Sampling-based sublinear low-rank matrix arithmetic framework for dequantizing quantum machine learning

Nai-Hui Chia* András Gilyén† Tongyang Li‡
Han-Hsuan Lin* Ewin Tang§ Chunhao Wang*

Abstract

We present an algorithmic framework for quantum-inspired classical algorithms on close-to-low-rank matrices, generalizing the series of results started by Tang's breakthrough quantum-inspired algorithm for recommendation systems [STOC'19]. Motivated by quantum linear algebra algorithms and the quantum singular value transformation (SVT) framework of Gilyén, Su, Low, and Wiebe [STOC'19], we develop classical algorithms for SVT that run in time independent of input dimension, under suitable quantum-inspired sampling assumptions. Our results give compelling evidence that in the corresponding QRAM data structure input model, quantum SVT does not yield exponential quantum speedups. Since the quantum SVT framework generalizes essentially all known techniques for quantum linear algebra, our results, combined with sampling lemmas from previous work, suffice to generalize all prior results about dequantizing quantum machine learning algorithms. In particular, our classical SVT framework recovers and often improves the dequantization results on recommendation systems, principal component analysis, supervised clustering, support vector machines, low-rank regression, and semidefinite program solving. We also give additional dequantization results on low-rank Hamiltonian simulation and discriminant analysis. Our improvements come from identifying the key feature of the quantum-inspired input model that is at the core of all prior quantum-inspired results: ℓ^2-norm sampling can approximate matrix products in time independent of their dimension. We reduce all our main results to this fact, making our exposition concise, self-contained, and intuitive.

*Department of Computer Science, University of Texas at Austin. Research supported by Scott Aaronson's Vannevar Bush Faculty Fellowship from the US Department of Defense. Email: {nai,linhh,chunhao}@cs.utexas.edu

†Alfréd Rényi Institute of Mathematics. Formerly at the Institute for Quantum Information and Matter, California Institute of Technology. Funding provided by Samsung Electronics Co., Ltd., for the project "The Computational Power of Sampling on Quantum Computers", and by the Institute for Quantum Information and Matter, an NSF Physics Frontiers Center (NSF Grant PHY-1733907), as well as by the EU's Horizon 2020 Marie Skłodowska-Curie program 891889-QuantOrder. Email: gilyen@renyi.hu

‡Department of Computer Science, Institute for Advanced Computer Studies, and Joint Center for Quantum Information and Computer Science, University of Maryland. Research supported by IBM PhD Fellowship, QISE-NET Triplet Award (NSF DMR-1747426), and the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Quantum Algorithms Teams program. Email: tongyang@cs.umd.edu

§University of Washington. This material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE-1762114. Email: ewint@cs.washington.edu

Contents

1 Introduction

1.1 Motivation

Quantum machine learning (QML) is a field of study with a rapidly growing number of proposals for how quantum computers could significantly speed up machine learning tasks [DW20, CHI+18]. If any of these proposals yield substantial practical speedups, it could be the killer application motivating the development of scalable quantum computers [Pre18]. At first glance, many applications of QML seem to admit exponential speedups. However, these exponential speedups are less likely to manifest in practice compared to, say, Shor's algorithm for factoring [Sho97], because unlike their classical counterparts, QML algorithms must make strong input assumptions and learn relatively little from their output [Aar15]. These caveats arise because both loading input data into a quantum computer and extracting amplitude data from an output quantum state are hard in their most generic forms.

A recent line of research analyzes the speedups of QML algorithms by developing classical counterparts that carefully exploit these restrictive input and output assumptions. This began with a breakthrough 2018 paper by Tang [Tan19] showing that the quantum recommendation systems algorithm [KP17], previously believed to be one of the strongest candidates for a practical exponential speedup in QML, does not give an exponential speedup. Specifically, Tang describes a "dequantized" algorithm that solves the same problem as the quantum algorithm and only suffers a polynomial slowdown. Tang's algorithm crucially exploits the input data structure assumed by the quantum algorithm, which is used for efficiently preparing states. Subsequent work relies on similar techniques to dequantize a wide range of QML algorithms, including those for principal component analysis and supervised clustering [Tan21], low-rank linear system solving [CGL+20], low-rank semidefinite program solving [CLLW20], support vector machines [DBH21], nonnegative matrix factorization [CLS+19], and minimal conical hull [DHLT20]. These results show that the advertised exponential speedups of many QML algorithms disappear when compared to classical algorithms with input assumptions analogous to the state preparation assumptions of the quantum algorithms, drastically changing our understanding of the landscape of potential QML algorithm speedups.

A recent line of work in quantum algorithms has worked to unify many quantum algorithms ranging from quantum walks to QML, under a quantum linear algebra framework called quantum singular value transformation (QSVT) [LC17, CGJ19, GSLW19]. Since this framework captures essentially all known linear algebraic QML techniques [MRTC21], including all prior dequantized QML algorithms (up to minor technical details), a natural question is whether this framework can be dequantized. One cannot hope to dequantize *all* of QSVT, because with sparse block-encodings of input data, QSVT can simulate Harrow, Hassidim, and Lloyd's pioneering poly-logarithmic time algorithm (HHL) for the BQP-complete problem of sampling from the solution of a sparse system of linear equations [HHL09]. However, one could hope to dequantize QSVT provided that the input data comes in the state preparation data structure used commonly for quantum linear algebra. This data structure allows for efficient QML when the input is close-to-low-rank, but as dequantized algorithms show, it also gives significant power to classical algorithms. Prior work [Tan21, CGL+20] has made similar speculations that these techniques could feasibly dequantize wide swathes of quantum linear algebra. In this work, we give evidence for these hopes by presenting a classical analogue of the QSVT framework and applying it to dequantize QML algorithms.

1.2 Results

We describe a *simple* framework for quantum-inspired classical algorithms with *wide applicability*, grasping the capabilities and limitations of these techniques. We use this framework to dequantize

many quantum linear algebra algorithms. We also prove QSVT-like extensibility properties of our framework, giving evidence that it can dequantize any QSVT algorithms in the QRAM input model.

Input model: Oversampling and query access. Our framework assumes a specific input model called *oversampling and query access*, which can be thought of as a classical analogue to quantum state preparation assumptions, i.e., the ability to prepare a quantum state $|v\rangle$ proportional to some input vector v. Our conceptual contribution is to define this generalization of sampling and query access, because it has better closure properties.

We have *sampling and query access* to a vector $v \in \mathbb{C}^n$, denoted $\mathrm{SQ}(v)$, if we can efficiently make the following kinds of queries (Definition 3.2): (1) given an index $i \in [n]$, output the corresponding entry $v(i)$; (2) sample an index $j \in [n]$ with probability $|v(j)|^2/\|v\|^2$; and (3) output the vector's ℓ^2-norm $\|v\|$. We have sampling and query access to a matrix $A \in \mathbb{C}^{m\times n}$, denoted $\mathrm{SQ}(A)$, if we have $\mathrm{SQ}(A(i,\cdot))$ for all rows of A, $A(i,\cdot)$, and also $\mathrm{SQ}(a)$ for a the vector of row norms (i.e., $a(i) \coloneqq \|A(i,\cdot)\|$). We have *$\phi$-oversampling and query access* to a vector v, denoted $\mathrm{SQ}_\phi(v)$, if (1) we can query for entries of v and (2) we have sampling and query access to an "entry-wise upper bound" vector $\tilde{v}$ satisfying $\|\tilde{v}\|^2 = \phi\|v\|^2$ and $|\tilde{v}(i)| \geq |v(i)|$ for all indices i; the definition for a matrix is analogous (Definition 3.4).

The parameter ϕ should be seen as a form of overhead that comes out in the runtime of algorithms: through rejection sampling, $\mathrm{SQ}_\phi(v)$ can do approximate versions of all the queries of $\mathrm{SQ}(v)$ with a factor ϕ of overhead (Lemma 3.5). In this paper, we most often think of ϕ as being independent of input size.

To motivate this definition, we make the following observations about this input model. First, as far as we know, if input data is given *classically*,[1] classical algorithms in the sampling and query model can be run whenever the corresponding algorithms in the quantum model can (Remark 3.11). For example, if input is loaded in the QRAM data structure, as commonly assumed in QML in order to satisfy state preparation assumptions [Pra14, CHI+18], then we have log-time sampling and query access to it. So, a fast classical algorithm for a problem in this classical model implies lack of quantum speedup for the problem, at least in the usual settings explored in the QML literature. In particular, a polynomial-time classical algorithm in this model implies lack of exponential quantum speedup. Second, oversampling and query access has many similarities to the notion of quantum block-encodings in quantum singular value transformation [GSLW19]. The commonly used data-structures that enable oversampling and query access to a matrix A also enable implementing an efficient quantum circuit whose unitary is a block-encoding of A. Further, in both input models one can perform efficient matrix arithmetic.

Matrix arithmetic. The thrust of our main results is to demonstrate that *oversampling and query access is approximately closed under arithmetic operations*. We argue that the essential power of quantum-inspired algorithms lies in their ability to leverage sampling and query access of the input matrices to provide oversampling and query access to complex arithmetic expressions as output (possibly with some approximation error), without paying the (at least) linear time necessary to compute such expressions in conventional ways. While the proven closure properties are important from a complexity theoretic point of view, some of them don't explicitly come into play in the demonstrated applications of our framework for dequantizing QML, since they often come with undesirable polynomial overhead. This is in contrast with quantum block-encodings, which generally compose with minimal overhead.

[1] This assumption is important. When input data is quantum (say, it is coming directly from a quantum system), a classical computer has little hope of performing linear algebra on it efficiently, see for example [ACQ22, HKP21].

We now list the closure properties that we show, along with the corresponding closure properties proven for block-encodings in [GSLW19]. For all of these, the query time for access to the output is just polynomial in the query times for access to the input, so in particular, these procedures run in time independent of input dimension. More specifically, we will compare what we call (sub)normalization "overhead" between the two, which is the value ϕ in the classical setting and what [GSLW19] denotes as α in the quantum setting. The two quantities are analogous, and roughly correspond to overheads in rejection sampling and post-selection, inducing a multiplicative factor in sampling times for both.

- Given access to a constant number of vectors $v_1, \ldots, v_\tau$, we have access to linear combinations $\sum_{t=1}^{\tau} \lambda_t v_t$, and analogously with linear combinations of matrices (Lemmas 3.6 and 3.9). This is a classical analogue to the "linear combinations of unitaries" technique for block-encodings [GSLW19, Lemma 52]. In the quantum setting, there is less overhead,[2] allowing for efficient block encodings for linear combinations of arbitrarily many matrices in certain settings.

- Given access to two matrices A, B with Frobenius norm at most one, we have access to a matrix Z ε-close to the product $A^\dagger B$ in Frobenius norm (Lemma 4.6 and Remark 4.7). In the quantum setting, closure of block-encodings under products is almost immediate [GSLW19, Lemma 53] and is not approximate. In both cases the individual input overheads of A and B are multiplied. With the same overheads one can also form Kronecker products $A \otimes B$ exactly—this is immediate both in the classical and quantum case [CVB20]. In particular, given access to two vectors u and v, we have access to their outer product $uv^\dagger$ (Lemma 3.8).

- Given access to a matrix A with Frobenius norm at most one and a Lipschitz function f, we have access to a matrix Z ε-close to $f(A^\dagger A)$ in Frobenius norm (Theorem 5.1)[3]. In the quantum setting, block-encodings are closed under even and odd polynomial singular value transformations [GSLW19, Lemmas 8, 10] without approximation, provided the polynomial is low-degree and bounded. This block-encoding closure property can be viewed as a corollary of the above two properties, but can also be achieved directly with some more efficient technical machinery.

  An even polynomial singular value transformation of A is precisely $f(A^\dagger A)$ for f a low-degree polynomial (which means f is Lipschitz), and odd polynomials can be decomposed into a product of an even polynomial with A, so our closure property is as strong as the quantum one. The full details are derived in Section 6.1.

To summarize, every arithmetic operation of matrices with block-encodings in [GSLW19] (with the possible exception of long linear combinations) can be mimicked by matrices with oversampling and query access, up to Frobenius norm error, provided that an input matrix in a block-encoding corresponds to having[4] $\mathrm{SQ}(A)$ and $\mathrm{SQ}(A^\dagger)$. The "linear combinations of vectors" and "outer products" classical closure properties have been used in prior work [Tan19, CGL+20]. However, without our new definition of oversampling and query access, it was not clear that these algorithms could be chained indefinitely as we show with these closure properties.

[2] In fact, already $\sqrt{\phi} \geq \alpha / \left\| \sum_{t=1}^{\tau} \lambda_t v_t \right\|$ which can be seen using that the root mean-square is at least the average.

[3] For a Hermitian matrix H and a function $f \colon \mathbb{R} \mapsto \mathbb{C}$, $f(H)$ denotes applying f to the eigenvalues of H. That is, $f(H) \coloneqq \sum_{i=1}^{n} f(\lambda_i) v_i v_i^\dagger$, for λ_i and v_i the eigenvalues and eigenvectors of H.

[4] We take some care here to distinguish whether we have oversampling and query access to A or $A^\dagger$. We don't need to: we show that having either one of them implies having the other, up to approximation (Remark 6.4). However, the accesses assumed in our closure properties are in some sense the most natural choices and require the least overhead.

Implications for quantum singular value transformation. Our results give compelling evidence that there is indeed no exponential speedup for QRAM-based QSVT, and show that oversampling and query access can be thought of as a classical analogue to block-encodings in the bounded Frobenius norm regime. Nevertheless, we do not rule out the possibility for large polynomial speedups, as the classical runtimes tend to have impractically large polynomial exponents. To elaborate more on this connection, we now recall the QSVT framework in more detail.

The QSVT framework of Gilyén, Su, Low, and Wiebe [GSLW19] assumes that the input matrix A is given by a *block-encoding*, which is a quantum circuit implementing a unitary transformation whose top-left block contains (up to scaling) A itself [LC17]. Given a block-encoding of A, one can apply it to a quantum state or form block-encodings of other expressions using the closure properties mentioned above. One can get a block-encoding of an input matrix A through various methods. If A is s-sparse with efficiently computable elements and $\|A\| \leq 1$, then one can directly get a block-encoding of A/s [GSLW19, Lemma 48]. If A is in the QRAM data structure (used for efficient state preparation for QML algorithms [Pra14]), one can directly get a block-encoding of $A/\|A\|_{\mathrm{F}}$ [GSLW19, Lemma 50]. We will use the term *QRAM-based QSVT* to refer to the family of quantum algorithms possible in the QSVT framework when all input matrices & vectors are given in the QRAM data structure.

The normalization in QRAM-based QSVT means that it has an implicit dependence on the Frobenius norm $\|A\|_{\mathrm{F}}$. Since $\|A\|_{\mathrm{F}}$ is also the key parameter in the complexity of our corresponding classical algorithms, this suggests that QRAM-based QSVT does not give inherent exponential quantum speedups (though, if input preparation/output analysis protocols have no classical analogues, they can act as a subroutine in an algorithm that does give an exponential quantum speedup). Our closure results confirm this: if input matrices and vectors are given in QRAM data structure, then on one (quantum) hand we can construct block-encodings of these matrices normalized to have Frobenius norm one and on the other (classical) hand we have sampling and query access to the input. The conclusion is that, up to some controllable approximation error, an algorithm using the block-encoding framework has a classical analogue in the oversampling and query access model. The classical algorithm's runtime is only polynomially slower than the corresponding quantum algorithm, except in the ε parameter.[5] One can argue similarly that there should be no exponential speedup for QSVT for block-encodings derived from (purifications of) density operators [GSLW19, Lemma 45] that come from some well-structured classical data (see Section 6.1). This stands in contrast to, for example, block-encodings that come from sparsity assumptions [GSLW19, Lemma 48], where the matrix in the block-encoding can have Frobenius norm as large as $\sqrt{sn}$ (where we take A to be $n \times n$), and so the classical techniques cannot be applied without incurring dependence on n in the runtime.

1.3 Technical overview

We now illustrate the flavor of the algorithmic ideas underlying our main results, by showing why the "oversampling" input model is closed under approximate matrix products. Suppose we are given sampling and query access to two matrices $A \in \mathbb{C}^{m \times n}$ and $B \in \mathbb{C}^{m \times p}$, and desire (over)sampling and query access to $A^\dagger B$. $A^\dagger B$ is a sum of outer products of rows of A with rows of B (that is, $A^\dagger B = \sum_{i=1}^{m} A(i,\cdot)^\dagger B(i,\cdot)$), so a natural idea is to use the outer product closure property to get access to each outer product individually, and then use the linear combination closure property to get access to their sum, which is $A^\dagger B$ as desired. However, there are m terms in the sum, which is

[5] The QML algorithms we discuss generally only incur $\mathrm{polylog}(\frac{1}{\varepsilon})$ terms, but need to eventually pay $\mathrm{poly}(1/\varepsilon)$ to extract information from output quantum states. So, we believe this exponential speedup is artificial. See the open questions section for more discussion of this error parameter.

too large: we can't even compute entries of $A^\dagger B$ in time independent of m. So, we use sampling to approximate this sum of m terms by a linear combination over far fewer terms, allowing us to get access to Z for $Z \approx A^\dagger B$. This type of matrix product approximation is well-known in the classical literature [DKM06]. Given $\mathrm{SQ}(A)$, we can pull samples $i_1, \ldots, i_s$ according to the row norms of A, a distribution we will denote p (so $p(i) = \|A(i,\cdot)\|^2/\|A\|_{\mathrm{F}}^2$). Consider $Z := \frac{1}{s}\sum_{k=1}^{s} \frac{1}{p(i_k)} A(i_k,\cdot)^\dagger B(i_k,\cdot)$. Z is an unbiased estimator of $A^\dagger B$: $\mathbb{E}[Z] = \frac{1}{s}\sum_{k=1}^{s}\sum_{\ell=1}^{m} p(\ell)\frac{A(\ell,\cdot)^\dagger B(\ell,\cdot)}{p(\ell)} = \sum_{\ell=1}^{m} A(\ell,\cdot)^\dagger B(\ell,\cdot) = A^\dagger B$. Further, the variance of this estimator is small. In the following computation, we consider $s = 1$, because the variance for general s decreases as $1/s$.

$$\begin{aligned}\mathbb{E}[\|A^\dagger B - Z\|_{\mathrm{F}}^2] &\le \sum_{i,j} \mathbb{E}[|Z(i,j)|^2] = \sum_{i,j}\sum_{\ell} p(\ell)\frac{1}{p(\ell)^2}|A(\ell,i)|^2|B(\ell,j)|^2 \\ &= \sum_{\ell}\frac{1}{p(\ell)}\|A(\ell,\cdot)\|^2\|B(\ell,\cdot)\|^2 = \sum_{\ell}\|A\|_{\mathrm{F}}^2\|B(\ell,\cdot)\|^2 = \|A\|_{\mathrm{F}}^2\|B\|_{\mathrm{F}}^2.\end{aligned}$$

By Chebyshev's inequality, we can choose $s = \mathcal{O}\left(\frac{1}{\varepsilon^2}\right)$ to get that $\|Z - A^\dagger B\|_{\mathrm{F}} < \varepsilon\|A\|_{\mathrm{F}}\|B\|_{\mathrm{F}}$ with probability 0.99. Since Z is a linear combination of s outer products, this gives us oversampling and query access to Z as desired. In our applications we would keep Z as an outer product $A'^\dagger B'$ for convenience. We have just sketched the proof of our key lemma: an approximate matrix product protocol.

Key lemma [DKM06] (informal version of Lemma 4.6). *Suppose we are given* $\mathrm{SQ}(X) \in \mathbb{C}^{m\times n}$ *and* $\mathrm{SQ}(Y) \in \mathbb{C}^{m\times p}$. *Then we can find normalized submatrices of* X *and* Y, $X' \in \mathbb{C}^{s\times n}$ *and* $Y' \in \mathbb{C}^{s\times p}$, *in* $\mathcal{O}(s)$ *time for* $s = \Theta(\frac{1}{\varepsilon^2}\log\frac{1}{\delta})$, *such that*

$$\Pr\left[\|X'^\dagger Y' - X^\dagger Y\|_{\mathrm{F}} \le \varepsilon\|X\|_{\mathrm{F}}\|Y\|_{\mathrm{F}}\right] > 1 - \delta.$$

We subsequently have $\mathcal{O}(s)$*-time* $\mathrm{SQ}(X'), \mathrm{SQ}(X'^\dagger), \mathrm{SQ}(Y'), \mathrm{SQ}(Y'^\dagger)$.

Prior quantum-inspired algorithms [Tan19, Tan21, CLW18, CLLW20] indirectly used this lemma by using [FKV04], which finds a low-rank approximation to the input matrix in the form of an approximate low-rank SVD and relies heavily on this lemma in the analysis.

One of our main results, mentioned earlier as our singular value transformation closure property, is that, given $\mathrm{SQ}(A) \in \mathbb{C}^{m\times n}$, in time independent of m and n, we can access an approximation of $f(A^\dagger A)$ for a Lipschitz-function f that, without loss of generality, satisfies $f(0) = 0$ (Theorem 5.1). One could use [FKV04] to give a classical algorithm for SVT, but a more efficient approach is to directly apply the key lemma twice to get an approximate decomposition of $f(A^\dagger A)$:

$$\begin{aligned} f(A^\dagger A) &\approx f(R^\dagger R) && \text{by key lemma, with } R \in \mathbb{C}^{r\times n} \text{ normalized rows of } A \\ &= R^\dagger \bar{f}(RR^\dagger)R && \text{by computation, where } \bar{f}(x) := f(x)/x \\ &\approx R^\dagger \bar{f}(CC^\dagger)R && \text{by key lemma, with } C \in \mathbb{C}^{r\times c} \text{ normalized columns of } R \end{aligned}$$

We call $R^\dagger\bar{f}(CC^\dagger)R$ an *RUR decomposition* because $R \in \mathbb{C}^{r\times n}$ is a subset of rows of the input matrix and U is a matrix with size independent of input dimension (R corresponds to the 'R' of the RUR decomposition, and $\bar{f}(CC^\dagger) \in \mathbb{C}^{r\times r}$ corresponds to the 'U'). In other words, an RUR decomposition expresses a desired matrix as a linear combination of r^2 outer products of rows of the input matrix ($\sum_{i,j}[\bar{f}(CC^\dagger)](i,j)R(i,\cdot)^\dagger R(j,\cdot)$, for example).[6] We want our output in the form

[6] This is the relevant variant of the notion of a *CUR decomposition* from the randomized numerical linear algebra and theoretical computer science communities [DMM08].

of an RUR decomposition, since we can describe such a decomposition implicitly just as a list of row indices and some additional coefficients, which avoids picking up a dependence on m or n in our runtimes. Further, having $\mathrm{SQ}(A)$ gives us $\mathrm{SQ}_\phi(R^\dagger U R)$ via closure properties, enabling efficient access to matrix-vector expressions like $R^\dagger U R b$.

More general results follow as corollaries of our main result on even SVT (Theorems 7.1 and 7.2). However, using only our main theorem about even SVT, we can directly recover most existing quantum-inspired machine learning algorithms without using these general results, yielding faster dequantization for QML algorithms. We now outline our results recovering such applications.

1.4 Applications: dequantizing QML & more

We use the results above to recover existing quantum-inspired algorithms for recommendation systems [Tan19], principal component analysis [Tan21], supervised clustering [Tan21], support vector machines [DBH21], low-rank matrix inversion [CGL+20], and semidefinite program solving [CLLW20]. We also propose new quantum-inspired algorithms for low-rank Hamiltonian simulation and discriminant analysis (dequantizing the quantum algorithm of Cong & Duan [CD16]).

Fig. 1 has a summary of our results, along with a comparison of runtimes to the corresponding quantum algorithms and prior quantum-inspired work, where it exists. All our results match or improve on prior dequantized algorithms apart from that for matrix inversion, where prior work gives an incomparable runtime that only holds for strictly low-rank matrices of rank k. Our results for matrix inversion and semidefinite program solving solve the problem in greater generality than prior work, without the restriction that the input matrices are strictly rank-k.[7]

We do *not* claim any meaningful breakthroughs for these problems in the classical literature: the problems that these QML algorithms solve differ substantially from their usual classical counterparts. For example, the quantum recommendation systems algorithm of Kerenidis and Prakash [KP17] performs sampling from a low-rank approximation of the input instead of low-rank matrix completion, which is the typical formalization of the recommendation systems problem [Tan19]. Evaluating these quantum algorithms' justifications for their versions of problems is outside the scope of this work: instead, we argue that these algorithms would likely not give exponential speedups when implemented, regardless of whether such implementations would be useful. The goal of our framework is to demonstrate what can be done classically and establish a classical frontier for quantum algorithms to push past.

The proofs for these dequantization results follow the same general structure: consider the quantum algorithm and formulate the problem that this algorithm solves, and in particular, the linear algebra expression that the quantum algorithm computes. From there, repeatedly use the SVT result and key lemma to approximate this expression by something like an RUR decomposition. Finally, use closure properties to gain oversampling and query access to that output decomposition. This procedure is relatively straightfoward and flexible. Also, unlike previous work [CGL+20, CLLW20], our results need not assume that the input is strictly low-rank. Instead, following [Tan19, GSLW19], our algorithms work on close-to-low-rank matrices by doing SVTs that smoothly threshold to effectively only operate on large-enough singular values.

1.5 Related work

Quantum-inspired algorithms. Our approach and analysis is much simpler than that of Frieze, Kannan, and Vempala [FKV04], while it also gives improved results in our applications, and has several other advantages. For example, the reduction to [FKV04] first given by Tang to get an

[7] For semidefinite program solving, $\|A^{(\cdot)}\|_{\mathrm{F}} \leq \sqrt{k}$, which makes the runtimes comparable.

	quantum algorithm	prior work	this work
simple QSVT [GSLW19], §6.1	$\frac{d\|A\|_{\mathrm{F}}\|b\|}{\|p^{(\mathrm{QV})}(A)b\|}$		$\frac{d^{22}\|A\|_{\mathrm{F}}^6\|b\|^6}{\varepsilon^6\|p^{(\mathrm{QV})}(A)b\|^6}$
recommendation systems [CGJ19], [Tan19], §6.2	$\frac{\|A\|_{\mathrm{F}}}{\sigma}$	$\frac{\|A\|_{\mathrm{F}}^{24}}{\sigma^{24}\varepsilon^{12}}$	$\frac{\|A\|_{\mathrm{F}}^6\|A\|^{10}}{\sigma^{16}\varepsilon^6}$
supervised clustering [LMR13], [Tan21], §6.3	$\frac{\|M\|_{\mathrm{F}}^2\|w\|^2}{\varepsilon}$ (♣)	$\frac{\|M\|_{\mathrm{F}}^4\|w\|^4}{\varepsilon^2}$	$\frac{\|M\|_{\mathrm{F}}^4\|w\|^4}{\varepsilon^2}$
principal component analysis [CGJ19], [Tan21], §6.4	$\frac{\|X\|_{\mathrm{F}}\|X\|}{\lambda_k\varepsilon}$	$\frac{\|X\|_{\mathrm{F}}^{36}}{\|X\|^{12}\lambda_k^{12}\eta^6\varepsilon^{12}}$	$\frac{\|X\|_{\mathrm{F}}^6}{\|X\|^2\lambda_k^2\eta^6\varepsilon^6}$
matrix inversion [GSLW19], [GLT18], §6.5	$\frac{\|A\|_{\mathrm{F}}}{\sigma}$	$\frac{\|A\|_{\mathrm{F}}^6\|A\|^{16}k^6}{\sigma^{22}\varepsilon^6}$ (♢)	$\frac{\|A\|_{\mathrm{F}}^6\|A\|^{22}}{\sigma^{28}\varepsilon^6}$
support vector machines [RML14], [DBH21], §6.6	$\frac{1}{\lambda^3\varepsilon^3}$ (♢)	$\mathrm{poly}\left(\frac{1}{\lambda},\frac{1}{\varepsilon}\right)$ (♣)	$\frac{1}{\lambda^{28}\varepsilon^6}$
Hamiltonian simulation [GSLW19], §6.7	$\|H\|_{\mathrm{F}}$		$\frac{\|H\|_{\mathrm{F}}^6\|H\|^{16}}{\max(1,\sigma^{16})\varepsilon^6}$
semidefinite program solving [vAG19], [CLLW20], §6.8	$\frac{\|A^{(\cdot)}\|_{\mathrm{F}}^7}{\varepsilon^{7.5}}+\frac{\sqrt{m}\|A^{(\cdot)}\|_{\mathrm{F}}^2}{\varepsilon^4}$	$\frac{mk^{57}}{\varepsilon^{92}}$ (♢)	$\frac{\|A^{(\cdot)}\|_{\mathrm{F}}^{22}}{\varepsilon^{46}}+\frac{m\|A^{(\cdot)}\|_{\mathrm{F}}^{14}}{\varepsilon^{28}}$
discriminant analysis [CD16], §6.9	$\frac{\|B\|_{\mathrm{F}}^7}{\varepsilon^3\sigma^7}+\frac{\|W\|_{\mathrm{F}}^7}{\varepsilon^3\sigma^7}$ (♢)		$\frac{\|B\|_{\mathrm{F}}^6\|B\|^4}{\varepsilon^6\sigma^{10}}+\frac{\|W\|_{\mathrm{F}}^6\|W\|^{10}}{\varepsilon^6\sigma^{16}}$

Figure 1: The time complexity for our algorithms, the quantum algorithms they are based on, and prior quantum-inspired algorithms (where they exist). We assume our sampling and query accesses to the input takes $\mathcal{O}(1)$ time. There are data structures that can support such queries (Remark 3.11), and if the input is in QRAM, the runtime only increases by at most a factor of log of input size.

We list the runtime of the algorithm, not including the time it takes to access the output (denoted with $\widetilde{\mathbf{sq}}$). The runtimes as listed ignore polylog terms, particularly those in error parameters (ε and δ) and dimension parameters (m and n). The matrices and vectors referenced in these runtimes are always the input, σ refers to a singular value threshold of the input matrices, λ refers to an eigenvalue threshold (which can be thought of here as σ^2), and $\eta > \varepsilon$ is a (dimensionless) gap parameter.

(♣) indicates that the error analyses of the corresponding results are incomplete; we list the runtime they achieve for completeness.

(♢) indicates that the corresponding results only hold in the restricted setting where the input matrices are strictly rank k. For the quantum algorithms with this tag, they allow for general matrices, but only have an informal error analysis arguing that singular values outside the range considered don't affect the final result.

SVT-based low-rank approximation bound from the standard notion of low-rank approximation [Tan19, Theorem 4.7] induces a quadratic loss in precision, which appears to be only an artifact of the analysis. Also, [FKV04] gives Frobenius norm error bounds, though for applications we often only need spectral norm bounds; our main theorem can get improved runtimes by taking advantage of the weaker spectral norm bounds. Finally, we take a reduced number of rows compared to columns, whereas [FKV04] approximates the input by taking the same number of rows and columns.

Randomized numerical linear algebra. All of the results presented here are more or less randomized linear algebra algorithms [Mah11, Woo14]. The kind of sampling we get from sampling and query access is called *importance sampling* or *length-square sampling* in that body of work: see the survey by Kannan and Vempala [KV17] for more on importance sampling. Importance sampling, and specifically, its approximate matrix product property, is the core primitive of this work. In addition to the low-rank approximation algorithms [FKV04] used in the quantum-inspired literature, others have used importance sampling for, e.g., orthogonal tensor decomposition [DM07, MMD08, SWZ16] (generalizing low-rank approximation [FKV04]) and support vector machines [HKS11].

The fundamental difference between quantum-inspired algorithms and traditional sketching algorithms is that we assume "we can perform quantum measurements" of states corresponding to input in time independent of input dimension (that is, we have efficient sampling and query access to input), and in exchange want algorithms that run in time independent of dimension and provide only (over)sampling and query access to the output. This quantum-inspired model is weaker than the standard sketching algorithm model (Remark 3.11): an algorithm taking T time in the quantum-inspired model for an input matrix A can be converted to a standard algorithm that runs in time $\mathcal{O}(\text{nnz}(A) + T)$, where $\text{nnz}(A)$ is the number of nonzero entries of A. So, we can also think about an $\mathcal{O}(T)$-time quantum-inspired algorithm as an $\mathcal{O}(\text{nnz}(A) + T)$-time sketching algorithm, where the $\text{nnz}(A)$ portion of the runtime can *only* be used to facilitate importance sampling.[8] This restriction makes for algorithms that may perform worse in generic sketching settings, but work in more settings, and so demonstrate lack of exponential quantum speedup for a wider range of problems.

A natural question is whether more modern sketching techniques can be used in our model. After all, importance sampling is only one of many sketching techniques studied in the large literature on sketching algorithms. Notably, though, *other types of sketches seem to fail in the input regimes where quantum machine learning succeeds*: assuming sampling and query access to input, importance sampling takes time independent of dimension, whereas other randomized linear algebra methods such as Count-Sketch and Johnson-Lindenstrauss still take time linear in input-sparsity.

Subsequent work by Chepurko, Clarkson, Horesh, Lin, and Woodruff [CCH+22] notes that importance sampling oversamples leverage score sampling, so usual analyses for leverage score sampling also hold for importance sampling, up to some small overhead. It is reasonable to suspect that this connection could lead to significant improvements over the results presented here. However, exploiting this connection for improved runtimes seems nontrivial, since most approaches using leverage score sampling requires performing $O(\text{nnz}(A))$-time pre-processing operations, even if one has $\text{SQ}(A)$. As a simple example, for low-rank approximation of an input matrix A, if we wish to adapt the algorithm of Clarkson and Woodruff [CW17, Woo14], importance sampling of A can replace CountSketch for one of the sketches [Woo14, Lemma 4.2], but importance sampling of SA cannot replace the other [Woo14, Theorem 4.3]. Versions of importance sampling may work for the second sketch (for example, sampling from a low-rank approximation of SA), but we are aware

[8] The same holds for quantum algorithms using the QRAM data structure input model: the data structure itself can be built during an $\mathcal{O}(\text{nnz}(A))$-time (classical) preprocessing phase.

of none that can be obtained easily from $\mathrm{SQ}(A)$. This may be a manifestation of the difficulty of achieving relative-error estimates to quantities like leverage scores and residuals in this model.

An alternative approach is to use a projection-cost preserving sketch (PCP) like *ridge* leverage score sampling to sketch A on both sides [CMM17]. The importance sampling from $\mathrm{SQ}(A)$ $\lambda/2$-oversamples ridge leverage score sampling, where $\lambda \coloneqq \|A\|_{\mathrm{F}}^2/\|A - A_k\|_{\mathrm{F}}^2 \geq \|A\|_{\mathrm{F}}^2/\sigma_{k+1}^2$, so using importance sampling in place of ridge leverage score sampling can give algorithms. This is how [CCH+22] gets their algorithms for low-rank sampling (recommendation systems) and quantum-inspired linear regression. This does appear to be a promising approach, with possibility to extend to dequantizing all of QSVT, but to the authors' knowledge, it is still an open question how to improve the algorithms presented in this work, with the exception of [CCH+22] improving over our recommendation systems algorithm. Their use of PCPs significantly improves the runtime for this low-rank sampling task down below $\frac{\|A\|_{\mathrm{F}}^6}{\sigma^6\varepsilon^6}$, but getting a similarly good runtime for all functions seems nontrivial. For example, their linear regression algorithm requires that the input matrix is strictly rank-k (or is regularized).

The quantum-like closure properties of importance sampling shown here may be useful in the context of classical sketching algorithms. This insight unlocks surprising power in importance sampling. For example, it reveals that Frieze, Kannan, and Vempala's low-rank approximation algorithm [FKV04], which, as stated, requires $\mathcal{O}(kmn)$ time to output the desired matrix, actually can produce useful results (samples and entries) in time independent of input dimension. To use the language in [FKV04], if Assumptions 1 and 2 hold for the input matrix, they also hold for the output matrix!

Classical algorithms for quantum problems. We are aware of two important prior results from before Tang's first paper [Tan19] that connect quantum algorithms to randomized numerical linear algebra. The first is Van den Nest's work on using probabilistic methods for quantum simulation [VdN11], which defines a notion of "computationally tractable" (CT) state equivalent to our notion of sampling and query access and then uses it to simulate restricted classes of quantum circuits. We share some essential ideas with this work, such as the simple sampling lemmas Lemmas 3.6 and 4.12, but also differ greatly since we focus on low-rank matrices relevant for QML, whereas [VdN11] focuses on simulating potentially large quantum circuits that correspond to high-rank matrices. The second is a paper by Rudi, Wossnig, Ciliberto, Rocchetto, Pontil, and Severini [RWC+20] that uses the Nyström method to simulate a sparse Hamiltonian H on a sparse input state in time poly-logarithmic in dimension and polynomial in $\|H\|_{\mathrm{F}}$, assuming sampling and query access to H. Our Hamiltonian simulation results do not require a sparsity assumption and still achieve a dimension-independent runtime, but get slightly larger exponents in exchange.

Practical implementation. A work by Arrazola, Delgado, Bardhan, and Lloyd [ADBL20] implements and benchmarks quantum-inspired algorithms for regression and recommendation systems. The aforementioned paper of Chepurko, Clarkson, Horesh, Lin, and Woodruff [CCH+22] does the same for the quantum-inspired algorithms they introduce. The former work makes various conclusions, including that the ε^2 scaling in the number of rows/columns taken in our recommendation systems algorithm is inherent and that the quantum-inspired algorithms performed slower and worse than direct computation for practical datasets. The latter work finds that their algorithms perform faster than direct algorithms, with an accompanying increase in error comparable to that of other sketching algorithms [DKW18]. This improvement appears to come from both a better-performing implementation as well as an algorithm with better asymptotic runtime. Nevertheless, it is difficult to draw definitive conclusions about the practicality of quantum-inspired algorithms as a whole

from these experimental results. Since quantum-inspired algorithms are a restricted, weaker form of computation than classical randomized numerical linear algebra algorithms (see the comparison made above), it seems possible that they perform worse than standard sketching algorithms, despite seemingly having exponentially improved runtime in theory.

Modern sketching algorithms use similar techniques to quantum-inspired algorithms, but are more natural to run on a classical computer and are likely to be faster. For example, Dahiya, Konomis, and Woodruff [DKW18] conducted an empirical study of sketching algorithms for low-rank approximation on both synthetic datasets and the movielens dataset, reporting that their implementation "finds a solution with cost at most 10 times the optimal one . . . but does so 10 times faster." Sketching algorithms like those in [DKW18] may become a relevant point of reference for benchmarking quantum linear algebra, when the implementation of these quantum algorithms on actual quantum hardware becomes possible. In a sense, our work shows using asymptotic runtime bounds that in many scenarios sketching and sampling techniques give similar computational power to quantum linear algebra, which is a counterintuitive point since the former typically leads to linear runtimes and the latter leads to poly-logarithmic ones.

Quantum machine learning. Our work has major implications for the landscape of quantum machine learning. Since we have presented many dequantized versions of QML algorithms, the question remains of what QML algorithms don't have such versions. In other words, what algorithms still have the potential to give exponential speedups?

There are two general paradigms for employing quantum linear algebra techniques in quantum machine learning: the low-rank approach and the high-rank approach. In both, we need to turn classical input data vectors to quantum states (via what's called an amplitude encoding), perform linear algebra operations on those vectors, and extract information about the output via sampling [Aar15]. Since preparing generic quantum states require a number of quantum gates that is proportional to the dimension of the vectors, we need some state preparation assumptions (like having QRAM with an appropriate data structure, etc., cf. Remark 3.11) in order to achieve sublinear runtimes. The main difference between the two paradigms is how the input matrices are given: in the low-rank approach, they are also given in QRAM, and so must be rescaled to have Frobenius norm one. QSVT with block-encodings coming from the QRAM data structure[9] or density operators is an example framework in this vein. The restrictions of this block-encoding method means that the rank needs to be small, but assuming the hardware needed for QRAM can be realized, this is still a flexible setting, one that QML researchers find interesting. In the high-rank approach, which is used in the HHL algorithm [HHL09] and its derivatives, the matrix needs to be represented by a concise quantum circuit and have a small (poly-logarithmic in input dimension) condition number in order to gain an exponential speedup over classical algorithms. This doesn't happen in typical datasets. The collection of these demanding requirements hamstrings most attempts to find applications of HHL [HHL09] with the potential for practical super-polynomial speedups.

Our results give evidence for the lack of exponential speedup for the former, low-rank approach. It is important to note however, that our results do not rule out the possibility of large polynomial quantum speedups. In order to assess the potential usefulness of QML algorithms in this regime it is important to improve on classical upper and lower bounds for these problems, which we leave as an open question. On the other hand, high-rank block-encodings, such as those coming from sparsity

[9]The QRAM data structure used here has alternatives which in some sense vary the "norm" in which one stores the input matrix [KP20, Theorem IV.4], [CGJ19, Lemma 25]. We do not expect that these alternatives would give sampling and query access to the matrix they store or could be otherwise dequantized, since these datastructures generalize and strengthen the sparse-access input model, which is known to be BQP-complete [HHL09].

assumptions in the original HHL algorithm [HHL09], remain impervious to our techniques. This suggests that the most promising way to get exponential quantum speedups for QML algorithms is by assuming sparse matrices as input, or utilizing other efficiently implementable high-rank quantum operation such as the Quantum Fourier Transform.

Works from Zhao, Fitzsimons, and Fitzsimons on Gaussian process regression [ZFF19]; from Lloyd, Garnerone, and Zanardi on topological data analysis [LGZ16, GCD20]; and from Yamasaki, Subramanian, Sonoda, and Koashi [YSSK20] on learning random features attempt to address these issues to get a super-polynomial quantum speedup. Though these works avoid the dequantization barrier to large quantum speedups, it remains to be seen how broad will be their impact on QML and whether these speedups manifest for data seen in practice.

Related independent work. Independently from our work, Jethwani, Le Gall, and Singh simultaneously derived similar results [JLGS20]. They implicitly derive a version of our even SVT result, and use it to achieve generic SVT (approximate $\mathrm{SQ}(b^\dagger f^{(\mathrm{SV})}(A))$ for a vector b) by writing $f^{(\mathrm{SV})}(A) = Ag(A^\dagger A)$ for $g(x) = f(\sqrt{x})/\sqrt{x}$ and then using sampling subroutines to get the solution from the resulting expression $b^\dagger AR^\dagger UR$. It is difficult to directly compare the main SVT results, because the parameters that appear in their runtime bounds are somewhat non-standard, but one can see that for typical choices of f, their results require a strictly low-rank A. In comparison our results apply to general A, and we also demonstrate how to apply them to (re)derive dequantized algorithms.

1.6 Open questions

Our framework recovers recent dequantization results, and we hope that it will be used for dequantizing more quantum algorithms. In the meantime, our work leaves several natural open questions:

(a) Is there an approach to QML that does not go through HHL (whose demanding assumptions make exponential speedups difficult to demonstrate even in theory) or a low-rank assumption (which, as we demonstrate, makes the tasks "easy" for classical computers) and yields a provable superpolynomial speedup for a practically relevant ML problem?

(b) Our algorithms still have significant slowdown as compared to their quantum counterparts. Can we shave condition number factors to get runtimes of the form $\widetilde{\mathcal{O}}\left(\frac{\|A\|_{\mathrm{F}}^6}{\sigma^6\varepsilon^6}\log^3\frac{1}{\delta}\right)$ (for the recommendation systems application, for instance), without introducing additional assumptions? Can we get even better runtimes by somehow avoiding SVD computation?

