^2 \epsilon'^2}\right) \\ = O\left(\left(\frac{d^{k+2}}{(k+1)!}\right)^2 \frac{d}{k \epsilon'^2}\right). \end{aligned} \quad (55)$$

where the second line follows from the factorial $k!$ dominating the exponential $2^{O(k)}$. Then, mirroring the proof of Theorem 5.1, in order to estimate $\tilde{S}_\alpha(\rho)$ to additive

error ϵ , it suffices to select $\epsilon' = s_\alpha \epsilon = s_\alpha \epsilon (\alpha - 1)$, which equates to a total number of measurements

$$O\left(\left(\frac{d^{k+2}}{(k+1)!}\right)^2 \frac{d}{k} \frac{1}{\tilde{s}_\alpha^2 \epsilon^2 (\alpha - 1)^2}\right). \quad (56)$$

Lastly, note that we have used the generic bounds of Eq. (38). Although these are worst-case, and could possibly be tightened for a specific polynomial approximation, we expect them to be relatively tight for polynomial approximations to functions with singularities like x^α . $\square$

As anticipated, parallel QSP reduces the depth by a factor of $O(k)$, and increases the number of measurements by a polynomial in d and a factor that decays quadratic-factorially in k . To precisely approximate the Rényi entropy using this result, we need to determine a sufficient polynomial approximation to x^α . Refs. [58, 5] provide an odd polynomial approximation to x^α that suffers additive error at most ϵ over $x \in [\delta, 1]$ for some $\delta > 0$, and show that the degree of this polynomial is $O(\alpha + \frac{1}{\delta} \log(1/\epsilon))$. Using this polynomial, there are two predominant approaches for Rényi entropy estimation.

First, as in Ref. [33], one can choose δ to be the smallest non-zero eigenvalue of ρ (or a lower bound thereof), such that $P(\rho) \approx \rho^\alpha$ over the support of ρ . This choice is equivalent to setting $\delta = 1/\kappa$ where κ is the condition number of ρ . Then, to estimate $S_\alpha(\rho)$ to error ϵ , we can select $\epsilon' = s_\alpha \epsilon |\alpha - 1|/2D$ where $D = \dim(\rho)$, corresponding to a degree $d = O(\alpha + \kappa \log(D/|\alpha - 1|s_\alpha \epsilon))$. This choice guarantees that $|\tilde{s}_\alpha - s_\alpha| = |\text{tr}(P(\rho)) - \text{tr}(\rho^\alpha)| \leq \text{tr}(\epsilon') = s_\alpha \epsilon |\alpha - 1|/2$, such that $|\tilde{S}_\alpha(\rho) - S_\alpha(\rho)| \leq \epsilon/2$ by Eq. (49). Therefore, to estimate $S_\alpha(\rho)$ to error ϵ , it suffices to estimate $\tilde{S}_\alpha(\rho)$ to error $\epsilon/2$. According to Theorem 5.4, the corresponding parallel QSP algorithm requires a number of measurements

$$\begin{aligned} O\left(\left(\frac{d^{k+2}}{(k+1)!}\right)^2 \frac{d}{k} \frac{1}{\tilde{s}_\alpha^2 \epsilon^2 (\alpha - 1)^2}\right) \\ = O\left(\frac{\text{poly}(\alpha, \kappa, \log(D))}{\tilde{s}_\alpha^2 \epsilon^2 (\alpha - 1)^2}\right). \end{aligned} \quad (57)$$

On the other hand, if the rank $r = \text{rank}(\rho)$ is known, then as Ref. [37] shows, one can alternatively select a value $\delta = O(1/r)$ while maintaining an accurate approximation of the Rényi entropy. Here, one can again select $\epsilon' = s_\alpha \epsilon |\alpha - 1|/2r$, such that $|\tilde{s}_\alpha - s_\alpha| = |\text{tr}(P(\rho)) - \text{tr}(\rho^\alpha)| \leq \text{tr}(\epsilon') = s_\alpha \epsilon |\alpha - 1|/2$, implying $|\tilde{S}_\alpha(\rho) - S_\alpha(\rho)| \leq \epsilon/2$ by Eq. (49). These choices correspond to a polynomial degree $d = O(\alpha + r \log(r/s_\alpha \epsilon |\alpha - 1|))$. As above, it suffices to estimate $\tilde{S}_\alpha(\rho)$ to error $\epsilon/2$, in

which case Theorem 5.4 indicates that parallel QSP necessitates a number of measurements

$$O\left(\frac{\text{poly}(\alpha, r)}{\tilde{s}_\alpha^2 \epsilon^2 (\alpha - 1)^2}\right). \quad (58)$$

This scales more favorably than the above approach leading to Eq. (57), because $r \leq \kappa$ for any density matrix.

In both cases, the query depth is reduced relative to the associated sequential algorithms [33, 37, 61], at the expense of increasing the measurement cost by a factor $O(\text{poly}(\alpha, \kappa, \log(D)))$ or $O(\text{poly}(\alpha, r))$. As such, these methods are best deployed on states with small condition number or low-rank, e.g. $\kappa, r = O(\text{polylog}(D))$, which arise in a variety of experimental [62, 63, 64], computational [65, 66, 67], and theoretical contexts [59, 68]. In these cases, quantum algorithms for Rényi entropy estimation achieve an exponential advantage over classical algorithms, which scale as $O(\text{poly}(D))$ [37, 61]. Fortunately, because the measurement cost of parallel QSP scales polynomially in κ and r , parallel QSP crucially retains this exponential speedup.

5.5 Von Neumann Entropy

Another ubiquitous quantity in quantum physics is the von Neumann entropy $S(\rho) = -\text{tr}(\rho \ln \rho)$, which characterizes entanglement [38], defines thermal states [69], describes phase transitions [70], and dictates the rate at which information can be transmitted across a quantum channel [71]. Given such broad interest, QSP-based algorithms have been put forth to estimate the von Neumann entropy [37, 58], which work by approximating $-\rho \ln \rho$ as a QSP polynomial and taking its trace. Similar to the situation of the non-integer α -Rényi entropy, the underlying function $-x \ln x$ is singular at $x = 0$, and can only be approximated by a polynomial away from this point. As such, polynomial approximations to $-x \ln x$ also exhibit coefficients whose 1-norm scales exponentially as $\|c\|_1 = 2^{O(d)}$ [23], thus rendering Theorem 5.2 impractical and suggests the use of Theorem 4.3 instead.

We can therefore invoke parallel QSP to prove the following theorem, analogous to Theorem 5.4. As above, we state this for a degree- d polynomial approximation of $-x \ln x$, where d will depend on the properties ρ , which we address after the proof.

Theorem 5.5 (Parallel QSP for Estimating von Neumann Entropy). *Let $P(x)$ be a degree- d polynomial approximation to $-x \ln x$. Then, parallel QSP enables the estimation of the von Neumann entropy $S(\rho) = -\text{tr}(\rho \ln \rho)$ as $\tilde{S}(\rho) = \text{tr}(P(\rho))$, with a circuit of query depth at most $\lfloor \frac{d-k}{2(k-1)} \rfloor + k - 2 = O(d/k + k)$ and width*

$O(k)$. The number of measurements necessary to estimate $\tilde{S}(\rho)$ to additive error ϵ is

$$O\left(\left(\frac{d^{k+2}}{(k+1)!}\right)^2 \frac{d}{k} \frac{1}{\epsilon^2}\right). \quad (59)$$

Proof. This proof is nearly identical to Theorem 5.4. In the absence of a specific polynomial approximation to $-x \ln(x)$ nor a factorization thereof, we again apply Theorem 4.3 to estimate $\text{tr}(P(\rho))$ to additive error ϵ , and insert the generic bounds of Eq. (38). This reduces the query depth to $\lfloor \frac{d-k}{2(k-1)} \rfloor + k - 2 = O(d/k + k)$ and necessitates a measurement cost

$$\begin{aligned} O\left(\left(\frac{d^k}{k!} \sqrt{\frac{d}{k}}\right)^2 \frac{d^4}{k^2} \frac{(1+\sqrt{2})^{4k}}{\epsilon^2}\right) = \\ O\left(\left(\frac{d^{k+2}}{(k+1)!}\right)^2 \frac{d}{k\epsilon^2}\right). \end{aligned} \quad (60)$$

Again, we expect the generic bounds to be nearly saturated for polynomial approximations to a singular function like $-x \ln x$. $\square$

As above, parallel QSP reduces the query depth by a factor $O(k)$ and increases the number of measurements by a factor $\text{poly}(d)$. What remains is to provide a sufficient polynomial approximation to $-x \ln x$. Similar to the function x^α , there exists an odd polynomial approximation to $-x \ln x$ that suffers additive error at most ϵ' over $x \in [\delta, 1]$ for some $\delta > 0$, and the requisite degree of this polynomial is $O(\frac{1}{\delta} \log(1/\epsilon'))$ [37, 58, 5].

Analogous to our treatment of the Rényi entropy, one can use this polynomial and select $\delta = 1/\kappa$ and $\epsilon' = \epsilon/2D$, where $D = \dim(\rho)$. This guarantees that $|\tilde{S}(\rho) - S(\rho)| \leq \epsilon/2$ and corresponds to a degree $d = O(\kappa \log(D/\epsilon))$. It then suffices to estimate $\tilde{S}(\rho)$ to error $\epsilon/2$, in which case Theorem 5.5 indicates that parallel QSP requires query depth $O(d/k + k)$ and a number of measurements $O(\frac{\text{poly}(\kappa, \log D)}{\epsilon^2})$.