(c) Do the matrix arithmetic closure properties we showed for ℓ_2-norm importance sampling hold for other kinds of sampling and sketching distributions, like leverage score or ℓ_p-norm sampling?

(d) In the quantum setting, linear algebra algorithms [GSLW19] can achieve logarithmic dependence on the precision ε. Can classical algorithms also achieve such exponentially improved dependence, when the goal is restricted to sampling from the output (i.e., without the requirement to query elements of the output)? If not, is there a mildly stronger classical model that can achieve this? Can one prove that this exponential advantage for sampling problems cannot be conferred to estimation/decision problems?

1.7 Organization

The paper proceeds as follows. Section 3 introduces the notion of (over)sampling and query access and some of its closure properties. Section 4 gives the fundamental idea of using sampling and query access to sketch matrices used for the approximation results in Section 4.1 and singular value transformation results in Section 5. These results form the framework that is used to dequantize QSVT in Section 6.1 and recover all the quantum-inspired results in Section 6. These applications of our framework contain various tricks and patterns that we consider to be "best practice" for coercing problems into our framework, since they have given us the best complexities and generality. More general results of SVT are shown in Section 7.

2 Preliminaries

To begin with, we define notation to be used throughout this paper. For $n \in \mathbb{N}$, $[n] \coloneqq \{1, \ldots, n\}$. For $z \in \mathbb{C}$, its absolute value is $|z| = \sqrt{z^* z}$, where z^* is the complex conjugate of z. $f \lesssim g$ denotes the ordering $f = \mathcal{O}(g)$ (and respectively for $\gtrsim$ and $\eqsim$). $\widetilde{\mathcal{O}}(g)$ is shorthand for $\mathcal{O}(g \operatorname{poly}(\log g))$. log refers to the natural logarithm. Finally, we assume that arithmetic operations (e.g., addition and multiplication of real numbers) and function evaluation oracles (computing $f(x)$ from x) take unit time, and that queries to oracles (like the queries to input discussed in Section 3) are at least unit time cost.

2.1 Linear algebra

In this paper, we consider complex matrices $A \in \mathbb{C}^{m \times n}$ for $m, n \in \mathbb{N}$. For $i \in [m], j \in [n]$, we let $A(i, \cdot)$ denote the i-th row of A, $A(\cdot, j)$ denote the j-th column of A, and $A(i, j)$ denote the (i, j)-th element of A. $(A \mid B)$ denotes the concatenation of matrices A and B and $\operatorname{vec}(A) \in \mathbb{C}^{mn}$ denotes the vector formed by concatenating the rows of A. For vectors $v \in \mathbb{C}^n$, $\|v\|$ denotes standard Euclidean norm (so $\|v\| \coloneqq (\sum_{i=1}^n |v(i)|^2)^{1/2}$). For a matrix $A \in \mathbb{C}^{m \times n}$, the *Frobenius norm* of A is $\|A\|_{\mathrm{F}} \coloneqq \|\operatorname{vec}(A)\| = (\sum_{i=1}^m \sum_{j=1}^n |A(i,j)|^2)^{1/2}$ and the *spectral norm* of A is $\|A\| \coloneqq \|A\|_{\mathrm{Op}} \coloneqq \sup_{x \in \mathbb{C}^n, \|x\|=1} \|Ax\|$. We say that U is an isometry if $\|Ux\| = \|x\|$ for all x, or equivalently, if U is a subset of columns of a unitary.

A *singular value decomposition* (SVD) of A is a representation $A = UDV^\dagger$, where for $N \coloneqq \min(m, n)$, $U \in \mathbb{C}^{m \times N}$ and $V \in \mathbb{C}^{n \times N}$ are isometries and $D \in \mathbb{R}^{N \times N}$ is diagonal with $\sigma_i \coloneqq D(i, i)$ and $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_N \geq 0$. We can also write this decomposition as $A = \sum_{i=1}^N \sigma_i u_i v_i^\dagger$, where $u_i \coloneqq U(\cdot, i)$ and $v_i \coloneqq V(\cdot, i)$. For Hermitian A, an *(unitary) eigendecomposition* of A is a singular value decomposition where $U = V$, except the entries of D are allowed to be negative.

Using SVD, we can define the rank-k approximation of A to be $A_k \coloneqq \sum_{i=1}^k \sigma_i u_i v_i^\dagger$ and the pseudoinverse of A to be $A^+ \coloneqq \sum_{i=1}^{\operatorname{rank}(A)} \frac{1}{\sigma_i} v_i u_i^\dagger$. We now formally define singular value transformation:

Definition 2.1. For a function $f \colon [0, \infty) \to \mathbb{C}$ such that $f(0) = 0$ and a matrix $A \in \mathbb{C}^{m \times n}$, we define the *singular value transform* of A via a singular value decomposition $A = \sum_{i=1}^{\min(m,n)} \sigma_i u_i v_i^\dagger$:

$$f^{(\mathrm{SV})}(A) \coloneqq \sum_{i=1}^{\min(m,n)} f(\sigma_i) u_i v_i^\dagger. \tag{1}$$

The requirement that $f(0) = 0$ ensures that the definition is independent of the (not necessarily unique) choice of SVD.

Definition 2.2. For a function $f : \mathbb{R} \to \mathbb{C}$ and a Hermitian matrix $A \in \mathbb{C}^{n\times n}$, we define the eigenvalue transform of A via a *unitary eigendecomposition* $A = \sum_{i=1}^{n} \lambda_i v_i v_i^\dagger$:

$$f^{(\mathrm{EV})}(A) \coloneqq \sum_{i=1}^{n} f(\lambda_i) v_i v_i^\dagger. \tag{2}$$

Since we only consider eigenvalue transformations of Hermitian matrices, where singular vectors/values and eigenvectors/values (roughly) coincide, the key difference between singular value transformation and eigenvalue transformation is that the latter can distinguish eigenvalue sign. As eigenvalue transformation is the standard notion of a matrix function, we will usually drop the superscript in notation: $f(A) \coloneqq f^{(\mathrm{EV})}(A)$.

We will use the following standard definition of a Lipschitz function.

Definition 2.3. We say $f\colon \mathbb{R} \to \mathbb{C}$ is *L-Lipschitz* on $\mathfrak{F} \subseteq \mathbb{R}$ if for all $x, y \in \mathfrak{F}$, $|f(x) - f(y)| \leq L|x - y|$.

We define approximate isometry as follows:[10]

Definition 2.4. Let $m, n \in \mathbb{N}$ and $m \geq n$. A matrix $V \in \mathbb{C}^{m\times n}$ is an *α-approximate isometry* if $\left\|V^\dagger V - I\right\| \leq \alpha$. It is an *$\alpha$-approximate projective isometry* if $\|V^\dagger V - \Pi\| \leq \alpha$ for Π an orthogonal projector.

If V is an α-approximate isometry, among other things, it implies that $|\|V\|^2 - 1| \leq \alpha$ and that there exists an isometry $U \in \mathbb{C}^{m\times n}$ with $\mathrm{im}(U) = \mathrm{im}(V)$ such that $\|U - V\| \leq \alpha$. We show this and other basic facts in the following lemma, whose proof is deferred to Appendix B.

Lemma 2.5. *If $\hat{X} \in \mathbb{C}^{m\times n}$ is an α-approximate isometry, then there is an exact isometry $X \in \mathbb{C}^{m\times n}$ with the same columnspace as $\hat{X}$ such that $\|\hat{X} - X\| \leq \alpha$. Furthermore, for any matrix $Y \in \mathbb{C}^{n\times n}$,*

$$\|\hat{X} Y \hat{X}^\dagger - X Y X^\dagger\| \leq (2\alpha + \alpha^2)\|Y\|.$$

If $\alpha < 1$, then $\|\hat{X}^+\| \leq (1-\alpha)^{-1}$ and

$$\|\hat{X} Y \hat{X}^\dagger - X Y X^\dagger\| \leq \alpha \frac{2-\alpha}{(1-\alpha)^2} \|\hat{X} Y \hat{X}^\dagger\|.$$

3 Sampling and query access oracles

Since we want our algorithms to run in time sublinear in input size, we must carefully define our access model. The sampling and query oracle we present below is unconventional, being designed as a reasonable classical analogue for the input model of some quantum algorithms. It will also be used heavily to move between intermediate steps of these quantum-inspired algorithms. First, as a warmup, we define a simple query oracle:

Definition 3.1 (Query access)**.** For a vector $v \in \mathbb{C}^n$, we have $\mathrm{Q}(v)$, *query access* to v, if for all $i \in [n]$, we can query for $v(i)$. Likewise, for a matrix $A \in \mathbb{C}^{m\times n}$, we have $\mathrm{Q}(A)$ if for all $(i, j) \in [m] \times [n]$, we can query for $A(i, j)$. Let $\mathbf{q}(v)$ (respectively $\mathbf{q}(A)$) denote the (time) cost of such a query.

For example, in the typical RAM access model, we are given our input $v \in \mathbb{C}^n$ as $\mathrm{Q}(v)$ with $\mathbf{q}(v) = 1$. For brevity, we will sometimes abuse this notation (and other access notations) and, for example, abbreviate "$\mathrm{Q}(A)$ for $A \in \mathbb{C}^{m\times n}$" as "$\mathrm{Q}(A) \in \mathbb{C}^{m\times n}$". We will also sometimes abuse complexity notation like $\mathbf{q}$ to refer to known bounds on the complexity, instead of the complexity itself.

[10] This is the notion of approximate orthonormality as given by the first arXiv version of [Tan19].

Definition 3.2 (Sampling and query access to a vector)**.** For a vector $v \in \mathbb{C}^n$, we have $\mathrm{SQ}(v)$, *sampling and query access* to v, if we can:

1. query for entries of v as in $\mathrm{Q}(v)$;
2. obtain independent samples $i \in [n]$ following the distribution $\mathcal{D}_v \in \mathbb{R}^n$, where $\mathcal{D}_v(i) \coloneqq |v(i)|^2/\|v\|^2$;
3. query for $\|v\|$.

Let $\mathbf{q}(v)$, $\mathbf{s}(v)$, and $\mathbf{n}(v)$ denote the cost of querying entries, sampling indices, and querying the norm respectively. Further define $\mathbf{sq}(v) \coloneqq \max(\mathbf{q}(v), \mathbf{s}(v), \mathbf{n}(v))$.

We will refer to these samples as *importance samples from* v, though one can view them as measurements of the quantum state $|v\rangle \coloneqq \frac{1}{\|v\|}\sum v_i|i\rangle$ in the computational basis.

Quantum-inspired algorithms typically don't give exact sampling and query access to the output vector. Instead, we get a more general version of sampling and query access, which assumes we can only access a sampling distribution that *oversamples* the correct distribution.[11]

Definition 3.3. For $p, q \in \mathbb{R}^n_{\geq 0}$ that are distributions, meaning $\sum_i p(i) = \sum_i q(i) = 1$, we say that p *ϕ-oversamples* q if, for all $i \in [n]$, $p(i) \geq q(i)/\phi$.

The motivation for this definition is the following: if p ϕ-oversamples q, then we can convert a sample from p to a sample from q with probability $1/\phi$ using rejection sampling: sample an i distributed as p, then accept the sample with probability $q(i)/(\phi p(i))$ (which is ≤ 1 by definition).

Definition 3.4 (Oversampling and query access)**.** For $v \in \mathbb{C}^n$ and $\phi \geq 1$, we have $\mathrm{SQ}_\phi(v)$, *ϕ-oversampling and query access* to v, if we have $\mathrm{Q}(v)$ and $\mathrm{SQ}(\tilde{v})$ for $\tilde{v} \in \mathbb{C}^n$ a vector satisfying $\|\tilde{v}\|^2 = \phi\|v\|^2$ and $|\tilde{v}(i)|^2 \geq |v(i)|^2$ for all $i \in [n]$. Denote $\mathbf{s}_\phi(v) \coloneqq \mathbf{s}(\tilde{v})$, $\mathbf{q}_\phi(v) \coloneqq \mathbf{q}(\tilde{v})$, $\mathbf{n}_\phi(v) \coloneqq \mathbf{n}(\tilde{v})$, and $\mathbf{sq}_\phi(v) \coloneqq \max(\mathbf{s}_\phi(v), \mathbf{q}_\phi(v), \mathbf{q}(v), \mathbf{n}_\phi(v))$.

The distribution $\mathcal{D}_{\tilde{v}}$ ϕ-oversamples $\mathcal{D}_v$, since for all $i \in [n]$,

$$\mathcal{D}_{\tilde{v}}(i) = \frac{|\tilde{v}_i|^2}{\|\tilde{v}\|^2} = \frac{|\tilde{v}_i|^2}{\phi\|v\|^2} \geq \frac{|v_i|^2}{\phi\|v\|^2} = \frac{1}{\phi}\mathcal{D}_v(i).$$

For this reason, we call $\mathcal{D}_{\tilde{v}}$ a *ϕ-oversampled importance sampling distribution* of v. $\mathrm{SQ}(v)$ is the same as $\mathrm{SQ}_1(v)$, by taking $\tilde{v} = v$. Note that we do not assume knowledge of ϕ (though it can be estimated, (though it can be estimated as shown in Lemma 3.5). However, we do need to know $\|\tilde{v}\|$ (even if $\|v\|$ is known), as it cannot be deduced from a small number of queries, samples, or probability computations. So, we will be choosing $\tilde{v}$ (and, correspondingly, ϕ) such that $\|\tilde{v}\|^2$ remains computable, even if potentially some $c\tilde{v}$ satisfies all our other requirements for some $c < 1$ (giving a smaller value of ϕ).

Intuitively speaking, estimators that use $\mathcal{D}_v$ can also use $\mathcal{D}_{\tilde{v}}$ via rejection sampling at the expense of a factor ϕ increase in the number of utilized samples. From this observation we can prove that oversampling access implies an approximate version of the usual sampling access:

Lemma 3.5. *Suppose we are given $\mathrm{SQ}_\phi(v)$ and some $\delta \in (0, 1]$. Denote $\widetilde{\mathbf{sq}}(v) \coloneqq \phi\,\mathbf{sq}_\phi(v)\log\frac{1}{\delta}$. We can sample from $\mathcal{D}_v$ with probability $\geq 1-\delta$ in $\mathcal{O}(\widetilde{\mathbf{sq}}(v))$ time. We can also estimate $\|v\|$ to ν multiplicative error for $\nu \in (0, 1]$ with probability $\geq 1-\delta$ in $\mathcal{O}\big(\frac{1}{\nu^2}\,\widetilde{\mathbf{sq}}(v)\big)$ time.*

[11] Oversampling turns out to be the "natural" form of approximation in this setting; other forms of error do not propagate through quantum-inspired algorithms well.

Proof. Consider the following rejection sampling algorithm to generate samples: sample an index i from $\tilde{v}$, and output it as the desired sample with probability $r(i) \coloneqq \frac{|v(i)|^2}{|\tilde{v}(i)|^2}$. Otherwise, restart. We can perform this: we can compute $r(i)$ in $\mathcal{O}\big(\mathbf{sq}_\phi(v)\big)$ time and $r(i) \le 1$ since $\tilde{v}$ bounds v.

The probability of accepting a sample in a round is $\sum_i \mathcal{D}_{\tilde{v}}(i) r(i) = \|v\|^2/\|\tilde{v}\|^2 = \phi^{-1}$ and, conditioned on a sample being accepted, the probability of it being i is $|v(i)|^2/\|v\|^2$, so the output distribution is $\mathcal{D}_v$ as desired. So, to get a sample with $\ge 1-\delta$ probability, run rejection sampling for at most $2\phi \log \frac{1}{\delta}$ rounds.

To estimate $\|v\|^2$, notice that we know $\|\tilde{v}\|^2$, so it suffices to estimate $\|v\|^2/\|\tilde{v}\|^2$ which is ϕ^{-1}. The probability of accepting the rejection sampling routine is ϕ^{-1}, so we run $3\nu^{-2}\phi \log \frac{2}{\delta}$ rounds of it for estimating ϕ^{-1}. Let Z denote the fraction of them which end in acceptance. Then, by a Chernoff bound we have

$$\Pr[|Z - \phi^{-1}| \ge \nu\phi^{-1}] \le 2\exp\Big(-\frac{\nu^2 z \phi^{-1}}{2+\nu}\Big) \le \delta,$$

so $Z\|\tilde{v}\|^2$ is a good multiplicative approximation to $\|v\|^2$ with probability $\ge 1-\delta$. □

Generally, compared to a quantum algorithm that can output (and measure) a desired vector $|v\rangle$, our algorithms will output $\mathrm{SQ}_\phi(u)$ such that $\|u - v\|$ is small. So, $\widetilde{\mathbf{sq}}(u)$ is the relevant complexity measure that we will analyze and bound: if we wish to mimic samples from the output of the quantum algorithm we dequantize, we will pay a one-time cost to run our quantum-inspired algorithm for "obtaining" $\mathrm{SQ}_\phi(u)$, and then pay $\widetilde{\mathbf{sq}}(u)$ cost per additional measurement. As for error, bounds on $\|u - v\|$ imply that measurements from u and v follow distributions that are close in total variation distance [Tan19, Lemma 4.1]. Now, we show that oversampling and query access of vectors is closed under taking small linear combinations.

Lemma 3.6 (Linear combinations, Proposition 4.3 of [Tan19])**.** *Given $\mathrm{SQ}_{\varphi_t}(v_t) \in \mathbb{C}^n$ and $\lambda_t \in \mathbb{C}$ for all $t \in [\tau]$, we have $\mathrm{SQ}_\phi(\sum_{t=1}^\tau \lambda_t v_t)$ for $\phi = \tau \frac{\sum \varphi_t \|\lambda_t v_t\|^2}{\|\sum \lambda_t v_t\|^2}$ and $\mathbf{sq}_\phi(\sum \lambda_t v_t) = \max_{t\in[\tau]} \mathbf{s}_{\varphi_t}(v_t) + \sum_{t=1}^\tau \mathbf{q}(v_t)$ (after paying $\mathcal{O}(\sum_{t=1}^\tau \mathbf{n}_{\varphi_t}(v_t))$ one-time pre-processing cost to query for norms).*

Proof. Denote $u \coloneqq \sum \lambda_t v_t$. To compute $u(s)$ for some $s \in [n]$, we just need to query $v_t(s)$ for all $t \in [\tau]$, paying $\mathcal{O}(\sum \mathbf{q}(v_t))$ cost. So, it suffices to get $\mathrm{SQ}(\tilde{u})$ for an appropriate bound $\tilde{u}$. We choose

$$\tilde{u}(s) = \sqrt{\tau \textstyle\sum_{t=1}^\tau |\lambda_t \tilde{v}_t(s)|^2},$$

so that $|\tilde{u}(s)| \ge |u(s)|$ by Cauchy–Schwarz, and $\|\tilde{u}\|^2 = \tau \sum_{t=1}^\tau \|\lambda_t \tilde{v}_t\|^2 = \tau \sum_{t=1}^\tau \varphi_t \|\lambda_t v_t\|^2$, giving the desired value of ϕ.

We have $\mathrm{SQ}(\tilde{u})$: we can compute $\|\tilde{u}\|^2$ by querying for all norms $\|\tilde{v}_t\|$, compute $\tilde{u}(s)$ by querying $\tilde{v}_t(s)$ for all $t \in [\tau]$. We can sample from $\tilde{u}$ by first sampling $t \in [\tau]$ with probability $\frac{\|\lambda_t \tilde{v}_t\|^2}{\sum_\ell \|\lambda_\ell \tilde{v}_\ell\|^2}$, and then taking our sample to be $j \in [n]$ from $\tilde{v}_t$. The probability of sampling $j \in [n]$ is correct:

$$\sum_{t=1}^\tau \frac{\|\lambda_t \tilde{v}_t\|^2}{\sum_\ell \|\lambda_\ell \tilde{v}_\ell\|^2} \frac{|\tilde{v}_t(j)|^2}{\|\tilde{v}_t\|^2} = \frac{\sum_{t=1}^\tau |\lambda_t \tilde{v}_t(j)|^2}{\sum_{\ell=1}^\tau \|\lambda_\ell \tilde{v}_\ell\|^2} = \frac{|\tilde{u}(j)|^2}{\|\tilde{u}\|^2}.$$

If we pre-process by querying all the norms $\|\tilde{v}_\ell\|$ in advance, we can sample from the distribution over i's in $\mathcal{O}(1)$ time, using an alias sampling data structure for the distribution (Remark 3.11), and we can sample from $\tilde{v}_t$ using our assumed access to it, $\mathrm{SQ}_{\varphi_t}(v_t)$. □

So, our general goal will be to express our output vector as a linear combination of a small number of input vectors that we have sampling and query access to. Then, we can get an approximate SQ access to our output using Lemma 3.5, where we pay an additional "cancellation constant" factor of $\phi = \tau \frac{\sum \varphi_t \|\lambda_t v_t\|^2}{\|\sum \lambda_t v_t\|^2}$. This factor is only large when the linear combination has significantly smaller norm than the components v_t in the sum suggest. Usually, in our applications, we can intuitively think about this overhead being small when the desired output vector mostly lies in a subspace spanned by singular vectors with large singular values in our low-rank input. Quantum algorithms also have the same kind of overhead. Namely, the QSVT framework encodes this in the subnormalization constant α of block-encodings, and the overhead from the subnormalization appears during post-selection [GSLW19]. When this cancellation is not too large, the resulting overhead typically does not affect too badly the runtime of our applications.

We also define oversampling and query access for a matrix. The same model (under an alternative definition) is also discussed in prior work [FKV04, DKR02] and is the right notion for the sampling procedures we will use.

Definition 3.7 (Oversampling and query access to a matrix)**.** For a matrix $A \in \mathbb{C}^{m\times n}$, we have $\mathrm{SQ}(A)$ if we have $\mathrm{SQ}(A(i,\cdot))$ for all $i \in [m]$ and $\mathrm{SQ}(a)$ for $a \in \mathbb{R}^m$ the vector of row norms $(a(i) \coloneqq \|A(i,\cdot)\|)$.

We have $\mathrm{SQ}_\phi(A)$ if we have $\mathrm{Q}(A)$ and $\mathrm{SQ}(\tilde{A})$ for $\tilde{A} \in \mathbb{C}^{m\times n}$ satisfying $\|\tilde{A}\|_{\mathrm{F}}^2 = \phi\|A\|_{\mathrm{F}}^2$ and $|\tilde{A}(i,j)|^2 \geq |A(i,j)|^2$ for all $(i,j) \in [m]\times[n]$.

The complexity of (over)sampling and querying from the matrix A is denoted by $\mathbf{s}_\phi(A) \coloneqq \max(\mathbf{s}(\tilde{A}(i,\cdot)), \mathbf{s}(\tilde{a}))$, $\mathbf{q}_\phi(A) \coloneqq \max(\mathbf{q}(\tilde{A}(i,\cdot)), \mathbf{q}(\tilde{a}))$, $\mathbf{q}(A) \coloneqq \max(\mathbf{q}(A(i,\cdot)))$, and $\mathbf{n}_\phi(A) \coloneqq \mathbf{n}(\tilde{a})$ respectively. We also denote $\mathbf{sq}_\phi(A) \coloneqq \max(\mathbf{s}_\phi(A), \mathbf{q}_\phi(A), \mathbf{q}(A), \mathbf{n}_\phi(A))$. We omit subscripts if $\phi = 1$.

Observe that access to a matrix, $\mathrm{SQ}_\phi(A)$, implies access to its vectorized version, $\mathrm{SQ}_\phi(\mathrm{vec}(A))$: we can take $\widetilde{\mathrm{vec}(A)} = \mathrm{vec}(\tilde{A})$, and the distribution for $\mathrm{vec}(\tilde{A})$ is sampled by sampling i from $\mathcal{D}_{\tilde{a}}$, and then sampling j from $\mathcal{D}_{\tilde{A}(i,\cdot)}$. This gives the output (i,j) with probability $|\tilde{A}(i,j)|^2/\|\tilde{A}\|_{\mathrm{F}}^2$. Therefore, one can think of $\mathrm{SQ}_\phi(A)$ as $\mathrm{SQ}_\phi(\mathrm{vec}(A))$, with the addition of having access to samples (i,j) from $\mathrm{vec}(A)$, conditioned on fixing a particular row i and also knowing the probabilities of these conditional samples.

Now we prove that oversampling and query access is closed under taking outer products. The same idea also extends to taking Kronecker products of matrices.

Lemma 3.8. *Given vectors $\mathrm{SQ}_{\varphi_u}(u) \in \mathbb{C}^m$ and $\mathrm{SQ}_{\varphi_v}(v) \in \mathbb{C}^n$, we have $\mathrm{SQ}_\phi(A)$ for their outer product $A \coloneqq uv^\dagger$ with $\phi = \varphi_u\varphi_v$ and $\mathbf{s}_\phi(A) = \mathbf{s}_{\varphi_u}(u) + \mathbf{s}_{\varphi_v}(v)$, $\mathbf{q}_\phi(A) = \mathbf{q}_{\varphi_u}(u) + \mathbf{q}_{\varphi_v}(v)$, $\mathbf{q}(A) = \mathbf{q}(u) + \mathbf{q}(v)$, and $\mathbf{n}_\phi(A) = \mathbf{n}_{\varphi_u}(u) + \mathbf{n}_{\varphi_v}(v)$,*

Proof. We can query an entry $A(i,j) = u(i)v(j)^\dagger$ by querying once from u and v. Our choice of upper bound is $\tilde{A} = \tilde{u}\tilde{v}^\dagger$. Clearly, this is an upper bound on $uv^\dagger$ and $\|\tilde{A}\|_{\mathrm{F}}^2 = \|\tilde{u}\|^2\|\tilde{v}\|^2 = \varphi_u\varphi_v\|A\|_{\mathrm{F}}^2$. We have $\mathrm{SQ}(\tilde{A})$ in the following manner: $\tilde{A}(i,\cdot) = \tilde{u}(i)\tilde{v}^\dagger$, so we have $\mathrm{SQ}(\tilde{A}(i,\cdot))$ from $\mathrm{SQ}(\tilde{v})$ after querying for $\tilde{u}(i)$, and $\tilde{a} = \|\tilde{v}\|^2\tilde{u}$, so we have $\mathrm{SQ}(\tilde{a})$ from $\mathrm{SQ}(\tilde{u})$ after querying for $\|\tilde{v}\|$. □

Using the same ideas as in Lemma 3.6, we can extend sampling and query access of input matrices to linear combinations of those matrices.

Lemma 3.9. *Given $\mathrm{SQ}_{\varphi^{(t)}}(A^{(t)}) \in \mathbb{C}^{m\times n}$ and $\lambda_t \in \mathbb{C}$ for all $t \in [\tau]$, we have $\mathrm{SQ}_\phi(A) \in \mathbb{C}^{m\times n}$ for $A \coloneqq \sum_{t=1}^\tau \lambda_t A^{(t)}$ with $\phi = \tau\frac{\sum_{t=1}^\tau \varphi^{(t)}\|\lambda_t A^{(t)}\|_{\mathrm{F}}^2}{\|A\|_{\mathrm{F}}^2}$ and $\mathbf{s}_\phi(A) = \max_{t\in[\tau]} \mathbf{s}_{\varphi^{(t)}}(A^{(t)}) + \sum_{t=1}^\tau \mathbf{q}_{\varphi^{(t)}}(A^{(t)})$,*

$\mathbf{q}_\phi(A) = \sum_{t=1}^{\tau} \mathbf{q}_{\varphi^{(t)}}(A^{(t)})$, $\mathbf{q}(A) = \sum_{t=1}^{\tau} \mathbf{q}(A^{(t)})$, *and* $\mathbf{n}_\phi(A) = 1$ *(after paying* $\mathcal{O}\Big(\sum_{t=1}^{\tau} \mathbf{n}_{\varphi^{(t)}}(A^{(t)})\Big)$ *one-time pre-processing cost).*

Proof. To compute $A(i,j) = \sum_{t=1}^{\tau} \lambda_t A^{(t)}(i,j)$ for $(i,j) \in [m] \times [n]$, we just need to query $A^{(t)}(i,j)$ for all $t \in [\tau]$, paying $\mathcal{O}\big(\sum_t \mathbf{q}(A^{(t)})\big)$ cost. So, it suffices to get $\mathrm{SQ}(\tilde{A})$ for an appropriate bound $\tilde{A}$. We choose

$$\tilde{A}(i,j) = \sqrt{\tau \sum_{t=1}^{\tau} |\lambda_t \tilde{A}^{(t)}(i,j)|^2}.$$

That $|\tilde{A}(i,j)| \geq |A(i,j)|$ follows from Cauchy–Schwarz, and we get the desired value of ϕ:

$$\|\tilde{A}\|_{\mathrm{F}}^2 = \tau \sum_{t=1}^{\tau} \|\lambda_i \tilde{A}^{(t)}\|_{\mathrm{F}}^2 = \tau \sum_{t=1}^{\tau} \varphi^{(t)} \|\lambda_i A^{(t)}\|_{\mathrm{F}}^2.$$

We have $\mathrm{SQ}(\tilde{A})$: we can compute $\|\tilde{A}\|_{\mathrm{F}}$ by querying for all norms $\|\tilde{A}^{(t)}\|_{\mathrm{F}}$, compute $\tilde{a}(i) = \|\tilde{A}(i,\cdot)\| = \sqrt{\tau \sum_{t=1}^{\tau} \|\lambda_t \tilde{A}^{(t)}(i,\cdot)\|^2}$ by querying $\tilde{a}^{(t)}(i)$ for all $t \in [\tau]$, and compute $\tilde{A}(i,j)$ by querying $\tilde{A}^{(t)}(i,j)$ for all $t \in [\tau]$. Analogously to Lemma 3.6, we can sample from $\tilde{a}$ by first sampling $s \in [\tau]$ with probability $\frac{\|\lambda_s \tilde{A}^{(s)}\|_{\mathrm{F}}^2}{\sum_t \|\lambda_t \tilde{A}^{(t)}\|_{\mathrm{F}}^2}$, then taking our sample to be $i \in [m]$ from $\mathcal{D}_{\tilde{a}^{(s)}}$. If we pre-process by querying all the Frobenius norms $\|\tilde{A}^{(t)}\|_{\mathrm{F}}$ in advance, we can sample from $\tilde{a}$ in $\mathcal{O}\Big(\max_{t\in[\tau]} \mathbf{s}_{\varphi^{(t)}}(A^{(t)})\Big)$ time. We can sample from $\tilde{A}(i,\cdot)$ by first sampling $s \in [\tau]$ with probability $\frac{\|\lambda_s \tilde{A}^{(s)}(i,\cdot)\|^2}{\sum_t \|\lambda_t \tilde{A}^{(t)}(i,\cdot)\|^2}$, then taking our sample to be $j \in [n]$ from $\mathcal{D}_{\tilde{A}^{(s)}(i,\cdot)}$. This takes $\mathcal{O}\Big(\sum_{t=1}^{\tau} \mathbf{q}_{\varphi^{(t)}}(A^{(t)}) + \max_{t\in[\tau]} \mathbf{s}_{\varphi^{(t)}}(A^{(t)})\Big)$ time. □

Remark 3.10. With the lemmas we've introduced, we can already get oversampling and query access to some modest expressions. For example, consider RUR decompositions, which show up frequently in our results: suppose we have $\mathrm{SQ}(A)$ for $A \in \mathbb{C}^{m\times n}$, $R \in \mathbb{C}^{r\times n}$ a (possibly normalized) subset of rows of A, and a matrix $U \in \mathbb{C}^{r\times r}$. Then

$$R^\dagger U R = \sum_{i=1}^{r} \sum_{j=1}^{r} U(i,j) R(i,\cdot)^\dagger R(j,\cdot),$$

which is a linear combination of r^2 outer products involving rows of A. So, by Lemma 3.8 and Lemma 3.9, we have $\mathrm{SQ}_\phi(R^\dagger U R)$.

For us, the most interesting scenario is when our sampling and query oracles take poly-logarithmic time, since this corresponds to the scenarios where quantum state preparation procedures can run in time $\mathrm{polylog}(n)$. In these scenarios, quantum machine learning have the potential to achieve exponential speedups. We can provide such classical access in various ways.

Remark 3.11. Below, we list settings where we have sampling and query access to input matrices and vectors, and whenever relevant, we compare the resulting runtimes to the time to prepare analogous quantum states. Note that because we do not analyze classical algorithms in the bit model, i.e., we do not count each operation bitwise, their runtimes may be missing log factors that should be counted for a fair comparison between classical and quantum.

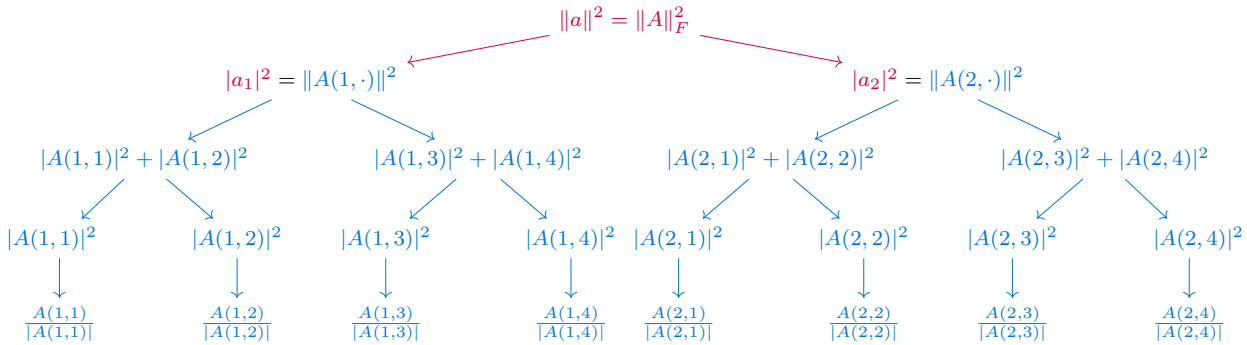

Figure 2: Dynamic data structure for a matrix $A \in \mathbb{C}^{2\times 4}$ discussed in Remark 3.11 part (b). We compose the data structure for a with the data structure for A's rows.

(a) (Data structure) Given $v \in \mathbb{C}^n$ in the standard RAM model, the alias method [Vos91] takes $\Theta(n)$ pre-processing time to output a data structure that uses $\Theta(n)$ space and can sample from v in $\Theta(1)$ time. In other words, we can get $\mathrm{SQ}(v)$ with $\mathbf{sq}(v) = \Theta(1)$ in $\mathcal{O}(n)$ time, and by extension, for a matrix $A \in \mathbb{C}^{m\times n}$, $\mathrm{SQ}(A)$ with $\mathbf{sq}(A) = \Theta(1)$ in $\mathcal{O}(mn)$ time.

If the input vector (resp. matrix) is given as a list of $\mathrm{nnz}(v)$ (resp. $\mathrm{nnz}(A)$) of its non-zero entries, then the pre-processing time is linear in that number of entries. Therefore, the quantum-inspired setting can be directly translated to a basic randomized numerical linear algebra algorithm. More precisely, with this data structure, a fast quantum-inspired algorithm (say, one running in time $\mathcal{O}(T\,\mathbf{sq}(A))$ for T independent of input size) implies an algorithm in the standard computational model (running in $\mathcal{O}(\mathrm{nnz}(A) + T)$ time).

(b) (Dynamic data structure) QML algorithms often assume that their input is in a data structure with a certain kind of quantum access [Pra14, KP20, GLM08, WZP18, RSW+19, CGJ19]. They argue that, since this data structure allows for circuits preparing input states with linear gate count but polylog *depth*, hardware called QRAM might be able to parallelize these circuits enough so that they run in effectively polylog *time*. In the interest of considering the best of all possible worlds for QML, we will treat circuit depth as runtime for QRAM and ignore technicalities.

This data structure (see Fig. 2) admits sampling and query access to the data it stores with just-as-good runtimes: specifically, for a matrix $A \in \mathbb{C}^{m\times n}$, we get $\mathrm{SQ}(A)$ with $\mathbf{q}(A) = \mathcal{O}(1)$, $\mathbf{s}(A) = \mathcal{O}(\log mn)$, and $\mathbf{n}(A) = \mathcal{O}(1)$. So, quantum-inspired algorithms can be used whenever QML algorithms assume this form of input.

Further, unlike the alias method stated above, this data structure supports updating entries in $\mathcal{O}(\log mn)$ time, which is used in applications of QML where data accumulates over time [KP17].

(c) (Integrability assumption) For $v \in \mathbb{C}^n$, suppose we can compute entries $v(i)$ and sums $\sum_{i\in I(b)}|v(i)|^2$ in time T, where $I(b) \subset [n]$ is the set of indices whose binary representation begins with the bitstring b. Then we have $\mathrm{SQ}(v)$ where $\mathbf{q}(v) = \mathcal{O}(T), \mathbf{s}(v) = \mathcal{O}(T\log n)$, and $\mathbf{n}(v) = \mathcal{O}(T)$. Analogously, the quantum state that encodes v in its amplitudes, $|v\rangle = \sum_i \frac{v_i}{\|v\|}|i\rangle$, can be prepared in time $\mathcal{O}(T\log n)$ via Grover-Rudolph state preparation [GR02]. (One can

think about the QRAM data structure as pre-computing all the necessary sums for this protocol.)

(d) (Uniformity assumption) Given $\mathcal{O}(1)$-time $\mathrm{Q}(v) \in \mathbb{C}^n$ and a β such that $\max|v(i)|^2 \leq \beta/n$, we have $\mathrm{SQ}_\phi(v)$ with $\phi = \beta/\|v\|^2$ and $\mathbf{sq}_\phi(v) = \mathcal{O}(1)$, by using the vector whose entries are all $\sqrt{\beta/n}$ as the upper bound $\tilde{v}$. Assuming the ability to query entries of v in superposition, a quantum state corresponding to v can be prepared in time $\mathcal{O}\big(\sqrt{\phi}\log n\big)$.

(e) (Sparsity assumption) If $A \in \mathbb{C}^{m\times n}$ has at most s non-zero entries per row (with efficiently computable locations) and the matrix elements are $|A(i,j)| \leq c$ (and efficiently computable), then we have $\mathrm{SQ}_\phi(A)$ for $\phi = c^2\frac{sm}{\|A\|_{\mathrm{F}}^2}$, simply by using the uniform distribution over non-zero entries for the oversampling and query oracles. For example, for $\mathrm{SQ}(\tilde{a})$ we can set $\tilde{a}(i) \coloneqq c\sqrt{s}$, and for $\tilde{A}(i,\cdot)$ we use the vector with entries c at the non-zeros of $A(i,\cdot)$ (potentially adding some "dummy" zero locations to have exactly s non-zeroes).

Note that similar sparse-access assumptions are often seen in the QML and Hamiltonian simulation literature [HHL09]. Also, if A is not much smaller than we expect, then ϕ can be independent of dimension. For example, if A has exactly s non-zero entries per row and $|A(i,j)| \geq c'$ for non-zero entries, then $\phi \leq (c/c')^2$.

(f) (CT states) In 2009, Van den Nest defined the notion of a "computationally tractable" (CT) state [VdN11]. Using our notation, $|\psi\rangle \in \mathbb{C}^n$ is a CT state if we have $\mathrm{SQ}(\psi)$ with $\mathbf{sq}(\psi) = \mathrm{polylog}(n)$. Van den Nest's paper identifies several classes of CT states, including product states, quantum Fourier transforms of product states, matrix product states of polynomial bond dimension, stabilizer states, and states from matchgate circuits. For more details on how can one get efficient sampling and query access to such vectors we direct the reader to [VdN11].

4 Matrix sketches

We now introduce the workhorse of our algorithms: the matrix sketch. Using sampling and query access, we can generate these sketches efficiently, and these allow one to reduce the dimensionality of a problem, up to some approximation. Most of the results presented in this section are known in the classical sketching literature: we present them here for completeness, and to restate them in the context of sampling and query access.

Definition 4.1. For a distribution $p \in \mathbb{R}^m$, we say that a matrix $S \in \mathbb{R}^{s\times m}$ is *sampled according to* p if each row of S is independently chosen to be $e_i/\sqrt{s\cdot p(i)}$ with probability $p(i)$, where e_i is the vector that is one in the ith position and zero elsewhere. If p is an ℓ_2-norm sampling distribution $\mathcal{D}_v$ as defined in Definition 3.2, then we also say S is *sampled according to* v.

We call S an *importance sampling sketch* for $A \in \mathbb{C}^{m\times n}$ if it is sampled according to A's row norms a, and we call S a *ϕ-oversampled importance sampling sketch* if it is sampled according to the bounding row norms from $\mathrm{SQ}_\phi(A)$, $\tilde{a}$ (or, more generally, from a ϕ-oversampled importance sampling distribution of a).

One should think of S as a description of how to sketch A down to SA. The following lemma shows that $\|SA\|_{\mathrm{F}}$ approximates $\|A\|_{\mathrm{F}}$, giving a simple example of the phenomenon that SA approximates A in certain senses: it shows that $\|SA\|_{\mathrm{F}} = \Theta(\|A\|_{\mathrm{F}})$ with probability ≥ 0.9 when S has $\Omega(\frac{1}{\phi^2})$ rows. We show later (Lemma 4.8) that a similar statement holds for spectral norm: $\|SA\| = \Theta(\|A\|)$ with probability ≥ 0.9 when S has $\tilde{\Omega}(\phi^2\|A\|_{\mathrm{F}}^2/\|A\|^2)$ rows.

Lemma 4.2 (Frobenius norm bounds for matrix sketches)**.** *Let $S \in \mathbb{C}^{r\times m}$ be a ϕ-oversampled importance sampling sketch of $A \in \mathbb{C}^{m\times n}$. Then $\|[SA](i,\cdot)\| \le \sqrt{\phi/r}\|A\|_{\mathrm{F}}$ for all $i \in [r]$, so $\|SA\|_{\mathrm{F}}^2 \le \phi\|A\|_{\mathrm{F}}^2$ (unconditionally). Equality holds when $\phi = 1$. Further,*

$$\Pr\left[\left|\|SA\|_{\mathrm{F}}^2 - \|A\|_{\mathrm{F}}^2\right| \ge \sqrt{\frac{\phi^2\log(2/\delta)}{2r}}\|A\|_{\mathrm{F}}^2\right] \le \delta.$$

Proof. Let p be the distribution used to create S, and let s_i be the sample from p used for row i of S. Then $\|SA\|_{\mathrm{F}}^2$ is the sum of the row norms $\|[SA](i,\cdot)\|^2$ over all $i \in [r]$, and

$$\|[SA](i,\cdot)\|^2 = \frac{\|A(s_i,\cdot)\|^2}{r\cdot p(s_i)} \le \frac{\phi}{r}\|A\|_{\mathrm{F}}^2$$
$$\mathbb{E}[\|[SA](i,\cdot)\|^2] = \sum_{s=1}^{m} p(s)\frac{\|A(s,\cdot)\|^2}{r\cdot p(s)} = \frac{1}{r}\|A\|_{\mathrm{F}}^2$$

The first equation shows the unconditional bounds on $\|SA\|_{\mathrm{F}}$. When $\phi = 1$, $p(i) = \|A(i,\cdot)\|^2/\|A\|_{\mathrm{F}}^2$ so the inequality becomes an equality. By the second equation, $\|SA\|_{\mathrm{F}}^2 - \|A\|_{\mathrm{F}}^2$ has expected value zero and is the sum of independent random variables bounded in $[-\|A\|_{\mathrm{F}}^2, (\phi-1)\|A\|_{\mathrm{F}}^2]$, so the probabilistic bound follows immediately from Hoeffding's inequality. □

In the standard algorithm setting, computing an importance sampling sketch requires reading all of A, since we need to sample from $\mathcal{D}_a$. If we have $\mathrm{SQ}_\phi(A)$, though, we can efficiently create a ϕ-oversampling sketch S in $\mathcal{O}\big(s(\mathbf{s}_\phi(A) + \mathbf{q}_\phi(A)) + \mathbf{n}_\phi(A)\big)$ time: for each row of S, we pull a sample from p, and then compute $\sqrt{p(i)}$. After finding this sketch S, we have an implicit description of SA: it is a normalized multiset of rows of A, so we can describe it with the row indices and corresponding normalization, $(i_1, c_1), \ldots, (i_s, c_s)$.

SA can be used to approximate matrix expressions involving A. Further, we can chain sketches using the lemma below, which shows that from $\mathrm{SQ}_\phi(A)$, we have $\mathrm{SQ}_{\le 2\phi}((SA)^\dagger)$, under a mild assumption on the size of the sketch S. This can be used to find a sketch $T^\dagger$ of $(SA)^\dagger$. The resulting expression SAT is small enough that we can compute functions of it in time independent of dimension, and so will be used extensively. When we discuss sketching A down to SAT, we are referring to the below lemma for the method of sampling T.

Lemma 4.3. *Consider $\mathrm{SQ}_\varphi(A) \in \mathbb{C}^{m\times n}$ and $S \in \mathbb{R}^{r\times m}$ sampled according to $\tilde{a}$, described as pairs $(i_1, c_1), \ldots, (i_r, c_r)$. If $r \ge 2\varphi^2 \log\frac{2}{\delta}$, then with probability $\ge 1-\delta$, we have $\mathrm{SQ}_\phi(SA)$ and $\mathrm{SQ}_\phi((SA)^\dagger)$ for some ϕ satisfying $\phi \le 2\varphi$. If $\varphi = 1$, then for all r, we have $\mathrm{SQ}(SA)$ and $\mathrm{SQ}((SA)^\dagger)$.*

The runtimes for $\mathrm{SQ}_\phi(SA)$ are $\mathbf{q}(SA) = \mathbf{q}(A)$, $\mathbf{s}_\phi(SA) = \mathbf{s}_\varphi(A)$, $\mathbf{q}_\phi(SA) = \mathbf{q}_\varphi(A)$, and $\mathbf{n}_\phi(SA) = \mathcal{O}(1)$, after $\mathcal{O}(\mathbf{n}_\varphi(A))$ pre-processing cost. The runtimes for $\mathrm{SQ}_\phi((SA)^\dagger)$ are $\mathbf{q}((SA)^\dagger) = \mathbf{q}(A)$, $\mathbf{s}_\phi((SA)^\dagger) = \mathbf{s}_\varphi(A) + r\,\mathbf{q}_\varphi(A)$, $\mathbf{q}_\phi((SA)^\dagger) = r\,\mathbf{q}_\varphi(A)$, and $\mathbf{n}_\phi((SA)^\dagger) = \mathbf{n}_\varphi(A)$.

Proof. By Lemma 4.2, $\|SA\|_{\mathrm{F}}^2 \ge \|A\|_{\mathrm{F}}^2/2$ with probability $\ge 1-\delta$. Suppose this bound holds. To get $\mathrm{SQ}_\phi(SA)$, we take $\widetilde{SA} = S\tilde{A}$, which bounds SA by inspection. Further, $\|S\tilde{A}\|_{\mathrm{F}}^2 = \|\tilde{A}\|_{\mathrm{F}}^2$ by Lemma 4.2, so $\phi = \|S\tilde{A}\|_{\mathrm{F}}^2/\|SA\|_{\mathrm{F}}^2 = \varphi\|A\|_{\mathrm{F}}^2/\|SA\|_{\mathrm{F}}^2 \le 2\varphi$. Analogously, $(S\tilde{A})^\dagger$ works as a bound for $\mathrm{SQ}_\phi((SA)^\dagger)$. We can query an entry of SA by querying the corresponding entry of A, so all that suffices is to show that we have $\mathrm{SQ}(S\tilde{A})$ and $\mathrm{SQ}((S\tilde{A})^\dagger)$ from $\mathrm{SQ}(\tilde{A})$. (When $\varphi = 1$, we can ignore the above argument: the rest of the proof will show that we have $\mathrm{SQ}(SA)$ and $\mathrm{SQ}((SA)^\dagger)$ from $\mathrm{SQ}(A)$.)

We have $\mathrm{SQ}(S\tilde{A})$. Because the rows of $S\tilde{A}$ are rescaled rows of $\tilde{A}$, we have SQ access to them from SQ access to $\tilde{A}$. Because $\|S\tilde{A}\|_{\mathrm{F}}^2 = \|\tilde{A}\|_{\mathrm{F}}^2$ and $\|[S\tilde{A}](i,\cdot)\|^2 = \|\tilde{A}\|_{\mathrm{F}}^2/r$, after precomputing

$\|\tilde{A}\|_{\mathrm{F}}^2$, we have SQ access to the vector of row norms of $S\tilde{A}$ (pulling samples simply by pulling samples from the uniform distribution).

We have $\mathrm{SQ}((S\tilde{A})^\dagger)$. (This proof is similar to one from [FKV04].) Since the rows of $(S\tilde{A})^\dagger$ are length r, we can respond to SQ queries to them by reading all entries of the row and performing some linear-time computation. $\|(S\tilde{A})^\dagger\|_{\mathrm{F}}^2 = \|\tilde{A}\|_{\mathrm{F}}^2$, so we can respond to a norm query by querying the norm of $\tilde{A}$. Finally, we can sample according to the row norms of $(S\tilde{A})^\dagger$ by first querying an index $i \in [r]$ uniformly at random, then outputting the index $j \in [n]$ sampled from $[S\tilde{A}](i,\cdot)$ (which we can sample from because it is a row of $\tilde{A}$). The distribution of the samples output by this procedure is correct: the probability of outputting j is

$$\frac{1}{r}\sum_{i=1}^{r}\frac{|[S\tilde{A}](i,j)|^2}{\|[S\tilde{A}](i,\cdot)\|^2} = \sum_{i=1}^{r}\frac{|[S\tilde{A}](i,j)|^2}{\|S\tilde{A}\|_{\mathrm{F}}^2} = \frac{\|[S\tilde{A}](\cdot,j)\|^2}{\|S\tilde{A}\|_{\mathrm{F}}^2}. \qquad \square$$

4.1 Approximation results

Here, we present approximation results on sketched matrices that we will use heavily throughout our results. We begin with a fundamental observation: given sampling and query access to a matrix A, we can approximate the matrix product $A^\dagger B$ by a sum of rank-one outer products. We formalize this with two variance bounds, which we can use together with Chebyshev's inequality.

Lemma 4.4 (Asymmetric matrix multiplication to Frobenius norm error, [DKM06, Lemma 4])**.** *Consider $X \in \mathbb{C}^{m\times n}, Y \in \mathbb{C}^{m\times p}$, and take $S \in \mathbb{R}^{r\times m}$ to be sampled according to $p \in \mathbb{R}^m$ a ϕ-oversampled importance sampling distribution from X or Y. Then,*

$$\mathbb{E}[\|X^\dagger S^\dagger SY - X^\dagger Y\|_{\mathrm{F}}^2] \le \frac{\phi}{r}\|X\|_{\mathrm{F}}^2\|Y\|_{\mathrm{F}}^2 \quad \textit{and} \quad \mathbb{E}\Big[\sum_{i=1}^{r}\|[SX](i,\cdot)\|^2\|[SY](i,\cdot)\|^2\Big] \le \frac{\phi}{r}\|X\|_{\mathrm{F}}^2\|Y\|_{\mathrm{F}}^2.$$

Proof. To show the first equation, we use that $\mathbb{E}[\|X^\dagger X^\dagger SY - X^\dagger Y\|_{\mathrm{F}}^2]$ is a sum of variances, one for each entry (i,j), since $\mathbb{E}[X^\dagger S^\dagger SY - XY]$ is zero in every entry. Furthermore, for every entry (i,j), the matrix expression is the sum of r independent, mean-zero terms, one for each row of S:

$$[X^\dagger S^\dagger SY - XY](i,j) = \sum_{s=1}^{r}\Big([SX](s,i)^\dagger[SY](s,j) - \frac{1}{r}[X^\dagger Y](i,j)\Big).$$

So, we can use standard properties of variances[12] to conclude that

$$\begin{aligned}\mathbb{E}[\|X^\dagger S^\dagger SY - X^\dagger Y\|_{\mathrm{F}}^2] &= r\cdot\mathbb{E}[\|[SX](1,\cdot)^\dagger[SY](1,\cdot) - \tfrac{1}{r}X^\dagger Y\|_{\mathrm{F}}^2] \le r\cdot\mathbb{E}[\|[SX](1,\cdot)^\dagger[SY](1,\cdot)\|_{\mathrm{F}}^2] \\ &= r\sum_{i=1}^{m}p(i)\frac{\|X(i,\cdot)^\dagger Y(i,\cdot)\|_{\mathrm{F}}^2}{r^2p(i)^2} = \frac{1}{r}\sum_{i=1}^{m}\frac{\|X(i,\cdot)\|^2\|Y(i,\cdot)\|^2}{p(i)} \le \frac{\phi}{r}\|X\|_{\mathrm{F}}^2\|Y\|_{\mathrm{F}}^2.\end{aligned}$$

The second other inequality follows by the same computation:

$$\mathbb{E}\Big[\sum_{i=1}^{r}\|[SX](i,\cdot)\|^2\|[SY](i,\cdot)\|^2\Big] = r\cdot\mathbb{E}[\|[SX](1,\cdot)\|^2\|[SY](1,\cdot)\|^2] \le \frac{\phi}{s}\|X\|_{\mathrm{F}}^2\|Y\|_{\mathrm{F}}^2. \qquad \square$$

The above result shows that, given $\mathrm{SQ}(X)$, $X^\dagger Y$ can be approximated by a sketch with constant failure probability. If we have $\mathrm{SQ}(X)$ *and* $\mathrm{SQ}(Y)$, we can make the failure probability exponential small. To show this tighter error bound, we use an argument of Drineas, Kannan, and Mahoney for approximating matrix multiplication. We state their result in a slightly stronger form, which is actually proved in their paper. For completeness, a proof of this statement is in the appendix.

[12] See the proof of Lemma 4.5 in Appendix B for this kind of computation done with more detail.

Lemma 4.5 (Matrix multiplication by subsampling [DKM06, Theorem 1])**.** *Suppose we are given $X \in \mathbb{C}^{n\times m}, Y \in \mathbb{C}^{n\times p}, r \in \mathbb{N}$ and a distribution $p \in \mathbb{R}^n$ satisfying the oversampling condition that, for some $\phi \geq 1$,*

$$p(k) \geq \frac{\|X(k,\cdot)\|\|Y(k,\cdot)\|}{\phi \sum_\ell \|X(\ell,\cdot)\|\|Y(\ell,\cdot)\|}.$$

Let $S \in \mathbb{R}^{r\times n}$ be sampled according to p. Then $X^\dagger S^\dagger S Y$ is an unbiased estimator for $X^\dagger Y$ and

$$\Pr\left[\|X^\dagger S^\dagger S Y - X^\dagger Y\|_{\mathrm{F}} < \sqrt{\frac{8\phi^2 \log(2/\delta)}{r}} \underbrace{\sum_\ell \|X(\ell,\cdot)\|\|Y(\ell,\cdot)\|}_{\leq \|X\|_{\mathrm{F}}\|Y\|_{\mathrm{F}}}\right] > 1-\delta.$$

From a simple application of Lemma 4.5, we get a key lemma used frequently in Section 6.

Lemma 4.6 (Approximating matrix multiplication to Frobenius norm error; corollary of [DKM06, Theorem 1])**.** *Consider $X \in \mathbb{C}^{m\times n}, Y \in \mathbb{C}^{m\times p}$, and take $S \in \mathbb{R}^{r\times m}$ to be sampled according to $q \coloneqq \frac{q_1+q_2}{2}$, where $q_1, q_2 \in \mathbb{R}^m$ are ϕ_1, ϕ_2-oversampled importance sampling distributions from x, y, the vector of row norms for X, Y, respectively. Then S is a $2\phi_1, 2\phi_2$-oversampled importance sampling sketch of X, Y, respectively. Further,*

$$\Pr\left[\|X^\dagger S^\dagger S Y - X^\dagger Y\|_{\mathrm{F}} < \sqrt{\frac{8\phi_1\phi_2 \log 2/\delta}{r}}\|X\|_{\mathrm{F}}\|Y\|_{\mathrm{F}}\right] > 1-\delta.$$

Proof. First, notice that $2q(i) \geq q_1(i)$ and $2q(i) \geq q_2(i)$, so q oversamples the importance sampling distributions for X and Y with constants $2\phi_1$ and $2\phi_2$, respectively. We get the bound by using Lemma 4.5; q satisfies the oversampling condition with $\phi = \frac{\sqrt{\phi_1\phi_2}\|X\|_{\mathrm{F}}\|Y\|_{\mathrm{F}}}{\sum_\ell \|X(\ell,\cdot)\|\|Y(\ell,\cdot)\|}$, using the inequality of arithmetic and geometric means:

$$\begin{aligned}
\frac{1}{q(i)}\frac{\|X(i,\cdot)\|\|Y(i,\cdot)\|}{\sum_\ell \|X(\ell,\cdot)\|\|Y(\ell,\cdot)\|} &= \frac{2}{q_1(i)+q_2(i)}\frac{\|X(i,\cdot)\|\|Y(i,\cdot)\|}{\sum_\ell \|X(\ell,\cdot)\|\|Y(\ell,\cdot)\|} \\
&\leq \frac{1}{\sqrt{q_1(i)q_2(i)}}\frac{\|X(i,\cdot)\|\|Y(i,\cdot)\|}{\sum_\ell \|X(\ell,\cdot)\|\|Y(\ell,\cdot)\|} \\
&\leq \frac{\sqrt{\phi_1\phi_2}\|X\|_{\mathrm{F}}\|Y\|_{\mathrm{F}}}{\|X(i,\cdot)\|\|Y(i,\cdot)\|}\frac{\|X(i,\cdot)\|\|Y(i,\cdot)\|}{\sum_\ell \|X(\ell,\cdot)\|\|Y(\ell,\cdot)\|} \\
&= \frac{\sqrt{\phi_1\phi_2}\|X\|_{\mathrm{F}}\|Y\|_{\mathrm{F}}}{\sum_\ell \|X(\ell,\cdot)\|\|Y(\ell,\cdot)\|}.
\end{aligned}$$

□

Remark 4.7. Lemma 4.6 implies that, given $\mathrm{SQ}_{\phi_1}(X)$ and $\mathrm{SQ}_{\phi_2}(Y)$, we can get $\mathrm{SQ}_\phi(M)$ for M a sufficiently good approximation to $X^\dagger Y$, with $\phi \leq \phi_1\phi_2\frac{\|X\|_{\mathrm{F}}^2\|Y\|_{\mathrm{F}}^2}{\|M\|_{\mathrm{F}}^2}$. This is an approximate closure property for oversampling and query access under matrix products.

Given the above types of accesses, we can compute the sketch S necessary for Lemma 4.6 by taking $p = \mathcal{D}_{\tilde{x}}$ and $q = \mathcal{D}_{\tilde{y}}$), thereby finding a desired $M \coloneqq X^\dagger S^\dagger S Y$. We can compute entries of M with only r queries each to X and Y, so all we need is to get $\mathrm{SQ}(\tilde{M})$ for $\tilde{M}$ the appropriate bound. We choose $|\tilde{M}(i,j)|^2 \coloneqq r\sum_{\ell=1}^r |[S\tilde{X}](\ell,i)^\dagger [S\tilde{Y}](\ell,j)|^2$; showing that we have $\mathrm{SQ}(M)$ follows from the proofs of Lemmas 3.8 and 3.9, since M is simply a linear combination of outer products of rows

of $\tilde{X}$ with rows of $\tilde{Y}$. Finally, this bound has the appropriate norm. Notating the rows sampled by the sketch as $s_1, \ldots, s_r$, we have

$$\|\tilde{M}\|_{\mathrm{F}}^2 = r\sum_{\ell=1}^{r} \|[S\tilde{X}](\ell,\cdot)\|^2\|[S\tilde{Y}](\ell,\cdot)\|^2 = r\sum_{\ell=1}^{r} \frac{\|\tilde{X}(s_\ell,\cdot)\|^2\|\tilde{Y}(s_\ell,\cdot)\|^2}{r^2(\frac{\|\tilde{X}(s_\ell,\cdot)\|^2}{2\|\tilde{X}\|_{\mathrm{F}}^2} + \frac{\|\tilde{Y}(s_\ell,\cdot)\|^2}{2\|\tilde{Y}\|_{\mathrm{F}}^2})^2}$$
$$\leq \sum_{\ell=1}^{r} \frac{\|\tilde{X}(s_\ell,\cdot)\|^2\|\tilde{Y}(s_\ell,\cdot)\|^2}{r(\frac{\|\tilde{X}(s_\ell,\cdot)\|\|\tilde{Y}(s_\ell,\cdot)\|}{\|\tilde{X}\|_{\mathrm{F}}\|\tilde{Y}\|_{\mathrm{F}}})^2} = \|\tilde{X}\|_{\mathrm{F}}^2\|\tilde{Y}\|_{\mathrm{F}}^2 = \phi_1\phi_2\|X\|_{\mathrm{F}}^2\|Y\|_{\mathrm{F}}^2.$$

If $X = Y$, we can get an improved spectral norm bound: instead of depending on $\|X\|_{\mathrm{F}}^2$, error depends on $\|X\|\|X\|_{\mathrm{F}}$.

Lemma 4.8 (Approximating matrix multiplication to spectral norm error [RV07, Theorem 3.1])**.** *Suppose we are given $A \in \mathbb{R}^{m\times n}, \varepsilon > 0, \delta \in [0,1]$, and $S \in \mathbb{R}^{r\times n}$ a ϕ-oversampled importance sampling sketch of A. Then*

$$\Pr\left[\|A^\dagger S^\dagger SA - A^\dagger A\| \lesssim \sqrt{\frac{\phi^2 \log r \log 1/\delta}{r}}\|A\|\|A\|_{\mathrm{F}}\right] > 1-\delta.$$

The above results can be used to approximate singular values, simply by directly translating the bounds on matrix product error to bounds on singular value error.

Lemma 4.9 (Approximating singular values)**.** *Given $\mathrm{SQ}_\phi(A) \in \mathbb{C}^{m\times n}$ and $\varepsilon \in (0,1]$, we can form importance sampling sketches $S \in \mathbb{R}^{r\times m}$ and $T^\dagger \in \mathbb{R}^{c\times n}$ in $\mathcal{O}\big((r+c)\,\mathbf{sq}_\phi(A)\big)$ time satisfying the following property. Take $r, c \geq s$ for some sufficiently large $s = \widetilde{\mathcal{O}}\Big(\frac{\phi^2}{\varepsilon^2}\log\frac{1}{\delta}\Big)$. Then, if σ_i and $\hat{\sigma}_i$ are the singular values of A and SAT, respectively (where $\hat{\sigma}_i = 0$ for $i > \min(r,c)$), we have with probability $\geq 1-\delta$ that*

$$\sqrt{\sum_{i=1}^{\min(m,n)} (\hat{\sigma}_i^2 - \sigma_i^2)^2} \leq \varepsilon\|A\|_{\mathrm{F}}^2.$$

If we additionally assume that $\varepsilon \lesssim \|A\|/\|A\|_{\mathrm{F}}$, we can conclude $|\sigma_i^2 - \hat{\sigma}_i^2| \leq \varepsilon\|A\|\|A\|_{\mathrm{F}}$ for all i.

This result follows from results bounding the error between singular values by errors of matrix products. For notation, let $\sigma_i(M)$ be the ith largest singular value of M. We will use the following inequalities relating norm error of matrices to error in their singular values:

Lemma 4.10 (Hoffman-Wielandt inequality [KV17, Lemma 2.7])**.** *For symmetric $X, Y \in \mathbb{R}^{n\times n}$,*

$$\sum |\sigma_i(X) - \sigma_i(Y)|^2 \leq \|X - Y\|_{\mathrm{F}}^2.$$

Lemma 4.11 (Weyl's inequality [Bha97, Corollary III.2.2])**.** *For $A, B \in \mathbb{C}^{m\times n}$, $|\sigma_k(A) - \sigma_k(B)| \leq \|A - B\|$. When A, B are Hermitian, the same bound holds for their eigenvalues.*[13]

Proof of Lemma 4.9. We use known theorems, plugging in the values of r and c. Using Lemma 4.6 for the sketch S, we know that

$$\Pr\left[\|A^\dagger S^\dagger SA - A^\dagger A\|_{\mathrm{F}} \leq \frac{\varepsilon}{2}\|A\|_{\mathrm{F}}^2\right] \geq 1-\delta;$$

[13] [Bha97, Corollary III.2.2] actually proves the Hermitian version. The result about singular values is an easy consequence, see for example the blog of Terence Tao [Tao10, Exercise 22(iv)].

by Lemma 4.3, $T^\dagger$ is an $\leq 2\phi$-oversampled importance sampling sketch of $(SA)^\dagger$, so by Lemma 4.6 for $T^\dagger$,

$$\Pr\left[\|SATT^\dagger A^\dagger S^\dagger - SAA^\dagger S^\dagger\|_{\mathrm{F}} \leq \frac{\varepsilon}{4}\|SA\|_{\mathrm{F}}^2\right] \geq 1-\delta,$$

and from Lemma 4.2,

$$\Pr\left[\|SA\|_{\mathrm{F}}^2 \leq 2\|A\|_{\mathrm{F}}^2\right] \geq 1-\delta.$$

By rescaling δ and union bounding, we can have all events happen with probability $\geq 1-\delta$. Then, from triangle inequality followed by Lemma 4.10,

$$\begin{aligned}\sqrt{\sum|\sigma_i(SAT)^2 - \sigma_i(A)^2|^2} &\leq \sqrt{\sum|\sigma_i(SAT)^2 - \sigma_i(SA)^2|^2} + \sqrt{\sum|\sigma_i(SA)^2 - \sigma_i(A)^2|^2} \\ &\leq \|(SAT)(SAT)^\dagger - (SA)(SA)^\dagger\|_{\mathrm{F}} + \|(SA)^\dagger(SA) - A^\dagger A\|_{\mathrm{F}} \\ &\leq \varepsilon\|A\|_{\mathrm{F}}^2.\end{aligned}$$

The analogous result holds for spectral norm via Lemma 4.8 and Lemma 4.11; the only additional complication is that we need to assert that $\|SA\| \lesssim \|A\|$. We use the following argument, using the upper bound on ε:

$$\|SA\|^2 = \|A^\dagger S^\dagger SA\| \leq \|A^\dagger S^\dagger SA - A^\dagger A\| + \|A^\dagger A\| \leq \|A\|^2 + \varepsilon\|A\|\|A\|_{\mathrm{F}} \lesssim \|A\|^2. \qquad \square$$

Finally, if we wish to approximate a vector inner product $u^\dagger v$, a special case of matrix product, we can do so with only sampling and query access to one of the vectors while still getting $\log\frac{1}{\delta}$ dependence on failure probability. The proof of this statement is in Appendix B.

Lemma 4.12 (Inner product estimation, [Tan19, Proposition 4.2])**.** *Given* $\mathrm{SQ}_\phi(u), \mathrm{Q}(v) \in \mathbb{C}^n$*, we can output an estimate* $c \in \mathbb{C}$ *such that* $|c - \langle u, v\rangle| \leq \varepsilon$ *with probability* $\geq 1-\delta$ *in time* $\mathcal{O}\big(\phi\|u\|^2\|v\|^2\frac{1}{\varepsilon^2}\log\frac{1}{\delta}(\mathbf{sq}_\phi(u) + \mathbf{q}(v))\big)$.

Remark 4.13. Lemma 4.12 also applies to higher-order tensor inner products:

(a) (Trace inner products, [GLT18, Lemma 11]) Given $\mathrm{SQ}_\phi(A) \in \mathbb{C}^{n\times n}$ and $\mathrm{Q}(B) \in \mathbb{C}^{n\times n}$, we can estimate $\mathrm{Tr}[AB^\dagger]$ to additive error ε with probability at least $1-\delta$ by using

$$\mathcal{O}\left(\phi\frac{\|A\|_{\mathrm{F}}^2\|B\|_{\mathrm{F}}^2}{\varepsilon^2}\big(\mathbf{sq}_\phi(A) + \mathbf{q}(B)\big)\log\frac{1}{\delta}\right)$$

time. To do this, note that $\mathrm{SQ}_\phi(A)$ and $\mathrm{Q}(B)$ imply $\mathrm{SQ}_\phi(\mathrm{vec}(A))$ and $\mathrm{Q}(\mathrm{vec}(B))$. $\mathrm{Tr}[AB] = \langle\mathrm{vec}(B), \mathrm{vec}(A)\rangle$, so we can just apply Lemma 4.12 to conclude.

(b) (Expectation values) Given $\mathrm{SQ}_\phi(A) \in \mathbb{C}^{n\times n}$ and $\mathrm{Q}(x), \mathrm{Q}(y) \in \mathbb{C}^n$, we can estimate $x^\dagger Ay$ to additive error ε with probability at least $1-\delta$ in

$$\mathcal{O}\left(\phi\frac{\|A\|_{\mathrm{F}}^2\|x\|^2\|y\|^2}{\varepsilon^2}\big(\mathbf{sq}_\phi(A) + \mathbf{q}(x) + \mathbf{q}(y)\big)\log\frac{1}{\delta}\right)$$

time. To do this, observe that $x^\dagger Ay = \mathrm{Tr}(x^\dagger Ay) = \mathrm{Tr}(Ayx^\dagger)$ and that $\mathrm{Q}(yx^\dagger)$ can be simulated with $\mathrm{Q}(x), \mathrm{Q}(y)$. So, we just apply the trace inner product procedure.

Finally, we observe a simple technique to convert importance sampling sketches into approximate isometries, by inserting the appropriate pseudoinverse. This will be used in some of the more involved applications.

Lemma 4.14. *Given $A \in \mathbb{C}^{m\times n}$, $S \in \mathbb{C}^{r\times m}$ sampled from a ϕ-oversampled importance sampling distribution of A, and $T^\dagger \in \mathbb{C}^{n\times c}$ sampled from an $\le \phi$-oversampled importance sampling distribution of $(SA)^\dagger$, let $R \coloneqq SA$ and $C \coloneqq SAT$. Let σ_k be the kth singular value of A. If, for $\alpha \in (0,1]$, $r = \tilde\Omega(\frac{\phi^2\|A\|^2\|A\|_{\mathrm F}^2}{\sigma_k^4}\log\frac{1}{\delta})$ and $c = \tilde\Omega(\frac{\phi^2\|A\|^2\|A\|_{\mathrm F}^2}{\sigma_k^4\alpha^2}\log\frac{1}{\delta})$, then with probability $\ge 1-\delta$, $((C_k)^+R)^\dagger$ is an α-approximate projective isometry onto the image of $(C_k)^+$. Further, $(DV^\dagger R)^\dagger$ is an α-approximate isometry, where $C_k^+ = UDV^\dagger$ is a singular value decomposition truncated so that $D \in \mathbb{R}^{k'\times k'}$ is full rank (so $k' \le \min(k, \operatorname{rank}(A))$).*

Proof. The following occurs with probability $\ge 1-\delta$. By Lemma 4.9, $\|C_k^+\| \lesssim \frac{1}{\sigma_k^2}$. By Lemma 4.8, $\|R^\dagger R - A^\dagger A\| \lesssim \|A\|^2$, which implies that $\|R\| \lesssim \|A\|$, and by Lemma 4.2, $\|R\|_{\mathrm F} \lesssim \|A\|_{\mathrm F}$. Further, $\|RR^\dagger - CC^\dagger\| \le \alpha\sigma_k^2\frac{\|R\|\|R\|_{\mathrm F}}{\|A\|\|A\|_{\mathrm F}} \lesssim \alpha\sigma_k^2$. Finally, $C_k^+C = C_k^+C_k$ is an orthogonal projector. So, with probability $\ge 1-\delta$,

$$\|(C_k^+R)(C_k^+R)^\dagger - (C_k^+C)(C_k^+C)^\dagger\| = \|C_k^+(RR^\dagger - CC^\dagger)(C_k^+)^\dagger\| \le \|C_k^+\|^2\|RR^\dagger - CC^\dagger\| = O(\alpha).$$

We get the computation for the α-approximate isometry by restricting attention to the span of U:

$$\|(DV^\dagger R)(DV^\dagger R)^\dagger - I\| = \|DV^\dagger(RR^\dagger - CC^\dagger)VD^\dagger\| \le \|UDV^\dagger\|^2\|RR^\dagger - CC^\dagger\| = O(\alpha). \qquad \square$$

One can also observe that, for a sufficiently good sketch C, $R \approx C_k(C_k)^+R$ in spectral norm, giving a generic way to approximate a sketch R by a product of a small matrix with an approximate projective isometry. We do not need it in our proofs, so this computation is not included.

5 Singular value transformation

Our main result is that, given $\mathrm{SQ}_\phi(A)$ and a smooth function f, we can approximate $f(A^\dagger A)$ by a decomposition $R^\dagger UR + f(0)I$. This primitive is based on the even singular value transformation used by Gilyén, Su, Low, and Wiebe [GSLW19].

Theorem 5.1 (Even singular value transformation)**.** *Let $A \in \mathbb{C}^{m\times n}$ and $f\colon \mathbb{R}^+ \to \mathbb{C}$ be such that $f(x)$ and $\bar f(x) \coloneqq (f(x)-f(0))/x$ are L-Lipschitz and $\bar L$-Lipschitz, respectively, on $\cup_{i=1}^{\min(m,n)}[\sigma_i^2 - d, \sigma_i^2 + d]$ for some $d > 0$. Take parameters ε and δ such that $0 < \varepsilon \lesssim \min(L\|A\|_*^2, \bar L\|A\|_*^2\|A\|^2)$ and $\delta \in (0,1]$. Choose a norm $* \in \{\mathrm F, \mathrm{Op}\}$.*

Suppose we have $\mathrm{SQ}_\phi(A)$. Consider the importance sampling sketch $S \in \mathbb{R}^{r\times m}$ corresponding to $\mathrm{SQ}_\phi(A)$ and the importance sampling sketch $T^\dagger \in \mathbb{R}^{c\times n}$ corresponding to $\mathrm{SQ}_{\le 2\phi}((SA)^\dagger)$ (which we have by Lemma 4.3). Then, for $R \coloneqq SA$ and $C \coloneqq SAT$, we can achieve the bound

$$\Pr\left[\|R^\dagger\bar f(CC^\dagger)R + f(0)I - f(A^\dagger A)\|_* > \varepsilon\right] < \delta, \tag{3}$$

if $r, c > \|A\|^2\|A\|_{\mathrm F}^2\phi^2\frac{1}{d^2}\log\frac{1}{\delta}$ (or, equivalently, $d > \bar\varepsilon \coloneqq \|A\|_*\|A\|_{\mathrm F}(\frac{\phi^2\log(1/\delta)}{\min(r,c)})^{1/2}$) and

$$r = \tilde\Omega\Big(\phi^2L^2\|A\|_*^2\|A\|_{\mathrm F}^2\frac{1}{\varepsilon^2}\log\frac{1}{\delta}\Big) \qquad c = \tilde\Omega\Big(\phi^2\bar L^2\|A\|^4\|A\|_*^2\|A\|_{\mathrm F}^2\frac{1}{\varepsilon^2}\log\frac{1}{\delta}\Big). \tag{4}$$

First, we make some technical remarks. The assumption that $\varepsilon \lesssim L\|A\|_*^2$ is for non-degeneracy: if $\varepsilon \ge L\|A\|^2$, then the naive approximation $f(0)I$ of $f(A^\dagger A)$ would suffice, since $\|f(0)I - f(A^\dagger A)\| \le L\|A\|^2 \le \varepsilon$ as desired.[14] The parameter d (or, rather, the parameter $\bar\varepsilon$) specifies the domain where

[14]The choice $f(0)I$ assumes that f is Lipschitz on $\{0, \|A\|^2\}$. More generally, we can choose $f(x)I$ for any $x \in \cup_{i=1}^{\min(m,n)}[\sigma_i^2 - d, \sigma_i^2 + d]$ in order to get a sufficiently good naive approximation.

$f(x)$ and $\bar{f}(x)$ should be smooth: the condition in the theorem is that they should be Lipschitz on the spectrum of $A^\dagger A$, with $\bar{\varepsilon}$ room for approximation. This will not come into play often, though, since we can often design our singular value transforms such that we can take $d = \infty$. For example, if our desired transform f becomes non-smooth outside the relevant interval $[0, \|A\|^2]$, we can apply Theorem 5.1 with $d = \infty$ and the function g such that $g(x) = g(\|A\|^2)$ for $x \geq \|A\|^2$ and $g(x) = f(x)$ otherwise. Then $g(A^\dagger A) = f(A^\dagger A)$ and g is smooth everywhere, so we do not need to worry about the d parameter. Finally, we note that no additional log terms are necessary (i.e., $\tilde{\Omega}$ becomes Ω) when the Frobenius norm is used.

By our discussion in Section 4, finding the sketches S and T for Theorem 5.1 takes time $\mathcal{O}\big((r+c)\,\mathbf{s}_\phi(A) + rc\,\mathbf{q}_\phi(A) + \mathbf{n}_\phi(A)\big)$, querying for all of the entries of C takes additional time $\mathcal{O}(rc\,\mathbf{q}(A))$, and computing $\bar{f}(CC^\dagger)$ takes additional time $\mathcal{O}\big(\min(r^2c, rc^2)\big)$ (if done naively). For our applications, this final matrix function computation will dominate the runtime, and the rest of the cost we will treat as $\mathcal{O}\big(rc\,\mathbf{sq}_\phi(A)\big)$.

For some intuition on error bounds and time complexity, we consider how the parameters in our main theorem behave in a restricted setting: suppose we have $\mathrm{SQ}(A)$ with minimum singular value σ and such that $\|A\|_\mathrm{F}/\sigma$ is dimension-independent.[15] This condition simultaneously bounds the rank and condition number of A. Further suppose[16] that f is L-Lipschitz on the interval $[0, \|A\|^2]$ and satisfies

$$L\|A\|^2 < \Gamma D \text{ where } D \coloneqq \max_{x\in[0,\|A\|^2]} f(x) - \min_{y\in[0,\|A\|^2]} f(y),$$

for some dimension-independent Γ. Γ must be at least one, so we can think about such an f as being at most Γ times "steeper" compared to the least possible "steepness". Under these assumptions, we can get a decomposition satisfying

$$\|R^\dagger \bar{f}(CC^\dagger)R + f(0)I - f(A^\dagger A)\| > \varepsilon D$$

with probability $\geq 1-\delta$ by taking

$$r = \tilde{\Theta}\Big(\Gamma^2 \frac{\|A\|_\mathrm{F}^2}{\|A\|^2}\frac{1}{\varepsilon^2}\log\frac{1}{\delta}\Big) \text{ and } c = \tilde{\Theta}\Big(\Gamma^2 \frac{\|A\|^2\|A\|_\mathrm{F}^2}{\sigma^4}\frac{1}{\varepsilon^2}\log\frac{1}{\delta}\Big).$$

The time to compute the decomposition is

$$\widetilde{\mathcal{O}}\Big(\frac{\|A\|_\mathrm{F}^6}{\|A\|^2\sigma^4}\frac{\Gamma^6}{\varepsilon^6}\log^3\frac{1}{\delta}\Big).$$

These quantities are all dimensionless. Dependence on σ arises because we bound $\bar{L} \leq L/\sigma^2$: our algorithm's dependence on $\bar{L}$ implicitly enforces a low-rank constraint in this case. All of our analyses give qualitatively similar results to this, albeit in more general settings allowing approximately low-rank input.

To perform error analyses, we will need bounds on the norms of the matrices in our decomposition. The following lemma gives the bounds we need for Section 6.

Lemma 5.2 (Norm bounds for even singular value transformation)**.** *Suppose the assumptions from Theorem 5.1 hold. Then with probability at least* $1-\delta$*, the event in Eq. (3) occurs (that is,*

[15] By a dimension-independent or dimensionless quantity, we mean a quantity that is both independent of the size of the input matrix and is scale-invariant, i.e., does not change under scaling $A \leftarrow \alpha A$.

[16] This criterion is fairly reasonable. For example, the polynomials used in QSVT satisfy it.

$R^\dagger \bar{f}(CC^\dagger)R \approx f(A^\dagger A) - f(0)I)$ *and moreover,* the following bounds also hold*:*

$$\|R\| = \mathcal{O}(\|A\|) \quad \textit{and} \quad \|R\|_{\mathrm{F}} = \mathcal{O}(\|A\|_{\mathrm{F}}), \tag{5}$$

$$\|\bar{f}(CC^\dagger)\| \leq \max\left\{|\bar{f}(x)| \,\middle|\, x \in \bigcup_{i=1}^{\min(r,c)} [\sigma_i^2 - \bar{\varepsilon}, \sigma_i^2 + \bar{\varepsilon}]\right\}, \tag{6}$$

$$\textit{when } * = \mathrm{Op}, \left\|R^\dagger \sqrt{\bar{f}(CC^\dagger)}\right\| \leq \sqrt{\|f(A^\dagger A) - f(0)I\| + \varepsilon}. \tag{7}$$

Eq. (7) is typically a better bound than combining Eqs. (5) and (6). For intuition, notice this is true if $\varepsilon, \bar{\varepsilon} = 0$: the left-hand and right-hand sides of the following inequality are the two ways to bound $\|R^\dagger \sqrt{\bar{f}(CC^\dagger)}\|^2$, up to constant factors (σ below runs over the singular values of A):

$$\|f(A^\dagger A) - f(0)I\| \leq \max_\sigma |f(\sigma^2) - f(0)| \leq \max_\sigma \sigma^2 \max_\sigma \frac{|f(\sigma^2) - f(0)|}{\sigma^2} = \|A\|^2 \max_\sigma |\bar{f}(\sigma^2)|.$$

The rest of this section will be devoted to proving Theorem 5.1 and Lemma 5.2. A mathematical tool we will need is a matrix version of the defining inequality of L-Lipschitz functions, $|f(x) - f(y)| \leq L|x - y|$ when f is L-Lipschitz. The Frobenius norm version of this bound (Lemma 5.3) follows by computing matrix derivatives; the spectral norm version (Lemma 5.4) has a far less obvious proof.

Lemma 5.3 ([Gil10, Corollary 2.3])**.** *Let A and B be Hermitian matrices and let $f\colon \mathbb{R} \to \mathbb{C}$ be L-Lipschitz continuous on the eigenvalues of A and B. Then $\|f^{(\mathrm{EV})}(A) - f^{(\mathrm{EV})}(B)\|_{\mathrm{F}} \leq L\|A - B\|_{\mathrm{F}}$.*

Lemma 5.4 ([AP11, Theorem 11.2])**.** *Let A and B be Hermitian matrices and let $f\colon \mathbb{R} \to \mathbb{C}$ be L-Lipschitz continuous on the eigenvalues of A and B. Then*

$$\left\|f^{(\mathrm{EV})}(A) - f^{(\mathrm{EV})}(B)\right\| \lesssim L\|A - B\| \log \min(\operatorname{rank} A, \operatorname{rank} B).$$

Proof of Theorem 5.1 and Lemma 5.2. Since $g(A^\dagger A) = f(A^\dagger A) + g(0)I$ for $f(x) \coloneqq g(x) - g(0)$, we can assume without loss of generality that $f(0) = 0$. As a reminder, in the statement of Theorem 5.1 we take

$$r = \tilde{\Omega}\Big(\phi^2 L^2 \|A\|_*^2 \|A\|_{\mathrm{F}}^2 \frac{1}{\varepsilon^2} \log \frac{1}{\delta}\Big) \qquad c = \tilde{\Omega}\Big(\phi^2 \bar{L}^2 \|A\|^4 \|A\|_*^2 \|A\|_{\mathrm{F}}^2 \frac{1}{\varepsilon^2} \log \frac{1}{\delta}\Big).$$

These values are chosen such that the following holds with probability $\geq 1 - \delta$ simultaneously.

1. The ith singular value of $CC^\dagger$ does not differ from the ith singular value of $A^\dagger A$ by more than $\bar{\varepsilon}$. This follows from Lemma 4.9 with error parameter $\varepsilon \|A\|_{\mathrm{F}}^{-1} \|A\|_*^{-1} \max(L^{-1}, \bar{L}^{-1}\|A\|^{-2})$. This immediately implies Eq. (6).

2. $\|R\|^2 = \mathcal{O}\big(\|A\|^2\big)$. This is the spectral norm bound in Eq. (5) (the Frobenius norm bound follows from Lemma 4.2). We use Lemma 4.8:

$$\|R\|^2 \leq \|A\|^2 + \|R^\dagger R - A^\dagger A\| \leq \|A\|^2 + \frac{\varepsilon \|A\|^2}{L\|A\|_*^2} = O(\|A\|^2).$$

3. $\|f(R^\dagger R) - f(A^\dagger A)\|_* = \mathcal{O}(\varepsilon)$. We need the polylog factors in our number of samples to deal with the $\log r$ that arises from Lemma 5.4 in the spectral norm case.

$$\begin{aligned}
&\|f(R^\dagger R) - f(A^\dagger A)\|_* \\
&\lesssim L\|R^\dagger R - A^\dagger A\|_* \log \operatorname{rank}(R^\dagger R) && \text{(Lemma 5.4 or Lemma 5.3)} \\
&\lesssim L\sqrt{\frac{\phi^2 \log r \log(1/\delta)}{r}} \|A\|_* \|A\|_{\mathrm{F}} \log r && \text{(Lemma 4.8 or Lemma 4.6)} \\
&\lesssim \varepsilon. && \text{(plugging in value for } r\text{)}
\end{aligned}$$

4. $\|\bar{f}(CC^\dagger) - \bar{f}(RR^\dagger)\|_* = \mathcal{O}\big(\varepsilon/\|A\|^2\big)$. This follows similarly to the above point.