Alternatively, if the rank $r = \text{rank}(\rho)$ is known, one can select $\delta = O(1/r)$ and $\epsilon' = O(\epsilon/r)$, while maintaining an accurate approximation to the von Neumann entropy [37]. This equates to a degree $d = O(r \log(r/\epsilon))$. Then, Theorem 5.5 readily implies that parallel QSP estimates $S(\rho)$ with query depth $O(d/k + k)$ and a number of measurements $O(\frac{\text{poly}(r)}{\epsilon^2})$.

Again, these methods are best deployed on states with $\kappa, r = O(\text{polylog}(D))$, in which case quantum algorithms provide an exponential improvement over classical algorithms [37, 61]. Because the measurement cost of parallel QSP scales as $\text{poly}(\kappa, \log(D))$ or $\text{poly}(r)$, parallel QSP retains this exponential speedup.

6 Discussion and Conclusion

In this work, we have presented a parallelized version of quantum signal processing, tailored to property estimation. Our algorithm parallelizes the computation of a property $\text{tr}(P(\rho))$ over k threads, and reduces query depth by a factor $O(k)$ while increasing circuit width by $O(k)$, characterizing a tradeoff between temporal and spatial resources in QSP. The core of our construction rests on the ability to factorize a polynomial of degree d into a product of k degree- $O(d/k)$ polynomials. Within a quantum circuit, these polynomials are prepared in parallel and then multiplied together using a generalized swap test. This methodology is applicable to general polynomials of a density matrix, with a measurement cost that depends on the chosen factorization. In justification of its utility, we apply parallel QSP to a variety of problems in property estimation, ranging from the estimation of entropies to partition functions.

Our work has important implications for intermediate-scale quantum computation. First, because parallel QSP reduces circuit depth, it is well suited for platforms limited to shallow circuits by short coherence times [72]. Secondly, parallel QSP opens up new opportunities for co-designing parallel quantum architectures, with applications in distributed quantum computing [26] and resource-intensive tasks on near-term devices [73]. In this context, separate devices perform QSP in parallel and then send their state to a central device that performs a swap test, with the key benefit that errors remain isolated across devices until the swap test. Lastly, parallel QSP can help to reduce the overhead incurred by error mitigation or error correction. For example, in the structured, algorithmic-level correction of coherent errors in QSP [24], parallel QSP reduces the overhead in circuit depth from $O(d^n)$ to $O((\frac{d}{k})^n)$ when errors are corrected to order n .

There exist numerous possible improvements to our parallel QSP algorithm. One limitation is the measurement overhead accrued when parallelizing over a generic polynomial, which does not always permit a factorization into polynomials of bounded magnitude on an interval. While here we circumvented this issue by decomposing into a well-behaved basis, it remains an open problem to bound the factorization constant for a generic polynomial and provide tighter bounds on the resulting measurement overhead. In addition, a theory of QSP on $\text{SU}(d)$ may prove useful in combining multiple monomials together [74, 75] with correspondingly better overhead. With an eye toward improving property estimation algorithms, it may also prove fruitful to integrate parallel QSP with randomized measurements techniques [19, 76].

Parallel QSP suggests multiple future directions for the design of further parallel quantum algorithms. First, while we discussed parallel QSP in the context of univariate polynomials, one could extend this algorithm to multivariate polynomials [23] to encompass relevant metrics like the fidelity $\text{tr}(\sqrt{\sqrt{\rho}\sigma\sqrt{\rho}})$ and trace distances $\frac{1}{2} \text{tr}(|\rho - \sigma|^n)$. Although parallel QSP can straightforwardly accommodate products of univariate polynomials by direct substitution into the parallel QSP circuit (Fig. 3), designing a generic multivariate polynomial is a more pressing challenge as a general theory of multivariate QSP has yet to be established [77, 78, 79]. Secondly, beyond polynomial factorization, other forms of parallelization are also possible. For example, by exploring spatial locality, one could use spline interpolations to approximate a well-behaved polynomial by lower-degree polynomials over restricted intervals; implementation of this idea may require access to projectors that divide the input space into restricted intervals. In addition, we note that there exists concurrent work considering depth reduction of QSP sequences, albeit in the setting of parameter estimation for calibration [73]. Understanding the relation of our methods to such ideas remains an interesting open question for future work.

Together, these prospective generalizations of parallel QSP, combined with parallel classical computing [80], offer utility in other areas, such as the parallel quantum simulation of quantum chemical systems and materials [81, 82]. Fundamentally, as properties of polynomial factorizations underlies parallel QSP, this suggests that other functional analytic properties could be leveraged to develop new and improved quantum algorithms through QSP. Given that polynomials have been extensively studied for centuries, boasting a rich array of properties from complex analysis to algebraic geometry, further exploration of the connections between polynomials and QSP may point to novel quantum algorithms.

Acknowledgements: The authors thank Patrick Rall, Danial Motlagh, and Alexander Zlokapa for useful discussion. JMM acknowledges support from the National Science Foundation Graduate Research Fellowship under Grant No. 2141064. This project was 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 number DE-SC0012704. YL also acknowledges support from the Department of Energy under contract number DE-SC0025384. This research was also supported in part by NTT Research.

References

- [1] 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).
- [2] Guang Hao Low, Theodore J. Yoder, and Isaac L. Chuang. “Methodology of resonant equiangular composite quantum gates”. *Physical Review X* **6** (2016).
- [3] Guang Hao Low and Isaac L. Chuang. “Optimal hamiltonian simulation by quantum signal processing”. *Physical Review Letters* **118** (2017).
- [4] Guang Hao Low and Isaac L. Chuang. “Hamiltonian simulation by qubitization”. *Quantum* **3**, 163 (2019).
- [5] András Gilyén, Yuan Su, Guang Hao Low, and Nathan 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).
- [6] John M Martyn, Zane M Rossi, Andrew K Tan, and Isaac L Chuang. “Grand unification of quantum algorithms”. *PRX Quantum* **2**, 040203 (2021).
- [7] Jeongwan Haah. “Product decomposition of periodic functions in quantum signal processing”. *Quantum* **3**, 190 (2019).
- [8] Rui Chao, Dawei Ding, Andras Gilyén, Cupjin Huang, and Mario Szegedy. “Finding angles for quantum signal processing with machine precision” (2020). [arXiv:2003.02831](https://arxiv.org/abs/2003.02831).
- [9] Yulong Dong, Xiang Meng, K. Birgitta Whaley, and Lin Lin. “Efficient phase-factor evaluation in quantum signal processing”. *Physical Review A* **103** (2021).
- [10] Yulong Dong, Lin Lin, Hongkang Ni, and Jiasu Wang. “Infinite quantum signal processing”. *Quantum* **8**, 1558 (2024).
- [11] Ashley Montanaro and Changpeng Shao. “Quantum and classical query complexities of functions of matrices”. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*. [Page 573–584](#). STOC 2024 New York, NY, USA (2024). Association for Computing Machinery.
- [12] Ewin Tang and Kevin Tian. “A CS guide to the quantum singular value transformation”. *Chapter : 2024 Symposium on Simplicity in Algorithms (SOSA)*, pages 121–143. SIAM. (2024).