When all of the above bounds hold, we can conclude:

$$\begin{aligned}
&\|R^\dagger \bar{f}(CC^\dagger)R - f(A^\dagger A)\|_* \\
&\leq \|R^\dagger \bar{f}(RR^\dagger)R - f(A^\dagger A)\|_* + \|R^\dagger(\bar{f}(RR^\dagger) - \bar{f}(CC^\dagger))R\|_* \\
&= \|f(R^\dagger R) - f(A^\dagger A)\|_* + \|R^\dagger(\bar{f}(RR^\dagger) - \bar{f}(CC^\dagger))R\|_* && \text{(Definition of } \bar{f}\text{)} \\
&\leq \|f(R^\dagger R) - f(A^\dagger A)\|_* + \|R\|^2 \|\bar{f}(RR^\dagger) - \bar{f}(CC^\dagger)\|_* \\
&\lesssim \varepsilon + \|R\|^2 \varepsilon/\|A\|^2 \\
&\lesssim \varepsilon.
\end{aligned}$$

This gives Eq. (3) after rescaling ε by an appropriate constant factor. When $* = \mathrm{Op}$, we also have Eq. (7), since

$$\left\|R^\dagger \sqrt{\bar{f}(CC^\dagger)}\right\| = \sqrt{\|R^\dagger \bar{f}(CC^\dagger)R\|} \leq \sqrt{\|f(A^\dagger A) - f(0)I\| + \varepsilon}. \qquad \square$$

We remark here that the log term in Lemma 5.4 unfortunately cannot be removed (because some Lipschitz functions are not operator Lipschitz). However, several bounds hold under various mild assumptions, and for particular functions, the log term can be improved to log log or completely removed. For example, the QSVT literature [GSLW19] cites the following result:

Lemma 5.5 ([AP10, Corollary 7.4])**.** *Let A and B be Hermitian matrices such that $aI \preceq A, B \preceq bI$, and let $f\colon \mathbb{R} \to \mathbb{C}$ be L-Lipschitz continuous on the interval $[a, b]$. Then*

$$\left\|f^{(\mathrm{EV})}(A) - f^{(\mathrm{EV})}(B)\right\| \lesssim L\|A - B\| \log\left(e\frac{b-a}{\|A-B\|}\right).$$

Though we will not use it, we can extend these results on eigenvalue transformation of Hermitian matrices to singular value transformation of general matrices via the reduction from [GSLW19, Corollary 21]. For example, Lemma 5.4 implies the following:

Lemma 5.6. *Let $A, B \in \mathbb{C}^{m\times n}$ be matrices and let $f\colon [0,\infty) \to \mathbb{C}$ be L-Lipschitz continuous on the singular values of A and B such that $f(0) = 0$. Then*

$$\|f^{(\mathrm{SV})}(A) - f^{(\mathrm{SV})}(B)\| \lesssim L\|A - B\| \log \min(\operatorname{rank} A, \operatorname{rank} B).$$

In Section 7, we prove results on generic singular value transformation and eigenvalue transformation by bootstrapping Theorem 5.1. Since these are slower, though, we will use primarily the even singular value transformation results that we just proved to recover "dequantized QML" results. This will be the focus of next section.

6 Applying the framework to dequantizing QML algorithms

Now, with our framework, we can recover previous dequantization results: recommendation systems (Section 6.2), supervised clustering (Section 6.3), principal component analysis (Section 6.4), low-rank matrix inversion (Section 6.5), support-vector machines (Section 6.6), and low-rank semidefinite programs (Section 6.8). We also propose new quantum-inspired algorithm for other applications, including QSVT (Section 6.1), Hamiltonian simulation (Section 6.7), and discriminant analysis

(Section 6.9). We give applications in roughly chronological order; this also happens to be a rough difficulty curve, with applications that follow more easily from our main results being first.

Everywhere it occurs, $K := \|A\|_{\mathrm{F}}^2/\sigma^2$, where A is the input matrix. $\kappa := \|A\|_2^2/\sigma^2$. For simplicity, we will often describe our runtimes as if we know spectral norms of input matrices (so, for example, we know κ). If we do not know the spectral norm, we can run Lemma 4.9 repeatedly with multiplicatively decreasing ε until we find a constant factor upper bound on the spectral norm, which suffices for our purposes. Alternatively, we can bound the spectral norm by the Frobenius norm, which we know from sampling and query access to input.

6.1 Dequantizing QSVT

We begin by dequantizing the quantum singular value transformation described by Gilyén, Su, Low, and Wiebe [GSLW19] for close-to-low-rank matrices.

Definition 6.1. For a matrix $A \in \mathbb{C}^{m\times n}$ and $p(x) \in \mathbb{C}[x]$ degree-d polynomial of parity-d (i.e., even if d is even and odd if d is odd), we define the notation $p^{(\mathrm{QV})}(A)$ in the following way:

1. If p is *even*, meaning that we can express $p(x) = q(x^2)$ for some polynomial $q(x)$, then
$$p^{(\mathrm{QV})}(A) := q(A^\dagger A) = p(\sqrt{A^\dagger A}).$$
2. If p is *odd*, meaning that we can express $p(x) = x \cdot q(x^2)$ for some polynomial $q(x)$, then
$$p^{(\mathrm{QV})}(A) := A \cdot q(A^\dagger A).$$

Theorem 6.2. *Suppose we are given a matrix $A \in \mathbb{C}^{m\times n}$ satisfying $\|A\|_{\mathrm{F}} = 1$ via the oracles for $\mathrm{SQ}(A)$ and $\mathrm{SQ}(A^\dagger)$ with $\mathbf{sq}(A), \mathbf{sq}(A^\dagger) = \mathcal{O}(\log(mn))$, a vector $\mathrm{SQ}(b) \in \mathbb{C}^n$ with $\|b\| = 1$ and $\mathbf{sq}(b) = \mathcal{O}(\log n)$, and a degree-$d$ polynomial $p(x)$ of parity-d such that $|p(x)| \le 1$ for all $x \in [-1, 1]$.*

Then with probability $\ge 1 - \delta$, for ε a sufficiently small constant, we can get $\mathrm{SQ}_\phi(v) \in \mathbb{C}^n$ such that $\|v - p^{(\mathrm{QV})}(A)b\| \le \varepsilon\|p^{(\mathrm{QV})}(A)b\|$ in $\mathrm{poly}\left(d, \frac{1}{\|p^{(\mathrm{QV})}(A)b\|}, \frac{1}{\varepsilon}, \frac{1}{\delta}, \log mn\right)$ time.

Specifically, for p even, the runtime is
$$\widetilde{\mathcal{O}}\left(\frac{d^{16}\|A\|^{10}}{(\varepsilon\|p^{(\mathrm{QV})}(A)b\|)^6}\log^3\frac{1}{\delta} + \frac{d^{12}\|A\|^8 + d^6\|A\|^2}{(\varepsilon\|p^{(\mathrm{QV})}(A)b\|)^4}\log^2\frac{1}{\delta}\log(mn)\right)$$
with
$$\widetilde{\mathbf{sq}}(v) = \widetilde{\mathcal{O}}\left(\frac{d^{12}\|A\|^4}{\varepsilon^4\|p^{(\mathrm{QV})}(A)b\|^6}\log(mn)\log^3\frac{1}{\delta}\right),$$
and for p odd, the runtime is
$$\widetilde{\mathcal{O}}\left(\frac{d^{22}\|A\|^{16}}{(\varepsilon\|p^{(\mathrm{QV})}(A)b\|)^6} + \frac{d^{16}\|A\|^{12} + d^{10}\|A\|^4\delta^{-1}}{(\varepsilon\|p^{(\mathrm{QV})}(A)b\|)^4}\log(mn)\right)$$
with
$$\widetilde{\mathbf{sq}}(v) = \widetilde{\mathcal{O}}\left(\frac{d^8}{\varepsilon^2\delta\|p^{(\mathrm{QV})}(A)b\|^4}\log(mn)\right).$$

From this result it follows that QSVT, as described in [GSLW19, Theorem 17], has no exponential speedup when the block-encoding of A comes from a quantum-accessible "QRAM" data structure as in [GSLW19, Lemma 50]. In the setting of QSVT, given A and b in QRAM, one can prepare $|b\rangle$ and construct a block-encoding for $A/\|A\|_{\mathrm{F}} = A$ in $\mathrm{polylog}(mn)$ time. Then one can apply (quantum) SVT by a degree-d polynomial on A and apply the resulting map to $|b\rangle$ with $d \cdot \mathrm{polylog}(mn)$ gates and finally project down to get the state $|p^{(\mathrm{QV})}(A)b\rangle$ with probability $\geq 1-\delta$ after $\Theta\big(\frac{1}{\|p^{(\mathrm{QV})}(A)b\|}\log\frac{1}{\delta}\big)$ iterations of the circuit. So, getting a sample from $|p^{(\mathrm{QV})}(A)b\rangle$ takes $\Theta\big(d\frac{1}{\|p^{(\mathrm{QV})}(A)b\|}\,\mathrm{polylog}(mn/\delta)\big)$ time. This circuit gives an exact outcome, possibly with some $\log(1/\varepsilon)$ factors representing the discretization error in truncating real numbers to finite precision (which we ignore, since we do not account for them in our classical algorithm runtimes).

Analogously, by Remark 3.11, having A and b in (Q)RAM implies having $\mathrm{SQ}(A)$ and $\mathrm{SQ}(b)$ with $\mathbf{sq}(A) = \mathcal{O}(\log mn)$ and $\mathbf{sq}(b) = \mathcal{O}(\log n)$. Since QSVT also needs to assume $\max_{x\in[-1,1]}|p(x)| \leq 1$, the classical procedure matches the assumptions for QSVT. Our algorithm runs only polynomially slower than the quantum algorithm, since the quantum runtime clearly depends on d, $\frac{1}{\|p^{(\mathrm{QV})}(A)b\|}$, and $\log(mn)$. We are exponentially slower in ε and δ (these errors are conflated for the quantum algorithm). However, this exponential advantage vanishes if the desired output is not a quantum state but some fixed value (or an estimate of one). In that case, the quantum algorithm must also pay $\frac{1}{\varepsilon}$ during the sampling or tomography procedures and the classical algorithm can boost a constant success probability to $\geq 1-\delta$, only paying a $\log\frac{1}{\delta}$ factor. Note that, unlike in the quantum output, we can query entries of the output, which a quantum algorithm cannot do without paying at least a $\frac{1}{\varepsilon}$ factor.

Theorem 6.2 also dequantizes QSVT for block-encodings of density operators when the density operator comes from some well-structured classical data. Indeed, [GSLW19, Lemma 45] assumes we can efficiently prepare a purification of the density operator ρ. The rough classical analogue is the assumption that we have sampling and query access to some $A \in \mathbb{C}^{m\times n}$ with $\rho = A^\dagger A$. Since $\mathrm{Tr}(\rho) = 1$, we have $\|A\|_{\mathrm{F}} = 1$. Then, $p^{(\mathrm{QV})}(\rho) = r^{(\mathrm{QV})}(A)$ for $r(x) = p(x^2)$ and $\|\rho\| = \|A\|^2$, so we can repeat the above argument to show the lack of exponential speedup for this input model too.

We can mimic the quantum algorithm with our techniques because low-degree polynomials are smooth, in the sense that we formalize with the following lemma (proven in Appendix B).

Lemma 6.3. *Consider $p(x)$ a degree-d polynomial of parity-d such that $|p(x)| \leq 1$ for $x \in [-1,1]$. Recall that, for a function $f : \mathbb{C} \to \mathbb{C}$, we define $\bar{f}(x) := (f(x) - f(0))/x$ (and $\bar{f}(0) = f'(0)$ when f is differentiable at zero).*

- *If p is even, then $\max_{x\in[0,1]}|q(x)| \leq 1$, $\max_{x\in[-1,1]}|q'(x)| \lesssim d^2$, $\max_{x\in[-1,1]}|\bar{q}(x)| \lesssim d^2$, and $\max_{x\in[-1,1]}|\bar{q}'(x)| \lesssim d^4$.*
- *If p is odd, then $\max_{x\in[-1,1]}|q(x)| \lesssim d$, $\max_{x\in[-1,1]}|q'(x)| \lesssim d^3$, $\max_{x\in[-1,1]}|\bar{q}(x)| \lesssim d^3$, and $\max_{x\in[-1,1]}|\bar{q}'(x)| \lesssim d^5$.*

These bounds are tight for Chebyshev polynomials. In general, these bounds can be loose, so for any particular QML application we recommend using our main results for faster algorithms.

Proof of Theorem 6.2. Consider the even case: take $p(x) = q(x^2)$ for q a degree-$d/2$ polynomial, so $p^{(\mathrm{QV})}(A) = q(A^\dagger A)$, and we have the correct form to apply Theorem 5.1. q is uncontrolled outside of $[-1,1]$, so we instead apply the singular value transformation which is constant outside of $[-1,1]$:

$$f(x) := \begin{cases} q(-1) & x \leq -1 \\ q(x) & -1 \leq x \leq 1 \\ q(1) & 1 \leq x \end{cases}.$$

We can do this because the singular values of A lie in $[0,1]$, so $q(A^\dagger A) = f(A^\dagger A)$. Then, by Lemma 6.3, f and $\bar{f}$ are Lipschitz with $L = \mathcal{O}(d^2), \bar{L} = \mathcal{O}(d^4)$. So, by Theorem 5.1, we can get $R \in \mathbb{C}^{r\times n}$ and $C \in \mathbb{C}^{r\times c}$ such that $\|R^\dagger \bar{f}(CC^\dagger)R + f(0)I - f(A^\dagger A)\| \le \varepsilon$, where

$$r = \widetilde{\mathcal{O}}\left(d^4\|A\|^2\|A\|_{\mathrm{F}}^2\frac{1}{\varepsilon^2}\log\frac{1}{\delta}\right) \qquad \text{and} \qquad c = \widetilde{\mathcal{O}}\left(d^8\|A\|^6\|A\|_{\mathrm{F}}^2\frac{1}{\varepsilon^2}\log\frac{1}{\delta}\right).$$

(We will later rescale ε; note that $\varepsilon\|p^{(\mathrm{QV})}(A)b\| \lesssim L\|A\|^2\|b\|$, so ε is small enough for the theorem assumption.) This reduces the problem to approximating $R^\dagger\bar{f}(CC^\dagger)Rb + f(0)b$. We further approximate $Rb \approx u \in \mathbb{C}^r$ such that $\|Rb - u\| \le \varepsilon/d$. Using Lemma 4.6, this needs $\mathcal{O}\left(\|A\|_{\mathrm{F}}^2\|b\|^2\frac{d^2}{\varepsilon^2}\log\frac{1}{\delta}\right)$ samples, which can be done in time $\mathcal{O}\left(\|A\|_{\mathrm{F}}^2\|b\|^2\frac{d^2}{\varepsilon^2}r\log(mn)\log\frac{1}{\delta}\right)$, using that $\|R\|_{\mathrm{F}} \lesssim \|A\|_{\mathrm{F}}$ (Eq. (5)) and $\mathbf{sq}(R^\dagger) = \mathcal{O}(r\,\mathbf{sq}(A))$ (Lemma 4.3). This suffices to maintain the error bound because (using Eqs. (6) and (7) and Lemma 6.3),

$$\begin{aligned}\|R^\dagger\bar{f}(CC^\dagger)(Rb-u)\| &\le \left\|R^\dagger\sqrt{\bar{f}(CC^\dagger)}\right\|\left\|\sqrt{\bar{f}(CC^\dagger)}\right\|\|Rb-u\| \\ &\le \sqrt{\|f(A^\dagger A) - f(0)I\| + \varepsilon}\sqrt{\max_x \bar{f}(x)}\frac{\varepsilon}{d} \le \sqrt{2+\varepsilon}d\frac{\varepsilon}{d} \lesssim \varepsilon.\end{aligned}$$

As a consequence, $v \coloneqq R^\dagger\bar{f}(CC^\dagger)u + f(0)b$ satisfies $\|v - p^{(\mathrm{QV})}(A)b\| \le \varepsilon$. Via Lemma 3.6, we can get $\mathrm{SQ}_\phi(v)$ with

$$\begin{aligned}\widetilde{\mathbf{sq}}(v) &= \phi\,\mathrm{SQ}_\phi(v)\log\frac{1}{\delta} \\ &= \left((r+1)\frac{\|R\|_{\mathrm{F}}^2\|\bar{f}(CC^\dagger)u\|^2 + p(0)^2\|b\|^2}{\|v\|^2}\right)\left((r+1)\log(mn)\right)\log\frac{1}{\delta} \\ &\lesssim r^2\frac{\|\bar{f}(CC^\dagger)\|^2(\|Rb\| + \|Rb-u\|)^2 + p(0)^2}{(\|p^{(\mathrm{QV})}(A)b\| - \|v - p^{(\mathrm{QV})}(A)\|)^2}\log(mn)\log\frac{1}{\delta} \\ &\lesssim r^2\frac{d^4(1+\varepsilon/d)^2 + 1}{(\|p^{(\mathrm{QV})}(A)b\| - \varepsilon)^2}\log\frac{1}{\delta} \\ &\lesssim \frac{r^2d^4}{\|p^{(\mathrm{QV})}(A)b\|^2}\log(mn)\log\frac{1}{\delta}.\end{aligned}$$

In the last step, we use that $\varepsilon \lesssim \|p^{(\mathrm{QV})}(A)b\|$; if we do not have that assumption, $\widetilde{\mathbf{sq}}(v) \lesssim \frac{r^2d^4}{\|v\|^2}\log(mn)\log\frac{1}{\delta}$. We rescale $\varepsilon \leftarrow \varepsilon\|p^{(\mathrm{QV})}(A)b\|$ to get the desired bound. The runtime is dominated by finding C in $\mathcal{O}(rc\log(mn))$ time, computing $\bar{f}(CC^\dagger)$ in $\mathcal{O}(r^2c)$ time, and estimating $R^\dagger b$ in $\mathcal{O}\left(r\frac{d^2}{\varepsilon^2}\log(mn)\log\frac{1}{\delta}\right)$ time. We also need to compute the matrix-vector product $\bar{f}(CC^\dagger)u$, but this can be done in $\mathcal{O}(rc)$ time by instead multiplying through with the expression $U\bar{f}(D^2)U^\dagger = \bar{f}(CC^\dagger)$, where $U \in \mathbb{C}^{r\times c}$ comes from the SVD of C.

Now for the odd case: we could use Theorem 7.1 here, but we will continue to use Theorem 5.1 here. Similarly to the even case, we take $g(x)$ to be $q(x)$ in $[-1,1]$ and held constant ouside it, so $p^{(\mathrm{QV})}(A) = A\cdot g(A^\dagger A)$. Then, by plugging in the smoothness parameters from Lemma 6.3, we get R, C such that $\|R^\dagger\bar{g}(CC^\dagger)R + g(0)I - g(A^\dagger A)\| < \frac{\varepsilon}{\|A\|}$ with probability $\ge 1-\delta$ where

$$r = \widetilde{\mathcal{O}}\left(d^6\|A\|^4\|A\|_{\mathrm{F}}^2\frac{1}{\varepsilon^2}\log\frac{1}{\delta}\right) \qquad c = \widetilde{\mathcal{O}}\left(d^{10}\|A\|^8\|A\|_{\mathrm{F}}^2\frac{1}{\varepsilon^2}\log\frac{1}{\delta}\right).$$

We now use the approximating matrix product lemmas Lemmas 4.4 and 4.6 three times.

1. We use Lemma 4.4 to approximate $AR^\dagger \approx A'R'^\dagger$ such that $\|AR^\dagger - A'R'^\dagger\| \le \varepsilon d^{-2}$. We can do this since we have $\mathrm{SQ}(A^\dagger)$ (by assumption) and $\mathrm{SQ}(R^\dagger)$, in $\mathcal{O}\big(\|A\|_{\mathrm{F}}^2\|R\|_{\mathrm{F}}^2 d^4\varepsilon^{-2}\delta^{-1}\big) = \mathcal{O}\big(d^4\varepsilon^{-2}\delta^{-1}\big)$ samples, with each sample costing $\mathcal{O}(r\log(mn))$ time (by Lemma 4.3). Then using Lemma 5.2 and the bounds on $|q(x)|$ and $|\bar{q}(x)|$ in Lemma 6.3,

$$\begin{aligned}\|(AR^\dagger - A'R'^\dagger)\bar{g}(CC^\dagger)R\| &\le \|AR^\dagger - A'R'^\dagger\|\|\sqrt{\bar{g}(CC^\dagger)}\|\|\sqrt{\bar{g}(CC^\dagger)}R\| \\ &\le (\varepsilon d^{-2})\sqrt{d^3}\sqrt{\|g(A^\dagger A) - g(0)I\| + \frac{\varepsilon}{\|A\|}} \lesssim \varepsilon.\end{aligned}$$

2. We use Lemma 4.6 to approximate $Rb \approx u$ such that $\|Rb - u\| \le \frac{\varepsilon}{d^2\|A\|}$, where we use $\mathcal{O}\Big(\|R\|_{\mathrm{F}}^2\|b\|^2\frac{d^4\|A\|^2}{\varepsilon^2}\log\frac{1}{\delta}\Big) = \mathcal{O}\big(d^4\|A\|^2\varepsilon^{-2}\log\frac{1}{\delta}\big)$ samples.

$$\begin{aligned}\|A'R'^\dagger\bar{g}(CC^\dagger)(Rb-u)\| &\le \|AR^\dagger\bar{g}(CC^\dagger)(Rb-u)\| + \|(A'R'^\dagger - AR^\dagger)\bar{g}(CC^\dagger)Rb\| \\ &\lesssim \|A\|\|R^\dagger\bar{g}(CC^\dagger)\|\|Rb-u\| + \varepsilon \lesssim \|A\|d^2(\varepsilon d^{-2}\|A\|^{-1}) + \varepsilon \lesssim \varepsilon.\end{aligned}$$

3. Using $\mathrm{SQ}(b)$ and Lemma 4.4, we approximate $Ab \approx A''b''$ such that $\|Ab - A''b''\| \le \varepsilon/d$ (and consequently, $q(0)\|Ab - A''b''\| \le \varepsilon$) with $\mathcal{O}\big(\|A\|_{\mathrm{F}}^2\|b\|^2 d^2\varepsilon^{-2}\delta^{-1}\big) = \mathcal{O}\big(d^2\varepsilon^{-2}\delta^{-1}\big)$ samples.

So, we have shown that $v := A'R'^\dagger\bar{g}(CC^\dagger)u + q(0)A''b''$ satisfies $\|v - p^{(\mathrm{QV})}(A)\| \lesssim \varepsilon$. v is a linear combination of columns of A; via Lemma 3.6, we can get $\mathrm{SQ}_\phi(v)$ with

$$\begin{aligned}\widetilde{\mathbf{sq}}(v) &= \phi\,\mathrm{SQ}_\phi(v)\log\frac{1}{\delta} \\ &= \widetilde{\mathcal{O}}\left(\frac{\sum_i\|A'(\cdot,i)\|^2|R'(\cdot,i)^\dagger\bar{g}(CC^\dagger)u\|^2 + \sum_j q(0)^2\|A''(\cdot,j)\|^2|b''(j)\|^2}{\|v\|^2}\Big(\frac{d^4+d^2}{\varepsilon^2\delta}\Big)^2\log(mn)\right) \\ &= \widetilde{\mathcal{O}}\left(\frac{\frac{\|A\|_{\mathrm{F}}^2\|R^\dagger\|_{\mathrm{F}}^2}{d^4\varepsilon^{-2}\delta^{-1}}(\|\bar{g}(CC^\dagger)R\|\|b\| + \|\bar{g}(CC^\dagger)\|\|Rb-u\|)^2 + q(0)^2\frac{\|A\|_{\mathrm{F}}^2\|b\|^2}{d^2\varepsilon^{-2}\delta^{-1}}}{(\|p^{(\mathrm{QV})}(A)b\| - \|p^{(\mathrm{QV})}(A)b - v\|)^2}\frac{d^8}{\varepsilon^4\delta^2}\log(mn)\right) \\ &= \widetilde{\mathcal{O}}\left(\frac{d^{-4}(d^2 + d^3\varepsilon d^{-2}\|A\|^{-1})^2 + 1}{(\|p^{(\mathrm{QV})}(A)b\| - \varepsilon)^2}\frac{d^8}{\varepsilon^2\delta}\log(mn)\right) \\ &= \widetilde{\mathcal{O}}\Big(d^8\|p^{(\mathrm{QV})}(A)b\|^{-2}\varepsilon^{-2}\delta^{-1}\log(mn)\Big).\end{aligned}$$

Above, we used that $\varepsilon \lesssim \|p^{(\mathrm{QV})}(A)b\| \le \|A\|\|q(A^\dagger A)\|\|b\| \le d\|A\|$. Now, we rescale $\varepsilon \leftarrow \varepsilon\|p^{(\mathrm{QV})}(A)b\|$ to get the desired statement. The runtime is dominated by the sampling for C in $\mathcal{O}(rc\log(mn))$ time, the computation of $\bar{g}(CC^\dagger)$ in $\mathcal{O}\big(r^2c\big)$ time, and the approximation of $AR^\dagger \approx A'R'^\dagger$ in $\mathcal{O}\big(rd^4\varepsilon^{-2}\delta^{-1}\log(mn)\big)$ time. □

Remark 6.4. Here, we make a brief remark about a technical detail we previously elided. Technically, QSVT can use $A^\dagger$ in QRAM instead of A (cf. [GSLW19, Lemma 50]), leaving open the possibility that there is a quantum algorithm that does not give an exponential speedup when A is in QRAM, but does when $A^\dagger$ is in QRAM. We sketch an argument why this is impossible by showing that, given $\mathrm{SQ}(A)$, we can simulate $\mathrm{SQ}_\phi(B)$ (and $\mathrm{SQ}_\phi(B^\dagger)$) for B such that $\|B - A^\dagger\| \le \varepsilon\|A\|$ with probability $\ge 1-\delta$. Unfortunately, this argument is fairly involved, so we defer it to Appendix A.

6.2 Recommendation systems

Our framework gives a simpler and faster variant of Tang's dequantization [Tan19] of Kerenidis and Prakash's quantum recommendation systems [KP17]. Tang's result is notable for being the first result in this line of work and for dequantizing what was previously believed to be the strongest candidate for practical exponential quantum speedups for a machine learning problem [Pre18].

We want to find a product $j \in [n]$ that is a good recommendation for a particular user $i \in [m]$, given incomplete data on user-product preferences. If we store this data in a matrix $A \in \mathbb{R}^{m \times n}$ with sampling and query access, in the strong model described by Kerenidis and Prakash [KP17], finding good recommendations reduces to the following:

Problem 6.5. For a matrix $A \in \mathbb{R}^{m \times n}$, given $\mathrm{SQ}(A)$ and a row index $i \in [m]$, sample from $\hat{A}(i, \cdot)$ up to δ error in total variation distance, where $\|\hat{A} - A_{\sigma,\eta}\|_{\mathrm{F}} \leq \varepsilon \|A\|_{\mathrm{F}}$.

Here, $A_{\sigma,\eta}$ is a certain type of low-rank approximation to A. The standard notion of low-rank approximation is that of $A_r \coloneqq \sum_{i=1}^{r} \sigma_i U(\cdot, i) V(\cdot, i)^\dagger$, which is the rank-$r$ matrix closest to A in spectral and Frobenius norms. Using singular value transformation, we define an analogous notion thresholding singular values instead of rank.

Definition 6.6 ($A_{\sigma,\eta}$)**.** We define $A_{\sigma,\eta}$ as a singular value transform of A satisfying:

$$A_{\sigma,\eta} \coloneqq P^{(\mathrm{SV})}_{\sigma,\eta}(A) \qquad P_{\sigma,\eta}(\lambda) \begin{cases} = \lambda & \lambda \geq \sigma(1+\eta) \\ = 0 & \lambda < \sigma(1-\eta) \\ \in [0, \lambda] & \text{otherwise} \end{cases}.$$

Note that $P_{\sigma,\eta}$ is not fully specified in the range $[\sigma(1-\eta), \sigma(1+\eta))$, so $A_{\sigma,\eta}$ is any of a family of matrices with error η.

For intuition, $P^{(\mathrm{SV})}_{\sigma,\eta}(A)$ is A for (right) singular vectors with value $\geq \sigma(1+\eta)$, zero for those with value $< \sigma(1-\eta)$, and something in between for the rest. Our analysis simplifies the original algorithm, which passes through the low-rank approximation guarantee of Frieze, Kannan, and Vempala [FKV04].

Our algorithm uses that we can rewrite our target low-rank approximation as $A \cdot t(A^\dagger A)$, where t is a smoothened projector. So, we can use our main theorem, Theorem 5.1, to approximate $t(A^\dagger A)$ by some $R^\dagger U R$. Then, the i^{th} row of our low-rank approximation is $A(i, \cdot) R^\dagger U R$, which is a product of a vector with an RUR decomposition. Thus, using the sampling techniques described in Section 4.1, we have $\mathrm{SQ}_\phi(A(i, \cdot) R^\dagger U R)$, so we can get the sample from this row as desired.

Corollary 6.7. *Suppose* $0 < \varepsilon \lesssim \|A\|/\|A\|_{\mathrm{F}}$ *and* $\eta \leq 0.99$*. A classical algorithm can solve Problem 6.5 in time*

$$\widetilde{\mathcal{O}}\left(\frac{K^3 \kappa^5}{\eta^6 \varepsilon^6} \log^3 \frac{1}{\delta} + \frac{K^2 \kappa \|A(i, \cdot)\|^2}{\eta^2 \varepsilon^2 \|\hat{A}(i, \cdot)\|^2} \log^2 \frac{1}{\delta} \right).$$

The assumption on ε is a weak non-degeneracy condition in the low-rank regime. For reference, $\eta = 1/6$ in the application of this algorithm to recommendation systems. So, supposing the first term of the runtime dominates, the runtime is $\widetilde{\mathcal{O}}\left(\frac{\|A\|_{\mathrm{F}}^6 \|A\|^{10}}{\sigma^{16} \varepsilon^6} \log^3 \frac{1}{\delta} \right)$, which improves on the previous runtime $\widetilde{\mathcal{O}}\left(\frac{\|A\|_{\mathrm{F}}^{24}}{\sigma^{24} \varepsilon^{12}} \log^3 \frac{1}{\delta} \right)$ of [Tan19]. The quantum runtime for this problem is $\widetilde{\mathcal{O}}\left(\frac{\|A\|_{\mathrm{F}}}{\sigma} \right)$, up to $\mathrm{polylog}(m, n)$ terms [CGJ19, Theorem 27].

Proof. Note that $A_{\sigma,\eta} = A \cdot t(A^\dagger A)$, where t is the thresholding function shown below.

$$t(x) = \begin{cases} 0 & x < (1-\eta)^2\sigma^2 \\ \frac{1}{4\eta\sigma^2}(x - (1-\eta)^2\sigma^2) & (1-\eta)^2\sigma^2 \le x < (1+\eta)^2\sigma^2 \\ 1 & x \ge (1+\eta)^2\sigma^2 \end{cases}.$$

We will apply Theorem 5.1 with error parameter ε to get matrices R, C such that $AR^\dagger \bar{t}(CC^\dagger)R$ satisfies

$$\|A_{\sigma,1/6} - AR^\dagger \bar{t}(CC^\dagger)R\|_{\mathrm{F}} \le \|A\|_{\mathrm{F}} \|t(A^\dagger A) - R^\dagger \bar{t}(CC^\dagger)R\| \le \varepsilon \|A\|_{\mathrm{F}}. \tag{8}$$

Since $t(x)$ is $(4\eta\sigma^2)^{-1}$-Lipschitz and $t(x)/x$ is $(4\eta(1-\eta)^2\sigma^4)^{-1}$-Lipschitz, the sizes of r and c are

$$r = \widetilde{\mathcal{O}}\left(L^2\|A\|^2\|A\|_{\mathrm{F}}^2 \frac{1}{\varepsilon^2}\log\frac{1}{\delta}\right) = \widetilde{\mathcal{O}}\left(\frac{\|A\|^2\|A\|_{\mathrm{F}}^2}{\sigma^4\eta^2\varepsilon^2}\log\frac{1}{\delta}\right) = \widetilde{\mathcal{O}}\left(\frac{K\kappa}{\eta^2\varepsilon^2}\log\frac{1}{\delta}\right);$$
$$c = \widetilde{\mathcal{O}}\left(\bar{L}^2\|A\|^6\|A\|_{\mathrm{F}}^2 \frac{1}{\varepsilon^2}\log\frac{1}{\delta}\right) = \widetilde{\mathcal{O}}\left(\frac{\|A\|^6\|A\|_{\mathrm{F}}^2}{\sigma^8\eta^2\varepsilon^2}\log\frac{1}{\delta}\right) = \widetilde{\mathcal{O}}\left(\frac{K\kappa^3}{\eta^2\varepsilon^2}\log\frac{1}{\delta}\right).$$

So, it suffices to compute the SVD of an $r \times c$ matrix, which has a runtime of

$$\widetilde{\mathcal{O}}\left(\frac{K^3\kappa^5}{\eta^6\varepsilon^6}\log^3\frac{1}{\delta}\right).$$

Next, we want to approximate $AR^\dagger \approx A'R'^\dagger$. If we had $\mathrm{SQ}(A^\dagger)$ (in particular, if we could compute column norms $\|A(\cdot, j)\|$), we could do this via Lemma 4.6, and if we were okay with paying factors of $\frac{1}{\delta}$, we could do this via Lemma 4.4. Here, we will instead implicitly define an approximation by approximating each row $[AR^\dagger](i,\cdot) = A(i,\cdot)R^\dagger$ via Lemma 4.6, since we then have $\mathrm{SQ}(A(i,\cdot)^\dagger)$ and $\mathrm{SQ}(R^\dagger)$. With this proposition, we can estimate $[AR^\dagger](i,\cdot) \approx (A(i,\cdot)S^\dagger S)R^\dagger$ to $\frac{\varepsilon}{\sqrt{K}}\|A(i,\cdot)\|\|R^\dagger\|_{\mathrm{F}} = \varepsilon\|A(i,\cdot)\|\sigma$ error using $r' := O(\varepsilon^{-2}K\log\frac{1}{\delta})$ samples[17]. Here, $A'(i,\cdot) := A(i,\cdot)S^\dagger S$ is our r'-sparse approximation, giving that

$$\|AR^\dagger - A'R^\dagger\|_{\mathrm{F}} = \sqrt{\textstyle\sum_{i=1}^m \|[AR^\dagger](i,\cdot) - [A'R^\dagger](i,\cdot)\|^2} \le \sqrt{\textstyle\sum_{i=1}^m \varepsilon^2\|A(i,\cdot)\|^2\sigma^2} = \varepsilon\sigma\|A\|_{\mathrm{F}}. \tag{9}$$

Using this and the observation that $\max_x \bar{t}(x) = (1+\eta)^{-2}\sigma^{-2} \le \sigma^{-2}$, we can bound the quality of our final approximation as

$$\begin{aligned}
\|\hat{A} - A_{\sigma,\eta}\|_{\mathrm{F}} &\le \|(A'R^\dagger - AR^\dagger)\bar{t}(CC^\dagger)R\|_{\mathrm{F}} + \|AR^\dagger\bar{t}(CC^\dagger)R - A_{\sigma,\eta}\|_{\mathrm{F}} && \text{by triangle inequality} \\
&\le \|A'R^\dagger - AR^\dagger\|_{\mathrm{F}} \left\|\sqrt{\bar{t}(CC^\dagger)}\right\| \left\|\sqrt{\bar{t}(CC^\dagger)}R\right\| + \varepsilon\|A\|_{\mathrm{F}} && \text{by Eq. (8)} \\
&\le \varepsilon\sigma\|A\|_{\mathrm{F}}\sigma^{-1}\sqrt{1+\varepsilon} + \varepsilon\|A\|_{\mathrm{F}} \lesssim \varepsilon\|A\|_{\mathrm{F}}. && \text{by Lemma 5.2 and Eq. (9)}
\end{aligned}$$

We can sample from $\hat{A}(i,\cdot) = A'(i,\cdot)R^\dagger\bar{f}(CC^\dagger)R$ by naively computing $x := A'(i,\cdot)R^\dagger\bar{f}(CC^\dagger)$, taking $O(r'r + rc)$ time. Then, we use Lemmas 3.5 and 3.6 to get a sample from xR with probability $\ge 1-\delta$ in $\mathcal{O}\big(\widetilde{\mathbf{sq}}_\phi(xR)\big)$ time, which is $\mathcal{O}\big(\phi\,\mathbf{sq}_\phi(xR)\log\frac{1}{\delta}\big)$, where $\mathbf{sq}_\phi(xR) = \mathcal{O}(r)$ and

$$\phi = r\frac{\sum_{j=1}^r |x(j)|^2\|R(j,\cdot)\|^2}{\|xR\|^2} \lesssim r\frac{\sum_{j=1}^r |x(j)|^2\|A\|_{\mathrm{F}}^2}{\|\hat{A}(i,\cdot)\|^2 r} = \frac{\|x\|^2\|A\|_{\mathrm{F}}^2}{\|\hat{A}(i,\cdot)\|^2}.$$

[17] Formally, to get a true approximation $AR \approx A'R$, we need to union bound the failure probability for each row, paying a $\log m$ factor in runtime. However, we will ignore this consideration: our goal is to sample from one row, so we only need to succeed in our particular row.

Then, using previously established bounds and bounds from Lemma 5.2, we have

$$\begin{aligned}\frac{\|x\|^2\|A\|_{\mathrm{F}}^2}{\|\hat{A}(i,\cdot)\|^2} &= \frac{\|A'(i,\cdot)R^\dagger\bar{t}(CC^\dagger)\|^2\|A\|_{\mathrm{F}}^2}{\|\hat{A}(i,\cdot)\|^2} \\ &\leq \left(\|A(i,\cdot)\|\left\|R^\dagger\sqrt{\bar{t}(CC^\dagger)}\right\|\left\|\sqrt{\bar{t}(CC^\dagger)}\right\| + \|A(i,\cdot)'R^\dagger - A(i,\cdot)R^\dagger\|\|\bar{t}(CC^\dagger)\|\right)^2 \frac{\|A\|_{\mathrm{F}}^2}{\|\hat{A}(i,\cdot)\|^2} \\ &\lesssim (\|A(i,\cdot)\|\sigma^{-1} + \varepsilon\sigma\|A(i,\cdot)\|\sigma^{-2})^2\frac{\|A\|_{\mathrm{F}}^2}{\|\hat{A}(i,\cdot)\|^2} \\ &\lesssim \frac{\|A(i,\cdot)\|^2\|A\|_{\mathrm{F}}^2}{\|\hat{A}(i,\cdot)\|^2\sigma^2}.\end{aligned}$$

This sampling procedure and the SVD dominate the runtime. Since the sampling is exact, the only error in total variation distance is the probability of failure. □

Remark 6.8. This algorithm implicitly assumes that the important singular values are $\geq \sigma$. Without such an assumption, we can take $\sigma = \varepsilon\|A\|_{\mathrm{F}}$ and $\eta = 1/2$, and have meaningful bounds on the output matrix $\hat{A}$. Observe that, for $p(x) = x(t(\sqrt{x}) - 1)$,

$$\|A \cdot t(A^\dagger A) - A\| = \|p^{(\mathrm{SV})}(A)\| \leq \frac{3}{2}\varepsilon\|A\|_{\mathrm{F}}.$$

So, our low-rank approximation output $\hat{A}$ satisfies $\|\hat{A} - A\| \lesssim \varepsilon\|A\|_{\mathrm{F}}$, with no assumptions on A, in $\widetilde{\mathcal{O}}\left(\frac{\|A\|_{\mathrm{F}}^6}{\|A\|^6\varepsilon^{22}}\log^3\frac{1}{\delta}\right)$ time. This can be subsequently used to get $\mathrm{SQ}_\phi(\hat{A}(i,\cdot)) = \mathrm{SQ}_\phi(e_i\hat{A})$ where $\|\hat{A}(i,\cdot) - A(i,\cdot)\| \lesssim \varepsilon\|A\|_{\mathrm{F}}$ (in a myopic sense, solving the same problem as Problem 6.5), or more generally, any product of $\hat{A}$ with a vector, in time independent of dimension.

6.3 Supervised clustering

The 2013 paper of Lloyd, Mohseni, and Rebentrost [LMR13] gives two algorithms for the machine learning problem of clustering. The first algorithm is a simple swap test procedure that was dequantized by Tang [Tan21] (the second is an application of the quantum adiabatic algorithm with no proven runtime guarantees). We will reproduce the algorithm from [Tan21] here: since the dequantization just uses the inner product protocol, so it rather trivially fits into our framework.

We have a dataset of points in $\mathbb{R}^d$ grouped into clusters, and we wish to classify a new data point by assigning it to the cluster with the nearest average, aka *centroid*. We do this by estimating the distance between the new point $p \in \mathbb{R}^d$ to the centroid of a cluster of points $q_1, \ldots, q_{n-1} \in \mathbb{R}^d$, namely, $\|p - \frac{1}{n-1}(q_1 + \cdots + q_{n-1})\|^2$. This is equal to $\|wM\|^2$, where

$$M \coloneqq \begin{bmatrix} p/\|p\| \\ -q_1/(\|q_1\|\sqrt{n-1}) \\ \vdots \\ -q_{n-1}/(\|q_{n-1}\|\sqrt{n-1}) \end{bmatrix} \in \mathbb{R}^{n\times d}, \qquad w \coloneqq \left[\|p\|, \frac{\|q_1\|}{\sqrt{n-1}}, \ldots, \frac{\|q_{n-1}\|}{\sqrt{n-1}}\right] \in \mathbb{R}^n.$$

Because the quantum algorithm assumes input in quantum states, we can assume sampling and query access to the data points, giving the problem

Problem 6.9. Given $\mathrm{SQ}(M) \in \mathbb{R}^{n\times d}, \mathrm{Q}(w) \in \mathbb{R}^n$, approximate $(wM)(wM)^T$ to additive ε error with probability at least $1 - \delta$.

Corollary 6.10 ([Tan21, Theorem 4])**.** *There is a classical algorithm to solve Problem 6.9 in* $\mathcal{O}\left(\|M\|_{\mathrm{F}}^4\|w\|^4\frac{1}{\varepsilon^2}\log\frac{1}{\delta}\right)$ *time.*

Note that $\|M\|_{\mathrm{F}}^2 = 2$ and $\|w\|^2 = \|p\|^2 + \frac{1}{n-1}\sum_{i=1}^{n-1}\|q_i\|^2$. The quantum algorithm has a quadratically faster runtime of $\mathcal{O}\big(\|M\|_{\mathrm{F}}^2\|w\|^2\frac{1}{\varepsilon}\big)$, ignoring polylog$(n,d)$ factors [LMR13, Tan21].

Proof. Recall our notation for the vector of row norms $m := [\|M(1,\cdot)\|,\ldots,\|M(n,\cdot)\|]$ coming from Definition 3.7. We can rewrite $(wM)(wM)^T$ as an inner product $\langle u, v\rangle$ where

$$u := \sum_{i=1}^{n}\sum_{j=1}^{d}\sum_{k=1}^{n} M(i,j)\|M(k,\cdot)\|e_i\otimes e_j\otimes e_k = M\otimes m$$
$$v := \sum_{i=1}^{n}\sum_{j=1}^{d}\sum_{k=1}^{n} \frac{w_i w_k M(j,k)}{\|M(k,\cdot)\|}e_i\otimes e_j\otimes e_k,$$

where u and v are three-dimension tensors. By flattening u and v, we can represent them as two vectors in $\mathbb{R}^{(n\cdot d\cdot n)\times 1}$. We clearly have $\mathrm{Q}(v)$ from queries to M and w. As for getting $\mathrm{SQ}(u)$ from $\mathrm{SQ}(M)$: to sample, we first sample i according to m, sample j according to $M(i,\cdot)$, and sample k according to m; to query, compute $u_{i,j,k} = M(i,j)m(k)$. Finally, we can apply Lemma 4.12 to estimate $\langle u, v\rangle$. $\|u\| = \|M\|_{\mathrm{F}}^2$ and $\|v\| = \|w\|^2$, so estimating $\langle u, v\rangle$ to ε additive error with probability at least $1-\delta$ requires $O(\|M\|_{\mathrm{F}}^4\|w\|^4\varepsilon^{-2}\log\frac{1}{\delta})$ samples. □

6.4 Principal component analysis

Principal component analysis (PCA) is an important data analysis tool, first proposed to be feasible via quantum computation by Lloyd, Mohseni, and Rebentrost [LMR14]. Given copies of states with density matrix $\rho = X^\dagger X$, the quantum PCA algorithm can prepare the state $\sum\lambda_i|v_i\rangle\langle v_i|\otimes|\hat{\lambda}_i\rangle\langle\hat{\lambda}_i|$, where λ_i and v_i are the eigenvalues and eigenvectors of $X^\dagger X$, and $\hat{\lambda}_i$ are eigenvalue estimates (up to additive error). See Prakash's PhD thesis [Pra14, Section 3.2] for a full analysis and Chakraborty, Gilyén, and Jeffery for a faster version of this algorithm in the block-encoding model [CGJ19]. Directly measuring the eigenvalue register is called *spectral sampling*, but such sampling is not directly useful for machine learning applications.

Though we do not know how to dequantize this protocol exactly, we can dequantize it in the low-rank setting, which is the only useful poly-logarithmic time application that Lloyd, Mohseni, and Rebentrost [LMR14] suggests for quantum PCA.

Problem 6.11 (PCA for low-rank matrices)**.** Given a matrix $\mathrm{SQ}(X)\in\mathbb{C}^{m\times n}$ such that $X^\dagger X$ has top k eigenvalues $\{\lambda_i\}_{i=1}^k$ and eigenvectors $\{v_i\}_{i=1}^k$, with probability $\geq 1-\delta$, compute eigenvalue estimates $\{\hat{\lambda}_i\}_{i=1}^k$ such that $\sum_{i=1}^k|\hat{\lambda}_i-\lambda_i|\leq\varepsilon\,\mathrm{Tr}(X^\dagger X)$ and eigenvectors $\{\mathrm{SQ}_\phi(\hat{v}_i)\}_{i=1}^k$ such that $\|\hat{v}_i - v_i\|\leq\varepsilon$ for all i.

Note that we should think of λ_i as σ_i^2, where σ_i is the ith largest singular value of X. To robustly avoid degeneracy conditions, our runtime must depend on parameters for condition number and spectral gap:

$$K := \mathrm{Tr}(X^\dagger X)/\lambda_k \geq k \quad\text{and}\quad \eta := \min_{i\in[k]}|\lambda_i-\lambda_{i+1}|/\|X\|^2. \tag{10}$$

We also denote $\kappa := \|X\|^2/\lambda_k$. Dependence on K and η are necessary to reduce Problem 6.11 to spectral sampling. If $K = \mathrm{poly}(n)$, then $\lambda_k = \mathrm{Tr}(X^\dagger X)/\mathrm{poly}(n)$, so distinguishing λ_k from λ_{k+1} necessarily takes $\mathrm{poly}(n)$ samples, and even sampling λ_k once takes $\mathrm{poly}(n)$ samples. As a result, learning v_k is also impossible. A straightforward coupon collector argument (given e.g. by Tang [Tan21]) shows that Problem 6.11 can be solved by a quantum algorithm performing spectral sampling[18], with runtime depending polynomially on K and $\frac{1}{\eta}$. We omit this argument for brevity.

[18]The quantum analogue to $\mathrm{SQ}(X)$ is efficient state preparation of X, a purification of ρ.

Classically, we can solve this PCA problem with quantum-inspired techniques, as first noted in [Tan21].

Corollary 6.12. *For $0 < \varepsilon \lesssim \eta\|X\|^2/\|X\|_{\mathrm{F}}^2$, we can solve Problem 6.11 in $\widetilde{\mathcal{O}}\Big(\frac{\|X\|_{\mathrm{F}}^6}{\lambda_k^2\|X\|^2}\eta^{-6}\varepsilon^{-6}\log^3\frac{k}{\delta}\Big)$ time to get $\mathrm{SQ}_\phi(\hat{v}_i)$ where $\widetilde{\mathbf{sq}}(\hat{v}_i) = \widetilde{\mathcal{O}}\Big(\frac{\|X\|_{\mathrm{F}}^4}{\lambda_i\|X\|^2}\eta^{-2}\varepsilon^{-2}\log^2\frac{1}{\delta}\Big)$.*

This improves significantly over prior work [Tan21, Theorem 8], which achieves the runtime of $\widetilde{\mathcal{O}}\Big(\frac{\|X\|_{\mathrm{F}}^{36}}{\|X\|^{12}\lambda_k^{12}}\eta^{-6}\varepsilon^{-12}\log^3\frac{k}{\delta}\Big)$.[19] The best quantum algorithm for this problem runs in $\widetilde{\mathcal{O}}\Big(\frac{\|X\|_{\mathrm{F}}\|X\|}{\lambda_k\varepsilon}\Big)$ time, up to factors of polylog(m,n) [CGJ19, Theorem 27].[20]

We approach the problem as follows. First, we use that an importance-sampled submatrix of X has approximately the same singular values as X itself (Lemma 4.9) to get our estimates $\{\hat{\lambda}_i\}_{i=1}^k$. With these estimates, we can define smoothened step functions f_i for $i \in [k]$ such that $f_i(X^\dagger X) = v_i^\dagger v_i$. We can then use our main theorem to find an RUR decomposition for $f_i(X^\dagger X)$. We use additional properties of the RUR description to argue that it is indeed a rank-1 outer product $\hat{v}_i^\dagger\hat{v}_i$, which is our desired approximation for the eigenvector. We have sampling and query access to $\hat{v}_i$ because it is $R^\dagger x$ for some vector x. Our runtime is quite good because these piecewise linear step functions have relatively tame derivatives, as opposed to the thresholded inverse function, whose Lipschitz constants must incur quadratic and cubic overheads in terms of condition number.

Proof. We will assume that we know λ_k and η. If both are unknown, then we can estimate them with the singular value estimation procedure described below (Lemma 4.9).

Notice that $\eta\|X\|^2 \le \lambda_k$ follows from our definition of η. The algorithm will proceed as follows: first, consider $C \coloneqq SXT \in \mathbb{C}^{c\times r}$ as described in Theorem 5.1, with parameters

$$r \coloneqq \widetilde{\mathcal{O}}\Big(\frac{\|X\|_{\mathrm{F}}^2}{\eta^2\|X\|^2\varepsilon^2}\log\frac{k}{\delta}\Big) \qquad c \coloneqq \widetilde{\mathcal{O}}\Big(\frac{\|X\|_{\mathrm{F}}^2\|X\|^2}{\eta^2\lambda_k^2\varepsilon^2}\log\frac{k}{\delta}\Big).$$

Consider computing the eigenvalues of $CC^\dagger$; denote the i^{th} eigenvalue $\hat{\lambda}_i$. Since $r, c = \Omega(\frac{\|X\|_{\mathrm{F}}^2}{\lambda_k\varepsilon^2}\log\frac{1}{\delta})$, by Lemma 4.9 with error parameter $\frac{\varepsilon\sqrt{\lambda_k}}{8\|X\|_{\mathrm{F}}}$, with probability $\ge 1-\delta$,

$$\sqrt{\sum\nolimits_{i=1}^{\min(m,n)}(\hat{\lambda}_i - \lambda_i)^2} \le \frac{\varepsilon\sqrt{\lambda_k}}{8\|X\|_{\mathrm{F}}}\|X\|_{\mathrm{F}}^2.$$

These $\hat{\lambda}_i$'s for $i \in [k]$ have the desired property for eigenvalue estimates:

$$\sum_{i=1}^k|\hat{\lambda}_i - \lambda_i| \le \sqrt{k}\sqrt{\sum\nolimits_{i=1}^k(\hat{\lambda}_i-\lambda_i)^2} \le \varepsilon\sqrt{k\lambda_k}\|X\|_{\mathrm{F}} \le \varepsilon\|X\|_{\mathrm{F}}^2.$$

[19] This runtime comes from taking $\varepsilon_\sigma = \varepsilon_v = \varepsilon$ and changing the normalization of the gap parameter $\eta = \eta\|X\|^2/\|X\|_{\mathrm{F}}^2$ to correspond to the problem as formulated here.

[20] Given X in QRAM, this follows from applying Theorem 27 to a quantum state with density matrix of $X^\dagger X$ with $\alpha = \|X\|_{\mathrm{F}}$ and $\Delta = \frac{\varepsilon\|X\|_{\mathrm{F}}^2}{\|X\|} \lesssim \frac{\eta\|X\|}{\|X\|_{\mathrm{F}}}$. The output is some estimate of $\sqrt{\lambda_i}$ to Δ error, which when squared is an estimate of λ_i to $\Delta\|X\| = \varepsilon\|X\|_{\mathrm{F}}^2$ error as desired. Then, the density matrix is a probability distribution over eigenvectors with their corresponding eigenvalue estimate (which is enough to identify the eigenvector). The coupon collector argument mentioned above gives us access to all the top k eigenvalues and eigenvectors by running this algorithm $\|X\|_{\mathrm{F}}^2/\lambda_k$ times [Tan21].

This bound also implies that, for all i, $|\hat{\lambda}_i - \lambda_i| \le \frac{\varepsilon}{8}\|X\|_{\mathrm{F}}^2$. Next, consider the eigenvalue transformations f_i for $i \in [k]$, defined

$$f_i(x) \coloneqq \begin{cases} 0 & x - \hat{\lambda}_i < -\frac{1}{4}\eta\|X\|^2 \\ 2 + \frac{8}{\eta\|X\|^2}(x - \hat{\lambda}_i) & -\frac{1}{4}\eta\|X\|^2 \le x - \hat{\lambda}_i < -\frac{1}{8}\eta\|X\|^2 \\ 1 & -\frac{1}{8}\eta\|X\|^2 \le x - \hat{\lambda}_i < \frac{1}{8}\eta\|X\|^2 \\ 2 - \frac{8}{\eta\|X\|^2}(x - \hat{\lambda}_i) & \frac{1}{8}\eta\|X\|^2 \le x - \hat{\lambda}_i < \frac{1}{4}\eta\|X\|^2 \\ 0 & \frac{1}{4}\eta\|X\|^2 \le x - \hat{\lambda}_i \end{cases}.$$

This is a function that is one when $|x - \hat{\lambda}_i| \le \frac{1}{8}\eta\|X\|^2$, zero when $|x - \hat{\lambda}_i| \ge \frac{1}{4}\eta\|X\|^2$, and interpolates between them otherwise. From the eigenvalue gap and the aforementioned bound $|\hat{\lambda}_i - \lambda_i| \le \frac{1}{8}\eta\|X\|^2$, we can conclude that $f_i(X^\dagger X) = v_i v_i^\dagger$ exactly. Further, by Theorem 5.1, we can conclude that $R^\dagger \bar{f}_i(CC^\dagger)R$ approximates $v_i v_i^\dagger$, with C, R the exact approximations used to estimate singular values. The conditions of Theorem 5.1 are satisfied because $\varepsilon \lesssim 8 \le \frac{8}{\eta} = L\|X\|^2$ for L the Lipschitz constant of f_i. The values of r, c are chosen so that $\|R^\dagger \bar{f}_i(CC^\dagger)R - f_i(X^\dagger X)\| \le \varepsilon/2$ (note $f_i(0) = 0$):

$$r = \widetilde{\mathcal{O}}\left(L^2\|X\|^2\|X\|_{\mathrm{F}}^2 \frac{1}{\varepsilon^2}\log\frac{1}{\delta}\right) = \widetilde{\mathcal{O}}\left(\frac{\|X\|_{\mathrm{F}}^2}{\|X\|^2\eta^2\varepsilon^2}\log\frac{1}{\delta}\right)$$
$$c = \widetilde{\mathcal{O}}\left(\bar{L}^2\|X\|^6\|X\|_{\mathrm{F}}^2 \frac{1}{\varepsilon^2}\log\frac{1}{\delta}\right) = \widetilde{\mathcal{O}}\left(\frac{\|X\|^6\|X\|_{\mathrm{F}}^2}{\eta^2\|X\|^4(\hat{\lambda}_i - \frac{1}{4}\eta\|X\|^2)^2\varepsilon^2}\log\frac{1}{\delta}\right) = \widetilde{\mathcal{O}}\left(\frac{\|X\|^2\|X\|_{\mathrm{F}}^2}{\eta^2\lambda_k^2\varepsilon^2}\log\frac{1}{\delta}\right).$$

Further, f_i is chosen with respect to $\hat{\lambda}_i$ such that $R^\dagger \bar{f}_i(CC^\dagger)R$ is rank one, since $CC^\dagger$ has one eigenvalue between $\hat{\lambda}_i - \frac{1}{4}\eta\|X\|^2$ and $\hat{\lambda}_i + \frac{1}{4}\eta\|X\|^2$. Thus, this approximation is an outer product, $R^\dagger \bar{f}_i(CC^\dagger)R = \hat{v}_i\hat{v}_i^\dagger$, and we take the corresponding vector to be our eigenvector estimate: $\|\hat{v}_i\| \le \sqrt{1 + \varepsilon/2} \le 1 + \varepsilon/4$, so

$$\begin{aligned} \varepsilon/2 &\ge \|(\hat{v}_i\hat{v}_i^\dagger - v_i v_i^\dagger)v_i\| && \text{by definition} \\ &= \|\langle \hat{v}_i, v_i\rangle \hat{v}_i - v_i\| && \text{by } \|v_i\|^2 = 1 \\ &\ge \|\hat{v}_i - v_i\| - (\langle \hat{v}_i, v_i\rangle - 1)\|\hat{v}_i\| && \text{by triangle inequality} \\ &\ge \|\hat{v}_i - v_i\| - (\|\hat{v}_i\|\|v_i\| - 1)\|u\| && \text{by Cauchy–Schwarz} \\ &\ge \|\hat{v}_i - v_i\| - (1 + \varepsilon/4 - 1)(1 + \varepsilon/4) && \text{by } \|\hat{v}_i\| \le 1 + \varepsilon/4 \\ &\ge \|\hat{v}_i - v_i\| - \varepsilon/2, \end{aligned}$$

which is the desired bound. By choosing failure probability δ/k, the bound can hold true for all k with probability $\ge 1 - \delta$.

Finally, we can get access to $\hat{v}_i = R^\dagger \bar{v}_i$, where $\bar{v}_i \in \mathbb{C}^r$ satisfies $\bar{v}_i^\dagger \bar{v}_i = \bar{f}_i(CC^\dagger)$. Since $\|\bar{v}_i^\dagger\| \le \sqrt{\max_x \bar{f}_i(x)} \lesssim \lambda_i^{-\frac{1}{2}}$, using Lemmas 3.5 and 3.6, we have $\mathrm{SQ}_\phi(\hat{v}_i)$ with

$$\phi = r\frac{\sum_{s=1}^r |\hat{v}_i(s)|^2\|R(s,\cdot)\|^2}{\|R^\dagger \bar{v}_i\|^2} = r\frac{\sum_{s=1}^r |\hat{v}_i(s)|^2\|X\|_{\mathrm{F}}^2}{\|R^\dagger \bar{v}_i\|^2 r} = \frac{\|\hat{v}_i\|^2\|X\|_{\mathrm{F}}^2}{\|R^\dagger \bar{v}_i\|^2} \lesssim \frac{\|X\|_{\mathrm{F}}^2}{\lambda_i(1-\varepsilon)^2} \lesssim \frac{\|X\|_{\mathrm{F}}^2}{\lambda_i},$$

so $\widetilde{\mathbf{sq}}_\phi(\hat{v}_i) = \phi\,\mathbf{sq}_\phi(v)\log\frac{1}{\delta} \lesssim \frac{\|X\|_{\mathrm{F}}^2}{\lambda_i} r\log\frac{1}{\delta}$. □

6.5 Matrix inversion and principal component regression

The low-rank matrix inversion algorithms given by Gilyén, Lloyd, and Tang [GLT18] and Chia, Lin, and Wang [CLW18] dequantize Harrow, Hassidim, and Lloyd's quantum matrix inversion algorithm (HHL) [HHL09] in the regime where the input matrix is *low-rank* instead of sparse. The corresponding quantum algorithm in this regime is given by Chakraborty, Gilyén, and Jeffery [CGJ19], among others. Since sparse matrix inversion is BQP-complete, it is unlikely that one can efficiently dequantize it. However, the variant of low-rank (non-sparse) matrix inversion appears often in quantum machine learning [Pra14, WZP18, RML14, CD16, RL18], making it an influential primitive in its own right.

Using our framework, we can elegantly derive the low-rank matrix inversion algorithm in a manner similar to prior quantum-inspired work [GLT18, CLW18]. Moreover, we can also handle the approximately low-rank regime and only invert the matrix on a well-conditioned subspace, solving principal component regression—for more discussion see [GSLW19]. Namely, we can find a thresholded pseudoinverse of an input matrix:

Definition 6.13 ($A^+_{\sigma,\eta}$)**.** We define $A^+_{\sigma,\eta}$ to be any singular value transform of A satisfying:

$$A^+_{\sigma,\eta} \coloneqq \mathrm{tinv}^{(\mathrm{SV})}_{\sigma,\eta}(A) \qquad \mathrm{tinv}_{\sigma,\eta}(\lambda)\begin{cases} = 1/\lambda & \lambda \geq \sigma \\ = 0 & \lambda < \sigma(1-\eta) \\ \in [0, \sigma^{-1}] & \text{otherwise} \end{cases}. \tag{11}$$

This definition is analogous to $A_{\sigma,\eta}$ in Section 6.2: it is A^+ for singular vectors with value $\geq \sigma$, zero for singular vectors with value $\leq \sigma(1-\eta)$, and a linear interpolation between the two in between.

Problem 6.14. Given $\mathrm{SQ}_\varphi(A) \in \mathbb{C}^{m\times n}, \mathrm{Q}(b) \in \mathbb{C}^m$, with probability $\geq 1-\delta$, get $\mathrm{SQ}_\phi(\hat{x})$ such that $\|\hat{x} - x^*\| \leq \varepsilon \|A\|^{-1}\|b\|$, where $x^* \coloneqq A^+_{\sigma,\eta} b$.

Corollary 6.15. *For $0 < \varepsilon \lesssim \frac{\|A\|^2}{\sigma^2}$ and $\eta \leq 0.99$, we can solve Problem 6.14 in $\widetilde{\mathcal{O}}\Big(\frac{\varphi^6 K^3 \kappa^{11}}{\eta^6 \varepsilon^6} \log^3 \frac{1}{\delta}\Big)$ time to give $\mathrm{SQ}_\phi(\hat{x})$ for $\widetilde{\mathbf{sq}}_\phi(\hat{x}) = \widetilde{\mathcal{O}}\Big(\frac{\varphi^4 K^2 \kappa^5}{\eta^2 \varepsilon^2} \frac{\|x^*\|^2}{\|\hat{x}\|^2} \log^2 \frac{1}{\delta}\Big)$.*

This should be compared to [GLT18], which applies only to strictly rank-k A with $\varphi = 1$ and gets the incomparable runtime of $\widetilde{\mathcal{O}}\Big(\frac{K^3 \kappa^8 k^6}{\eta^6 \varepsilon^6} \log^3 \frac{1}{\delta}\Big)$. The corresponding quantum algorithm using block-encodings takes $\mathcal{O}(\|A\|_{\mathrm{F}}/\sigma)$ time, up to $\mathrm{polylog}(m,n)$ factors, to get this result for constant η [GSLW19, Theorem 41].

If we further assume that $\varepsilon < 0.99$ and b is in the image of A, then $\widetilde{\mathbf{sq}}_\phi(\hat{x})$ can be simplified, since $\|\hat{x}\| \geq \|x^*\| - \varepsilon\|A\|^{-1}\|b\| \geq (1-\varepsilon)\|x^*\|$, so $\frac{\|x^*\|}{\|\hat{x}\|} \leq 100$. However, this algorithm also works for larger ε; namely, if we only require that $\|\hat{x} - x^*\| \leq \varepsilon\sigma^{-1}\|b\|$ (a "worst-case" error bound), then this algorithm works with runtime smaller by a factor of κ^3 (and $\widetilde{\mathbf{sq}}_\phi(\hat{x})$ smaller by a factor of κ).

The algorithm comes from rewriting $A^+_{\sigma,\eta} b = \iota(A^\dagger A) A^\dagger b$ for ι a function encoding a thresholded inverse. Namely, $\iota(x) = 1/x$ for $x \geq \sigma^2$, $\iota(x) = 0$ for $x \leq (1-\eta)^2\sigma^2$, and is a linear interpolation between the endpoints for $x \in [(1-\eta)^2\sigma^2, \sigma^2]$. By our main theorem, we can find an RUR decomposition for $\iota(A^\dagger A)$, from which we can then get $\mathrm{SQ}(R^\dagger U R A^\dagger b)$ via sampling techniques.

Proof. We will solve our problem for $x^* = A^+_{\sigma,\eta} b = \iota(A^\dagger A) A^\dagger b$ where

$$\iota(x) \coloneqq \begin{cases} 0 & x < \sigma^2(1-\eta)^2 \\ \frac{1}{(2\eta - \eta^2)\sigma^4}(x - \sigma^2(1-\eta)^2) & \sigma^2(1-\eta)^2 \leq x < \sigma^2 \\ \frac{1}{x} & \sigma^2 \leq x \end{cases}.$$

So, if we can estimate $\iota(A^\dagger A)$ such that $\|\iota(A^\dagger A) - R^\dagger \bar{\iota}(CC^\dagger)R\| \le \frac{\varepsilon}{\|A\|^2}$, then as desired,

$$\|A^+_{\sigma,\eta} b - R^\dagger \bar{\iota}(CC^\dagger) R A^\dagger b\| \le \frac{\varepsilon}{\|A\|}\|b\| \le \varepsilon \|A^+_{\sigma,\eta} b\|.$$

By Theorem 5.1 with $L = \frac{1}{(2\eta-\eta^2)\sigma^4}$ and $\bar{L} = \frac{1}{(1-\eta)^2(2\eta-\eta^2)\sigma^6}$, we can find such R and C with

$$r = \widetilde{\mathcal{O}}\left(\varphi^2 \frac{\|A\|^2\|A\|_{\mathrm{F}}^2}{(2\eta-\eta^2)^2\sigma^8 \frac{\varepsilon^2}{\|A\|^4}} \log\frac{1}{\delta}\right) = \widetilde{\mathcal{O}}\left(\frac{\varphi^2 K\kappa^3}{\eta^2\varepsilon^2}\log\frac{1}{\delta}\right)$$
$$c = \widetilde{\mathcal{O}}\left(\varphi^2 \frac{\|A\|^6\|A\|_{\mathrm{F}}^2}{(1-\eta)^4(2\eta-\eta^2)^2\sigma^{12} \frac{\varepsilon^2}{\|A\|^4}} \log\frac{1}{\delta}\right) = \widetilde{\mathcal{O}}\left(\frac{\varphi^2 K\kappa^5}{\eta^2\varepsilon^2}\log\frac{1}{\delta}\right).$$

Computing the SVD of a matrix of this size dominates the runtime, giving the complexity in the theorem statement. Next, we would like to further approximate $R^\dagger\bar{\iota}(CC^\dagger)RA^\dagger b$. We will do this by estimating $RA^\dagger b$ by some vector u to $\varepsilon\sigma^3\|A\|^{-1}\|b\| = \varepsilon\|A\|_{\mathrm{F}}^2\|b\|K^{-1}\kappa^{-\frac{1}{2}}$ error, since then, using the bounds from Lemma 5.2,

$$\begin{aligned}\|R^\dagger\bar{\iota}(CC^\dagger)RA^\dagger b - R^\dagger\bar{\iota}(CC^\dagger)u\| &\le \left\|R^\dagger\sqrt{\bar{\iota}(CC^\dagger)}\right\|\left\|\sqrt{\bar{\iota}(CC^\dagger)}\right\|\|RA^\dagger b - u\| \\ &\lesssim \sqrt{\sigma^{-2} + \tfrac{\varepsilon}{\|A\|^2}}\sigma^{-2}(\varepsilon\sigma^3\|A\|^{-1}\|b\|) \lesssim \varepsilon\|A\|^{-1}\|b\|.\end{aligned}$$

We use Remark 4.13 to estimate $u(i) = R(i,\cdot)A^\dagger b$, for all $i \in [r]$, to $\varepsilon\|R(i,\cdot)\|\|A\|_{\mathrm{F}}\|b\|K^{-1}\kappa^{-\frac{1}{2}}$ error, with probability $\ge 1-\delta/r$. This takes $\mathcal{O}\left(\varphi\frac{K^2\kappa}{\varepsilon^2}\log\frac{r}{\delta}\right)$ samples for each of the r entries. This implies that $\hat{x} \coloneqq R^\dagger\bar{\iota}(CC^\dagger)u$ has the desired error and failure probability. Finally, we can use Lemmas 3.5 and 3.6 with matrix $R^\dagger$ and vector $\bar{\iota}(CC^\dagger)u$ to get $\mathrm{SQ}_\phi(\hat{x})$ for

$$\begin{aligned}\phi &= \varphi r\frac{\sum_{s=1}^r |[\bar{\iota}(CC^\dagger)u](s)|^2\|R(s,\cdot)\|^2}{\|\hat{x}\|^2} \\ &= \varphi^2\frac{\|\bar{\iota}(CC^\dagger)u\|^2\|A\|_{\mathrm{F}}^2}{\|\hat{x}\|^2} && \text{by } \|R(s,\cdot)\| \le \|A\|_{\mathrm{F}}\sqrt{\varphi/r} \\ &\le \varphi^2\frac{(\|\bar{\iota}(CC^\dagger)R\|\|A^\dagger\|\|b\| + \|\bar{\iota}(CC^\dagger)\|\|RA^\dagger b - u\|)^2\|A\|_{\mathrm{F}}^2}{\|\hat{x}\|^2} && \text{by linear algebra} \\ &\lesssim \varphi^2\frac{(\sigma^{-3}\|A\|\|b\| + \sigma^{-4}\varepsilon\sigma^3\|b\|/\|A\|)^2\|A\|_{\mathrm{F}}^2}{\|\hat{x}\|^2} && \text{by prior bounds} \\ &\lesssim \varphi^2\frac{\sigma^{-6}\|A\|^2\|b\|^2\|A\|_{\mathrm{F}}^2}{\|\hat{x}\|^2} && \text{by } \varepsilon \lesssim \|A\|^2/\sigma^2 \\ &\le \varphi^2 K\kappa^2\frac{\|x^*\|^2}{\|\hat{x}\|^2}, && \text{by } \|A\|^{-1}\|b\| \le \|x^*\|\end{aligned}$$

so $\widetilde{\mathbf{sq}}_\phi(\hat{x}) = \phi\,\mathbf{sq}_\phi(\hat{x})\log\frac{1}{\delta} = \mathcal{O}\left(r\varphi^2 K\kappa^2\frac{\|x^*\|^2}{\|\hat{x}\|^2}\log\frac{1}{\delta}\right)$. □

6.6 Support vector machines

In this section, we use our framework to dequantize Rebentrost, Mohseni, and Lloyd's quantum support vector machine [RML14], which was previously noted to be possible by Ding, Bao, and

Huang [DBH21]. Mathematically, the support vector machine is a simple machine learning model attempting to label points in $\mathbb{R}^m$ as $+1$ or -1. Given input data points $x_1, \ldots, x_m \in \mathbb{R}^n$ and their corresponding labels $y \in \{\pm 1\}^m$. Let $w \in \mathbb{R}^n$ and $b \in \mathbb{R}$ be the specification of hyperplanes separating these points. It is possible that no such hyperplane satisfies all the constraints. To resolve this, we add a slack vector $e \in \mathbb{R}^m$ such that $e(j) \geq 0$ for $j \in [m]$. We want to minimize the squared norm of the residuals:

$$\begin{aligned} \min_{w,b} \quad & \frac{1}{2}\frac{\|w\|^2}{2} + \frac{\gamma}{2}\|e\|^2 \\ \text{s.t.} \quad & y(i)(w^T x_i + b) = 1 - e(i), \quad \forall i \in [m]. \end{aligned}$$

The dual of this problem is to maximize over the Karush-Kuhn-Tucker multipliers of a Lagrange function, taking partial derivatives of which yields the linear system

$$\begin{bmatrix} 0 & \vec{1}^T \\ \vec{1} & XX^T + \gamma^{-1} I \end{bmatrix} \begin{bmatrix} b \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 \\ y \end{bmatrix}, \tag{12}$$

where $\vec{1}$ is the all-ones vector and $X = \{x_1, \ldots, x_m\} \in \mathbb{C}^{m \times n}$. Call the above $m+1 \times m+1$ matrix F, and $\hat{F} \coloneqq F / \operatorname{Tr}(F)$.

The quantum algorithm, given X and y in QRAM, outputs a quantum state $|\hat{F}^+_{\lambda,0.01}\begin{bmatrix} 0 \\ y \end{bmatrix}\rangle$ (Definition 6.13) in $\widetilde{\mathcal{O}}\big(\frac{1}{\lambda^3 \varepsilon^3} \operatorname{polylog}(mn)\big)$ time. The quantum-inspired analogue is as follows.

Problem 6.16. Given $\mathrm{SQ}(X) \in \mathbb{R}^{m \times n}$ and $\mathrm{SQ}(y) \in \mathbb{R}^m$, for $\|\hat{F}\| \leq 1$, output $\mathrm{SQ}_\phi(v) \in \mathbb{R}^{m+1}$ such that $\|\hat{x} - \hat{F}^+_{\lambda,\eta}\begin{bmatrix} 0 \\ y \end{bmatrix}\| \leq \varepsilon \|\hat{F}^+_{\lambda,\eta}\begin{bmatrix} 0 \\ y \end{bmatrix}\|$ with probability $\geq 1 - \delta$.

Note that we must assume $\|\hat{F}\| \leq 1$; the quantum algorithm makes the same assumption[21]. Another dequantization was reported in [DBH21], which, assuming X is strictly low-rank (with minimum singular value σ), outputs a description of $(XX^T)^+ y$ that can be used to classify points. This can be done neatly in our framework: express $(XX^T)^+$ (or, more generally, $(XX^T)^+_{\sigma,\eta}$) as $Xf(X^TX)X^T$ for the appropriate choice of f. Then, use Theorem 5.1 to approximate $f(X^TX) \approx R^T Z R$ and use Lemma 4.4 to approximate $XR^T \approx CW^T$. This gives an approximate "CUC" decomposition of the desired matrix, since $Xf(X^TX)X^T \approx XR^TZRX^T \approx CW^TZWC^T$, which we can use for whatever purpose we like.

For our solution to Problem 6.16, though, we simply reduce to matrix inversion as described in Section 6.5: we first get $\mathrm{SQ}_\phi(\hat{F})$, and then we apply Corollary 6.15 to complete. Section VI.C of [DBH21] claims to dequantize this version, but gives no correctness bounds[22] or runtime bounds (beyond arguing it is polynomial in the desired parameters).

Corollary 6.17. *For $0 < \varepsilon \lesssim 1$ and $\eta \leq 0.99$, we can solve Problem 6.16 in $\widetilde{\mathcal{O}}\big(\lambda^{-28}\eta^{-6}\varepsilon^{-6}\log^3\frac{1}{\delta}\big)$ time, where we get $\mathrm{SQ}_\phi(v)$ for $\widetilde{\mathbf{sq}}_\phi(v) = \widetilde{\mathcal{O}}\big(\lambda^{-14}\eta^{-2}\varepsilon^{-4}\log^2(\frac{1}{\delta})\log(\frac{m}{\delta})\big)$.*

The runtimes in the statement are not particularly tight, but we chose the form to mirror the runtime of the QSVM algorithm, which similarly depends polynomially on $\frac{1}{\lambda}$ and $\frac{1}{\eta}$.

Proof. Consider constructing $\mathrm{SQ}_\varphi(K) \in \mathbb{C}^{m \times m}$ as follows. To query an entry $K(i,j)$, we estimate $X(i,\cdot)X(j,\cdot)^T$ to $\varepsilon\|X(i,\cdot)\|\|X(j,\cdot)\|$ error. We define $K(i,j)$ to be this estimate. Using Lemma 4.12,

[21] The algorithm as written in [RML14] assumes that $\|F\| \leq 1$; we confirmed with an author that this is a typo.

[22] The correctness of this dequantization is unclear, since the approximations performed in this section incur significant errors.

we can do this in $\mathcal{O}\big(\frac{1}{\varepsilon^2}\log\frac{q}{\delta}\big)$ time. q here refers to the number of times the query oracle is used, so in total the subsequent algorithm will only have an errant query with probability $\geq 1-\delta$. (q will not appear in the runtime because it's folded into a polylog term.) Then, we can take $\tilde{K} \coloneqq xx^T$, where $x \in \mathbb{R}^m$ is the vector of row norms of X, since by Cauchy–Schwarz,

$$K(i,j) \leq X(i,\cdot)X(j,\cdot)^T + \varepsilon\|X(i,\cdot)\|\|X(j,\cdot)\| \leq (1+\varepsilon)\|X(i,\cdot)\|\|X(j,\cdot)\| = \tilde{K}(i,j).$$

Since we have $\mathrm{SQ}(x)$ from $\mathrm{SQ}(X)$, we have $\mathrm{SQ}(\tilde{K})$ with $\mathbf{sq}(\tilde{K}) = \mathcal{O}(1)$ by Lemma 3.8. $\|\tilde{K}\|_{\mathrm{F}}^2 = (1+\varepsilon)^2\|X\|_{\mathrm{F}}^4$, so we have $\mathrm{SQ}_\varphi(K)$ for $\varphi = (1+\varepsilon)^2\frac{\|X\|_{\mathrm{F}}^4}{\|K\|_{\mathrm{F}}^2}$. We can trivially get $\mathrm{SQ}(L)$ for $L \coloneqq \left[\begin{smallmatrix} 0 & \vec{1}^T \\ \vec{1} & \gamma^{-1}I \end{smallmatrix}\right]$ with $\mathbf{sq}(L) = \mathcal{O}(1)$. Our approximation to $\hat{F}$ is

$$M \coloneqq \frac{1}{\mathrm{Tr}(F)}\left(L + \left[\begin{smallmatrix} 0 & \vec{0}^T \\ \vec{0} & K \end{smallmatrix}\right]\right); \qquad \|M - \hat{F}\| \leq \frac{1}{\mathrm{Tr}(F)}\|K - XX^T\|_{\mathrm{F}} \leq \frac{1}{\mathrm{Tr}(F)}\varepsilon\|X\|_{\mathrm{F}}^2 \leq \varepsilon.$$

Using Lemma 3.9, we have $\mathrm{SQ}_{\varphi'}(M)$ with

$$\varphi' = \frac{2((1+\varepsilon)^2\frac{\|X\|_{\mathrm{F}}^4}{\|K\|_{\mathrm{F}}^2}\|K\|_{\mathrm{F}}^2 + \|L\|_{\mathrm{F}}^2)}{\mathrm{Tr}(F)^2\|M\|_{\mathrm{F}}^2} \lesssim \frac{\|X\|_{\mathrm{F}}^4 + \gamma^{-2}m + 2m}{(\|X\|_{\mathrm{F}}^2 + m\gamma^{-1})^2\|M\|_{\mathrm{F}}^2} \lesssim \frac{1}{\|M\|_{\mathrm{F}}^2},$$

where the last inequality uses that $\mathrm{Tr}(F) \geq \sqrt{m}$, which follows from $\|\hat{F}\| \leq 1$:

$$1 = \|\hat{F}\|\left\|\left[\begin{smallmatrix} 0 \\ \vec{1}/\sqrt{m} \end{smallmatrix}\right]\right\| \geq \left\|\hat{F}\left[\begin{smallmatrix} 0 \\ \vec{1}/\sqrt{m} \end{smallmatrix}\right]\right\| \geq \frac{\sqrt{m}}{\mathrm{Tr}(F)}.$$

Note that we can compute $\mathrm{Tr}(F)$ given $\mathrm{SQ}(X)$. So, applying Corollary 6.15, we can get the desired $\mathrm{SQ}_\phi(v)$ in runtime

$$\widetilde{\mathcal{O}}\left(\frac{\varphi^6\|M\|_{\mathrm{F}}^6\|M\|^{22}}{\lambda^{28}\eta^6\varepsilon^6}\log^3\frac{1}{\delta}\right) \lesssim \widetilde{\mathcal{O}}\left(\frac{\|M\|^{22}}{\|M\|_{\mathrm{F}}^6\lambda^{28}\eta^6\varepsilon^6}\log^3\frac{1}{\delta}\right) \lesssim \widetilde{\mathcal{O}}\left(\frac{1}{\lambda^{28}\eta^6\varepsilon^6}\log^3\frac{1}{\delta}\right).$$

Here, we used that $\|M\| \leq \|M\|_{\mathrm{F}} \lesssim 1$, which we know since $\varphi' \geq 1$ (by our definition of oversampling and query access). That $\mathrm{Q}(M) = \mathcal{O}\big(\frac{1}{\varepsilon^2}\log\frac{q}{\delta}\big)$ does not affect the runtime, since the dominating cost is still the SVD. On the other hand, this does come into play for the runtime for sampling:

$$\widetilde{\mathbf{sq}}_\phi(v) = \widetilde{\mathcal{O}}\left(\frac{\varphi^4\|M\|_{\mathrm{F}}^4\|M\|^{10}}{\eta^2\varepsilon^2}\log^2\left(\frac{1}{\delta}\right)\frac{1}{\varepsilon^2}\log\left(\frac{m}{\delta}\right)\right).$$

We take $q = m$ to guarantee that all future queries will be correct with probability $\geq 1-\delta$. □

The normalization used by the quantum and quantum-inspired SVM algorithms means that these algorithms fail when X has too small Frobenius norm, since then the singular values from XX^T are all filtered out. In Appendix B, we describe an alternative method that relies less on normalization assumptions, instead simply computing F^+. This is possible if we depend on $\|X\|_{\mathrm{F}}^2\gamma$ in the runtime. Recall from Eq. (12) that we regularize by adding $\gamma^{-1}I$, so γ^{-1} acts as a singular value lower bound and $\|X\|_{\mathrm{F}}^2\gamma$ implicitly constrains.

Corollary 6.18. *Given $\mathrm{SQ}(X^T)$ and $\mathrm{SQ}(y)$, with probability $\geq 1-\delta$, we can output a real number $\hat{b}$ such that $|b-\hat{b}| \leq \varepsilon(1+b)$ and $\mathrm{SQ}_\phi(\hat{\alpha})$ such that $\|\hat{\alpha}-\alpha\| \leq \varepsilon\gamma\|y\|$, where α and b come from Eq. (12). Our algorithm runs in $\widetilde{\mathcal{O}}\big(\|X\|_{\mathrm{F}}^6\|X\|^{16}\gamma^{11}\varepsilon^{-6}\log^3\frac{1}{\delta}\big)$ time, with $\widetilde{\mathbf{sq}}_\phi(\hat{\alpha}) = \widetilde{\mathcal{O}}\Big(\|X\|_{\mathrm{F}}^4\|X\|^6\gamma^5\frac{\gamma^2 m}{\|\hat{\alpha}\|^2}\varepsilon^{-4}\log^2\frac{1}{\delta}\Big)$. Note that when $\gamma^{-1/2}$ is chosen to be sufficiently large (e.g. $O(\|X\|_{\mathrm{F}})$) and $\|\alpha\| = \Omega(\gamma\|y\|)$, this runtime is dimension-independent.*

Notice that $\varepsilon\gamma\|y\|$ is the right notion, since γ is an upper bound on the spectral norm of the inverse of the matrix in Eq. (12). We assume $\mathrm{SQ}(X^T)$ instead of $\mathrm{SQ}(X)$ for convenience, though both are possible via the observation that $f(XX^T) = Xf(X^TX)X^T$.

6.7 Hamiltonian simulation

The problem of simulating the dynamics of quantum systems was the original motivation for quantum computers proposed by Feynman [Fey82]. Specifically, given a Hamiltonian H, a quantum state $|\psi\rangle$, a time $t > 0$, and a desired error $\varepsilon > 0$, we ask to prepare a quantum state $|\psi_t\rangle$ such that

$$\||\psi_t\rangle - e^{iHt}|\psi\rangle\| \leq \varepsilon.$$

This problem, known as Hamiltonian simulation, sees wide application, including in quantum physics and quantum chemistry. A rich literature has developed on quantum algorithms for Hamiltonian simulation [Llo96, ATS03, BCK15], with an optimal quantum algorithm for simulating sparse Hamiltonians given in [LC17]. In this subsection, we apply our framework to develop classical algorithms for Hamiltonian simulation. Specifically, we ask:

Problem 6.19. Consider a Hermitian matrix $H \in \mathbb{C}^{n\times n}$, a unit vector $b \in \mathbb{C}^n$, and error parameters $\varepsilon, \delta > 0$. Given $\mathrm{SQ}(H)$ and $\mathrm{SQ}(b)$, output $\mathrm{SQ}_\phi(\hat{b})$ with probability $\geq 1-\delta$ for some $\hat{b} \in \mathbb{C}^n$ satisfying $\|\hat{b} - e^{iH}b\| \leq \varepsilon$.

We give two algorithms that are fundamentally the same, but operate in different regimes: the first works for low-rank H, and the second for arbitrary H.

Corollary 6.20. *Suppose H has minimum singular value σ and $\varepsilon < \min(0.5, \sigma)$. We can solve Problem 6.19 in $\widetilde{\mathcal{O}}\Big(\frac{\|H\|_{\mathrm{F}}^6\|H\|^{16}}{\sigma^{16}\varepsilon^6}\log^3\frac{1}{\delta}\Big)$ time, giving $\mathrm{SQ}_\phi(\hat{b})$ with $\widetilde{\mathbf{sq}}_\phi(\hat{b}) = \widetilde{\mathcal{O}}\Big(\frac{\|H\|_{\mathrm{F}}^4\|H\|^8}{\sigma^8\varepsilon^4}\log^3\frac{1}{\delta}\Big)$.*

This runtime is dimensionless in a certain sense. The natural error bound to require is that $\|\hat{b} - e^{iH}b\| \leq \varepsilon\|H\|$, since $|\frac{d}{dx}(e^{-i\|H\|x})| = \|H\|$. So, if we rescale ε to $\varepsilon\|H\|$, the runtime is $\widetilde{\mathcal{O}}\Big(\frac{\|H\|_{\mathrm{F}}^6\|H\|^{10}}{\sigma^{16}\varepsilon^6}\log^3\frac{1}{\delta}\Big)$, which is dimensionless. The runtime of the algorithm in the following corollary does not have this property, so its scaling with $\|H\|$ is worse, despite being faster for, say, $\|H\| = 1$.

Corollary 6.21. *For $\varepsilon < \min(0.5, \|H\|^3)$, we can solve Problem 6.19 in $\widetilde{\mathcal{O}}\big(\|H\|^{16}\|H\|_{\mathrm{F}}^6\varepsilon^{-6}\log^3\frac{1}{\delta}\big)$ time, giving $\mathrm{SQ}_\phi(\hat{b})$ with $\widetilde{\mathbf{sq}}_\phi(\hat{b}) = \widetilde{\mathcal{O}}\big(\|H\|^8\|H\|_{\mathrm{F}}^4\varepsilon^{-4}\log^3\frac{1}{\delta}\big)$.*

Our strategy proceeds as follows: consider a generic function $f(x)$ and Hermitian H. We can write $f(x)$ as a sum of an even function $a(x) := \frac{1}{2}(f(x) + f(-x))$ and an odd function $b(x) := \frac{1}{2}(f(x) - f(-x))$. For the even function, we can use Theorem 5.1 to approximate it via the function $f_a(x) := a(\sqrt{x})$; the odd function can be written as H times an even function, which we approximate using Theorem 5.1 for $f_b(x) := b(\sqrt{x})/\sqrt{x}$. In other words, $f(H) = f_a(H^\dagger H) + f_b(H^\dagger H)H$. Since $|a'(x)|, |b'(x)| \leq |f'(x)|$, the Lipschitz constants don't blow up by splitting f into even and odd parts.