- [13] Michel Alexis, Lin Lin, Gevorg Mnatsakanyan, Christoph Thiele, and Jiasu Wang. “Infinite quantum signal processing for arbitrary Szegő functions” (2024). [arXiv:2407.05634](#).
- [14] Bjorn K. Berntson and Christoph Sünderhauf. “Complementary polynomials in quantum signal processing” (2024). [arXiv:2406.04246](#).
- [15] Kyle DeBry, Jasmine Sinanan-Singh, Colin D. Bruzewicz, David Reens, May E. Kim, Matthew P. Roychowdhury, Robert McConnell, Isaac L. Chuang, and John Chiaverini. “Experimental quantum channel discrimination using metastable states of a trapped ion”. *Phys. Rev. Lett.* **131**, 170602 (2023).
- [16] Yuta Kikuchi, Conor Mc Keever, Luuk Coopmans, Michael Lubasch, and Marcello Benedetti. “Realization of quantum signal processing on a noisy quantum computer”. *npj Quantum Information* **9**, 93 (2023).
- [17] David A Patterson, Frederick P Brooks Jr, Ivan E Sutherland, and Charles P Thacker. “Computer architecture”. Elsevier Science. (2011).
- [18] Sonika Johri, Damian S. Steiger, and Matthias Troyer. “Entanglement spectroscopy on a quantum computer”. *Physical Review B* **96** (2017).
- [19] A. Elben, B. Vermersch, M. Dalmonte, J.I. Cirac, and P. Zoller. “Rényi entropies from random quenches in atomic hubbard and spin models”. *Physical Review Letters* **120** (2018).
- [20] Harry Buhrman, Richard Cleve, John Watrous, and Ronald de Wolf. “Quantum fingerprinting”. *Physical Review Letters* **87** (2001).
- [21] Michał Oszmaniec, Daniel J Brod, and Ernesto F Galvão. “Measuring relational information between quantum states, and applications”. *New Journal of Physics* **26**, 013053 (2024).
- [22] Guang Hao Low. “Quantum signal processing by single-qubit dynamics”. PhD thesis. Massachusetts Institute of Technology. (2017). url: <http://hdl.handle.net/1721.1/115025>.
- [23] Yihui Quek, Eneet Kaur, and Mark M. Wilde. “Multivariate trace estimation in constant quantum depth”. *Quantum* **8**, 1220 (2024).
- [24] Andrew K. Tan, Yuan Liu, Minh C. Tran, and Isaac L. Chuang. “Perturbative model of noisy quantum signal processing”. *Phys. Rev. A* **107**, 042429 (2023).
- [25] Zane M Rossi, Jeffery Yu, Isaac L Chuang, and Sho Sugiura. “Quantum advantage for noisy channel discrimination”. *Physical Review A* **105**, 032401 (2022).
- [26] Marcello Caleffi, Michele Amoretti, Davide Ferrari, Jessica Illiano, Antonio Manzalini, and Angela Sara Cacciapuoti. “Distributed quantum computing: A survey”. *Computer Networks* **254**, 110672 (2024).
- [27] Yihui Quek, Daniel Stilck França, Sumeet Khatri, Johannes Jakob Meyer, and Jens Eisert. “Exponentially tighter bounds on limitations of quantum error mitigation”. *Nature Physics* **20**, 1648–1658 (2024).
- [28] Tianyi Peng, Aram W. Harrow, Maris Ozols, and Xiaodi Wu. “Simulating large quantum circuits on a small quantum computer”. *Physical Review Letters* **125** (2020).
- [29] Danial Motlagh and Nathan Wiebe. “Generalized quantum signal processing”. *PRX Quantum* **5**, 020368 (2024).
- [30] Yulong Dong, Xiang Meng, K Birgitta Whaley, and Lin Lin. “Efficient phase-factor evaluation in quantum signal processing”. *Physical Review A* **103**, 042419 (2021).
- [31] Lexing Ying. “Stable factorization for phase factors of quantum signal processing”. *Quantum* **6**, 842 (2022).
- [32] Shuntaro Yamamoto and Nobuyuki Yoshioka. “Robust angle finding for generalized quantum signal processing” (2024). [arXiv:2402.03016](#).
- [33] Sathyawageeswar Subramanian and Min-Hsiu Hsieh. “Quantum algorithm for estimating α -renyi entropies of quantum states”. *Physical Review A* **104** (2021).
- [34] András Gilyén and Tongyang Li. “Distributional property testing in a quantum world” (2019). [arXiv:1902.00814](#).
- [35] Dorit Aharonov, Vaughan Jones, and Zeph Landau. “A polynomial quantum algorithm for approximating the jones polynomial”. In *Proceedings of the Thirty-Eighth Annual ACM Symposium on Theory of Computing*. Page 427–436. STOC ’06 New York, NY, USA (2006). Association for Computing Machinery.
- [36] András Gilyén and Alexander Poremba. “Improved quantum algorithms for fidelity estimation” (2022). [arXiv:2203.15993](#).
- [37] Qisheng Wang, Ji Guan, Junyi Liu, Zhicheng Zhang, and Mingsheng Ying. “New quantum algorithms for computing quantum entropies and distances”. *IEEE Transactions on Information Theory* **70**, 5653–5680 (2024).

- [38] Michael A Nielsen and Isaac L Chuang. “Quantum computation and quantum information”. Cambridge University Press. (2010).
- [39] Gilles Brassard, Peter Høyer, Michele Mosca, and Alain Tapp. “Quantum amplitude amplification and estimation”. *Quantum Computation and Information* Page 53–74 (2002).
- [40] Artur K. Ekert, Carolina Moura Alves, Daniel K. L. Oi, Michał Horodecki, Paweł Horodecki, and L. C. Kwek. “Direct estimations of linear and non-linear functionals of a quantum state”. *Physical Review Letters* **88** (2002).
- [41] Matthew B. Hastings, Iván González, Ann B. Kallin, and Roger G. Melko. “Measuring renyi entanglement entropy in quantum monte carlo simulations”. *Physical Review Letters* **104** (2010).
- [42] Todd A. Brun. “Measuring polynomial functions of states” (2004). [arXiv:quant-ph/0401067](https://arxiv.org/abs/quant-ph/0401067).
- [43] Paweł Horodecki and Artur Ekert. “Method for direct detection of quantum entanglement”. *Physical Review Letters* **89** (2002).
- [44] Yiğit Subaşı, Lukasz Cincio, and Patrick J Coles. “Entanglement spectroscopy with a depth-two quantum circuit”. *Journal of Physics A: Mathematical and Theoretical* **52**, 044001 (2019).
- [45] Justin Yirka and Yiğit Subaşı. “Qubit-efficient entanglement spectroscopy using qubit resets”. *Quantum* **5**, 535 (2021).
- [46] Alan Edelman and Hiroshi Murakami. “Polynomial roots from companion matrix eigenvalues”. *Mathematics of Computation* **64**, 763–776 (1995).
- [47] Angus Lowe, Matija Medvidović, Anthony Hayes, Lee J. O’Riordan, Thomas R. Bromley, Juan Miguel Arrazola, and Nathan Killoran. “Fast quantum circuit cutting with randomized measurements”. *Quantum* **7**, 934 (2023).
- [48] Allan Tosta, Thais de Lima Silva, Giancarlo Camilo, and Leandro Aolita. “Randomized semi-quantum matrix processing”. *npj Quantum Information* **10** (2024).
- [49] Yue Wang and Qi Zhao. “Faster quantum algorithms with "fractional"-truncated series” (2024). [arXiv:2402.05595](https://arxiv.org/abs/2402.05595).
- [50] John M. Martyn and Patrick Rall. “Halving the cost of quantum algorithms with randomization”. *npj Quantum Information* **11** (2025).
- [51] Dominic W. Berry, Graeme Ahokas, Richard Cleve, and Barry C. Sanders. “Efficient quantum algorithms for simulating sparse hamiltonians”. *Communications in Mathematical Physics* **270**, 359–371 (2006).
- [52] Nai-Hui Chia, Kai-Min Chung, Yao-Ching Hsieh, Han-Hsuan Lin, Yao-Ting Lin, and Yu-Ching Shen. “On the impossibility of general parallel fast-forwarding of hamiltonian simulation”. In Proceedings of the Conference on Proceedings of the 38th Computational Complexity Conference. CCC ’23 Dagstuhl, DEU (2023). Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.
- [53] Shelby Kimmel, Cedric Yen-Yu Lin, Guang Hao Low, Maris Ozols, and Theodore J. Yoder. “Hamiltonian simulation with optimal sample complexity”. *npj Quantum Information* **3** (2017).
- [54] Kevin Cheng. “Parallel QSP”. <https://github.com/kevinchengg/parallelQSP> (2024).
- [55] Tiff Brydges, Andreas Elben, Petar Jurcevic, Benoît Vermersch, Christine Maier, Ben P. Lanyon, Peter Zoller, Rainer Blatt, and Christian F. Roos. “Probing rényi entanglement entropy via randomized measurements”. *Science* **364**, 260–263 (2019).
- [56] N. M. Linke, S. Johri, C. Figgatt, K. A. Landsman, A. Y. Matsuura, and C. Monroe. “Measuring the rényi entropy of a two-site fermi-hubbard model on a trapped ion quantum computer”. *Physical Review A* **98** (2018).
- [57] Mohamed Hibat-Allah, Martin Ganahl, Lauren E. Hayward, Roger G. Melko, and Juan Carrasquilla. “Recurrent neural network wave functions”. *Physical Review Research* **2** (2020).
- [58] Youle Wang, Lei Zhang, Zhan Yu, and Xin Wang. “Quantum phase processing and its applications in estimating phase and entropies”. *Physical Review A* **108** (2023).
- [59] Seth Lloyd, Masoud Mohseni, and Patrick Rebentrost. “Quantum principal component analysis”. *Nature Physics* **10**, 631–633 (2014).
- [60] Dominic W. Berry, Andrew M. Childs, Richard Cleve, Robin Kothari, and Rolando D. Somma. “Simulating hamiltonian dynamics with a truncated Taylor series”. *Physical Review Letters* **114** (2015).
- [61] Youle Wang, Benchu Zhao, and Xin Wang. “Quantum algorithms for estimating quantum entropies”. *Phys. Rev. Appl.* **19**, 044041 (2023).
- [62] David Gross, Yi-Kai Liu, Steven T. Flammia, Stephen Becker, and Jens Eisert. “Quantum state tomography via compressed sensing”. *Physical Review Letters* **105** (2010).
- [63] Cristina Butucea, Mădălin Guță, and Theodore Kypraios. “Spectral thresholding quantum tomography for low rank states”. *New Journal of Physics* **17**, 113050 (2015).

- [64] Israel F. Araujo, Carsten Blank, Ismael C. S. Araujo, and Adenilton J. da Silva. “Low-rank quantum state preparation”. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* **43**, 161–170 (2024).
- [65] Jacob C Bridgeman and Christopher T Chubb. “Hand-waving and interpretive dance: an introductory course on tensor networks”. *Journal of Physics A: Mathematical and Theoretical* **50**, 223001 (2017).
- [66] D. Perez-Garcia, F. Verstraete, M. M. Wolf, and J. I. Cirac. “Matrix product state representations” (2007). [arXiv:quant-ph/0608197](#).
- [67] Marcin Jarzyna and Rafał Demkowicz-Dobrzański. “Matrix product states for quantum metrology”. *Physical Review Letters* **110** (2013).
- [68] Nic Ezzell, Zoë Holmes, and Patrick J. Coles. “The quantum low-rank approximation problem” (2022). [arXiv:2203.00811](#).
- [69] Anirban N. Chowdhury, Guang Hao Low, and Nathan Wiebe. “A variational quantum algorithm for preparing quantum gibbs states” (2020). [arXiv:2002.00055](#).
- [70] Brian Skinner, Jonathan Ruhman, and Adam Nahum. “Measurement-induced phase transitions in the dynamics of entanglement”. *Physical Review X* **9** (2019).
- [71] Benjamin Schumacher. “Quantum coding”. *Phys. Rev. A* **51**, 2738–2747 (1995).
- [72] Aaron Somoroff, Quentin Ficheux, Raymond A. Mencia, Haonan Xiong, Roman Kuzmin, and Vladimir E. Manucharyan. “Millisecond coherence in a superconducting qubit”. *Phys. Rev. Lett.* **130**, 267001 (2023).
- [73] Yulong Dong, Jonathan A. Gross, and Murphy Yuezhen Niu. “Optimal low-depth quantum signal-processing phase estimation”. *Nature Communications* **16** (2025).
- [74] Lorenzo Laneve. “Quantum signal processing over $SU(N)$ ” (2024). [arXiv:2311.03949](#).
- [75] Xi Lu, Yuan Liu, and Hongwei Lin. “Quantum signal processing and quantum singular value transformation on $U(N)$ ” (2024). [arXiv:2408.01439](#).
- [76] Andreas Elben, Steven T. Flammia, Hsin-Yuan Huang, Richard Kueng, John Preskill, Benoît Vermersch, and Peter Zoller. “The randomized measurement toolbox”. *Nature Reviews Physics* **5**, 9–24 (2022).
- [77] Zane M. Rossi and Isaac L. Chuang. “Multivariable quantum signal processing (M-QSP): prophecies of the two-headed oracle”. *Quantum* **6**, 811 (2022).
- [78] Balázs Németh, Blanka Kövér, Boglárka Kulcsár, Roland Botond Miklósi, and András Gilyén. “On variants of multivariate quantum signal processing and their characterizations” (2023). [arXiv:2312.09072](#).
- [79] Niladri Gomes, Hokiat Lim, and Nathan Wiebe. “Multivariable qsp and bosonic quantum simulation using iterated quantum signal processing” (2024). [arXiv:2408.03254](#).
- [80] Yuri Alexeev, Maximilian Amsler, Marco Antonio Barroca, Sanzio Bassini, Torey Battelle, Daan Camps, David Casanova, Young jai Choi, Frederic T Chong, Charles Chung, et al. “Quantum-centric supercomputing for materials science: A perspective on challenges and future directions”. *Future Generation Computer Systems* (2024).
- [81] Yuan Liu, Oinam R Meitei, Zachary E Chin, Arkopal Dutt, Max Tao, Isaac L Chuang, and Troy Van Voorhis. “Bootstrap embedding on a quantum computer”. *Journal of Chemical Theory and Computation* **19**, 2230–2247 (2023).
- [82] Bela Bauer, Sergey Bravyi, Mario Motta, and Garnet Kin-Lic Chan. “Quantum algorithms for quantum chemistry and quantum materials science”. *Chemical Reviews* **120**, 12685–12717 (2020).
- [83] Dominic W. Berry, Andrew M. Childs, Richard Cleve, Robin Kothari, and Rolando D. Somma. “Exponential improvement in precision for simulating sparse hamiltonians”. *Proceedings of the 46th Annual ACM Symposium on Theory of Computing* (2014).
- [84] Dominic W. Berry, Andrew M. Childs, and Robin Kothari. “Hamiltonian simulation with nearly optimal dependence on all parameters”. In 2015 IEEE 56th Annual Symposium on Foundations of Computer Science. *Page 792–809*. IEEE (2015).
- [85] Isaac Chuang, Andrew Tan, and John M Martyn. “PyQSP: Python Quantum Signal Processing”. <https://github.com/ichuang/pyqsp> (2021).
- [86] Theodore J Rivlin. “Chebyshev polynomials”. Courier Dover Publications. (2020).
- [87] John P Boyd. “Chebyshev and fourier spectral methods”. Courier Corporation. (2001).
- [88] Q. I. Rahman and G. Schmeisser. “Analytic theory of polynomials”. Clarendon Press, Oxford. (2002).
- [89] George B Arfken, Hans J Weber, and Frank E Harris. “Mathematical methods for physicists: a comprehensive guide”. Academic press. (2011).
- [90] J Oliver. “A note on the signs of truncated chebyshev polynomials”. *BIT Numerical Mathematics* **18**, 233–235 (1978).

A Implementation of Arbitrary Polynomials with Linear-Combination-of-Unitaries and Generalized QSP

As we mentioned in Sec. 2.1, QSP generates polynomials that are restricted to obey the conditions of Eq. (6), such as having definite parity. However, there exist techniques to expand this class of polynomials and implement arbitrary polynomials, provided they are bounded in magnitude.

One method of achieving this is with linear-combination-of-unitaries (LCU) circuits [83, 60, 84]. In this context, an LCU circuit is composed of controlled QSP sequences and allows one to sum together polynomials. This can be used to construct a polynomial of indefinite parity by summing together its even and odd components, as we illustrate in Fig. 4. The LCU construction is sequential, implying that for a polynomial of degree d , the requisite query depth is $2d$. However, an LCU circuit rescales the sum by a constant, and thus requires amplitude amplification or additional measurements to accommodate for this rescaling. In the context of parallel QSP, where block encodings are multiplied together, this rescaling will accumulate exponentially in the number of threads and increase the measurement cost.

LCU Circuit

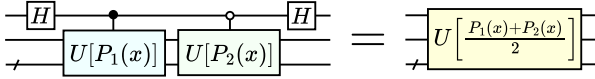


Figure 4: The linear-combination-of-unitaries (LCU) circuit that encodes a sum of two block encodings. Here $U[P_1(x)]$ and $U[P_2(x)]$ block-encode polynomials $P_1(x)$ and $P_2(x)$, and are realized through QSP. Note how the sum of these polynomials is rescaled by a factor of 2. Also observe that this construction is sequential, and thus doubles the query depth.

A more promising approach is the recently introduced construction of *generalized QSP* [29]. Generalized QSP proposes a generalization of QSP sequence (Eq. (4)) that instead uses controlled block encoding:

$$\begin{bmatrix} U[A] & 0 \\ 0 & I \end{bmatrix}, \quad \begin{bmatrix} I & 0 \\ 0 & U[A]^\dagger \end{bmatrix}. \quad (61)$$

By interspersing these with general $SU(2)$ rotations (in contrast to z -rotations of Eq. (4)), Ref. [29] shows how the resulting sequence block-encodes a polynomial $P(U[A])$, restricted only by the constraint $\|P\|_{[-1,1]} \leq 1$. In particular, this construction requires $2d$ queries to the block encodings of Eq. (61) to generate a degree- d polynomial. This alleviates the parity constraint of or-

dinary QSP, while also avoiding the rescaling imposed by the LCU method.

In the context of ordinary QSP, where we assume access to a block encoding, generalized QSP applies as follows. By a technique known as qubitization [4, 5, 6], which itself underlies QSP, there exists a privileged basis in which a block encoding of A is equivalent to an x -rotation:

$$U[A] = \begin{bmatrix} A & i\sqrt{1-A^2} \\ i\sqrt{1-A^2} & A \end{bmatrix} = e^{iX \arccos(A)}. \quad (62)$$

By then controlling on this block encoding and its inverse with an additional qubit, we can construct the inputs of Eq. (61) needed for generalized QSP. Then, executing generalized QSP on this, we can encode a polynomial with coefficients c_n :

$$P(U[A]) = \sum_{n=0}^d c_n U[A]^n = \sum_{n=0}^d c_n e^{iXn \arccos(A)}. \quad (63)$$

By projecting the remaining qubit into the $|0\rangle\langle 0|$ element, we attain the polynomial

$$\begin{aligned} \langle 0|P(U[A])|0\rangle &= \sum_{n=0}^d c_n \cos(n \arccos(A)) \\ &= \sum_{n=0}^d c_n T_n(A) =: \tilde{P}(A) \end{aligned} \quad (64)$$

where $T_n(x)$ is the order n Chebyshev polynomial (see Appendix C for review). As a linear combination of Chebyshev polynomials, this can represent an arbitrary degree- d polynomial $\tilde{P}(A)$, provided it is bounded as $\|\tilde{P}\|_{[-1,1]} \leq 1$.

Together, these results are summarised as follows. A degree- d polynomial of definite parity can be implemented directly through QSP with a query depth d . On the other hand, a degree- d polynomial of indefinite parity can be implemented via generalized QSP at the expense of an additional control qubit, access to the inverse block-encoding unitary $U[A]^\dagger$, and an increased query depth $2d$.

B Discussion of Code Implementation

The code for our implementation of parallel QSP can be found at Ref. [54]. With the goal of applying parallel QSP to the estimation of the property $w = \text{tr}(P(\rho))$, this code takes as input a degree- d polynomial $P(x)$ and an integer $k \geq 1$ corresponding to the number of threads over which to parallelize over. The code then executes the construction of Theorem 4.1: it decomposes $P(x)$ into constituent polynomials as per Eq. (35), assuming

that $P(x) \geq 0$ is non-negative over the real axis. It then numerically finds the roots of $P_{\geq k}(x)$ to factorize it into k factor polynomials according to Theorem 3.2. This factorization is done arbitrarily and is not optimized to minimize the factorization constant $\mathcal{K}(P_{\geq k})$; this optimization would be an interesting problem to study for future work.

For each resulting factor polynomials, we run a QSP phase finding algorithm. Using the $|+\rangle\langle+|$ block of a QSP sequence, this algorithm determines the QSP phases corresponding to the even/odd and real/imaginary components of the factor polynomials, such that the factor polynomials can be constructed via an LCU circuit. As the degree of each factor polynomial is reduced to $O(d/k)$, we can employ a simple phase finding algorithm mirroring that of Ref. [85]: we estimate the QSP phases by minimizing the sum of the squares error across $x \in [-1, 1]$.