Now, we specialize to Hamiltonian simulation. We first rewrite the problem, using the function $\mathrm{sinc}(x) := \sin(x)/x$.

$$e^{iH}b = \cos(H)b + i\cdot\mathrm{sinc}(H)Hb = f_{\cos}(H^\dagger H)b + f_{\mathrm{sinc}}(H^\dagger H)Hb,$$

where $f_{\cos}(\lambda) := \cos(\sqrt{\lambda})$ and $f_{\mathrm{sinc}}(\lambda) := i\cdot\mathrm{sinc}(\sqrt{\lambda})$. When applying Theorem 5.1 on $f_{\cos}$ and f_{sinc},

we will use the following bounds on the smoothness of $f_{\cos}$ and f_{sinc}.

$$
\begin{aligned}
|f'_{\cos}(x)| &= \left|\frac{\sin(\sqrt{x})}{2\sqrt{x}}\right| \le \min\left(\frac{1}{2}, \frac{1}{2\sqrt{x}}\right) \\
|\bar{f}'_{\cos}(x)| &= \left|\frac{2 - 2\cos(\sqrt{x}) - \sqrt{x}\sin(\sqrt{x})}{2x^2}\right| \le \min\left(\frac{1}{24}, \frac{5}{2x^{3/2}}\right) \\
|f'_{\mathrm{sinc}}(x)| &= \left|\frac{\sqrt{x}\cos(\sqrt{x}) - \sin(\sqrt{x})}{2x^{3/2}}\right| \le \min\left(\frac{1}{4}, \frac{1}{x}\right) \\
|\bar{f}'_{\mathrm{sinc}}(x)| &= \left|\frac{2\sqrt{x} + \sqrt{x}\cos(\sqrt{x}) - 3\sin(\sqrt{x})}{2x^{5/2}}\right| \le \min\left(\frac{1}{60}, \frac{3}{x^2}\right)
\end{aligned}
$$

We separate these bounds into the case where $x \ge 1$, which we use when we assume H has a minimum singular value, and the case where $x < 1$, which we use for arbitrary H.

Proof of Corollary 6.21. Using the Lipschitz bounds above with Theorem 5.1, we can find $R_{\cos} \in \mathbb{C}^{r_{\cos} \times n}, C_{\cos} \in \mathbb{C}^{r_{\cos} \times c_{\cos}}, R_{\mathrm{sinc}} \in \mathbb{C}^{r_{\mathrm{sinc}} \times n}, C_{\mathrm{sinc}} \in \mathbb{C}^{r_{\mathrm{sinc}} \times c_{\mathrm{sinc}}}$ such that

$$\|R_{\cos}^\dagger \bar{f}_{\cos}(C_{\cos} C_{\cos}^\dagger) R_{\cos} + I - f_{\cos}(H^\dagger H)\| \le \varepsilon \tag{13}$$

$$\|R_{\mathrm{sinc}}^\dagger \bar{f}_{\mathrm{sinc}}(C_{\mathrm{sinc}} C_{\mathrm{sinc}}^\dagger) R_{\mathrm{sinc}} + i \cdot I - f_{\mathrm{sinc}}(H^\dagger H)\| \le \frac{\varepsilon}{\|H\|} \tag{14}$$

where, using that our Lipschitz constants are all bounded by constants,

$$
\begin{aligned}
r_{\cos} &= \widetilde{\mathcal{O}}\left(\|H\|_{\mathrm{F}}^2 \|H\|^2 \varepsilon^{-2} \log\frac{1}{\delta}\right) & c_{\cos} &= \widetilde{\mathcal{O}}\left(\|H\|_{\mathrm{F}}^2 \|H\|^6 \varepsilon^{-2} \log\frac{1}{\delta}\right) \\
r_{\mathrm{sinc}} &= \widetilde{\mathcal{O}}\left(\|H\|_{\mathrm{F}}^2 \|H\|^4 \varepsilon^{-2} \log\frac{1}{\delta}\right) & c_{\mathrm{sinc}} &= \widetilde{\mathcal{O}}\left(\|H\|_{\mathrm{F}}^2 \|H\|^8 \varepsilon^{-2} \log\frac{1}{\delta}\right).
\end{aligned}
$$

As a consequence,

$$\left\| e^{iH} b - \left(R_{\cos}^\dagger \bar{f}_{\cos}(C_{\cos} C_{\cos}^\dagger) R_{\cos} b + b + R_{\mathrm{sinc}}^\dagger \bar{f}_{\mathrm{sinc}}(C_{\mathrm{sinc}} C_{\mathrm{sinc}}^\dagger) R_{\mathrm{sinc}} H b + iHb \right) \right\| \lesssim \varepsilon.$$

Note that, by Lemma 5.2, $\|R_{\cos}\| \lesssim \|H\|$, $\|\bar{f}_{\cos}(C_{\cos} C_{\cos}^\dagger)\| \lesssim 1$, and $\|R_{\cos}^\dagger \sqrt{\bar{f}_{\cos}(C_{\cos} C_{\cos}^\dagger)}\| \lesssim 1$; the same bounds hold for the sinc analogues. We now approximate using Lemma 4.6 four times.

1. We approximate $R_{\cos} b \approx u$ to $\varepsilon\|b\|$ error, requiring $\mathcal{O}\big(\|H\|_{\mathrm{F}}^2 \varepsilon^{-2} \log\frac{1}{\delta}\big)$ samples.
2. We approximate $R_{\mathrm{sinc}} H \approx WC$ to ε error, requiring $\mathcal{O}\big(\|H\|_{\mathrm{F}}^4 \varepsilon^{-2} \log\frac{1}{\delta}\big)$ samples.
3. We approximate $Cb \approx v$ to $\varepsilon\|H\|_{\mathrm{F}}^{-1}\|b\|$ error, requiring $\mathcal{O}\big(\|H\|_{\mathrm{F}}^4 \varepsilon^{-2} \log\frac{1}{\delta}\big)$ samples.
4. We approximate $Hb \approx R^\dagger w$ to $\varepsilon\|b\|$ accuracy, requiring $r := \mathcal{O}\big(\|H\|_{\mathrm{F}}^2 \varepsilon^{-2} \log\frac{1}{\delta}\big)$ samples.

Our output will be

$$\hat{b} := R_{\cos}^\dagger \bar{f}_{\cos}(C_{\cos} C_{\cos}^\dagger) u + b + R_{\mathrm{sinc}}^\dagger \bar{f}_{\mathrm{sinc}}(C_{\mathrm{sinc}} C_{\mathrm{sinc}}^\dagger) W v + i R^\dagger w,$$

which is close to $e^{iH} b$ because

$$
\begin{aligned}
&\left\| \hat{b} - \left(R_{\cos}^\dagger \bar{f}_{\cos}(C_{\cos} C_{\cos}^\dagger) R_{\cos} b + b + R_{\mathrm{sinc}}^\dagger \bar{f}_{\mathrm{sinc}}(C_{\mathrm{sinc}} C_{\mathrm{sinc}}^\dagger) R_{\mathrm{sinc}} H b + iHb \right) \right\| \\
&\le \|R_{\cos}^\dagger \bar{f}_{\cos}(C_{\cos} C_{\cos}^\dagger)(u - R_{\cos} b)\| + \|R_{\mathrm{sinc}}^\dagger \bar{f}_{\mathrm{sinc}}(C_{\mathrm{sinc}} C_{\mathrm{sinc}}^\dagger)(R_{\mathrm{sinc}} H - WC) b\| \\
&\qquad + \|R_{\mathrm{sinc}}^\dagger \bar{f}_{\mathrm{sinc}}(C_{\mathrm{sinc}} C_{\mathrm{sinc}}^\dagger) W (Cb - v)\| + \|iR^\dagger w - iHb\| \\
&\lesssim \|u - R_{\cos} b\| + \|R_{\mathrm{sinc}} H - WC\|\|b\| + \|H\|_{\mathrm{F}} \|Cb - v\| + \|R^\dagger w - Hb\| \le 4\varepsilon\|b\|.
\end{aligned}
$$

Now, we have expressed $\hat{b}$ as a linear combination of a small number of vectors, all of which we have sampling and query access to. We can complete using Lemmas 3.5 and 3.6, where the matrix is the concatenation $(R_{\cos}^\dagger \mid b \mid R_{\text{sinc}}^\dagger \mid i \cdot R^\dagger)$, and the vector is the concatenation $(\bar{f}_{\cos}(C_{\cos}C_{\cos}^\dagger)u \mid 1 \mid \bar{f}_{\text{sinc}}(C_{\text{sinc}}C_{\text{sinc}}^\dagger)Wv \mid w)$. The length of this vector is $r_{\cos} + 1 + r_{\text{sinc}} + r \lesssim r_{\text{sinc}}$. We get $\mathrm{SQ}_\phi(\hat{b})$ where

$$\begin{aligned}
\phi &\lesssim r_{\text{sinc}}\Big(\frac{\|H\|_{\mathrm{F}}^2}{r_{\cos}}\|\bar{f}_{\cos}(C_{\cos}C_{\cos}^\dagger)u\|^2 + \|b\|^2 + \frac{\|H\|_{\mathrm{F}}^2}{r_{\text{sinc}}}\|\bar{f}_{\text{sinc}}(C_{\text{sinc}}C_{\text{sinc}}^\dagger)Wv\|^2 + \frac{\|H\|_{\mathrm{F}}^2}{r}\|w\|^2\Big)\|\hat{b}\|^{-2} \\
&\lesssim \Big(\frac{r_{\text{sinc}}}{r_{\cos}}\|H\|_{\mathrm{F}}^2(1+\varepsilon)^2\|b\|^2 + r_{\text{sinc}}\|b\|^2 + \|H\|_{\mathrm{F}}^2(1+\varepsilon)^2\|b\|^2 + \frac{r_{\text{sinc}}}{r}\|H\|_{\mathrm{F}}^2\|b\|^2\Big)\|b\|^{-2} \\
&= \widetilde{\mathcal{O}}\big(\|H\|_{\mathrm{F}}^2\|H\|^2 + r_{\text{sinc}} + \|H\|_{\mathrm{F}}^2 + \|H\|_{\mathrm{F}}^2\|H\|^4\big) = \widetilde{\mathcal{O}}(r_{\text{sinc}}).
\end{aligned}$$

In the second inequality, we use the same bounds for proving $\|\hat{b} - e^{iH}b\| \le \varepsilon$, repurposed to argue that all approximations are sufficiently close to the values they are estimating, up to relative error. So, $\widetilde{\mathbf{sq}}_\phi(\hat{b}) = \widetilde{\mathcal{O}}(r_{\text{sinc}}^2 \log \frac{1}{\delta})$. □

Proof of Corollary 6.20. Our approach is the same, though with different parameters. For Theorem 5.1, we use that in the interval $[\sigma^2/2, \infty)$, $f_{\cos}$ has Lipschitz constants of $L = O(1/\sigma)$ and $\bar{L} = O(1/\sigma^3)$ and f_{sinc} has $L = O(1/\sigma^2)$ and $\bar{L} = O(1/\sigma^4)$. So, if we take

$$\begin{aligned}
r_{\cos} &= \widetilde{\mathcal{O}}\left(\|H\|^2\frac{\|H\|_{\mathrm{F}}^2}{\sigma^2}\varepsilon^{-2}\log\frac{1}{\delta}\right) & c_{\cos} &= \widetilde{\mathcal{O}}\left(\|H\|^2\frac{\|H\|_{\mathrm{F}}^2\|H\|^4}{\sigma^6}\varepsilon^{-2}\log\frac{1}{\delta}\right) \\
r_{\text{sinc}} &= \widetilde{\mathcal{O}}\left(\|H\|^2\frac{\|H\|_{\mathrm{F}}^2\|H\|^2}{\sigma^4}\varepsilon^{-2}\log\frac{1}{\delta}\right) & c_{\text{sinc}} &= \widetilde{\mathcal{O}}\left(\|H\|^2\frac{\|H\|_{\mathrm{F}}^2\|H\|^6}{\sigma^8}\varepsilon^{-2}\log\frac{1}{\delta}\right),
\end{aligned}$$

all the conditions of Theorem 5.1 are satisfied: in particular, $\sigma^2/2 > \bar{\varepsilon}$ in both cases, up to rescaling ε by a constant factor:

$$\begin{aligned}
\bar{\varepsilon}_{\cos} &\lesssim \|H\|\|H\|_{\mathrm{F}}\frac{\varepsilon\sigma}{\|H\|\|H\|_{\mathrm{F}}} = \varepsilon\sigma \le \sigma^2 \\
\bar{\varepsilon}_{\text{sinc}} &\lesssim \|H\|\|H\|_{\mathrm{F}}\frac{\varepsilon\sigma^2}{\|H\|^2\|H\|_{\mathrm{F}}} = \varepsilon\sigma^2\|H\|^{-1} \le \sigma^2
\end{aligned}$$

Here, we used our initial assumption that $\varepsilon \le \sigma$. So, the bounds Eqs. (13) and (14) hold. Note that, by Lemma 5.2, $\|R_{\cos}\| \lesssim \|H\|$, $\|\bar{f}_{\cos}(C_{\cos}C_{\cos}^\dagger)\| \lesssim \sigma^{-2}$, and $\|R_{\cos}^\dagger\sqrt{\bar{f}_{\cos}(C_{\cos}C_{\cos}^\dagger)}\| \le 1$; the same bounds hold for the sinc analogues. We now approximate using Lemma 4.6 four times.

1. We approximate $R_{\cos}b \approx u$ to $\varepsilon\sigma\|b\|$ error, requiring $\mathcal{O}\big(\|H\|_{\mathrm{F}}^2\sigma^{-2}\varepsilon^{-2}\log\frac{1}{\delta}\big)$ samples.
2. We approximate $R_{\text{sinc}}H \approx WC$ to $\varepsilon\sigma$ error, requiring $\mathcal{O}\big(\|H\|_{\mathrm{F}}^4\sigma^{-2}\varepsilon^{-2}\log\frac{1}{\delta}\big)$ samples.
3. We approximate $Cb \approx v$ to $\varepsilon\sigma\|H\|_{\mathrm{F}}^{-1}\|b\|$ error, requiring $\mathcal{O}\big(\|H\|_{\mathrm{F}}^4\sigma^{-2}\varepsilon^{-2}\log\frac{1}{\delta}\big)$ samples.
4. We approximate $Hb \approx R^\dagger w$ to $\varepsilon\|b\|$ accuracy, requiring $r := \mathcal{O}\big(\|H\|_{\mathrm{F}}^2\varepsilon^{-2}\log\frac{1}{\delta}\big)$ samples.

Our output will be

$$\hat{b} := R_{\cos}^\dagger\bar{f}_{\cos}(C_{\cos}C_{\cos}^\dagger)u + b + R_{\text{sinc}}^\dagger\bar{f}_{\text{sinc}}(C_{\text{sinc}}C_{\text{sinc}}^\dagger)Wv + iR^\dagger w,$$

which is close to $e^{iH}b$ by the argument

$$\begin{aligned}
&\left\|\hat{b} - \left(R_{\cos}^\dagger \bar{f}_{\cos}(C_{\cos}C_{\cos}^\dagger)R_{\cos}b + b + R_{\sin c}^\dagger \bar{f}_{\sin c}(C_{\sin c}C_{\sin c}^\dagger)R_{\sin c}Hb + iHb\right)\right\| \\
&\leq \|R_{\cos}^\dagger \bar{f}_{\cos}(C_{\cos}C_{\cos}^\dagger)(u - R_{\cos}b)\| + \|R_{\sin c}^\dagger \bar{f}_{\sin c}(C_{\sin c}C_{\sin c}^\dagger)(R_{\sin c}H - WC)b\| \\
&\qquad + \|R_{\sin c}^\dagger \bar{f}_{\sin c}(C_{\sin c}C_{\sin c}^\dagger)W(Cb - v)\| + \|iR^\dagger w - iHb\| \\
&\lesssim \sigma^{-1}\|u - R_{\cos}b\| + \sigma^{-1}\|R_{\sin c}H - WC\|\|b\| + \sigma^{-1}\|H\|_{\mathrm{F}}\|Cb - v\| + \|R^\dagger w - Hb\| \leq 4\varepsilon\|b\|
\end{aligned}$$

Now, we have expressed $\hat{b}$ as a linear combination of a small number of vectors, all of which we have sampling and query access to. We can complete using Lemmas 3.5 and 3.6, where the matrix is the concatenation $(R_{\cos}^\dagger \mid b \mid R_{\sin c}^\dagger \mid i \cdot R^\dagger)$, and the vector is the concatenation $(\bar{f}_{\cos}(C_{\cos}C_{\cos}^\dagger)u \mid 1 \mid \bar{f}_{\sin c}(C_{\sin c}C_{\sin c}^\dagger)Wv \mid w)$. The length of this vector is $r_{\cos} + 1 + r_{\sin c} + r \lesssim r_{\sin c}$. We get $\mathrm{SQ}_\phi(\hat{b})$ where

$$\begin{aligned}
\phi &\lesssim r_{\sin c}\left(\frac{\|H\|_{\mathrm{F}}^2}{r_{\cos}}\|\bar{f}_{\cos}(C_{\cos}C_{\cos}^\dagger)u\|^2 + \|b\|^2 + \frac{\|H\|_{\mathrm{F}}^2}{r_{\sin c}}\|\bar{f}_{\sin c}(C_{\sin c}C_{\sin c}^\dagger)Wv\|^2 + \frac{\|H\|_{\mathrm{F}}^2}{r}\|w\|^2\right)\|\hat{b}\|^{-2} \\
&\lesssim \left(\frac{r_{\sin c}}{r_{\cos}}\|H\|_{\mathrm{F}}^2\sigma^{-2}\|b\|^2 + r_{\sin c}\|b\|^2 + \|H\|_{\mathrm{F}}^2\sigma^{-2}\|b\|^2 + \frac{r_{\sin c}}{r}\|H\|_{\mathrm{F}}^2\|b\|^2\right)\|b\|^{-2} \\
&= \widetilde{\mathcal{O}}\left(\|H\|_{\mathrm{F}}^2\|H\|^2\sigma^{-4} + r_{\sin c} + \|H\|_{\mathrm{F}}^2\sigma^{-2} + \|H\|^4\sigma^{-4}\|H\|_{\mathrm{F}}^2\right) = \widetilde{\mathcal{O}}\left(r_{\sin c} + \frac{t^2\|H\|_{\mathrm{F}}^2}{\sigma^4}\right).
\end{aligned}$$

So, $\widetilde{\mathbf{sq}}_\phi(\hat{b}) = \widetilde{\mathcal{O}}\left(r_{\sin c}(r_{\sin c} + \|H\|_{\mathrm{F}}^2\|H\|^2\sigma^{-4})\log\frac{1}{\delta}\right)$. Since $\varepsilon < \sigma$, the $r_{\sin c}^2$ term dominates. □

Remark 6.22. In the case where H is not low-rank, we could still run a modified version of Corollary 6.20 to compute a modified "$\exp_{\sigma,\eta}(iH)$" where singular values below σ are smoothly thresholded away. Following the same logic as Definition 6.13, we could redefine $f_{\cos}$ such that $f_{\cos}(x) = 1$ for $x < \sigma^2(1-\eta)$, $f_{\cos}(x) = \cos(\sqrt{\lambda})$ for $x \geq \sigma^2$, and is a linear interpolation between the endpoints for the x in between (and $f_{\sin c}$ similarly). These functions have the same Lipschitz constants as their originals, up to factors of $\frac{1}{\eta}$, and give the desired behavior of "smoothing away" small singular values (though we do keep the 0th and 1st order terms of the exponential).

Remark 6.23. Our result generalizes those of Ref. [RWC+20], which achieves essentially the same result only in the much easier regime where H and b are sparse. They achieve a significant speedup due to these assumptions: note that when H is sparse, and a subsample of rows R is taken, $RR^\dagger$ can be computed in time independent of dimension; so, we only need to take a subsample of rows, and not of columns. More corners can be cut from our algorithm in this fashion. In summary, though our algorithm is significantly slower, their sparsity assumptions are essential for their fast runtime, and our framework can identify where these tradeoffs occur.

6.8 Semidefinite program solving

A recent line of inquiry in quantum computing [BS17, vAGGdW20, BKL+19, vAG19] focuses on finding quantum speedups for *semidefinite programs* (SDPs), a central topic in the theory of convex optimization with applications in algorithms design, operations research, and approximation algorithms. Chia, Li, Lin, and Wang [CLLW20] first noticed that quantum-inspired algorithms could dequantize these quantum algorithms in certain regimes. We improve on their result, giving an algorithm which is as general as the quantum algorithms, if the input is given classically (e.g., in a data-structure in RAM). Our goal is to solve the ε-feasibility problem; solving an SDP reduces by binary search to solving $\log(1/\varepsilon)$ instances of this feasibility problem.

Problem 6.24 (SDP ε-feasibility)**.** Given an $\varepsilon > 0$, m real numbers $b_1, \ldots, b_m \in \mathbb{R}$, and Hermitian $n \times n$ matrices $\mathrm{SQ}(A^{(1)}), \ldots, \mathrm{SQ}(A^{(m)})$ such that $-I \preceq A^{(i)} \preceq I$ for all $i \in [m]$, we define $\mathcal{S}_\varepsilon$ as the set of all X satisfying[23]

$$\begin{aligned} \mathrm{Tr}[A^{(i)}X] &\leq b_i + \varepsilon \quad \forall\, i \in [m]; \\ X &\succeq 0; \\ \mathrm{Tr}[X] &= 1. \end{aligned}$$

If $\mathcal{S}_\varepsilon = \varnothing$, output "infeasible". If $\mathcal{S}_0 \neq \varnothing$, output an $X \in \mathcal{S}_\varepsilon$. (If neither condition holds, either output is acceptable.)

Corollary 6.25. *Let $F \geq \max_{j\in[m]}(\|A^{(j)}\|_{\mathrm{F}})$, and suppose*[24] *$F = \Omega(1)$. Then we can solve Problem 6.24 with success probability $\geq 1 - \delta$ in cost*

$$\widetilde{\mathcal{O}}\left(\left(\frac{F^{18}}{\varepsilon^{40}}\log^{20}(n)\,\mathbf{sq}(A) + \frac{F^{22}}{\varepsilon^{46}}\log^{23}(n) + m\frac{F^{8}}{\varepsilon^{18}}\log^{8}(n)\,\mathbf{q}(A) + m\frac{F^{14}}{\varepsilon^{28}}\log^{13}(n)\right)\log^3\frac{1}{\delta}\right),$$

providing sampling and query access to a solution.

Assuming $\mathbf{sq}(A) = \widetilde{\mathcal{O}}(1)$, this runtime is $\widetilde{\mathcal{O}}\left(\frac{F^{22}}{\varepsilon^{46}}\log^{23}(n) + m\frac{F^{14}}{\varepsilon^{28}}\log^{13}(n)\right)$. For the same feasibility problem, the previous quantum-inspired SDP solver [CLLW20] proved a complexity bound $\widetilde{\mathcal{O}}\big(mr^{57}\varepsilon^{-92}\log^{37}(n)\big)$, assuming that the constraint matrices have rank at most r. Since the rank constraint implies that $\left\|A^{(\cdot)}\right\|_{\mathrm{F}} \leq \sqrt{r}$, under this assumption our algorithm has complexity $\widetilde{\mathcal{O}}\big(r^{11}\varepsilon^{-46}\log^{23}(n) + mr^{7}\varepsilon^{-28}\log^{13}(n)\big)$. So, our new algorithm both solves a more general problem and also greatly improves the runtime. The paper with the current best runtime for SDP solving does not discuss this precise model, but if we use the runtime they achieve in quantum state input model, making reasonable substitutions of $\gamma \to \frac{1}{\varepsilon}$ and $B \to F^2$, the corresponding quantum runtime is $\widetilde{\mathcal{O}}\left(\frac{F^7}{\varepsilon^{7.5}} + \frac{\sqrt{m}F^2}{\varepsilon^4}\right)$, up to polylog$(n)$ factors.

Like prior work on quantum algorithms for SDP-solving, we use the matrix multiplicative weights (MMW) framework [AK16, Kal07] to solve Problem 6.24. Corollary 6.25 immediately follows from running the algorithm this framework admits (Algorithm 1), where we solve an instance of the problem described in Lemma 6.26 with precision $\theta = \varepsilon/4$ in each of the $\mathcal{O}\big(\log(n)/\varepsilon^2\big)$ iterations.

MMW works as a zero-sum game with two players, where the first player wants to provide an $X \in \mathcal{S}_\varepsilon$, and the second player wants to find a violation for any proposed X, i.e., a $j \in [m]$ such that $\mathrm{Tr}[A^{(j)}X] > b_j + \varepsilon$. At the t^{th} round of the game, if the second player points out a violation j_t for the current solution X_t, the first player proposes a new solution

$$X_{t+1} \propto \exp[-\varepsilon(A^{(j_1)} + \cdots + A^{(j_t)})].$$

Solutions of this form are also known as *Gibbs states*. It is known that MMW solves the SDP ε-feasibility problem in $\mathcal{O}\left(\frac{\log n}{\varepsilon^2}\right)$ iterations; a proof can be found, e.g., in the work of Brandão, Kalev, Li, Lin, Svore, and Wu [BKL+19, Theorem 3] or in Lee, Raghavendra and Steurer [LRS15, Lemma 4.6].

Our task is to execute Lines 3 and 4 of Algorithm 1, for an implicitly defined matrix with the form given in Line 6.

[23] For simplicity, we assume here that X is normalized to have trace 1. This can be relaxed; for an example, see [vAGGdW20].

[24] Because of the normalization assumption that $\|A^{(\cdot)}\| \leq 1$, F is effectively a dimensionless "stable rank"-type constant, normalized by $\max_i \|A^{(i)}\|$.

Algorithm 1: MMW based feasibility testing algorithm for SDPs

Set $X_1 \coloneqq \frac{I_n}{n}$, and the number of iterations $T \coloneqq \frac{16 \log n}{\varepsilon^2}$;
for $t = 1, \ldots, T$ **do**
  find a $j_t \in [m]$ such that $\operatorname{Tr}[A^{(j_t)} X_t] > b_{j_t} + \frac{\varepsilon}{2}$
    or conclude correctly that $\operatorname{Tr}[A^{(j_t)} X_t] \leq b_{j_t} + \varepsilon$ for all $j \in [m]$
  if a $j_t \in [m]$ is found **then**
    $X_{t+1} \coloneqq \exp[-\frac{\varepsilon}{4} \sum_{i=1}^{t} A^{(j_i)}] / \operatorname{Tr}[\exp[-\frac{\varepsilon}{4} \sum_{i=1}^{t} A^{(j_i)}]]$
  else conclude that $X_t \in \mathcal{S}_\varepsilon$
    return X_t
end
If no solution found, conclude that the SDP is infeasible and terminate the algorithm

Lemma 6.26 ("Efficient" trace estimation)**.** *Consider the setting described in Corollary 6.25. Given* $\theta \in (0, 1]$, $t \leq \frac{\log(n)}{\theta^2}$ *and* $j_i \in [m]$ *for* $i \in [t]$, *defining* $H \coloneqq \exp[-\theta \sum_{i=1}^{t} A^{(j_i)}]$, *we can estimate* $\operatorname{Tr}(A^{(i)} H) / \operatorname{Tr}(H)$ *with success probability* $\geq 1 - \delta$ *for all* $i \in [m]$ *to precision* θ *in cost*

$$\widetilde{\mathcal{O}}\left(\left[\frac{F^{18}}{\theta^{38}} \log^{19}(n)\, \mathbf{sq}(A) + \frac{F^{22}}{\theta^{44}} \log^{22}(n) + m \frac{F^{8}}{\theta^{16}} \log^{7}(n)\, \mathbf{q}(A) + m \frac{F^{14}}{\theta^{26}} \log^{12}(n) \right] \log^3 \frac{1}{\delta} + \frac{\log(n)}{\theta^2}\, \mathbf{n}(A) \right),$$

where $\mathbf{sq}(A) = \max_{j \in [m]} \mathbf{sq}(A^{(j)})$, *and* $\mathbf{s}(A)$, $\mathbf{q}(A)$, $\mathbf{n}(A)$ *are defined analogously.*

To estimate $\operatorname{Tr}[A^{(i)} H]$, we first notice that we have $\mathrm{SQ}_\phi(\theta \sum_{i=1}^{t} A^{(j_i)})$, since it is a linear combination of matrices that we have sampling and query access to (Lemma 3.9). Then, we can find approximations of the Gibbs state by applying eigenvalue transformation (Theorem 7.2) according to the exponential function to get $\exp[-\theta \sum_{i=1}^{t} A^{(j_i)}]$ as an RUR decomposition. Then the estimation of $\operatorname{Tr}[A^{(i)} H]$ can be performed by usual techniques (namely, Remark 4.13).

In order to understand how precisely we need to approximate the matrix in Line 6 we prove the following lemmas. Our first lemma will show that, to estimate $\operatorname{Tr}(A^{(i)} H) / \operatorname{Tr}(H)$ to θ precision, it suffices to estimate both $\operatorname{Tr}(A^{(i)} H)$ and $\operatorname{Tr}(H)$ to $\frac{1}{3}\theta \operatorname{Tr}(H)$ precision.

Lemma 6.27. *Suppose that* $\theta \in [0, 1]$ *and* $a, \tilde{a}, Z, \tilde{Z}$ *are such that* $|a| \leq Z$, $|a - \tilde{a}| \leq \frac{\theta}{3} Z$, *and* $|Z - \tilde{Z}| \leq \frac{\theta}{3} Z$, *then*

$$\left| \frac{\tilde{a}}{\tilde{Z}} - \frac{a}{Z} \right| \leq \theta.$$

Proof.

$$\left| \frac{\tilde{a}}{\tilde{Z}} - \frac{a}{Z} \right| = \left| \frac{\tilde{a} Z}{Z \tilde{Z}} - \frac{a \tilde{Z}}{Z \tilde{Z}} \right| \leq \left| \frac{\tilde{a} Z - a Z}{Z \tilde{Z}} \right| + \left| \frac{a Z - a \tilde{Z}}{Z \tilde{Z}} \right| \leq \frac{3}{2Z} |\tilde{a} - a| + \frac{3a}{2Z^2} |Z - \tilde{Z}| \leq \frac{1}{2}\theta + \frac{1}{2}\theta \leq \theta. \quad \square$$

Next, we will prove that the approximations we will use to $\operatorname{Tr}(A^{(i)} H)$ and $\operatorname{Tr}(H)$ suffice. We introduce some useful properties of matrix norms. For a matrix $A \in \mathbb{C}^{m \times n}$ and $p \in [1, \infty]$, we denote by $\|A\|_p$ the *Schatten p-norm*, which is the ℓ^p-norm of the singular values $(\sum_i \sigma_i^p(A))^{1/p}$. In particular, $\|A\|_{\mathrm{F}} = \|A\|_2$ and $\|A\|_{\mathrm{Op}} = \|A\|_\infty$. We recall some useful inequalities [Bha97, Section IV.2]. *Hölder's inequality* states that for all $B \in \mathbb{C}^{n \times k}$ and $r, p, q \in (0, \infty]$ such that $\frac{1}{p} + \frac{1}{q} = \frac{1}{r}$, we have $\|AB\|_r \leq \|A\|_p \|B\|_q$. The *trace-norm inequality* states that if $n = m$, then $|\operatorname{Tr}(A)| \leq \|A\|_1$.

Lemma 6.28 (Perturbations of the partition function)**.** *For all Hermitian matrices $H, \tilde{H} \in \mathbb{C}^{n\times n}$,*

$$\left|\operatorname{Tr}(e^{\tilde{H}}) - \operatorname{Tr}(e^{H})\right| \le \left\|e^{\tilde{H}} - e^{H}\right\|_1 \le \left(e^{\|\tilde{H}-H\|} - 1\right) \operatorname{Tr}(e^{H}).$$

The bound in the above lemma is tight, as shown by the example $\tilde{H} \coloneqq H + \varepsilon I$. The proof is in the appendix. Before proving the following lemma, we observe that for any Hermitian matrix $H \in \mathbb{C}^{n\times n}$ with $\|H\|_{\mathrm{F}}^2 \le \frac{n}{4}$, we have by Hölder's inequality that

$$\operatorname{Tr}(e^{H}) = n + \operatorname{Tr}(e^{H} - I) = n + \sum_i (e^{\lambda_i} - 1) \ge n + \sum_i \lambda_i = n + \operatorname{Tr}(H) \ge n - \sqrt{n}\|H\|_{\mathrm{F}} \ge n/2. \quad (15)$$

Lemma 6.29. *Consider a Hermitian matrix $H \in \mathbb{C}^{n\times n}$ such that $\|H\|_{\mathrm{F}}^2 \le \frac{n}{4}$. Let H have an approximate eigendecomposition in the following sense: for $r \le n$, suppose we have a diagonal matrix $D \in \mathbb{R}^{r\times r}$ and $\widetilde{U} \in \mathbb{C}^{r\times n}$ that satisfy $\|\widetilde{U}\widetilde{U}^\dagger - I\| \le \delta$ and $\|H - \widetilde{U}^\dagger D \widetilde{U}\| \le \varepsilon$ for $\varepsilon \le \frac{1}{2}$ and $\delta \le \min(\frac{\varepsilon}{4(\|H\|+\varepsilon)}, \frac{\varepsilon}{2})$. Then we have*

$$|(\operatorname{Tr}(e^{D}) + n - r) - \operatorname{Tr}(e^{H})| \le 2(e-1)\varepsilon \operatorname{Tr}(e^{H}), \quad (16)$$

and, moreover, for all $A \in \mathbb{C}^{n\times n}$ we have

$$|\operatorname{Tr}(A\widetilde{U}^\dagger(e^{D} - I)\widetilde{U}) + \operatorname{Tr}(A) - \operatorname{Tr}(Ae^{H})| \lesssim \varepsilon\|A\| \operatorname{Tr}(e^{H}).$$

Proof. First, recall that, by Lemma 2.5, there is unitary U such that $\|\widetilde{U} - U\| \le \delta$. Consequently, also using facts from Lemma 2.5, along with bounds on δ,

$$\|H - U^\dagger D U\| \le \|H - \widetilde{U}^\dagger D \widetilde{U}\| + \delta\frac{2-\delta}{(1-\delta)^2}\|\widetilde{U}^\dagger D \widetilde{U}\| \le \varepsilon + 4\delta(\|H\| + \varepsilon) \le 2\varepsilon. \quad (17)$$

By Lemma 6.28 we have

$$\left\|e^{U^\dagger D U} - e^{H}\right\|_1 \le (e^{2\varepsilon} - 1) \operatorname{Tr}(e^{H}) \le 2(e-1)\varepsilon \operatorname{Tr}(e^{H}),$$

and since $e^{U^\dagger D U} = U^\dagger(e^{D} - I)U + I$, by the linearity of trace, the trace-norm inequality, and Hölder's inequality,

$$\begin{aligned}&|\operatorname{Tr}(AU^\dagger(e^{D} - I)U) + \operatorname{Tr}(A) - \operatorname{Tr}(Ae^{H})| \\ &\qquad = |\operatorname{Tr}(A(e^{U^\dagger D U} - e^{H}))| \le \|A\|\|e^{U^\dagger D U} - e^{H}\|_1 \le 2(e-1)\|A\|\varepsilon \operatorname{Tr}(e^{H}). \quad (18)\end{aligned}$$

In particular, setting $A = I$, we get the first desired bound

$$|(\operatorname{Tr}(e^{D}) + n - r) - \operatorname{Tr}(e^{H})| = |\operatorname{Tr}(U^\dagger(e^{D} - I)U + I) - \operatorname{Tr}(e^{H})| \le 2(e-1)\varepsilon \operatorname{Tr}(e^{H}).$$

Note that the two identity matrices in the equation above refer to identities of two different sizes. Now, if we show that $\operatorname{Tr}(AU^\dagger(e^{D} - I)U) - \operatorname{Tr}(A\widetilde{U}^\dagger(e^{D} - I)\widetilde{U})$ is sufficiently small, then the second desired bound follows by Eq. (18) and triangle inequality.

$$\begin{aligned}&|\operatorname{Tr}(AU^\dagger(e^{D} - I)U) - \operatorname{Tr}(A\widetilde{U}^\dagger(e^{D} - I)\widetilde{U})| \\ &= |\operatorname{Tr}((UAU^\dagger - \widetilde{U}A\widetilde{U}^\dagger)(e^{D} - I))| \\ &\le \left\|UAU^\dagger - \widetilde{U}A\widetilde{U}^\dagger\right\|\left\|e^{D} - I\right\|_1 && \text{by trace-norm and Hölder's inequality} \\ &\le (2\delta + \delta^2)\|A\|\left\|e^{D} - I\right\|_1 && \text{by Lemma 2.5} \\ &\le 2\varepsilon\|A\|\left\|e^{D} - I\right\|_1 && \text{by assumption that } \delta \le \varepsilon/2 \\ &\le 2\varepsilon\|A\|\left(\operatorname{Tr}(e^{D}) + r\right) && \text{by triangle inequality} \\ &\lesssim \varepsilon\|A\| \operatorname{Tr}(e^{H}). && \text{by Eqs. (15) and (16)}\end{aligned}$$

□

Now we are ready to devise our upper bound on the trace estimation subroutine.

Proof of Lemma 6.26. By Lemma 6.27, it suffices to find estimates of $\operatorname{Tr}(e^H)$ and $\operatorname{Tr}(Ae^H)$ for all $A = A^{(i)}$, to $\frac{\theta}{3}\operatorname{Tr}(e^H)$ additive precision. Recall from the statement that $H \coloneqq -\theta\sum_{i=1}^{t} A^{(j_i)}$. By triangle inequality, $\|H\|_{\mathrm{F}} \le \frac{F}{\theta}\log(n)$. Because H is a linear combination of matrices, by Lemma 3.9, after paying $\frac{\log(n)}{\theta^2}\mathbf{n}(A)$ cost, we can obtain $\mathrm{SQ}_\phi(H)$ for $\phi \le \frac{F^2\log^2(n)}{\theta^2\|H\|_{\mathrm{F}}^2}$ with $\mathbf{q}(H) = \mathbf{q}_\phi(H) \le \frac{\log(n)}{\theta^2}\mathbf{q}(A)$ and $\mathbf{s}_\phi(H) = \mathbf{s}(A)$.

If $\frac{F}{\theta}\log(n) > \sqrt{n}/18$, then we simply compute the sum H by querying all matrix elements of every $A^{(j_i)}$ in the sum, costing $\mathcal{O}\big(tn^2\,\mathbf{q}(A)\big)$. Then we compute e^H and its trace $\operatorname{Tr}(e^H)$ all in time $\mathcal{O}\big(n^3\big)$ [PC99]. Finally, we compute all the traces $\operatorname{Tr}(e^H A^{(m)})$ in time $\mathcal{O}\big(mn^2\big)$. The overall complexity is $\mathcal{O}\big(n^2(t\,\mathbf{q}(A)+n+m)\big) = \widetilde{\mathcal{O}}\Big(\frac{F^6}{\theta^6}\,\mathbf{q}(A)\log^6(n) + m\frac{F^4}{\theta^4}\log^4(n)\Big)$.

If $\frac{F}{\theta}\log(n) \le \sqrt{n}/18$ we do the following. Note that if $\|H\| \le 1$, then $\operatorname{Tr}(e^H) \ge n/e$ and $\operatorname{Tr}(A^{(i)}e^H) \le \big\|A^{(i)}\big\|_{\mathrm{F}}\big\|e^H\big\|_{\mathrm{F}} \le Fe\sqrt{n}$, so $\operatorname{Tr}(A^{(i)}e^H)/\operatorname{Tr}(e^H) \le e^2F/\sqrt{n} \le \theta$, and outputting 0 as estimates is acceptable. We use Theorem 7.2 (with $f(x) = x$, so that $L = 1$, and choosing $\varepsilon \coloneqq \Theta(\theta)$) to find a diagonal matrix $D \in \mathbb{R}^{s\times s}$ with $s = \widetilde{\mathcal{O}}\Big(\phi^2\|H\|_{\mathrm{F}}^6/\varepsilon^6\log(1/\delta)\Big) = \widetilde{\mathcal{O}}\big(F^6\theta^{-6}\log^6(n)\varepsilon^{-6}\log(1/\delta)\big) = \widetilde{\mathcal{O}}\big(F^6\theta^{-12}\log^6(n)\log(1/\delta)\big)$ together with an approximate isometry $\widetilde{U} = N(SH) \in \mathbb{C}^{s\times n}$ such that $\Big\|H - \widetilde{U}^\dagger D\widetilde{U}\Big\| \le \mathcal{O}(\varepsilon)$. If every diagonal element is less than $3/4$, then we conclude that $\|H\| \le 1$, and return 0. Otherwise we have $\|H\| \ge 1/2$ and thus by Theorem 7.2 we have $\Big\|\widetilde{U}\widetilde{U}^\dagger - I\Big\| \lesssim \varepsilon^3\|H\|^{-3} \lesssim \frac{\varepsilon}{\|H\|+\varepsilon} + \varepsilon$ with probability at least $1 - \frac{\delta}{2}$. As per Theorem 7.2, the cost of this is $\log^3(1/\delta)$ times at most

$$\begin{aligned}
\widetilde{\mathcal{O}}\bigg(\frac{\|H\|_{\mathrm{F}}^{18}}{\varepsilon^{18}}\phi^7\,\mathbf{sq}_\phi(H) + \frac{\|H\|_{\mathrm{F}}^{22}}{\varepsilon^{22}}\phi^6\bigg) &= \widetilde{\mathcal{O}}\bigg(\frac{\|H\|_{\mathrm{F}}^{4}}{\varepsilon^{18}}\frac{F^{14}}{\theta^{14}}\log^{14}(n)\,\mathbf{sq}_\phi(H) + \frac{\|H\|_{\mathrm{F}}^{10}}{\varepsilon^{22}}\frac{F^{12}}{\theta^{12}}\log^{12}(n)\bigg)\\
&= \widetilde{\mathcal{O}}\bigg(\frac{\|H\|_{\mathrm{F}}^{4}}{\varepsilon^{18}}\frac{F^{14}}{\theta^{16}}\log^{15}(n)\,\mathbf{sq}(A) + \frac{\|H\|_{\mathrm{F}}^{10}}{\varepsilon^{22}}\frac{F^{12}}{\theta^{12}}\log^{12}(n)\bigg)\\
&= \widetilde{\mathcal{O}}\bigg(\frac{1}{\varepsilon^{18}}\frac{F^{18}}{\theta^{20}}\log^{19}(n)\,\mathbf{sq}(A) + \frac{1}{\varepsilon^{22}}\frac{F^{22}}{\theta^{22}}\log^{22}(n)\bigg)\\
&= \widetilde{\mathcal{O}}\bigg(\frac{F^{18}}{\theta^{38}}\log^{19}(n)\,\mathbf{sq}(A) + \frac{F^{22}}{\theta^{44}}\log^{22}(n)\bigg).
\end{aligned}$$

By Lemma 6.29 we have[25] that $\operatorname{Tr}(e^D) + (n-s)$ is a multiplicative $\frac{\theta}{3}$-approximation of $\operatorname{Tr}(e^H)$ as desired, and for all $A = A^{(i)}$, $\operatorname{Tr}((e^D - I)\widetilde{U}A\widetilde{U}^\dagger) + \operatorname{Tr}(A)$ is an additive $(\frac{\theta}{9}\operatorname{Tr}(e^H))$-approximation of $\operatorname{Tr}(Ae^H)$. We can ignore the $\operatorname{Tr}(A)$ in our approximation: by Eq. (15) we have

$$\operatorname{Tr}(A) \le \|A\|_{\mathrm{F}}\|I\|_{\mathrm{F}} \le F\sqrt{n} \le \theta n/18 \le \theta\operatorname{Tr}(e^H)/9,$$

so $|\operatorname{Tr}((e^D - I)\widetilde{U}A\widetilde{U}^\dagger) - \operatorname{Tr}(Ae^H)| \le \frac{2\theta}{9}\operatorname{Tr}(e^H))$. So, it suffices to compute an additive $(\frac{\theta}{9}\operatorname{Tr}(e^H))$-approximation of $\operatorname{Tr}((e^D - I)\widetilde{U}A\widetilde{U}^\dagger) = \operatorname{Tr}(A\widetilde{U}^\dagger(e^D - I)\widetilde{U})$ to obtain the $(\frac{\theta}{3}\operatorname{Tr}(e^H))$-approximation of $\operatorname{Tr}(Ae^H)$ we seek.