C Upper Bounds on the Magnitude of Constituent Polynomials

Here we will prove bounds on properties of the constituent polynomials discussed in Sec. 4. Consider a real degree- d polynomial $P(x)$ that is bounded in magnitude by 1 over $x \in [-1, 1]$:

$$P(x) = \sum_{n=0}^d a_n x^n, \quad \|P\|_{[-1,1]} \leq 1. \quad (65)$$

As in Sec. 4, we split this polynomial into two constituent polynomials, $P_{<k}(x)$ and $P_{\geq k}(x)$, as

$$\begin{aligned} P(x) &= \sum_{n=0}^{k-1} a_n x^n + x^k \sum_{n=k}^d a_n x^{n-k} \\ &=: P_{<k}(x) + x^k P_{\geq k}(x), \end{aligned} \quad (66)$$

where

$$P_{<k}(x) := \sum_{n=0}^{k-1} a_n x^n, \quad P_{\geq k}(x) := \sum_{n=0}^{d-k} a_{n+k} x^n, \quad (67)$$

are the constituent polynomials of $P(x)$, of degree $k-1$ and $d-k$, respectively.

In order to estimate the trace $z = \text{tr}(P(\rho))$ using parallel QSP, we estimate $w_{<k} = \text{tr}(P_{<k}(\rho))$ and $w_{\geq k} = \text{tr}(P_{\geq k}(\rho))$ separately. As shown in Sec. 4, the measurement cost of obtaining these estimates depends on the magnitudes of the constituent polynomials $P_{<k}(x)$ and $P_{\geq k}(x)$. Although $\|P\|_{[-1,1]} \leq 1$, $P_{<k}(x)$ and $P_{\geq k}(x)$ are not necessarily bounded in magnitude by 1. Below, we show that for any such polynomial

$P(x)$, the constituent polynomials are bounded as

$$\begin{aligned} \|P_{<k}\|_{[-1,1]} &\leq O\left(\frac{d^{k-1}}{(k-1)!}\right), \\ \|P_{\geq k}\|_{[-1,1]} &\leq O\left(\frac{d^k}{k!} \sqrt{\frac{d}{k}}\right). \end{aligned} \quad (68)$$

Therefore, $\|P_{<k}\|_{[-1,1]}, \|P_{\geq k}\|_{[-1,1]} \leq O(\text{poly}(d))$. This implies that the measurement cost of parallel QSP for property estimation is at worst polynomial in d .

In proving these bounds we will invoke the Chebyshev polynomials [86]. For context, recall that the n th Chebyshev polynomial $T_n(x)$ is a degree n polynomial defined on $|x| \leq 1$ as

$$T_n(x) = \cos(n \arccos(x)). \quad (69)$$

It is well-established that $T_n(x)$ is a degree n polynomial of fixed parity (either even or odd, depending on n) and bounded magnitude $|T_n(x)|_{[-1,1]} = 1$ [87, 86]. They may be expressed as polynomials as

$$T_n(x) = \sum_{j=0}^n t_{n,j} x^j, \quad (70)$$

where $t_{n,j}$ are the corresponding coefficients. It can also be shown that $T_n(x)$ can be re-expressed as

$$T_n(x) = \frac{1}{2} \left((x - \sqrt{x^2 - 1})^n + (x + \sqrt{x^2 - 1})^n \right). \quad (71)$$

The Chebyshev polynomials provide a convenient basis for expanding functions on $x \in [-1, 1]$. A function $F(x)$ can be expanded as

$$F(x) = \frac{c_0}{2} + \sum_{n=1}^{\infty} c_n T_n(x), \quad (72)$$

where

$$c_n = \frac{2}{\pi} \int_{-1}^1 \frac{F(x) T_n(x)}{\sqrt{1-x^2}} dx \quad (73)$$

are the Chebyshev coefficients for all $n \geq 0$. These coefficients can be bounded as

$$|c_n| \leq \frac{2}{\pi} \int_{-1}^1 \frac{\|F\|_{[-1,1]}}{\sqrt{1-x^2}} dx = \|F\|_{[-1,1]}. \quad (74)$$

Moreover, if $F(x)$ is degree- d polynomial, then this series truncates at order d .

C.1 Bound on the Magnitude of $P_{<k}(x)$

To bound the magnitude of $P_{<k}(x)$, we begin by noting the result of the following theorems from Ref. [88]:

Theorem C.1 (Bound on Coefficients of Bounded Polynomials; Theorems 16.3.1 and 16.3.2 of Ref. [88]). Let $T_d(x) = \sum_{n=0}^d t_{d,n} x^n$ denote the Chebyshev polynomial of degree d . Let $P(x) = \sum_{n=0}^d a_n x^n$ be a degree- d polynomial that is bounded at the Chebyshev nodes as

$$\begin{aligned} |P(\cos(\frac{\pi\nu}{d}))| &\leq 1 \quad \text{for } \nu = 0, 1, \dots, d, \\ |P(\cos(\frac{\pi\nu}{d-1}))| &\leq 1 \quad \text{for } \nu = 0, 1, \dots, d-1. \end{aligned} \quad (75)$$

Then, for even d , the coefficients are bounded as

$$\begin{aligned} |a_{2j}| &\leq |t_{d,2j}| \\ |a_{2j-1}| &\leq |t_{d-1,2j-1}| \end{aligned} \quad (76)$$

for $j = 0, 1, \dots, d/2$. Analogously, for odd d , the coefficients are bounded as

$$\begin{aligned} |a_{2j}| &\leq |t_{d-1,2j}| \\ |a_{2j-1}| &\leq |t_{d,2j-1}| \end{aligned} \quad (77)$$

for $j = 0, 1, \dots, (d-1)/2$. Equality is achieved for $P(x) = T_d(x)$.

Applicability of this theorem demands that $P(x)$ be bounded by 1 at the Chebyshev nodes; this condition is satisfied for a bounded polynomial $\|P\|_{[-1,1]} \leq 1$ as we consider here. Therefore, Theorem C.1 indicates that the coefficients of any such bounded polynomial are necessarily upper bounded in magnitude by the coefficients of Chebyshev polynomials. To employ this result in practice, it will be useful to also show that the coefficients of the Chebyshev polynomials scale as $|t_{d,n}| \leq O(\frac{d^n}{n!})$:

Lemma C.1 (Bound on Coefficients of the Chebyshev Polynomials). The coefficients of the Chebyshev polynomials are bounded in magnitude as

$$|t_{d,n}| \leq \frac{(d+n)^n}{n!} = O\left(\frac{d^n}{n!}\right) \quad (78)$$

Proof. Explicitly, the Chebyshev coefficients are [89]

$$t_{d,n} = (-1)^{\frac{d-n}{2}} 2^{n-1} d^{\frac{(d+n)}{2}} \frac{(\frac{d+n}{2}-1)!}{(\frac{d-n}{2})! n!}. \quad (79)$$

This is upper bounded as

$$\begin{aligned} |t_{d,n}| &= 2^{n-1} d^{\frac{(d+n)}{2}} \frac{(\frac{d+n}{2}-1)!}{(\frac{d-n}{2})! n!} = \frac{2^{n-1} d^{\frac{(d+n)}{2}}}{\frac{d-n}{2}} \binom{\frac{d+n}{2}}{n} \\ &\leq 2^n \frac{(\frac{d+n}{2})^n}{n!} = \frac{(d+n)^n}{n!} = O\left(\frac{d^n}{n!}\right), \end{aligned} \quad (80)$$

where we have used that $\binom{a}{b} \leq \frac{a^b}{b!}$. $\square$

We can then merge Theorem C.1 and Lemma C.1 to bound $P_{<k}(x)$ as follows:

Theorem C.2. For any degree- d polynomial $P(x)$ that is bounded as $\max_{x \in [-1,1]} |P(x)| \leq 1$, its constituent polynomial $P_{<k}(x)$ (as defined in Eq. 67) is necessarily bounded as

$$\|P_{<k}\|_{[-1,1]} \leq O\left(\frac{d^{k-1}}{(k-1)!}\right) = \text{poly}(d). \quad (81)$$

Proof. To bound the magnitude of $P_{<k}(x)$, we first note that the triangle inequality implies

$$\|P_{<k}\|_{[-1,1]} = \max_{x \in [-1,1]} \left| \sum_{n=0}^{k-1} a_n x^n \right| \leq \sum_{n=0}^{k-1} |a_n|. \quad (82)$$

In conjunction with Theorem C.1 and Lemma C.1, we can bound this as

$$\sum_{n=0}^{k-1} |a_n| \leq \sum_{n=0}^{k-1} |t_{d,n}| \leq \sum_{n=0}^{k-1} O\left(\frac{d^n}{n!}\right) = O\left(\frac{d^{k-1}}{(k-1)!}\right). \quad (83)$$

$\square$

Therefore, $P_{<k}(x)$ is necessarily bounded in magnitude by $O(d^{k-1}/(k-1)!)$. While a precise bound on $P_{<k}(x)$ depends on the coefficients of the polynomial $P(x)$ (and can grow slower than $O(d^{k-1}/(k-1)!)$), Theorem C.2 indicates that even in the worst case, the magnitude of $P_{<k}(x)$ only grows polynomially in d . We used this result in Sec. 4 to prove that the estimation of $w_{<k} = \text{tr}(P_{<k}(\rho))$ requires at most $\text{poly}(d)$ measurements.

C.2 Bound on the Magnitude of $P_{\geq k}(x)$

To bound $P_{\geq k}(x)$, it will not suffice to consider the sum of the magnitudes of the corresponding coefficients, as we did for $P_{<k}(x)$. In general, this sum can grow exponentially with d , which we aim to avoid. For example, for the Chebyshev polynomials, this sum is $\sum_n |t_{d,n}| = 2^{O(d)}$.

Instead, we will derive our result by first considering the constituent polynomials of the Chebyshev polynomials:

$$T_n(x) =: T_{n;<k}(x) + x^k T_{n;\geq k}(x). \quad (84)$$

We will also consider the polynomial constructed from truncating the low order terms of a Chebyshev polynomial:

$$\mathcal{T}_{n;k}(x) := \sum_{j=k}^n t_{n,j} x^j \quad (85)$$

This is related to the constituent polynomial $T_{n;\geq k}(x)$ as

$$T_{n;\geq k}(x) = \mathcal{T}_{n;k}(x)/x^k. \quad (86)$$

Because the Chebyshev polynomials have fixed parity, it is only relevant to consider $\mathcal{T}_{n;k}$ for n and k of the same

parity (i.e., both either even or odd). Ref. [90] studied these truncated Chebyshev polynomials and showed they have definite sign:

Theorem C.3 (Main Result of Ref. [90]). *The truncated Chebyshev polynomials $\mathcal{T}_{n;k}(x)$ have definite sign over $x \in [0, 1]$:*

$$(-1)^l \mathcal{T}_{n;n-2l}(x) \geq 0 \text{ for } x \in [0, 1], \quad (87)$$

for all $l = 0, 1, \dots, \lfloor \frac{n}{2} \rfloor - 1$.

This implies that the sign of $\mathcal{T}_{n;n-2l}(x)$ over $x \in [0, 1]$ is equal to the sign of the coefficient $t_{n,n-2l}$. In the region $x \in [-1, 0]$, the sign of is determined by the parity of $\mathcal{T}_{n;n-2l}(x)$, or equivalently the parity of n . We can use this result to prove the following bound on $T_{n;\geq k}(x)$.

Corollary C.3.1 (Maximum of $T_{n;\geq k}(x)$). *For n and k of the same parity (i.e., both even or odd), the maximum magnitude of the constituent polynomial $T_{n;\geq k}(x)$ over $x \in [-1, 1]$ occurs at $x = 0$ and takes the value:*

$$\max_{x \in [-1, 1]} |T_{n;\geq k}(x)| = |T_{n;\geq k}(0)| = |t_{n,k}| = O\left(\frac{n^k}{k!}\right). \quad (88)$$

Proof. The proof will proceed by induction on increasing n . First, note that for n and k of the same parity, the partial sum polynomial $T_{n;\geq k}(x)$ consists of only even powers, and hence is an even function. According to Theorem C.3 and Eq. (86), this function is of constant sign over $-1 \leq x \leq 1$, and this sign is $\text{sign}(t_{n,k}) = (-1)^{\frac{n-k}{2}}$.

Moving to the proof by induction, we will make the inductive hypothesis that the maximum magnitudes of the constituent polynomials are achieved at $x = 0$:

$$\forall n' \leq n, |T_{n';\geq k}(0)| \geq |T_{n';\geq k}(x)|, \quad (89)$$

for all $k = 0, 2, \dots, n'$ if n' is even, or $k = 1, 3, \dots, n'$ if n' is odd.

Let us show that the base cases $n' = 0$ and $n' = 1$ are satisfied. For $n' = 0$, we have $|T_{0;\geq 0}(x)| = 1 \leq |T_{0;\geq 0}| = 1$ is satisfied trivially. For $n' = 1$, we similarly have $|T_{1;\geq 1}(x)| = |t_{1,1}| \leq |T_{1;\geq 1}(0)| = |t_{1,1}|$. It is then straightforward to consider larger values of n' , such as $n' = 2$ in which case we have

$$\begin{aligned} |T_{2;\geq 0}(x)| &\leq 1 = |T_{2;\geq 0}(0)|, \text{ and} \\ |T_{2;\geq 2}(x)| &= |t_{2,2}| \leq |T_{2;\geq 2}(0)| = |t_{2,2}|. \end{aligned} \quad (90)$$

at $k = 0$ and $k = 2$, respectively.

Then, supposing that the inductive hypothesis is true up to $n' = n$, we can show that it is also true for $n' = n + 1$. To prove this, note that the Chebyshev polynomials obey the recursion relation

$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x). \quad (91)$$

This implies that, for k of the same parity as $n + 1$, the constituent polynomials obey

$$T_{n+1;\geq k}(x) = 2T_{n;\geq k-1}(x) - T_{n-1;\geq k}(x) \quad (92)$$

for $k \leq n - 1$ (which ensures that the polynomial $T_{n-1;\geq k}(x)$ exists). As per Theorem C.3 and Eq. (86), $T_{n;\geq k-1}(x)$ and $T_{n-1;\geq k}(x)$ have constant and opposite sign over $-1 \leq x \leq 1$. This implies that

$$\begin{aligned} |T_{n+1;\geq k}(x)| &\leq 2|T_{n;\geq k-1}(x)| + |T_{n-1;\geq k}(x)| \\ &\leq 2|T_{n;\geq k-1}(0)| + |T_{n-1;\geq k}(0)| \\ &= |2T_{n;\geq k-1}(0) - T_{n-1;\geq k}(0)| \\ &= |T_{n+1;\geq k}(0)|, \end{aligned} \quad (93)$$

where this first inequality is an application of the triangle inequality, the second inequality is the inductive hypothesis, and the last equalities follow from the constant sign of the constituent polynomials. This holds true for $k = 0, \dots, n - 1$ as per Eq. (92). We can prove the remaining cases $k = n$ and $k = n + 1$ as follows. For $k = n$, the recurrence relation of Eq. (91) corresponds to

$$T_{n+1;\geq n}(x) = 2T_{n;\geq n-1}(x), \quad (94)$$

such that

$$\begin{aligned} |T_{n+1;\geq n}(x)| &= 2|T_{n;\geq n-1}(x)| \\ &\leq 2|T_{n;\geq n-1}(0)| = |T_{n+1;\geq n}(0)|. \end{aligned} \quad (95)$$

For $k = n + 1$, the hypothesized inequality is trivially satisfied because the constituent polynomial is a constant: $|T_{n+1;\geq n+1}(0)| = |t_{n+1,n+1}| \geq |T_{n+1;\geq n+1}(0)| = |t_{n+1,n+1}|$.

Therefore, by induction on increasing n , this completes the proof that the maximum of the constituent polynomial $T_{n;\geq k}(x)$ occurs at $x = 0$. Moreover, the value at $x = 0$ is $|T_{n;\geq k}(0)| = |t_{n,k}| = O\left(\frac{n^k}{k!}\right)$. $\square$

We can use Corollary C.3.1 in conjunction with the Chebyshev decomposition of a polynomial to prove the following bound on an arbitrary constituent polynomial:

Theorem C.4 (Bound on constituent polynomial $P_{\geq k}(x)$). *For a polynomial $P(x)$ that is bounded as $\|P\|_{[-1,1]} \leq 1$, its partial sum polynomial is necessarily bounded as*

$$\|P_{\geq k}\|_{[-1,1]} \leq O\left(\frac{d^k}{k!} \sqrt{\frac{d}{k}}\right) = O(\text{poly}(d)). \quad (96)$$

Proof. To prove this result, first decompose $P(x)$ into the basis of Chebyshev polynomials:

$$P(x) = \sum_{n=0}^d c_n T_n(x) \quad (97)$$

where c_n are the Chebyshev coefficients of $P(x)$. The Chebyshev polynomials obey the orthogonality relation $\frac{2-\delta_{n,0}}{\pi} \int_{-1}^1 \frac{T_n(x)T_m(x)}{\sqrt{1-x^2}} dx = \delta_{nm}$, such that the coefficients c_n are given by

$$c_n = \frac{2-\delta_{n,0}}{\pi} \int_{-1}^1 dx \frac{P(x)T_n(x)}{\sqrt{1-x^2}}. \quad (98)$$

In this basis, the constituent polynomial can be expressed as

$$P_{\geq k}(x) = \sum_{n=k}^d c_n T_{n;\geq k}(x). \quad (99)$$

Next, we can employ the Cauchy-Schwarz inequality to show that

$$\begin{aligned} |P^{2,k}(x)| &= \left| \sum_{n=k}^d c_n T_{n;\geq k}(x) \right| \\ &\leq \sqrt{\sum_{n=k}^d |c_n|^2} \times \sqrt{\sum_{n=k}^d |T_{n;\geq k}(x)|^2} \end{aligned} \quad (100)$$

Applying Parseval's theorem with the inner product of the Chebyshev polynomials, the 2-norm of c_n is upper bounded as

$$\begin{aligned} \frac{2}{\pi} \int_{-1}^1 dx \frac{|P(x)|^2}{\sqrt{1-x^2}} &= \frac{|c_0|^2}{2} + \sum_{n=1}^d |c_n|^2 \\ &\leq \frac{2}{\pi} \int_{-1}^1 dx \frac{1}{\sqrt{1-x^2}} = 1, \end{aligned} \quad (101)$$

and therefore $\sum_{n=k}^d |c_n|^2 \leq 2 = O(1)$. On the other hand, the 2-norm of the Chebyshev polynomials can be upper bounded by invoking Corollary C.3.1:

$$\begin{aligned} \max_{x \in [-1,1]} \sum_{n=k}^d |T_{n;\geq k}(x)|^2 &\leq \sum_{n=k}^d |t_{n,k}|^2 \leq \sum_{n=k}^d O\left(\frac{n^{2k}}{(k!)^2}\right) \\ &= O\left(\frac{1}{(k!)^2} \cdot \frac{d^{2k+1}}{2k+1}\right) = O\left(\left(\frac{d^k}{k!}\right)^2 \frac{d}{k}\right), \end{aligned} \quad (102)$$

where we have used that $\sum_{n=0}^d n^{2k} = \frac{d^{2k+1}}{2k+1} + O(d^{2k})$. Inputting these upper bound into Eq. (100), we obtain

$$|P_{\geq k}(x)| \leq O\left(\frac{d^k}{k!} \sqrt{\frac{d}{k}}\right), \quad (103)$$

for $x \in [-1, 1]$. This completes the proof of the stated result.