We use Remark 4.13 to estimate $\operatorname{Tr}(A\widetilde{U}^\dagger(e^D - I)\widetilde{U})$ to additive precision $(\frac{\theta}{9}\operatorname{Tr}(e^H))$. Note that by Lemma 6.29 and Eq. (15) we have

$$\Big\|\widetilde{U}^\dagger(e^D - I)\widetilde{U}\Big\|_{\mathrm{F}} \le \|\widetilde{U}\|^2\|e^D - I\|_{\mathrm{F}} \le 2\|e^D - I\|_{\mathrm{F}} \le 2\|e^D - I\|_1 \lesssim \operatorname{Tr}(e^H),$$

[25] In case applying Theorem 7.2 would result in $s > n$, we instead directly diagonalize H ensuring $s \le n$.

and since $s = \widetilde{\mathcal{O}}\big(F^6\theta^{-12}\log^6(n)\log(1/\delta)\big)$ and $\mathbf{q}(H) \leq \frac{\log(n)}{\theta^2}\,\mathbf{q}(A)$, we also have

$$\begin{aligned}\mathbf{q}(\widetilde{U}^\dagger(e^D - I)\widetilde{U}) &= \mathbf{q}((SH)^\dagger N^\dagger(e^D - I)N(SH)) \\ &= \mathcal{O}\big(s\cdot\mathbf{q}(H) + s^2\big) \\ &= \widetilde{\mathcal{O}}\big(F^6\theta^{-14}\log^7(n)\log(1/\delta)\,\mathbf{q}(A) + F^{12}\theta^{-24}\log^{12}(n)\log^2(1/\delta)\big).\end{aligned}$$

Therefore, Remark 4.13 tells us that given $\mathrm{SQ}(A)$, a $(\frac{\theta}{9}\operatorname{Tr}(e^H))$-approximation of $\operatorname{Tr}(A\widetilde{U}^\dagger(e^D - I)\widetilde{U})$ can be computed with success probability at least $1 - \frac{\delta}{2m}$ in time

$$\mathcal{O}\bigg(\frac{\|A\|_{\mathrm{F}}^2}{\theta^2}\big(\mathbf{sq}(A) + s\cdot\mathbf{q}(H) + s^2\big)\log\frac{m}{\delta}\bigg).$$

Since we do this for all $i \in [m]$, the overall complexity of obtaining the desired estimates $\operatorname{Tr}(A^{(i)}e^H)$ with success probability at least $1 - \frac{\delta}{2}$ is m times

$$\widetilde{\mathcal{O}}\bigg(\frac{F^8}{\theta^{16}}\log^7(n)\log(1/\delta)\log(m/\delta)\,\mathbf{q}(A) + \frac{F^{14}}{\theta^{26}}\log^{12}(n)\log^2(m/\delta)\log(m/\delta)\bigg). \qquad \square$$

6.9 Discriminant analysis

Discriminant analysis is used for dimensionality reduction and classification over large data sets. Cong and Duan introduced a quantum algorithm to perform both with Fisher's linear discriminant analysis [CD16], a generalization of principal component analysis to data separated into classes.

The problem is as follows: given classified data, we wish to project our data onto a subspace that best explains between-class variance, while minimizing within-class variance. Suppose there are M input data points $\{x_i \in \mathbb{R}^N : 1 \leq i \leq M\}$ each belonging to one of k classes. Let μ_c denote the centroid (mean) of class $c \in [k]$, and $\bar{x}$ denote the centroid of all data points. Following the notation of [CD16], let

$$S_B = \sum_{c=1}^{k}(\mu_c - \bar{x})(\mu_c - \bar{x})^T \text{ and } S_W = \sum_{c=1}^{k}\sum_{x\in c}(\mu_c - x)(\mu_c - x)^T.$$

denote the between-class and within-class scatter matrices of the dataset respectively. The original goal is to solve the generalized eigenvalue problem $S_B v_i = \lambda_i S_W v_i$ and output the top eigenvalues and eigenvectors; for dimensionality reduction using linear discriminant analysis, we would project onto these top eigenvectors. If S_W would be full-rank, this problem would be equivalent to finding the eigenvalues of $S_W^{-1}S_B$. However, this does not happen in general, and therefore various relaxations are considered in the literature [BHK97, Wel09]. For example, Welling [Wel09] considers the eigenvalue problem of

$$S_B^{\frac{1}{2}}S_W^{-1}S_B^{\frac{1}{2}}. \tag{19}$$

Cong and Duan further relax the problem, as they ignore small eigenvalues of S_W and S_B, and only compute approximate eigenvalues of Eq. (19) (after truncating eigenvalues), leading to inexact eigenvectors. We construct a classical analogue of their quantum algorithm.[26] Cong and Duan also describe a quantum algorithm for discriminant analysis classification; this algorithm does a matrix inversion procedure very similar to those described in Section 6.5 and Section 6.6, so for brevity we will skip dequantizing this algorithm.

[26] Analyzing whether or not the particular relaxation used in this and other quantum machine learning papers provides a meaningful output is unfortunately beyond the scope of our paper.

To formally analyze this algorithm, we could, as in Section 6.4, assume the existence of an eigenvalue gap, so the eigenvectors are well-conditioned. However, let us instead use a different convention: if we can find diagonal D and an approximate isometry U such that $S_B^{\frac{1}{2}} S_W^{-1} S_B^{\frac{1}{2}} U \approx UD$, then we say we have found approximate eigenvalues and eigenvectors of $S_W^+ S_B$.

Problem 6.30 (Linear discriminant analysis)**.** Consider the functions

$$\operatorname{sqrt}(x) = \begin{cases} 0 & x < \sigma^2/2 \\ 2x/\sigma - \sigma & \sigma^2/2 \le x < \sigma^2 \\ \sqrt{x} & x \ge \sigma^2 \end{cases} \qquad \operatorname{inv}(x) = \begin{cases} 0 & x < \sigma^2/2 \\ 2x/\sigma^4 - 1/\sigma^2 & \sigma^2/2 \le x < \sigma^2 \\ 1/x & x \ge \sigma^2 \end{cases}.$$

Given $\mathrm{SQ}(B, W) \in \mathbb{C}^{m \times n}$, with $S_W \coloneqq W^\dagger W$ and $S_B \coloneqq B^\dagger B$, find an α-approximate isometry $U \in \mathbb{C}^{n \times p}$ and diagonal $D \in \mathbb{C}^{p \times p}$ such that we have $\mathrm{SQ}_\phi(U(\cdot, i))$ for all i, $|D_{ii} - \lambda_i| \le \varepsilon \|B\|^2/\sigma^2$ for λ_i the eigenvalues of $\operatorname{sqrt}(S_B) \operatorname{inv}(S_W) \operatorname{sqrt}(S_B)$, and

$$\| \operatorname{sqrt}(S_B) \operatorname{inv}(S_W) \operatorname{sqrt}(S_B) U - UD \| \le \varepsilon \| \operatorname{sqrt}(S_B) \|^2 \| \operatorname{inv}(S_W) \| \le \varepsilon \|B\|^2/\sigma^2.$$

The choice of error bound is natural, since $\|B\|^2/\sigma^2$ is essentially $\| \operatorname{sqrt}(S_B)\|^2 \| \operatorname{inv}(S_W)\|$: we aim for additive error. The quantum algorithm achieves a runtime of $\widetilde{\mathcal{O}}\left(\frac{\|B\|_{\mathrm{F}}^7}{\varepsilon^3 \sigma^7} + \frac{\|W\|_{\mathrm{F}}^7}{\varepsilon^3 \sigma^7}\right)$, up to $\operatorname{polylog}(m, n)$ factors [CD16, Theorem 2].[27]

Corollary 6.31. *For $\varepsilon < \sigma/\|B\|$, we can solve Problem 6.30 in $\widetilde{\mathcal{O}}\left(\left(\frac{\|B\|_{\mathrm{F}}^6 \|B\|^4}{\varepsilon^6 \sigma^{10}} + \frac{\|W\|_{\mathrm{F}}^6 \|W\|^{10}}{\varepsilon^6 \sigma^{16}}\right) \log^3 \frac{1}{\delta}\right)$ time, with $\widetilde{\mathbf{sq}}_\phi(U(\cdot, i)) = \widetilde{\mathcal{O}}\left(\frac{\|B\|_{\mathrm{F}}^4}{\varepsilon^2 \sigma^4} \log^2 \frac{1}{\delta}\right)$.*

We prove this by using Theorem 5.1 to approximate $\operatorname{sqrt}(W^\dagger W) \approx R_W^\dagger U_W R_W$ and $\operatorname{inv}(B^\dagger B) \approx R_B^\dagger U_B R_B$ by RUR decompositions. Then, we use Lemma 4.6 to approximate $R_W R_B^\dagger$ by small submatrices $R'_W R'^\dagger_B$. This yields an approximate RUR decomposition of the matrix whose eigenvalues and vectors we want to find, $R_W^\dagger U R_W$ for $U = U_W R'_W R'^\dagger_B U_B R'_B R'^\dagger_W U_W$.

Finding eigenvectors from an RUR decomposition follows from an observation (Lemma 4.14): for a matrix C_W formed by sampling columns from R_W (using $\mathrm{SQ}(W)$), and $[C_W]_k$ the rank-k approximation to C_W (which can be computed because C_W has size independent of dimension), $(([C_W]_k)^+ R_W)^\dagger$ has singular values either close to zero or close to one. This roughly formalizes the intuition of C_W preserving the left singular vectors and singular values of R_W. We can rewrite $R_W^\dagger U R_W = R_W^\dagger (C_k^+)^\dagger C_k^\dagger U C_k C_k^+ R_W$, which holds by choosing k sufficiently large and choosing C to be the same sketch used for U. Then, we can compute the eigendecomposition of the center $C_k^\dagger U C_k = V D V^\dagger$, which gives us an approximate eigendecomposition for $R_W^\dagger U R_W$: $(C_k^+ R_W)^\dagger V$ is an approximate isometry, so we choose its columns to be our eigenvectors, and our eigenvalues are the diagonal entries of D. We show that this has the approximation properties analogous to the quantum algorithm.

Proof. By Theorem 5.1, we can find R_B, C_B, R_W, C_W such that

$$\| \operatorname{sqrt}(B^\dagger B) - R_B^\dagger \overline{\operatorname{sqrt}}(C_B C_B^\dagger) R_B \| \le \varepsilon \|B\|$$
$$\| \operatorname{inv}(W^\dagger W) - R_W^\dagger \overline{\operatorname{inv}}(C_W C_W^\dagger) R_W \| \le \varepsilon/\sigma^2$$

[27] This is the runtime of Step 2 of Algorithm 1. The normalization factor of $\max(\|B\|_{\mathrm{F}}, \|S\|_{\mathrm{F}})$ is implicit there, κ_{eff} corresponds to $\frac{\max(\|B\|_{\mathrm{F}}, \|S\|_{\mathrm{F}})}{\sigma^2}$, and the error bound the algorithm achieves is the one we describe here, since the authors must implicitly rescale the inverse and square root function by a cumulative factor of $\|B\|^2/\sigma^2$ to apply their Theorem 1.

with

$$r_B = \widetilde{\mathcal{O}}\left(\frac{\|B\|_{\mathrm{F}}^2}{\varepsilon^2\sigma^2}\log\frac{1}{\delta}\right) \qquad c_B = \widetilde{\mathcal{O}}\left(\frac{\|B\|^4\|B\|_{\mathrm{F}}^2}{\varepsilon^2\sigma^6}\log\frac{1}{\delta}\right)$$
$$r_W = \widetilde{\mathcal{O}}\left(\frac{\|W\|^2\|W\|_{\mathrm{F}}^2}{\varepsilon^2\sigma^4}\log\frac{1}{\delta}\right) \qquad c_W = \widetilde{\mathcal{O}}\left(\frac{\|W\|^6\|W\|_{\mathrm{F}}^2}{\varepsilon^2\sigma^8}\log\frac{1}{\delta}\right).$$

Let $Z_B \coloneqq \overline{\mathrm{sqrt}}(C_B C_B^\dagger)$ and $Z_W \coloneqq \overline{\mathrm{inv}}(C_W C_W^\dagger)$. These approximations suffice for us:

$$\begin{aligned}
&\|\operatorname{sqrt}(S_B)\operatorname{inv}(S_W)\operatorname{sqrt}(S_B) - R_B^\dagger Z_B R_B R_W^\dagger Z_W R_W R_B^\dagger Z_B R_B\| \\
&\qquad \le \|\operatorname{sqrt}(S_B) - R_B^\dagger Z_B R_B\|\|\operatorname{inv}(S_W)\operatorname{sqrt}(S_B)\| \\
&\qquad + \|R_B^\dagger Z_B R_B\|\|\operatorname{inv}(S_W) - R_W^\dagger Z_W R_W\|\|\operatorname{sqrt}(S_B)\| \\
&\qquad\qquad + \|R_B^\dagger Z_B R_B R_W^\dagger Z_W R_W\|\|\operatorname{sqrt}(S_B) - R_B^\dagger Z_B R_B\|,
\end{aligned}$$

each of which is bounded by $\varepsilon\|B\|^2/\sigma^2$. Next, we approximate $\|R_B R_W^\dagger - R'_B R_W'^\dagger\|_{\mathrm{F}} \le \varepsilon\sigma^{3/2}\sqrt{\|B\|}$, since then

$$\begin{aligned}
&\|\bar\Sigma_B^{\frac12}\bar U_B^\dagger R_B R_W^\dagger \bar U_W \bar\Sigma_W^{\frac12} - \bar\Sigma_B^{\frac12}\bar U_B^\dagger R'_B R_W'^\dagger \bar U_W \bar\Sigma_W^{\frac12}\| \\
&\le \|\bar\Sigma_B^{\frac12}\bar U_B^\dagger\|\|R_B R_W^\dagger - R'_B R_W'^\dagger\|\|\bar U_W\bar\Sigma_W^{\frac12}\| \\
&\le \sigma^{-\frac12}\|R_B R_W^\dagger - R'_B R_W'^\dagger\|\sigma^{-2} \\
&\le \varepsilon\sqrt{\|B\|/\sigma^2},
\end{aligned}$$

and so

$$\|R_B^\dagger Z_B R_B R_W^\dagger Z_W R_W R_B^\dagger Z_B R_B - R_B^\dagger Z_B R'_B R_W'^\dagger Z_W R'_W R_B'^\dagger Z_B R_B\| \lesssim \varepsilon\|B\|^2/\sigma^2.$$

Now, we can compute $Z \coloneqq Z_B R'_B R_W'^\dagger Z_W R'_W R_B'^\dagger Z_B$ and, using that $Z_B = Z_B[C_B]_{\frac{\sigma}{\sqrt2}}[C_B]^+_{\frac{\sigma}{\sqrt2}}$, rewrite

$$R_B^\dagger Z R_B = R_B^\dagger([C_B]^+_{\frac{\sigma}{\sqrt2}})^\dagger[C_B]^\dagger_{\frac{\sigma}{\sqrt2}} Z [C_B]_{\frac{\sigma}{\sqrt2}}[C_B]^+_{\frac{\sigma}{\sqrt2}} R_B.$$

By Lemma 4.14, $([C_B]^+_{\frac{\sigma}{\sqrt2}} R_B)^\dagger$ is an $\varepsilon\sigma/\|B\|$-approximate projective isometry[28] onto the image of $[C_B]^+_{\frac{\sigma}{\sqrt2}}$ (where we use that $\varepsilon < \sigma/\|B\|$). To turn this approximate projective isometry into an isometry, we compute the eigendecomposition $[C_B]^\dagger_{\frac{\sigma}{\sqrt2}} Z [C_B]_{\frac{\sigma}{\sqrt2}} = V\Sigma V^\dagger$, where we truncate so that V is full rank. Consequently, $U \coloneqq R_B^\dagger([C_B]^+_{\frac{\sigma}{\sqrt2}})^\dagger V$ is full rank—the image of V is contained in the image of $[C_B]^+_{\frac{\sigma}{\sqrt2}}$—and thus is an $\varepsilon\sigma/\|B\|$-approximate isometry. So, our eigenvectors are U and our eigenvalues are $D \coloneqq \Sigma$. This satisfies the desired bounds because

$$\begin{aligned}
\|\operatorname{sqrt}(S_B)\operatorname{inv}(S_W)\operatorname{sqrt}(S_B)U - UD\| &\le \|\operatorname{sqrt}(S_B)\operatorname{inv}(S_W)\operatorname{sqrt}(S_B)U - UDU^\dagger U\| + \|UDU^\dagger U - UD\| \\
&\le \varepsilon\frac{\|B\|^2}{\sigma^2}\|U\| + \|UD\|\|U^\dagger U - I\| \lesssim \varepsilon\frac{\|B\|^2}{\sigma^2}.
\end{aligned}$$

[28] We get more than we need here: an ε-approximate projective isometry would suffice for the subsequent arguments.

The eigenvalues are correct because, by the approximate isometry condition, $\|U - \tilde{U}\| \lesssim \varepsilon \frac{\sigma}{\|B\|}$ for $\tilde{U}$ an isometry, and so we can use Lemma 2.5 to conclude

$$\begin{aligned}
&\| \operatorname{sqrt}(S_B) \operatorname{inv}(S_W) \operatorname{sqrt}(S_B) - \tilde{U} D \tilde{U}^\dagger \| \\
&\leq \| \operatorname{sqrt}(S_B) \operatorname{inv}(S_W) \operatorname{sqrt}(S_B) - U D U^\dagger \| + \| U D U^\dagger - \tilde{U} D \tilde{U}^\dagger \| \\
&\lesssim \varepsilon \frac{\|B\|^2}{\sigma^2} + \varepsilon \frac{\sigma}{\|B\|} \|D\| \lesssim \varepsilon \frac{\|B\|^2}{\sigma^2}.
\end{aligned}$$

$\tilde{U} D \tilde{U}^\dagger$ is an eigendecomposition. Furthermore, this is an approximation of a Hermitian PSD matrices, where singular value error bounds align with eigenvalue error bounds. So, Weyl's inequality (Lemma 4.11) implies the desired bound $|D_{ii} - \lambda_i| \lesssim \varepsilon \frac{\|B\|^2}{\sigma^2}$ for λ_i the true eigenvalues.

We have $\mathrm{SQ}_\phi(U(\cdot, i))$ by Lemmas 3.5 and 3.6, since $U(\cdot, i) = R_B^\dagger([C_B]^+_{\frac{\sigma}{\sqrt{2}}})^\dagger V(\cdot, i)$. The runtime is $\widetilde{\mathbf{sq}}_\phi(U(\cdot, i)) = r_B \phi \log \frac{1}{\delta}$, where

$$\phi = r_B \frac{\sum_{j=1}^{r_B} \|R_B(j, \cdot)\|^2 |([[C_B]^+_{\frac{\sigma}{\sqrt{2}}})^\dagger V(\cdot, i)](j)|^2}{\|U(\cdot, i)\|} \lesssim \|B\|_{\mathrm{F}}^2 \|([C_B]^+_{\frac{\sigma}{\sqrt{2}}})^\dagger V(\cdot, i)\|^2 \lesssim \frac{\|B\|_{\mathrm{F}}^2}{\sigma^2}.$$

This gives the stated runtime. □

7 More singular value transformation

In this section, we present more general versions of our algorithm for even SVT to get results for generic SVT (Theorem 7.1) and eigenvalue transformation (Theorem 7.2). In applications we mainly use even SVT to allow for more fine-tuned control over runtime, but we do use eigenvalue transformation in Section 6.8.

For generic SVT: consider a matrix $A \in \mathbb{C}^{m \times n}$ and a function $f : [0, \infty) \to \mathbb{C}$ satisfying $f(0) = 0$ (so the singular value transformation $f^{(\mathrm{SV})}(A)$ is well-defined as in Definition 2.1). Given $\mathrm{SQ}(A)$ and $\mathrm{SQ}(A^\dagger)$, we give an algorithm to output a CUR decomposition approximating $f^{(\mathrm{SV})}(A)$.

Theorem 7.1 (Generic singular value transformation)**.** *Let $A \in \mathbb{C}^{m \times n}$ be given with both $\mathrm{SQ}_\phi(A)$ and $\mathrm{SQ}_\phi(A^\dagger)$ and let $f : [0, \infty) \to \mathbb{C}$ be a function such that $f(0) = 0$, $g(x) := f(\sqrt{x})/\sqrt{x}$ is L-Lipschitz, and $\bar{g}(x) := (g(x) - g(0))/x$ is $\bar{L}$-Lipschitz. Then, for $0 < \varepsilon \leq \min(L\|A\|^3, \bar{L}\|A\|^5)$, we can output sketches $R := SA \in \mathbb{C}^{r \times n}$ and $C := AT \in \mathbb{C}^{m \times c}$, along with $M \in \mathbb{C}^{r \times c}$ such that*

$$\Pr\left[\|CMR + g(0)A - f^{(\mathrm{SV})}(A)\| > \varepsilon\right] < \delta,$$

with $r = \widetilde{\mathcal{O}}\big(\phi^2 L^2 \|A\|^2 \|A\|_{\mathrm{F}}^4 \frac{1}{\varepsilon^2} \log \frac{1}{\delta}\big)$ and $c = \widetilde{\mathcal{O}}\big(\phi^2 L^2 \|A\|^4 \|A\|_{\mathrm{F}}^2 \frac{1}{\varepsilon^2} \log \frac{1}{\delta}\big)$. Finding S, M, and T takes time

$$\begin{aligned}
\tilde{\mathcal{O}}\Big(&(\bar{L}^2 \|A\|^8 \|A\|_{\mathrm{F}}^2 + L^2 \|A\|^2 \|A\|_{\mathrm{F}}^4) \frac{\phi^2}{\varepsilon^2} \log \frac{1}{\delta} (\mathbf{s}_\phi(A) + \mathbf{s}_\phi(A^\dagger) + \mathbf{q}_\phi(A) + \mathbf{q}_\phi(A^\dagger)) \\
&+ (L^2 \bar{L}^2 \|A\|^{12} \|A\|_{\mathrm{F}}^4 + L^4 \|A\|^6 \|A\|_{\mathrm{F}}^6) \frac{\phi^4}{\varepsilon^4} \log^2 \frac{1}{\delta} \, \mathbf{q}(A) \\
&+ (L^4 \bar{L}^2 \|A\|^{16} \|A\|_{\mathrm{F}}^6 + L^6 \|A\|^{10} \|A\|_{\mathrm{F}}^8) \frac{\phi^6}{\varepsilon^6} \log^3 \frac{1}{\delta} + \mathbf{n}_\phi(A) \Big).
\end{aligned}$$

If we only wish to assume $\mathrm{SQ}_\phi(A)$, we can do so by using Lemma 4.4 instead of Lemma 4.6 in our proof, paying an additional factor of $\frac{1}{\delta}$.

Note that if $\mathbf{sq}_\phi(A), \mathbf{sq}_\phi(A^\dagger) = \mathcal{O}(1)$, then this runtime is dominated by the last term. Moreover, if A is strictly low-rank, with minimum singular value σ, or essentially equivalently, if $f(x) = 0$ for $x \le \sigma$ and so $g(x) = 0$ for $x \le \sigma^2$, then $L \le \ell/\sigma^2$ and $\bar{L} = 2\ell/\sigma^4$ for ℓ the Lipschitz constant of f. In this case the complexity is

$$\widetilde{\mathcal{O}}\left(\left(\frac{\|A\|^{10}\|A\|_{\mathrm{F}}^6}{\sigma^{16}} + \frac{\|A\|^4\|A\|_{\mathrm{F}}^8}{\sigma^{12}}\right)\left(\frac{\ell\|A\|}{\varepsilon}\right)^6 \phi^6 \log^3\frac{1}{\delta}\right). \tag{20}$$

Importantly, when $\varepsilon = \mathcal{O}(\ell\|A\|)$ (that is, if we want relative error), this runtime is independent of dimension. If one desires greater generality, where we only need to depend on the Lipschitz constant of f, we can use a simple trick: as we aim for a spectral norm bound, we can essentially treat A as if it had strictly low rank. Consider the variant of f, $f_{\ge\sigma}$, which is zero below $\sigma/2$, f above σ, and is a linear interpolation in between.

$$f_{\ge\sigma}(x) \coloneqq \begin{cases} 0 & 0 \le x < \sigma/2 \\ (2x/\sigma - 1)f(\sigma) & \sigma/2 \le x < \sigma \\ f(x) & \sigma \le x \end{cases}$$

Then $\|f^{(\mathrm{SV})}(A) - f^{(\mathrm{SV})}_{\ge\varepsilon/\ell}(A)\| \le \varepsilon$, because $f(\varepsilon/\ell) \le \varepsilon$. Further, the Lipschitz constant of $f_{\ge\varepsilon/\ell}$ is at most 2ℓ: the slope of the linear interpolation is $2f(\sigma)/\sigma \le 2\ell\sigma/\sigma$. So, we can run our algorithm for arbitrary ℓ-Lipschitz f in the time given by Eq. (20), with $\sigma = \varepsilon/\ell$.

Our proof strategy is to apply our main result Theorem 5.1 to $g(A^\dagger A)$, for $g(x) \coloneqq f(\sqrt{x})/\sqrt{x}$, and subsequently approximate matrix products with Lemma 4.6 to get an approximation of the form $A'R'^\dagger UR + g(0)A$:

$$f^{(\mathrm{SV})}(A) = Ag(A^\dagger A) \approx AR^\dagger UR + A(g(0)I) \approx A'R'^\dagger UR + g(0)A.$$

Here, $A'R'^\dagger UR$ is a CUR decomposition as desired, since A' is a normalized subset of columns of A. One could further approximate $g(0)A$ by a CUR decomposition if necessary (e.g. by adapting the eigenvalue transformation result below).

We do not use this theorem in our applications. Sometimes we implicitly use a similar strategy (e.g. in Section 6.5), but because we apply our matrix to a vector ($f(A^\dagger)b$) we can use Lemma 4.12 instead of Lemma 4.6 when approximating. This allows for the algorithm to work with only $\mathrm{SQ}_\phi(A)$ and still achieve a poly-logarithmic dependence on $\frac{1}{\delta}$.

Proof. If we want to compute $\hat{f}^{(\mathrm{SV})}(A)$, we can work with $f(x) \coloneqq \hat{f}(x) - g(0)x$, so that $g(0) = 0$ without loss of generality. Notice that $\hat{f}^{(\mathrm{SV})}(A) = f^{(\mathrm{SV})}(A) + g(0)A$, so if we get a CUR decomposition for $f^{(\mathrm{SV})}(A)$ we can add $g(0)A$ after to get the decomposition in the theorem statement.

Consider the SVT $g(x) \coloneqq f(\sqrt{x})/\sqrt{x}$, so that $f^{(\mathrm{SV})}(A) = Ag(A^\dagger A)$. First, use Theorem 5.1 to get $SA \in \mathbb{C}^{r\times n}$, $SAT \in \mathbb{C}^{r\times c}$ such that, with probability $\ge 1 - \delta/4$,

$$\|(SA)^\dagger \bar{g}((SAT)(SAT)^\dagger)SA - g(A^\dagger A)\| \le \frac{\varepsilon}{2\|A\|}. \tag{21}$$

Second, use Lemma 4.6 to get a sketch $T'^\dagger \in \mathbb{C}^{c'\times n}$ such that, with probability $\ge 1 - \delta/4$,

$$\|A(SA)^\dagger - AT'(SAT')^\dagger\| \le \varepsilon(3L\|A\|)^{-1}. \tag{22}$$

The choices of parameters necessary are as follows (using that $\|SA\|_{\mathrm{F}} = O(\|A\|_{\mathrm{F}})$ by Eq. (5) and we have a 2ϕ-oversampled distribution for $(SA)^\dagger$ by Lemma 4.3):

$$r = \tilde{\Theta}\Big(\phi^2 L^2 \|A\|^4 \|A\|_{\mathrm{F}}^2 \frac{1}{\varepsilon^2} \log \frac{1}{\delta}\Big)$$
$$c = \tilde{\Theta}\Big(\phi^2 \bar{L}^2 \|A\|^8 \|A\|_{\mathrm{F}}^2 \frac{1}{\varepsilon^2} \log \frac{1}{\delta}\Big)$$
$$c' = \tilde{\Theta}\Big(\phi^2 L^2 \|A\|^2 \|A\|_{\mathrm{F}}^4 \frac{1}{\varepsilon^2} \log \frac{1}{\delta}\Big)$$

This implies the desired bound through the following sequence of approximations:

$$\begin{aligned} f^{(\mathrm{SV})}(A) &= Ag(A^\dagger A) \\ &\approx A(SA)^\dagger \bar{g}((SAT)(SAT)^\dagger) SA \\ &\approx \underbrace{AT'}_{C} \underbrace{(SAT')^\dagger \bar{g}((SAT)(SAT)^\dagger)}_{M} \underbrace{SA}_{R}. \end{aligned}$$

This gives us a CUR decomposition of $f^{(\mathrm{SV})}(A)$. These two approximations only incur $\mathcal{O}(\varepsilon)$ error in spectral norm; for the first, notice that

$$\begin{aligned} &\|Ag(A^\dagger A) - A(SA)^\dagger \bar{g}((SAT)(SAT)^\dagger) SA\| \\ &\le \|A\| \|(SA)^\dagger \bar{g}((SAT)(SAT)^\dagger) SA - g(A^\dagger A)\| \le \frac{\varepsilon}{2}. \qquad \text{(by (21))} \end{aligned}$$

For the second approximation observe that $|g(x)| \le L|x|$ (and, by corollary, $\bar{g}(x) \le L$) due to g being L-Lipschitz and $g(0) = 0$, therefore

$$\begin{aligned} &\|(A(SA)^\dagger - AT'(SAT')^\dagger) \bar{g}((SAT)(SAT)^\dagger) SA\| \\ &\le \|AT'(SAT')^\dagger - A(SA)^\dagger\| \Big\|\sqrt{\bar{g}((SAT)(SAT)^\dagger)}\Big\| \Big\|\sqrt{\bar{g}((SAT)(SAT)^\dagger)} SA\Big\| \\ &\le \varepsilon (3L\|A\|)^{-1} \Big\|\sqrt{\bar{g}((SAT)(SAT)^\dagger)}\Big\| \Big\|\sqrt{\bar{g}((SAT)(SAT)^\dagger)} SA\Big\| \qquad \text{(by (22))} \\ &\le \varepsilon (3L\|A\|)^{-1} \sqrt{L} \sqrt{\|g(A^\dagger A)\| + \frac{\varepsilon}{2\|A\|}} \qquad \text{(since } \bar{g}(x) \le L \text{ and by (21))} \\ &\le \varepsilon (3L\|A\|)^{-1} \sqrt{\frac{3}{2}} L\|A\| < \frac{\varepsilon}{2}. \qquad \text{(since } |g(x)| \le L|x| \text{ and } \varepsilon \le L\|A\|^3) \end{aligned}$$

The time complexity of this procedure is

$$\mathcal{O}\Big(\mathbf{n}_\phi(A) + (r+c+c')(\mathbf{s}_\phi(A) + \mathbf{q}_\phi(A)) + c'(\mathbf{s}_\phi(A^\dagger) + \mathbf{q}_\phi(A^\dagger)) + (rc + rc')\,\mathbf{q}(A) + r^2 c + r^2 c'\Big),$$

which comes from producing sketches, querying all the relevant entries of SAT and SAT', the singular value transformation of SAT, and the matrix multiplication in M. We get r factors in the latter two terms because we can separate $\bar{g}((SAT)(SAT)^\dagger) = \sqrt{\bar{g}}(SAT)(\sqrt{\bar{g}}(SAT))^\dagger$ where $\sqrt{\bar{g}}(x) \coloneqq \sqrt{\bar{g}(x)}$. □

As for eigenvalue transformation, consider a function $f : \mathbb{R} \to \mathbb{C}$ and a Hermitian matrix $A \in \mathbb{C}^{n \times n}$, given $\mathrm{SQ}(A)$. If f is even (so $f(x) = f(-x)$), then $f(A) = f(\sqrt{A^\dagger A})$, so we can use Theorem 5.1 to compute the eigenvalue transform $f(A)$. For non-even f, we cannot use this result, and present the following algorithm to compute it.

Theorem 7.2 (Eigenvalue transformation)**.** *Suppose we are given a Hermitian* $\mathrm{SQ}_\phi(A) \in \mathbb{C}^{n\times n}$ *with eigenvalues* $\lambda_1 \geq \cdots \geq \lambda_n$*, a function* $f : \mathbb{R} \to \mathbb{C}$ *that is* L*-Lipschitz on* $\cup_{i=1}^n [\lambda_i - d, \lambda_i + d]$ *for some* $d > \frac{\varepsilon}{L}$*, and some* $\varepsilon \in (0, L\|A\|/2]$*. Then we can output matrices* $S \in \mathbb{C}^{r\times n}$*,* $N \in \mathbb{C}^{s\times r}$*, and* $D \in \mathbb{C}^{s\times s}$*, with* $r = \widetilde{\mathcal{O}}\Big(\phi^2\|A\|^4\|A\|_{\mathrm{F}}^2 \frac{L^6}{\varepsilon^6} \log\frac{1}{\delta}\Big)$ *and* $s = \widetilde{\mathcal{O}}\Big(\|A\|_{\mathrm{F}}^2 \frac{L^2}{\varepsilon^2}\Big)$*, such that*

$$\Pr\Big[\|(SA)^\dagger N^\dagger D N (SA) + f(0)I - f^{(\mathrm{EV})}(A)\| > \varepsilon\Big] < \delta,$$

in time

$$\begin{aligned}\tilde{\mathcal{O}}\Big((L^{10}\varepsilon^{-10}\|A\|^8\|A\|_{\mathrm{F}}^2\phi^2 \log\frac{1}{\delta} + L^6\varepsilon^{-6}\|A\|_{\mathrm{F}}^6\phi\log\frac{1}{\delta})(\mathbf{s}_\phi(A) + \mathbf{q}_\phi(A)) \\ + (L^{16}\varepsilon^{-16}\|A\|^{12}\|A\|_{\mathrm{F}}^4\phi^4\log^2\frac{1}{\delta} + L^{18}\varepsilon^{-18}\|A\|^8\|A\|_{\mathrm{F}}^{10}\phi^5\log^3\frac{1}{\delta})\,\mathbf{q}(A) \\ + L^{22}\varepsilon^{-22}\|A\|^{16}\|A\|_{\mathrm{F}}^6\phi^6\log^3\frac{1}{\delta} + \mathbf{n}_\phi(A)\Big).\end{aligned}$$

Moreover, this decomposition satisfies the following two properties. First, NSA *is an approximate isometry:* $\|(NSA)(NSA)^\dagger - I\| \leq (\frac{\varepsilon}{L\|A\|})^3$*. Second,* D *is a diagonal matrix and its diagonal entries satisfy* $|D(i,i) + f(0) - f(\lambda_i)| \leq \varepsilon$ *for all* $i \in [n]$ *(where* $D(i,i) \coloneqq 0$ *for* $i > s$*).*

Under the reasonable assumptions[29] that $\mathbf{sq}(A)$ and ϕ are small ($\mathcal{O}(1)$, say) and $\varepsilon \leq L\|A\|\frac{\|A\|}{\|A\|_{\mathrm{F}}}$, the complexity of this theorem is $\widetilde{\mathcal{O}}\big(L^{22}\varepsilon^{-22}\|A\|^{16}\|A\|_{\mathrm{F}}^6\log^3\frac{1}{\delta}\big)$.

We now outline our proof. Our strategy is similar to the one used for quantum-inspired semidefinite programming [CLLW20]: first we find the eigenvectors and eigenvalues of A and then apply f to the eigenvalues. Let $\pi(x)$ be a (smoothened) step function designed so it can zeroes out small eigenvalues of A (in particular, eigenvalues smaller than $\varepsilon/\sqrt{2}L$). Then

$$\begin{aligned} A &\approx \pi(A^\dagger A)A\pi(A^\dagger A) && \text{by definition of } \pi \\ &\approx R^\dagger\bar{\pi}(CC^\dagger)RAR^\dagger\bar{\pi}(CC^\dagger)R && \text{by Theorem 5.1} \\ &\approx R^\dagger\bar{\pi}(CC^\dagger)M\bar{\pi}(CC^\dagger)R && \text{by sketching } M \approx RAR^\dagger \\ &= R^\dagger(C_\sigma C_\sigma^+)^\dagger\bar{\pi}(CC^\dagger)M\bar{\pi}(CC^\dagger)C_\sigma C_\sigma^+ R. && \text{where } \sigma = \varepsilon/\sqrt{2}L \end{aligned}$$

Here, C_σ is the low-rank approximation of C formed by transforming C according to the "filter" function on x that is 0 for $x < \sigma$ and x otherwise. $\hat{U} \coloneqq C_\sigma^+ R \in \mathbb{C}^{c\times n}$ is an approximate isometry by Lemma 4.14. We are nearly done now: since the rest of the matrix expression, $C_\sigma^\dagger\bar{\pi}(CC^\dagger)M\bar{\pi}(CC^\dagger)C_\sigma \in \mathbb{C}^{c\times c}$, consists of submatrices of A of size independent of n, we can directly compute its unitary eigendecomposition $UDU^\dagger$. This gives the approximate decomposition $A \approx (\hat{U}U)D(\hat{U}U)^\dagger$, with $\hat{U}U$ and D acting as approximate eigenvectors and eigenvalues of A, respectively. An application of Lemma 5.4 shows that $f(A) \approx (\hat{U}U)f(D)(\hat{U}U)^\dagger$ in the desired sense. Therefore, our output approximation of $f(A)$ comes in the form of an RUR decomposition that can be rewritten in the form of an approximate eigendecomposition. The only major difference between this proof sketch and the proof below is that we perform our manipulations on the SVD of C_σ, to save on computation time: note that the SVD can be made small in dimension, using that the rank of C_σ is bounded by $\|C\|_{\mathrm{F}}^2/\sigma^2$.

[29]The correct way to think about ε is as some constant fraction of $L\|A\|$. If $\varepsilon > L\|A\|$ then $f(0)I$ is a satisfactory approximation. The bound we give says that we want an at least $\|A\|_{\mathrm{F}}/\|A\|$ improvement over trivial, which is modest in the close-to-low-rank regime that we care about. Similar assumptions appear in Section 6.

Proof. Throughout this proof ε is not dimensionless; if choices of parameters are confusing, try replacing ε with $\varepsilon\|A\|$. We will take $f(0) = 0$ without loss of generality. First, consider the "smooth projection" singular value transformation

$$\pi(x) = \begin{cases} 0 & x < \frac{\varepsilon^2}{2L^2} \\ \frac{2L^2}{\varepsilon^2}x - 1 & \frac{\varepsilon^2}{2L^2} \le x < \frac{\varepsilon^2}{L^2} \\ 1 & \frac{\varepsilon^2}{L^2} \le x \end{cases}$$

Since π is a projector onto the large eigenvectors of A, we can add these projectors to our expression without incurring too much spectral norm error.

$$\|\pi(A^\dagger A)A\pi(A^\dagger A) - A\| = \max_{i\in[n]}|\pi(\lambda_i^2)\lambda_i\pi(\lambda_i^2) - \lambda_i| \le \varepsilon/L$$

Second, use Theorem 5.1 to get $SA \in \mathbb{C}^{r\times n}$, $SAT \in \mathbb{C}^{r\times c}$ such that, with probability $\ge 1-\delta$,

$$\|(SA)^\dagger\bar{\pi}((SAT)(SAT)^\dagger)SA - \pi(A^\dagger A)\| \le \frac{\varepsilon}{L\|A\|}.$$

The necessary sizes for these bounds to hold are as follows (Lipschitz constants for π are $2L^2/\varepsilon^2$ and $4L^4/\varepsilon^4$, $\|SA\|_{\mathrm{F}} = O(\|A\|_{\mathrm{F}})$ by Eq. (5), and we have a 2ϕ-oversampled distribution for $(SA)^\dagger$ by Lemma 4.3):[30]

$$r = \tilde{\Theta}\Big(\phi^2\frac{L^4}{\varepsilon^4}\|A\|^2\|A\|_{\mathrm{F}}^2\frac{L^2\|A\|^2}{\varepsilon^2}\log\frac{1}{\delta}\Big) = \tilde{\Theta}\Big(\phi^2\|A\|^4\|A\|_{\mathrm{F}}^2\frac{L^6}{\varepsilon^6}\log\frac{1}{\delta}\Big),$$
$$c = \tilde{\Theta}\Big(\phi^2\frac{L^8}{\varepsilon^8}\|A\|^6\|A\|_{\mathrm{F}}^2\frac{L^2\|A\|^2}{\varepsilon^2}\log\frac{1}{\delta}\Big) = \tilde{\Theta}\Big(\phi^2\|A\|^8\|A\|_{\mathrm{F}}^2\frac{L^{10}}{\varepsilon^{10}}\log\frac{1}{\delta}\Big).$$

This approximation does not incur too much error:

$$\begin{aligned}
&\|R^\dagger\bar{\pi}(CC^\dagger)RAR^\dagger\bar{\pi}(CC^\dagger)R - \pi(A^\dagger A)A\pi(A^\dagger A)\| \\
&\le \|\pi(A^\dagger A)A(\pi(A^\dagger A) - R^\dagger\bar{\pi}(CC^\dagger)R)\| + \|(\pi(A^\dagger A) - R^\dagger\bar{\pi}(CC^\dagger)R)AR^\dagger\bar{\pi}(CC^\dagger)R\| \\
&\le \frac{\varepsilon}{L\|A\|}\Big(\|\pi(A^\dagger A)\|\|A\| + \|A\|\|R^\dagger\bar{\pi}(CC^\dagger)R\|\Big) \le \frac{\varepsilon}{L\|A\|}\Big(\|A\| + \|A\|\Big(1 + \frac{\varepsilon}{L\|A\|}\Big)\Big) \le 3\frac{\varepsilon}{L}.
\end{aligned}$$

Third, apply Remark 4.13(b) r^2 times to approximate each entry of $RAR^\dagger$: pull t samples from $\mathrm{SQ}_\phi(A)$ for $t := \mathcal{O}\Big(\phi\|A\|_{\mathrm{F}}^6\frac{L^6}{\varepsilon^6}\log\frac{r^2}{\delta}\Big)$ such that, given some $\mathrm{Q}(x), \mathrm{Q}(y)$, with probability $\ge 1 - \frac{\delta}{r^2}$, one can output an estimate of $x^\dagger Ay$ up to $\frac{\varepsilon^3\|x\|\|y\|}{L^3\|A\|_{\mathrm{F}}^2}$ additive error with *no additional queries to* $\mathrm{SQ}_\phi(A)$. Then, by union bound, with probability $\ge 1-\delta$, using the same t samples from A each time, one can output an estimate of $R(i,\cdot)AR(j,\cdot)^\dagger$ up to $\frac{\varepsilon^3\|R(i,\cdot)\|\|R(j,\cdot)\|}{L^3\|A\|_{\mathrm{F}}^2}$ error for all $i, j \in [r]$ such that $i \le j$. Let M be the matrix of these estimates. Then, using that $\|R\|_{\mathrm{F}} = \mathcal{O}(\|A\|_{\mathrm{F}})$ from Eq. (5),

$$\|M - RAR^\dagger\|_{\mathrm{F}}^2 \le \sum_{i=1}^r\sum_{j=1}^r\Big(\frac{\varepsilon^3\|R(i,\cdot)\|\|R(j,\cdot)\|}{L^3\|A\|_{\mathrm{F}}^2}\Big)^2 = \frac{\varepsilon^6\|R\|_{\mathrm{F}}^4}{L^6\|A\|_{\mathrm{F}}^4} \lesssim \frac{\varepsilon^6}{L^6}.$$

[30] The constraint on the size of ε from Theorem 5.1 here is $\varepsilon(L\|A\|)^{-1} \lesssim \min(L^2\|A\|^2/\varepsilon^2, L^4\|A\|^4/\varepsilon^4)$, which is true since $\varepsilon \le L\|A\|/2$.