As an aside, we suspect this bound could be sharpened to $\|P_{\geq k}\|_{[-1,1]} \leq O\left(\frac{d^k}{k!}\right)$, which is saturated by the Chebyshev polynomials. $\square$

D Augmenting Trace Estimation with Importance Sampling

Importance sampling can be utilized to expand the class of functions whose traces can be estimated. To demonstrate this for functions of a density matrix, suppose that we have the ability to estimate the trace of a set of basis functions $\{B_j(\rho)\}_{j=1}^d$, by using QSP or other techniques. For example, one could have $B_j(\rho) = \rho^j$ be monomials, or even $B_j(\rho) = T_j(\rho)$ be the Chebyshev polynomials. In practice, the trace of a basis function is approximated by repeatedly measuring an estimator $\hat{B}_j$, whose expectation value is the desired trace:

$$\mathbb{E}[\hat{B}_j] = \text{tr}(B_j(\rho)) \quad (104)$$

For example, in the QSP test (Sec. 2.3), the estimator is $\hat{B} = m \in \{0, 1\}$, where m is the measurement of the block-encoding qubit.

Given this ability, one can expand the class of functions whose traces can be estimated by incorporating importance sampling. Consider a function $f(\rho)$ expanded in the basis $\{B_j(\rho)\}_{j=1}^d$ as

$$f(\rho) = \sum_{j=1}^d c_j B_j(\rho), \quad (105)$$

with complex coefficients c_j . In order to estimate the trace $\text{tr}(f(\rho))$, first define a probability distribution $p(j) = |c_j|/\|c\|_1$, where $\|c\|_1 = \sum_j |c_j|$ is the 1-norm of the coefficients. Then, by sampling an integer $j \sim p(j)$ and evaluating the corresponding estimator $\hat{B}_j$, one can construct the following quantity whose expectation value yields the desired trace:

$$\mathbb{E}_{j \sim p} \left[\frac{c_j}{|c_j|} \hat{B}_j \right] = \sum_{j=1}^d p_j \frac{c_j}{|c_j|} \text{tr}(B_j(\rho)) = \frac{\text{tr}(f(\rho))}{\|c\|_1}. \quad (106)$$

Due to the rescaling by $\|c\|_1$, estimating $\text{tr}(f(\rho))$ to additive error ϵ requires $O(\|c\|_1^2/\epsilon^2)$ measurements.

Ref. [48] uses this importance sampling procedure to estimate a large class of traces and expectation values, provided the ability to generate Chebyshev polynomials $B_j(\rho) = T_j(\rho)$ with QSP. A similar sampling procedure was also used in Ref. [23] to estimate the trace of functions of a density matrix, given only the ability to estimate the trace of the monomials $B_j(\rho) = \rho^j$. In both references, because the measurement cost depends quadratically on the 1-norm of the coefficients, the authors note that this method is best suited for well-behaved functions whose 1-norm is not prohibitively large (i.e., scales only polynomially in d rather than exponentially).

In our work, we use importance sampling to extend parallel QSP from the limited scope of Theorem 3.2 to a larger class of property estimation problems according to Theorem 4.2. We achieve this by decomposing a trace $\text{tr}(P(\rho))$ into a linear combination of terms that are each amenable to parallel QSP, and applying importance sampling to this sum. We discuss the proof of this theorem in the following section.

E Proofs for Parallel QSP for Property Estimation

Here we prove Theorem 4.2, which provides a general scheme for estimating a property $w = \text{tr}(P(\rho))$ with parallel QSP. As we mentioned in the main text, we achieve this by decomposing $P(x)$ into a linear combination of polynomials that are each amenable to parallel QSP. We can then apply importance sampling to this linear combination to extract w .

The basis that we choose to decompose into is the basis of Chebyshev polynomials. In this basis, we can present and prove the theorem:

Theorem E.1 (Parallel QSP for Property Estimation: Definite Parity). *Let $P(x)$ be a real polynomial of degree d and definite parity, that is bounded as $\|P\|_{[-1,1]} \leq 1$. By invoking parallel QSP across k threads, where k has the same parity as d , we can estimate*

$$w = \text{tr}(P(\rho)) \quad (107)$$

with a circuit of width $O(k)$ and query depth at most $\lfloor \frac{d-k}{2k} \rfloor + k - 1 = O(d/k + k)$. The number of measurements required to resolve w to additive error ϵ is

$$\begin{aligned} & O\left(\frac{\|P_{<k}\|_{[-1,1]}^2}{\epsilon^2} + \frac{\|P_{<k}\|_{[-1,1]}^2 d^4 (1 + \sqrt{2})^{4k}}{k^2 \epsilon^2}\right) \\ &= O\left(\frac{\|P_{<k}\|_{[-1,1]}^2 + \|P_{\geq k}\|_{[-1,1]}^2 d^4 2^{O(k)}}{\epsilon^2}\right). \end{aligned} \quad (108)$$

Proof. Suppose $P(x)$ is of definite parity, and that k has the same parity. Then, by decomposing $P(x) = P_{<k}(x) + x^k P_{\geq k}(x)$, $P_{<k}(x)$ has the same parity as $P(x)$, whereas $P_{\geq k}(x)$ is necessarily even. As in the proof of Theorem 4.1, we seek to estimate $w_{<k}$ and $w_{\geq k}$ to error $\epsilon/2$ each, such that their sum approximates w to error ϵ .

First, we can estimate $w_{<k}$ with standard QSP. Because $P_{<k}(x)$ is real and of definite parity, the requisite query depth is $k - 1$ and the requisite number of measurements is $O\left(\frac{\|P_{<k}\|_{[-1,1]}^2}{\epsilon^2}\right)$.

Next, because $P_{\geq k}(x)$ is even, we can expand it in the basis of even Chebyshev polynomials:

$$\begin{aligned} P_{\geq k}(x) &= \sum_{j=0}^{\frac{d-k}{2}} c_{2j} T_{2j}(x) \\ &= \sum_{a=0}^{\lfloor \frac{d-k}{2k} \rfloor} \sum_{b=0}^{k-1} c_{a \cdot 2k + 2b} T_{a \cdot 2k + 2b}(x). \end{aligned} \quad (109)$$

where we have judiciously recast this as a sum over Chebyshev polynomials with orders expressed as multiples of $2k$, i.e. order $2j = a \cdot 2k + b$ for $a, b \geq 0$. Going forward, our approach will be to use properties of the Chebyshev polynomials to recast $P_{\geq k}(x)$ as a linear combination of polynomials that are each amenable to parallel QSP. Then, one can estimate the corresponding trace of the terms in this linear combination to furnish an approximation to $w_{\geq k}$.

In more detail, we will use the following properties of the Chebyshev polynomials [86]:

$$\begin{aligned} T_{mn}(x) &= T_m(T_n(x)) \\ T_{m+n}(x) &= 2T_m(x)T_n(x) - T_{|m-n|}(x). \end{aligned} \quad (110)$$

These properties enable, for instance, a Chebyshev polynomial of degree $4n$ to be recast as

$$T_{4n}(x) = T_4(T_n(x)) = 8T_n(x)^4 - 8T_n(x)^2 + 1. \quad (111)$$

The polynomials comprising this linear combination ($T_n(x)^4$, $T_n(x)^2$, and 1) are all positive definite and factorize trivially as products of Chebyshev polynomials of degree at most n . Thus, each term in this expression is amenable to parallel QSP, and the degree is reduced by a factor of 4. This is the strategy we will use going forward, but with the degree reduced by a factor of $2k$ instead of 4.

Applying this strategy to Eq. (109), we begin by writing

$$T_{a \cdot 2k + 2b}(x) = 2T_{a \cdot 2k}(x)T_{2b}(x) - T_{a \cdot 2k - 2b}(x) \quad (112)$$

for $b > 0$ and $a \cdot 2k - 2b \geq 0$. Inserting this expression into Eq. (109), each term $-T_{a \cdot 2k - 2b}(x)$ can be re-included into the sum by modifying the corresponding coefficient $c_{a \cdot 2k - 2b}$. Starting with the highest degree $a = \lfloor (d-k)/2k \rfloor$, this allows us to write

$$\begin{aligned} & \sum_{b=1}^{k-1} c_{a \cdot 2k + 2b} T_{a \cdot 2k + 2b}(x) = \\ & \sum_{b=1}^{k-1} c_{a \cdot 2k + 2b} (2T_{a \cdot 2k}(x)T_{2b}(x) - T_{(a-1) \cdot 2k + 2(k-b)}(x)). \end{aligned} \quad (113)$$

By re-including these terms into the sum of Eq. (109), this effectively changes the coefficients to (for $b > 0$)

$$c_{a \cdot 2k+2b} \mapsto 2c_{a \cdot 2k+b} \quad (114)$$

$$c_{(a-1) \cdot 2k+2(k-b)} \mapsto c_{(a-1) \cdot 2k+2(k-b)} - c_{a \cdot 2k+2b}.$$

We now decrement a , and recurse this procedure, updating the coefficients appropriately. Note that the second mapping in Eq. (114) is equivalent to $c_{a \cdot 2k+b} \mapsto c_{a \cdot 2k+b} - c_{(a+1) \cdot 2k+2(k-b)}$, which becomes the recursion

$$c_{a \cdot 2k+2b} \mapsto c_{a \cdot 2k+2b} - c_{(a+1) \cdot 2k+2(k-b)} + c_{(a+2) \cdot 2k+2b} - c_{(a+3) \cdot 2k+2(k-b)} \dots \quad (115)$$

Unfolding this recursion for each term in Eq. (109), we find that the new coefficients are

$$\tilde{c}_{a \cdot 2k+2b} = \begin{cases} c_{a \cdot 2k} & b = 0 \\ c_{2b} & a = 0 \\ 2 \sum_{j=0}^{\lfloor \frac{d-k}{2k} \rfloor - a} (-1)^j c_{(a+j) \cdot 2k+2\alpha_j} & a, b \geq 1, \end{cases} \quad (116)$$

where

$$\alpha_j = \begin{cases} b & j \text{ even} \\ k-b & j \text{ odd}, \end{cases} \quad (117)$$

such that we may rewrite $P_{\geq k}(x)$ as

$$P_{\geq k}(x) = \sum_{a=0}^{\lfloor \frac{d-k}{2k} \rfloor} \sum_{b=0}^{k-1} \tilde{c}_{a \cdot 2k+2b} T_{a \cdot 2k}(x) T_{2b}(x). \quad (118)$$

Because $|c_{2j}| \leq \|P_{\geq k}\|_{[-1,1]}$ as per Eq. (74), the magnitude of these coefficients is

$$|\tilde{c}_{a \cdot 2k+2b}| \leq 2\|P_{\geq k}\|_{[-1,1]} \left(\left\lfloor \frac{d-k}{2k} \right\rfloor - a \right). \quad (119)$$

Next, let us denote the Chebyshev polynomials of even degree as

$$T_{2n}(x) = \sum_{j=0}^n t_{2n,2j} x^{2j}. \quad (120)$$

Accordingly, we can express the product $T_{a \cdot 2k}(x) T_{2b}(x)$ in Eq. (118) as

$$\begin{aligned} T_{a \cdot 2k}(x) T_{2b}(x) &= T_{2k}(T_a(x)) T_2(T_b(x)) \\ &= \sum_{j=0}^k \sum_{l=0}^1 t_{2k,2j} t_{2,2l} T_a(x)^{2j} T_b(x)^{2l}. \end{aligned} \quad (121)$$

such that

$$\begin{aligned} P_{\geq k}(x) &= \sum_{a=0}^{\lfloor \frac{d-k}{2k} \rfloor} \sum_{b=0}^{k-1} \sum_{j=0}^k \sum_{l=0}^1 \tilde{c}_{a \cdot 2k+2b} t_{2k,2j} t_{2,2l} T_a(x)^{2j} T_b(x)^{2l} \\ &=: \sum_{a=0}^{\lfloor \frac{d-k}{2k} \rfloor} \sum_{b=0}^{k-1} \sum_{j=0}^k \sum_{l=0}^1 C_{abjl}^{(k)} T_a(x)^{2j} T_b(x)^{2l}, \end{aligned} \quad (122)$$

where we have defined the coefficients $C_{abjl}^{(k)} = \tilde{c}_{a \cdot 2k+2b} t_{2k,2j} t_{2,2l}$.

This allows us to write $w_{\geq k}$ as

$$\begin{aligned} w_{\geq k} &= \text{tr}(\rho^k P_{\geq k}(\rho)) \\ &=: \sum_{a=0}^{\lfloor \frac{d-k}{2k} \rfloor} \sum_{b=0}^{k-1} \sum_{j=0}^k \sum_{l=0}^1 C_{abjl}^{(k)} \text{tr}(\rho^k |T_a(\rho)^j T_b(\rho)^l|^2). \end{aligned} \quad (123)$$

This is now re-expressed as a linear combination of terms that are each amenable to parallel QSP. That is, the trace $\text{tr}(\rho^k |T_a(\rho)^j T_b(\rho)^l|^2)$ obeys the conditions of Theorem 3.2: the polynomial $R(x) = (T_a(x)^j T_b(x)^l)^2$ is real and non-negative, and trivially factorizes into k factor polynomials as

$$R(x) = \prod_{s=1}^k |\mathcal{R}_s(x)|^2 \quad (124)$$

where

$$\mathcal{R}_s(x) = \begin{cases} T_a(x) T_b(x) & s = 1, \text{ for } j \geq 1, l = 1, \\ T_b(x) & s = 1, \text{ for } j = 0, l = 1, \\ T_a(x) & s \geq 1, j > l, \\ 1 & s \geq j, l. \end{cases} \quad (125)$$

These factor polynomials are all real-valued, of definite parity, and have degree at most $a + b \leq \lfloor \frac{d-k}{2k} \rfloor + k - 1$. Therefore, they can each be directly implemented through QSP, with query depth at most $\lfloor \frac{d-k}{2k} \rfloor + k - 1 = O(d/k + k)$, and achieve a factorization constant $\mathcal{K} = 1$.

Using this strategy, we can estimate $w_{\geq k}$ by estimating the terms in the linear combination of Eq. (123) with parallel QSP. A particularly efficient way to do this is to use importance sampling according to Appendix D. That is, define a probability distribution $p(a, b, j, l) = |C_{a,b,j,l}^{(k)}| / \|C^{(k)}\|_1$. Then, by sampling $a, b, j, l \sim p(a, b, j, l)$ and evaluating the corresponding estimator of $\text{tr}(\rho^k |T_a(\rho)^j T_b(\rho)^l|^2)$ (i.e. the measurement of the parallel QSP circuit associated with this polynomial), we can construct a quantity whose expectation value converges to $w_{\geq k}$. The associated cost of estimating $w_{\geq k}$ to additive error ϵ is $O(\|C^{(k)}\|_1^2 / \epsilon^2)$.

In order to bound this cost, we can upper bound the 1-norm $\|C^{(k)}\|_1$ by invoking the identity

$$\sum_{j=0}^n |t_{2n,2j}| = \frac{1}{2}(1+\sqrt{2})^{2n} + \frac{1}{2}(1-\sqrt{2})^{2n} = O((1+\sqrt{2})^{2n}). \quad (126)$$

This follows from the fact that $t_{2n,2j} = (-1)^j (-1)^n |t_{2n,2j}|$ (see Eq. (79)), such that the 1-norm of the Chebyshev polynomial coefficients can be

expressed as

$$(-1)^n T_n(i) = (-1)^n \sum_{j=0}^n t_{2n,2j}(i)^{2j} = \sum_{j=0}^n |t_{2n,2j}|. \quad (127)$$

Using Eq. (71), this evaluates to $(-1)^n T_n(i) = \frac{1}{2}(1 + \sqrt{2})^{2n} + \frac{1}{2}(1 - \sqrt{2})^{2n}$. Therefore, using this identity, we can upper bound $\|C^{(k)}\|_1$ as

$$\begin{aligned} \|C^{(k)}\|_1 &= \sum_{a=0}^{\lfloor \frac{d-k}{2k} \rfloor} \sum_{b=0}^{k-1} \sum_{j=0}^k \sum_{l=0}^1 |\tilde{c}_{a \cdot 2k + 2b} t_{2k,2j} t_{2,2l}| \\ &= \sum_{a=0}^{\lfloor \frac{d-k}{2k} \rfloor} \sum_{b=0}^{k-1} |\tilde{c}_{a \cdot 2k + 2b}| \sum_{j=0}^k |t_{2k,2j}| \cdot \sum_{l=0}^1 |t_{2,2l}| \\ &= \|\tilde{c}\|_1 \cdot O((1 + \sqrt{2})^{2k}). \end{aligned} \quad (128)$$

Using Eq. (119), we can upper bound as $\|\tilde{c}\|_1$

$$\begin{aligned} \|\tilde{c}\|_1 &= \sum_{a=0}^{\lfloor \frac{d-k}{2k} \rfloor} \sum_{b=0}^{k-1} |\tilde{c}_{a \cdot 2k + 2b}| \\ &\leq \sum_{a=0}^{\lfloor \frac{d-k}{2k} \rfloor} \sum_{b=0}^{k-1} 2 \|P_{\geq k}\|_{[-1,1]} \left(\left\lfloor \frac{d-k}{2k} \right\rfloor - a \right) \\ &= 2k \|P_{\geq k}\|_{[-1,1]} \sum_{a=0}^{\lfloor \frac{d-k}{2k} \rfloor} \left(\left\lfloor \frac{d-k}{2k} \right\rfloor - a \right) \\ &= O\left(\|P_{\geq k}\|_{[-1,1]} \frac{d^2}{k} \right). \end{aligned} \quad (129)$$

And therefore we obtain a measurement cost

$$\begin{aligned} O\left(\frac{\|C^{(k)}\|_1^2}{\epsilon^2}\right) &= O\left(\frac{\|P_{\geq k}\|_{[-1,1]}^2 d^4 (1 + \sqrt{2})^{4k}}{k^2 \epsilon^2}\right) \\ &= O\left(\frac{\|P_{\geq k}\|_{[-1,1]}^2 d^4 2^{O(k)}}{\epsilon^2}\right). \end{aligned} \quad (130)$$

Lastly, we suspect that this bound could be tightened by improving the bound of Eq. (119), which is rather loose.

□