From Eqs. (6) and (7),

$$\|R^\dagger \bar{\pi}(CC^\dagger)(RAR^\dagger - M)\bar{\pi}(CC^\dagger)R\| \lesssim \frac{\varepsilon^3}{L^3}\|R^\dagger \bar{\pi}(CC^\dagger)\|^2 \leq \frac{\varepsilon^3}{L^3}\Big(1 + \frac{\varepsilon}{L\|A\|}\Big)\frac{L^2}{\varepsilon^2} \lesssim \frac{\varepsilon}{L}.$$

So far, we have shown that we can find an RUR approximation to A, with

$$\|R^\dagger \bar{\pi}(CC^\dagger)M\bar{\pi}(CC^\dagger)R - A\| \lesssim \frac{\varepsilon}{L}$$

However, if we wish to apply an eigenvalue transformation to A, we need to access the eigenvalues of A as well. To do this, we will express this decomposition as an approximate unitary eigendecomposition. Using that $\bar{\pi}$ zeroes out singular values that are smaller than $\frac{\varepsilon^2}{2L^2}$, we can write our expression as $\hat{U}\hat{D}\hat{U}^\dagger$, for $\hat{U} \in \mathbb{C}^{n\times s}$ and $\hat{D} \in \mathbb{C}^{s\times s}$:

$$\begin{aligned}
&R^\dagger \bar{\pi}(CC^\dagger)M\bar{\pi}(CC^\dagger)R \\
&= \big(C^+_{\frac{\varepsilon}{\sqrt{2}L}}R\big)^\dagger \big(C^\dagger_{\frac{\varepsilon}{\sqrt{2}L}}\bar{\pi}(CC^\dagger)M\bar{\pi}(CC^\dagger)C_{\frac{\varepsilon}{\sqrt{2}L}}\big)\big(C^+_{\frac{\varepsilon}{\sqrt{2}L}}R\big) \\
&= \underbrace{\big(R^\dagger U^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}}(D^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}})^{-1}\big)}_{\hat{U}}\underbrace{\big(D^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}}(U^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}})^\dagger \bar{\pi}(CC^\dagger)M\bar{\pi}(CC^\dagger)U^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}}D^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}}\big)}_{\hat{D}}\underbrace{\big((D^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}})^{-1}(U^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}})^\dagger R\big)}_{\hat{U}^\dagger}.
\end{aligned} \tag{23}$$

Here, we are using an SVD of C truncated to ignore singular values smaller than $\frac{\varepsilon}{\sqrt{2}L}$, where $U^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}} \in \mathbb{C}^{r\times s}$, $D^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}} \in \mathbb{C}^{s\times s}$, $V^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}} \in \mathbb{C}^{c\times s}$, where s is the number of singular values of C that are at least $\frac{\varepsilon}{\sqrt{2}L}$. Note that, as a result, $s \leq \|C\|_{\mathrm{F}}^2/(\varepsilon/\sqrt{2}L)^2 \lesssim \|A\|_{\mathrm{F}}^2 L^2\varepsilon^{-2}$ and $s \leq \min(r, c, n)$. By Lemma 4.14 with our values of r and c, we get that $\hat{U} \coloneqq R^\dagger U^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}}(D^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}})^{-1}$ is an $\mathcal{O}\Big(\frac{\varepsilon^3}{L^3\|A\|^3}\Big)$-approximate isometry; we rescale ε until this is at most $\frac{1}{2}$.

The rest of this error analysis will show that, since $\hat{U}$ is an approximate isometry, $f(A) \approx \hat{U}f(\hat{D})\hat{U}^\dagger$ in the senses required for the theorem statement. Though $\hat{D}$ is not diagonal, since it is $s\times s$, we can compute its unitary eigendecomposition $U^{(\hat{D})}D^{(\hat{D})}(U^{(\hat{D})})^\dagger$; so, we can take $D \coloneqq f(D^{(\hat{D})})$ and $N \coloneqq (U^{(\hat{D})})^\dagger(D^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}})^+(U^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}})^\dagger$ to get the decomposition in the theorem statement. (Including the isometry $(U^{(\hat{D})})^\dagger$ in our expression for $\hat{U}$ does not change the value of α).

First, consider the eigenvalues of $\hat{D}$. Note that $\|\hat{U}^\dagger\hat{U} - I\| \leq \alpha$ since $\hat{U}$ is an α-approximate isometry, and by Lemma 2.5, there exists an isometry U such that $\|U - \hat{U}\| \leq \alpha$. We first observe that, using Lemma 2.5 and our bound on α,

$$\begin{aligned}
\|A - U\hat{D}U^\dagger\| &\leq \|A - \hat{U}\hat{D}\hat{U}^\dagger\| + \|\hat{U}\hat{D}\hat{U}^\dagger - U\hat{D}U^\dagger\| \\
&\leq \frac{\varepsilon}{L} + \alpha\frac{2-\alpha}{(1-\alpha)^2}\|\hat{U}\hat{D}\hat{U}^\dagger\| \\
&\leq \frac{\varepsilon}{L} + \alpha\frac{2-\alpha}{(1-\alpha)^2}\Big(\|A\| + \frac{\varepsilon}{L}\Big) \lesssim \frac{\varepsilon}{L}.
\end{aligned}$$

Consequently, by Weyl's inequality (Lemma 4.11), the eigenvalues of $U\hat{D}U^\dagger$, $\hat{\lambda}_1 \geq \cdots \geq \hat{\lambda}_n$ satisfy $|\lambda_i - \hat{\lambda}_i| \lesssim \frac{\varepsilon}{L}$ for all $i \in [n]$, and by assumption, f is L-Lipschitz on the spectrums of A and $U\hat{D}U$. From this, we can conclude that we can compute estimates for the eigenvalues of $f(A)$, since the eigenvalues of $U\hat{D}U^\dagger$ are the eigenvalues of $\hat{D}$ (padded with zero eigenvalues) which we know from our eigendecomposition of $\hat{D}$ Further, our estimates $f(\hat{\lambda}_i)$ satisfy the desired bound

$|f(\hat{\lambda}_i) - f(\lambda_i)| \le \varepsilon$. Finally, since f is Lipschitz on our spectrums of concern, the desired error bound for our approximation holds by the following computation (which uses Lemma 2.5 extensively):

$$\begin{aligned}
\|f(A) - \hat{U} f(\hat{D}) \hat{U}^\dagger\| &\le \|f(A) - U f(\hat{D}) U^\dagger\| + \|U f(\hat{D}) U^\dagger - \hat{U} f(\hat{D}) \hat{U}^\dagger\| \\
&\le \|f(A) - U f(\hat{D}) U^\dagger\| + (2\alpha + \alpha^2)\|f(\hat{D})\| \\
&\lesssim L(\|A - U\hat{D}U^\dagger\| + (2\alpha + \alpha^2)\|\hat{D}\|) \log s && \text{by Lemma 5.4} \\
&\le L(\|A - \hat{U}\hat{D}\hat{U}^\dagger\| + 2(2\alpha + \alpha^2)\|\hat{D}\|) \log s \\
&\le L\Big(\frac{\varepsilon}{L} + \frac{2(2\alpha + \alpha^2)}{(1-\alpha)^2}\|\hat{U}\hat{D}\hat{U}^\dagger\|\Big) \log s \\
&\le L\Big(\frac{\varepsilon}{L} + \frac{2(2\alpha + \alpha^2)}{(1-\alpha)^2}\Big(\|A\| + \frac{\varepsilon}{L}\Big)\Big) \log s \lesssim \varepsilon \log s. && \text{by } \alpha \le \tfrac{\varepsilon}{L\|A\|}
\end{aligned}$$

Finally, we rescale ε down by $\log^2 s$ so that this final bound is $\mathcal{O}(\varepsilon)$. This term is folded into the polylog terms of r, c, and s. (We need to scale by more than $\log s$ because s has a dependence on $\frac{1}{\varepsilon^2}$.) This completes the error analysis.

The complexity analysis takes some care: we want to compute our matrix expressions in the correct order. First, we will sample to get S and T, and then compute the *truncated* singular value decomposition of $C := SAT$, which we have denoted $C_{\frac{\varepsilon}{\sqrt{2}L}} = U^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}} D^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}} (V^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}})^\dagger$ for $U^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}} \in \mathbb{C}^{r \times s}$, $D^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}} \in \mathbb{C}^{s \times s}$, $V^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}} \in \mathbb{C}^{c \times s}$. Then, we will perform the inner product estimation protocol r^2 times to get our estimate $M \in \mathbb{C}^{r \times r}$, and compute the eigendecomposition of

$$\begin{aligned}
\hat{D} &= D^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}} (U^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}})^\dagger \bar{\pi}(CC^\dagger) M \bar{\pi}(CC^\dagger) U^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}} D^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}} \\
&= D^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}} \bar{\pi}((D^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}})^2)(U^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}})^\dagger M U^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}} \bar{\pi}((D^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}})^2) D^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}}
\end{aligned}$$

via the final expression above, with the truncations propagated through the matrices, to get $\hat{D} = U^{(\hat{D})} D^{(\hat{D})} (U^{(\hat{D})})^\dagger$. Then, we compute and output $D = \hat{D}$ and $N = (U^{(\hat{D})})^\dagger (D^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}})^+ (U^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}})^\dagger$. By evaluating the expression for $\hat{D}$ from left-to-right, we only need to perform matrix multiplications that (naively) take s^3 or sr^2 time. The only cost of c we incur is in computing the SVD of C. The runtime is

$$\mathcal{O}\big((r + c + t)(\mathbf{s}_\phi(A) + \mathbf{q}_\phi(A)) + (rc + r^2 t)\,\mathbf{q}(A) + s^3 + r^2 s + r^2 c + \mathbf{n}_\phi(A)\big). \qquad \square$$

Acknowledgments

ET thanks Craig Gidney for the reference to alias sampling. AG is grateful to Saeed Mehraban for insightful suggestions about proving perturbation bounds on partition functions. Part of this work was done while visiting the Simons Institute for the Theory of Computing. We gratefully acknowledge the Institute's hospitality.

References

[Aar15] Scott Aaronson. Read the fine print. *Nature Physics*, 11(4):291, 2015.

[ACQ22] Dorit Aharonov, Jordan Cotler, and Xiao-Liang Qi. Quantum algorithmic measurement. *Nature Communications*, 13(1):1–9, 2022. arXiv: 2101.04634

[ADBL20] Juan Miguel Arrazola, Alain Delgado, Bhaskar Roy Bardhan, and Seth Lloyd. Quantum-inspired algorithms in practice. *Quantum*, 4:307, 2020. arXiv: 1905.10415

[AK16] Sanjeev Arora and Satyen Kale. A combinatorial, primal-dual approach to semidefinite programs. *Journal of the ACM*, 63(2):12:1–12:35, 2016. Earlier version in STOC'07.

[AP10] Alexei B. Aleksandrov and Vladimir V. Peller. Operator Hölder–Zygmund functions. *Advances in Mathematics*, 224(3):910–966, 2010. arXiv: 0907.3049

[AP11] Alexei B. Aleksandrov and Vladimir V. Peller. Estimates of operator moduli of continuity. *Journal of Functional Analysis*, 261(10):2741–2796, 2011. arXiv: 1104.3553

[vAG19] Joran van Apeldoorn and András Gilyén. Improvements in quantum SDP-solving with applications. In *Proceedings of the 46th International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 99:1–99:15, 2019. arXiv: 1804.05058

[vAGGdW20] Joran van Apeldoorn, András Gilyén, Sander Gribling, and Ronald de Wolf. Quantum SDP-solvers: Better upper and lower bounds. *Quantum*, 4:230, 2020. Earlier version in FOCS'17. arXiv: 1705.01843

[ATS03] Dorit Aharonov and Amnon Ta-Shma. Adiabatic quantum state generation and statistical zero knowledge. In *Proceedings of the 35th ACM Symposium on the Theory of Computing*, pages 20–29, New York, NY, USA, 2003. ACM, Association for Computing Machinery. arXiv: quant-ph/0301023

[BCK15] Dominic W. Berry, Andrew M. Childs, and Robin Kothari. Hamiltonian simulation with nearly optimal dependence on all parameters. In *Proceedings of the 56th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 792–809, 2015. arXiv: 1501.01715

[BE95] Peter Borwein and Tamás Erdélyi. *Polynomials and polynomial inequalities*, volume 161 of *Graduate Texts in Mathematics*. Springer, New York, NY, USA, 1995.

[Bel97] R. Bellman. *Introduction to matrix analysis*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, second edition, 1997.

[Bha97] Rajendra Bhatia. *Matrix Analysis*, volume 169 of *Graduate Texts in Mathematics*. Springer, 1997.

[BHK97] Peter N. Belhumeur, João P. Hespanha, and David J. Kriegman. Eigenfaces vs. Fisherfaces: recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):711–720, 1997.

[BKL+19] Fernando G. S. L. Brandão, Amir Kalev, Tongyang Li, Cedric Yen-Yu Lin, Krysta M. Svore, and Xiaodi Wu. Quantum SDP solvers: Large speed-ups, optimality, and applications to quantum learning. In *Proceedings of the 46th International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 27:1–27:14, 2019. arXiv: 1710.02581

[BS17] Fernando G. S. L. Brandão and Krysta M. Svore. Quantum speed-ups for solving semidefinite programs. In *Proceedings of the 58th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 415–426, 2017. arXiv: 1609.05537

[CCH+22] Nadiia Chepurko, Kenneth Clarkson, Lior Horesh, Honghao Lin, and David Woodruff. Quantum-inspired algorithms from randomized numerical linear algebra. In *International Conference on Machine Learning*, pages 3879–3900. PMLR, 2022. arXiv: 2011.04125

[CD16] Iris Cong and Luming Duan. Quantum discriminant analysis for dimensionality reduction and classification. *New Journal of Physics*, 18(7):073011, jul 2016. arXiv: 1510.00113

[CGJ19] Shantanav Chakraborty, András Gilyén, and Stacey Jeffery. The power of block-encoded matrix powers: Improved regression techniques via faster Hamiltonian simulation. In *Proceedings of the 46th International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 33:1–33:14, 2019. arXiv: 1804.01973

[CGL+20] Nai-Hui Chia, András Gilyén, Han-Hsuan Lin, Seth Lloyd, Ewin Tang, and Chunhao Wang. Quantum-inspired algorithms for solving low-rank linear equation systems with logarithmic dependence on the dimension. In *Proceedings of the 31st International Symposium on Algorithms and Computation (ISAAC)*, pages 47:1–47:17, 2020.

[CHI+18] Carlo Ciliberto, Mark Herbster, Alessandro Davide Ialongo, Massimiliano Pontil, Andrea Rocchetto, Simone Severini, and Leonard Wossnig. Quantum machine learning: a classical perspective. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 474(2209):20170551, January 2018.

[CLLW20] Nai-Hui Chia, Tongyang Li, Han-Hsuan Lin, and Chunhao Wang. Quantum-inspired sublinear algorithm for solving low-rank semidefinite programming. In *Proceedings of the 45th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 23:1–23:15, 2020. arXiv: 1901.03254

[CLS+19] Zhihuai Chen, Yinan Li, Xiaoming Sun, Pei Yuan, and Jialin Zhang. A quantum-inspired classical algorithm for separable non-negative matrix factorization. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 4511–4517, Palo Alto, CA, USA, 2019. AAAI Press, AAAI. arXiv: 1907.05568

[CLW18] Nai-Hui Chia, Han-Hsuan Lin, and Chunhao Wang. Quantum-inspired sublinear classical algorithms for solving low-rank linear systems. arXiv: 1811.04852, 2018.

[CMM17] Michael B. Cohen, Cameron Musco, and Christopher Musco. Input sparsity time low-rank approximation via ridge leverage score sampling. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1758–1777, Philadelphia, PA, USA, January 2017. Society for Industrial and Applied Mathematics.

[CVB20] Daan Camps and Roel Van Beeumen. Approximate quantum circuit synthesis using block encodings. *Physical Review A*, 102(5):052411, 2020. arXiv: 2007.01417

[CW17] Kenneth L. Clarkson and David P. Woodruff. Low-rank approximation and regression in input sparsity time. *J. ACM*, 63(6), January 2017.

[DBH21] Chen Ding, Tian-Yi Bao, and He-Liang Huang. Quantum-inspired support vector machine. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–13, 2021. arXiv: 1906.08902

[DHLT20] Yuxuan Du, Min-Hsiu Hsieh, Tongliang Liu, and Dacheng Tao. Quantum-inspired algorithm for general minimum conical hull problems. *Physical Review Research*, 2(3):033199, 2020. arXiv: 1907.06814

[DKM06] Petros Drineas, Ravi Kannan, and Michael W. Mahoney. Fast Monte Carlo algorithms for matrices I: Approximating matrix multiplication. *SIAM Journal on Computing*, 36(1):132–157, 2006.

[DKR02] Petros Drineas, Iordanis Kerenidis, and Prabhakar Raghavan. Competitive recommendation systems. In *Proceedings of the 34th ACM Symposium on the Theory of Computing (STOC)*, pages 82–90, New York, NY, USA, 2002. Association for Computing Machinery.

[DKW18] Yogesh Dahiya, Dimitris Konomis, and David P Woodruff. An empirical evaluation of sketching for numerical linear algebra. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1292–1300, New York, NY, USA, 2018. ACM, Association for Computing Machinery.

[DM07] Petros Drineas and Michael W. Mahoney. A randomized algorithm for a tensor-based generalization of the singular value decomposition. *Linear Algebra and its Applications*, 420(2-3):553–571, 2007.

[DMM08] Petros Drineas, Michael W. Mahoney, and S. Muthukrishnan. Relative-error CUR matrix decompositions. *SIAM Journal on Matrix Analysis and Applications*, 30(2):844–881, January 2008.

[DW20] Vedran Dunjko and Peter Wittek. A non-review of Quantum Machine Learning: trends and explorations. *Quantum Views*, 4:32, March 2020.

[Fey51] Richard P. Feynman. An operator calculus having applications in quantum electrodynamics. *Physical Review*, 84:108–128, 1951.

[Fey82] Richard P. Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21(6-7):467–488, 1982.

[FKV04] Alan Frieze, Ravi Kannan, and Santosh Vempala. Fast Monte-Carlo algorithms for finding low-rank approximations. *Journal of the ACM*, 51(6):1025–1041, 2004.

[GCD20] Casper Gyurik, Chris Cade, and Vedran Dunjko. Towards quantum advantage via topological data analysis, 2020. arXiv: 2005.02607

[Gil10] Michael I. Gil. Perturbations of functions of diagonalizable matrices. *Electronic Journal of Linear Algebra*, 20:303–313, 2010.

[GLM08] Vittorio Giovannetti, Seth Lloyd, and Lorenzo Maccone. Quantum random access memory. *Physical Review Letters*, 100(16):160501, 2008. arXiv: 0708.1879

[GLT18] András Gilyén, Seth Lloyd, and Ewin Tang. Quantum-inspired low-rank stochastic regression with logarithmic dependence on the dimension. arXiv: 1811.04909, 2018.

[GR02] Lov Grover and Terry Rudolph. Creating superpositions that correspond to efficiently integrable probability distributions. arXiv: quant-ph/0208112, 2002.

[GSLW19] András Gilyén, Yuan Su, Guang Hao Low, and Nathan Wiebe. Quantum singular value transformation and beyond: Exponential improvements for quantum matrix arithmetics. In *Proceedings of the 51st ACM Symposium on the Theory of Computing (STOC)*, pages 193–204, 2019. arXiv: 1806.01838

[HHL09] Aram W. Harrow, Avinatan Hassidim, and Seth Lloyd. Quantum algorithm for linear systems of equations. *Physical Review Letters*, 103(15):150502, 2009. arXiv: 0811.3171

[HKP21] Hsin-Yuan Huang, Richard Kueng, and John Preskill. Information-theoretic bounds on quantum advantage in machine learning. *Physical Review Letters*, 126(19):190505, 2021. arXiv: 2101.02464

[HKS11] Elad Hazan, Tomer Koren, and Nati Srebro. Beating SGD: Learning SVMs in sublinear time. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 1233–1241, Red Hook, NY, USA, 2011. Curran Associates, Inc.

[JLGS20] Dhawal Jethwani, François Le Gall, and Sanjay K. Singh. Quantum-inspired classical algorithms for singular value transformation. In *Proceedings of the 45th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 53:1–53:14, 2020. arXiv: 1910.05699

[Kal07] Satyen Kale. *Efficient algorithms using the multiplicative weights update method*. PhD thesis, Princeton University, 2007.

[KP17] Iordanis Kerenidis and Anupam Prakash. Quantum recommendation systems. In *Proceedings of the 8th Innovations in Theoretical Computer Science Conference (ITCS)*, pages 49:1–49:21, 2017. arXiv: 1603.08675

[KP20] Iordanis Kerenidis and Anupam Prakash. Quantum gradient descent for linear systems and least squares. *Physical Review A*, 101(2):022316, 2020. arXiv: 1704.04992

[KS48] Robert Karplus and Julian Schwinger. A note on saturation in microwave spectroscopy. *Physical Review*, 73:1020–1026, 1948.

[KV17] Ravindran Kannan and Santosh Vempala. Randomized algorithms in numerical linear algebra. *Acta Numerica*, 26:95–135, 2017.

[LC17] Guang Hao Low and Isaac L. Chuang. Optimal Hamiltonian simulation by quantum signal processing. *Physical Review Letters*, 118(1):010501, 2017. arXiv: 1606.02685

[LGZ16] Seth Lloyd, Silvano Garnerone, and Paolo Zanardi. Quantum algorithms for topological and geometric analysis of data. *Nature Communications*, 7:10138, 2016. arXiv: 1408.3106

[Llo96] Seth Lloyd. Universal quantum simulators. *Science*, 273(5278):1073–1078, 1996.

[LMR13] Seth Lloyd, Masoud Mohseni, and Patrick Rebentrost. Quantum algorithms for supervised and unsupervised machine learning, 2013. arXiv: 1307.0411

[LMR14] Seth Lloyd, Masoud Mohseni, and Patrick Rebentrost. Quantum principal component analysis. *Nature Physics*, 10:631–633, 2014. arXiv: 1307.0401

[LRS15] James R. Lee, Prasad Raghavendra, and David Steurer. Lower bounds on the size of semidefinite programming relaxations. In *Proceedings of the 47th ACM Symposium on the Theory of Computing (STOC)*, pages 567–576, 2015. arXiv: 1411.6317

[Mah11] Michael W. Mahoney. Randomized algorithms for matrices and data. *Foundations and Trends® in Machine Learning*, 3(2):123–224, 2011.

[McD89] Colin McDiarmid. *On the method of bounded differences*, page 148–188. London Mathematical Society Lecture Note Series. Cambridge University Press, Cambridge, England, 1989.

[MMD08] Michael W. Mahoney, Mauro Maggioni, and Petros Drineas. Tensor-CUR decompositions for tensor-based data. *SIAM Journal on Matrix Analysis and Applications*, 30(3):957–987, 2008.

[MRTC21] John M. Martyn, Zane M. Rossi, Andrew K. Tan, and Isaac L. Chuang. Grand unification of quantum algorithms. *Physical Review X*, 2(4):040203, 2021. arXiv: 2105.02859

[PC99] Victor Y. Pan and Zhao Q. Chen. The complexity of the matrix eigenproblem. In *Proceedings of the 31st ACM Symposium on the Theory of Computing (STOC)*, page 507–516, 1999.

[Pra14] Anupam Prakash. *Quantum Algorithms for Linear Algebra and Machine Learning*. PhD thesis, University of California at Berkeley, 2014.

[Pre18] John Preskill. Quantum Computing in the NISQ era and beyond. *Quantum*, 2:79, 2018. arXiv: 1801.00862

[RL18] Patrick Rebentrost and Seth Lloyd. Quantum computational finance: quantum algorithm for portfolio optimization. arXiv: 1811.03975, 2018.

[RML14] Patrick Rebentrost, Masoud Mohseni, and Seth Lloyd. Quantum support vector machine for big data classification. *Physical Review Letters*, 113(13):130503, 2014. arXiv: 1307.0471

[RSW+19] Patrick Rebentrost, Maria Schuld, Leonard Wossnig, Francesco Petruccione, and Seth Lloyd. Quantum gradient descent and Newton's method for constrained polynomial optimization. *New Journal of Physics*, 21(7):073023, 2019. arXiv: 1612.01789

[RV07] Mark Rudelson and Roman Vershynin. Sampling from large matrices: An approach through geometric functional analysis. *Journal of the ACM (JACM)*, 54(4):21–es, July 2007.

[RWC+20] Alessandro Rudi, Leonard Wossnig, Carlo Ciliberto, Andrea Rocchetto, Massimiliano Pontil, and Simone Severini. Approximating Hamiltonian dynamics with the Nyström method. *Quantum*, 4:234, 2020. arXiv: 1804.02484

[Sho97] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, 1997. Earlier version in FOCS'94. arXiv: quant-ph/9508027

[SWZ16] Zhao Song, David Woodruff, and Huan Zhang. Sublinear time orthogonal tensor decomposition. In *Advances in Neural Information Processing Systems 29*, pages 793–801. Curran Associates, Inc., Red Hook, NY, USA, 2016.

[Tan19] Ewin Tang. A quantum-inspired classical algorithm for recommendation systems. In *Proceedings of the 51st ACM Symposium on the Theory of Computing (STOC)*, pages 217–228, 2019. arXiv: 1807.04271

[Tan21] Ewin Tang. Quantum principal component analysis only achieves an exponential speedup because of its state preparation assumptions. *Physical Review Letters*, 127(6):060503, 2021. arXiv: 1811.00414

[Tao10] Terence Tao. 254a, Notes 3a: Eigenvalues and sums of Hermitian matrices, 2010. https://terrytao.wordpress.com/2010/01/12/254a-notes-3a-eigenvalues-and-sums-of-hermitian-matrices/.

[VdN11] Maarten Van den Nest. Simulating quantum computers with probabilistic methods. *Quantum Information and Computation*, 11(9&10):784–812, 2011. arXiv: 0911.1624

[Vos91] Michael D. Vose. A linear algorithm for generating random numbers with a given distribution. *IEEE Transactions on Software Engineering*, 17(9):972–975, 1991.

[Wel09] Max Welling. Fisher linear discriminant analysis. https://www.ics.uci.edu/~welling/teaching/273ASpring09/Fisher-LDA.pdf, 2009.

[Woo14] David P. Woodruff. Sketching as a tool for numerical linear algebra. *Foundations and Trends® in Theoretical Computer Science*, 10(1–2):1–157, 2014.

[WZP18] Leonard Wossnig, Zhikuan Zhao, and Anupam Prakash. Quantum linear system algorithm for dense matrices. *Physical Review Letters*, 120(5):050502, 2018. arXiv: 1704.06174

[YSSK20] Hayata Yamasaki, Sathyawageeswar Subramanian, Sho Sonoda, and Masato Koashi. Learning with optimized random features: Exponential speedup by quantum machine learning without sparsity and low-rank assumptions. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 13674–13687. Curran Associates, Inc., Red Hook, NY, USA, 2020. arXiv: 2004.10756

[ZFF19] Zhikuan Zhao, Jack K. Fitzsimons, and Joseph F. Fitzsimons. Quantum-assisted Gaussian process regression. *Physical Review A*, 99(5):052331, 2019. arXiv: 1512.03929

A Proof sketch for Remark 6.4

Recall that we wish to show that, given $\mathrm{SQ}(A)$, we can simulate $\mathrm{SQ}_\phi(B)$ for B such that $\|B - A^\dagger\| \leq \varepsilon\|A\|$ with probability $\geq 1 - \delta$.

Following the argument from Remark 6.8, we can find a $B \coloneqq AR^\dagger \bar{t}(CC^\dagger)R$ satisfying the above property in $\widetilde{\mathcal{O}}\Big(\frac{\|A\|_{\mathrm{F}}^{28}}{\|A\|^{28}\varepsilon^{22}} \log^3 \frac{1}{\delta}\Big)$ (rescaling ε appropriately). Here, R and $\bar{t}(CC^\dagger)$ come from an application of Theorem 5.1 with $t : \mathbb{R} \to \mathbb{C}$ a smooth step function that goes from zero to one around $(\varepsilon\|A\|)^2$. If we had sampling and query access to the columns of $AR^\dagger$, we would be done, since then

$B = \sum_{i=1}^{r}\sum_{j=1}^{r}[\bar{t}(CC^\dagger)](i,j)[A'R'^\dagger](\cdot,i)R(j,\cdot)$, and we can express B as a sum of r^2 outer products of vectors that we have sampling and query access to. This gives us both $\mathrm{SQ}_\phi(B)$ and $\mathrm{SQ}_\phi(B^\dagger)$.

We won't get exactly this, but using that $\bar{t}(CC^\dagger) = (C^+_{\varepsilon\|A\|/2})^\dagger t(C^\dagger C)C^+_{\varepsilon\|A\|/2}$, for $UDV^\dagger$ the SVD of C and $U_{\varepsilon\|A\|/2}D_{\varepsilon\|A\|/2}V^\dagger_{\varepsilon\|A\|/2}$ the SVD truncated to singular values at least $\varepsilon\|A\|/2$, we can rewrite

$$B = A(R^\dagger U_{\varepsilon\|A\|/2}D^+_{\varepsilon\|A\|/2})(t(D^2)D^+_{\varepsilon\|A\|/2}U^\dagger_{\varepsilon\|A\|/2})R.$$

Now it suffices to get sampling and query access to the columns of $A(R^\dagger U_{\varepsilon\|A\|/2}D^+_{\varepsilon\|A\|/2})$, and by Lemma 4.14, $R^\dagger U_{\varepsilon\|A\|/2}D^+_{\varepsilon\|A\|/2}$ is an ε^3-approximate isometry. Further, we can lower bound the norms of these columns, using that $R^\dagger R \approx A^\dagger A$ and $CC^\dagger \approx RR^\dagger$.

$$\begin{aligned}
\|A(R^\dagger U_{\varepsilon\|A\|/2}D^+_{\varepsilon\|A\|/2})\|^2 &= \|(U_{\varepsilon\|A\|/2}D^+_{\varepsilon\|A\|/2})^\dagger RA^\dagger AR^\dagger(U_{\varepsilon\|A\|/2}D^+_{\varepsilon\|A\|/2})\| \\
&\approx \|(U_{\varepsilon\|A\|/2}D^+_{\varepsilon\|A\|/2})^\dagger RR^\dagger RR^\dagger(U_{\varepsilon\|A\|/2}D^+_{\varepsilon\|A\|/2})\| \\
&= \|RR^\dagger(U_{\varepsilon\|A\|/2}D^+_{\varepsilon\|A\|/2})\|^2 \\
&\approx \|CC^\dagger U_{\varepsilon\|A\|/2}D^+_{\varepsilon\|A\|/2}\|^2 \\
&= \|UD^2U^\dagger U_{\varepsilon\|A\|/2}D^+_{\varepsilon\|A\|/2}\|^2 \\
&\geq \varepsilon^2\|A\|^2
\end{aligned}$$

Consider one particular column $v \coloneqq [R^\dagger U_{\varepsilon\|A\|/2}D^+_{\varepsilon\|A\|/2}](\cdot,\ell)$; summarizing our prior arguments, we know $\|v\| \geq \frac{1}{2}$ from approximate orthonormality and $\|Av\| \gtrsim \varepsilon\|A\|$, which we just showed. We can also query for entries of v since it is a linear combination of rows of R. We make one more approximation $Av \approx u$, using Lemma 4.12 as we do in Corollary 6.7. That is, if we want to know $[Av](i) = A(i,\cdot)v$, we use our inner product protocol to approximate it to $\gamma\|A(i,\cdot)\|\|v\|$ error, and declare it to be $u(i)$. This implicitly defines u via an algorithm to compute its entries from $\mathrm{SQ}(A)$ and $\mathrm{Q}(v)$. Let B' be the version of B, with the columns of $AR^\dagger U_{\varepsilon\|A\|/2}D^+_{\varepsilon\|A\|/2}$ replaced with their u versions. One can set γ such that the correctness bound $\|B' - A^\dagger\| \lesssim \varepsilon$ and our lower bound $u \gtrsim \varepsilon\|A\|$ both still hold. All we need now to get $\mathrm{SQ}_\phi(u)$ (thereby completing our proof sketch) is a bound $\tilde{u}$ such that we have $\mathrm{SQ}(\tilde{u})$. We will take $\tilde{u}(i) \coloneqq 2\|A(i,\cdot)\|$. We have $\mathrm{SQ}(\tilde{u})$ immediately from $\mathrm{SQ}(A)$, $\phi = \|\tilde{u}\|^2/\|u\|^2 \lesssim \varepsilon^2\|A\|_{\mathrm{F}}^2/\|A\|^2$ (from our lower bound on $\|u\|$), and $|\tilde{u}(i)| \geq \|A(i,\cdot)\| + \gamma\|A(i,\cdot)\|v\| \geq |u(i)|$ (from our correctness bound from Lemma 4.12).

B Deferred proofs

Lemma 2.5. *If $\hat{X} \in \mathbb{C}^{m\times n}$ is an α-approximate isometry, then there is an exact isometry $X \in \mathbb{C}^{m\times n}$ with the same columnspace as $\hat{X}$ such that $\|\hat{X} - X\| \leq \alpha$. Furthermore, for any matrix $Y \in \mathbb{C}^{n\times n}$,*

$$\|\hat{X}Y\hat{X}^\dagger - XYX^\dagger\| \leq (2\alpha + \alpha^2)\|Y\|.$$

If $\alpha < 1$, then $\|\hat{X}^+\| \leq (1-\alpha)^{-1}$ and

$$\|\hat{X}Y\hat{X}^\dagger - XYX^\dagger\| \leq \alpha\frac{2-\alpha}{(1-\alpha)^2}\|\hat{X}Y\hat{X}^\dagger\|.$$

Proof. Let $\hat{X} = UDV^\dagger$ be a singular value decomposition of $\hat{X}$, with singular values $\sigma_1, \ldots, \sigma_n$ and $U \in \mathbb{C}^{m\times n}$, $D \in \mathbb{R}^{n\times n}$, $V \in \mathbb{C}^{n\times n}$. We set $X \coloneqq UV^\dagger$ which is an isometry since $(UV^\dagger)^\dagger UV^\dagger = I$,

and has the same columnspace as $\hat{X}$, and

$$\begin{aligned}\|UV^\dagger - \hat{X}\| = \|UV^\dagger - UDV^\dagger\| = \|I - D\| &= \max_{i\in[n]}|1-\sigma_i| \le \max_{i\in[n]}|1-\sigma_i||1+\sigma_i| \\ &= \max_{i\in[n]}|1-\sigma_i^2| = \|I - D^2\| = \|I - VDU^\dagger UDV^\dagger\| = \|I - \hat{X}^\dagger\hat{X}\| \le \alpha.\end{aligned}$$

Consequently,

$$\begin{aligned}\|\hat{X}Y\hat{X}^\dagger - XYX^\dagger\| &\le \|\hat{X}Y(\hat{X}-X)^\dagger\| + \|(\hat{X}-X)YX\| \\ &\le \alpha(\|\hat{X}Y\| + \|YX\|) \\ &\le \alpha(\|XY\| + \alpha\|Y\| + \|YX\|) \\ &= (2\alpha+\alpha^2)\|Y\|\end{aligned}$$

Suppose $\alpha < 1$, ruling out the possibility that $\hat{X}$ is the zero matrix. Then by Lemma 4.11 we have

$$\begin{aligned}\|\hat{X}^+\| = \max_{i\in[n]}\frac{1}{\sigma_i} &\le \frac{1}{1-\alpha}, \text{ and consequently} \\ \|\hat{X}Y\hat{X}^\dagger - XYX^\dagger\| &\le \alpha(\|\hat{X}Y\| + \|Y\|) \\ &\le \alpha(\|\hat{X}Y\hat{X}^\dagger\|\|\hat{X}^+\| + \|\hat{X}Y\hat{X}^\dagger\|\|\hat{X}^+\|^2) \\ &\le \alpha\frac{1-\alpha+1}{(1-\alpha)^2}\|\hat{X}Y\hat{X}^\dagger\|.\end{aligned}$$

□

Lemma 4.5 (Matrix multiplication by subsampling [DKM06, Theorem 1])**.** *Suppose we are given $X \in \mathbb{C}^{n\times m}, Y \in \mathbb{C}^{n\times p}, r \in \mathbb{N}$ and a distribution $p \in \mathbb{R}^n$ satisfying the oversampling condition that, for some $\phi \ge 1$,*

$$p(k) \ge \frac{\|X(k,\cdot)\|\|Y(k,\cdot)\|}{\phi\sum_\ell \|X(\ell,\cdot)\|\|Y(\ell,\cdot)\|}.$$

Let $S \in \mathbb{R}^{r\times n}$ be sampled according to p. Then $X^\dagger S^\dagger SY$ is an unbiased estimator for $X^\dagger Y$ and

$$\Pr\left[\|X^\dagger S^\dagger SY - X^\dagger Y\|_{\mathrm{F}} < \sqrt{\frac{8\phi^2\log(2/\delta)}{r}}\underbrace{\sum_\ell \|X(\ell,\cdot)\|\|Y(\ell,\cdot)\|}_{\le\|X\|_{\mathrm{F}}\|Y\|_{\mathrm{F}}}\right] > 1-\delta.$$

Proof. Using that the rows of S are selected independently, we can conclude the following:

$$\begin{aligned}
\mathbb{E}[(SX)^\dagger(SY)] &= r \cdot \mathbb{E}[[SX](1,\cdot)^\dagger[SY](1,\cdot)] = r\sum_{i=1}^{n} p(i)\frac{X(i,\cdot)^\dagger Y(i,\cdot)}{rp(i)} = X^\dagger Y \\
\mathbb{E}[\|X^\dagger S^\dagger SY - X^\dagger Y\|_{\mathrm{F}}^2] &= \sum_{i=1}^{m}\sum_{j=1}^{p}\mathbb{E}\big[\big|[X^\dagger S^\dagger SY - X^\dagger Y](i,j)\big|^2\big] \\
&= r\sum_{i=1}^{m}\sum_{j=1}^{p}\mathbb{E}\big[\big|[SX](1,i)^\dagger[SY](1,j) - [X^\dagger Y](i,j)\big|^2\big] \\
&\le r\sum_{i=1}^{m}\sum_{j=1}^{p}\mathbb{E}\big[\big|[SX](1,i)^\dagger[SY](1,j)\big|^2\big] \\
&= r\mathbb{E}\big[\|[SX](1,\cdot)\|^2\|[SY](1,\cdot)\|^2\big] \\
&= r\sum_{k=1}^{n} p(k)\frac{\|X(k,\cdot)\|^2}{r\cdot p(k)}\frac{\|Y(k,\cdot)\|^2}{r\cdot p(k)} \\
&= \frac{1}{r}\sum_{k=1}^{n}\frac{1}{p(k)}\|X(k,\cdot)\|^2\|Y(k,\cdot)\|^2 \\
&\le \frac{1}{r}\sum_{k=1}^{n}\frac{\phi\sum_\ell \|X(\ell,\cdot)\|\|Y(\ell,\cdot)\|}{\|X(k,\cdot)\|\|Y(k,\cdot)\|}\|X(k,\cdot)\|^2\|Y(k,\cdot)\|^2 \\
&= \frac{\phi}{r}\Big(\sum_k \|X(k,\cdot)\|\|Y(k,\cdot)\|\Big)^2.
\end{aligned}$$

To prove concentration, we use McDiarmid's "independent bounded difference inequality" [McD89].

Lemma B.1 ([McD89, Lemma (1.2)])**.** *Let $X_1,\dots,X_c$ be independent random variables with X_s taking values in a set A_s for all $s\in[c]$. Suppose that f is a real valued measurable function on the product set $\Pi_s A_s$ such that $|f(x)-f(x')|\le b_s$ whenever the vectors x and x' differ only in the s-th coordinate. Let Y be the random variable $f[X_1,\dots,X_c]$. Then for any $\gamma>0$:*

$$\Pr[|Y-\mathbb{E}[Y]|\ge\gamma] \le 2\exp\Big(-\frac{2\gamma^2}{\sum_s b_s^2}\Big).$$

To use Lemma B.1, we think about this expression as a function of the indices that are randomly chosen from p. That is, let f be the function $[n]^r\to\mathbb{R}$ defined to be

$$f(i_1,i_2,\dots,i_r) \coloneqq \Big\|X^\dagger Y - \sum_{s=1}^{r}\frac{1}{r\cdot p(i_s)}X(i_s,\cdot)^\dagger Y(i_s,\cdot)\Big\|_{\mathrm{F}},$$

Then, by Jensen's inequality, we have

$$\mathbb{E}[f] = \mathbb{E}[\|X^\dagger S^\dagger SY - X^\dagger Y\|_{\mathrm{F}}] \le \sqrt{\mathbb{E}[\|X^\dagger S^\dagger SY - XY\|_{\mathrm{F}}^2]} \le \sqrt{\frac{\phi}{r}}\sum_k \|X(k,\cdot)\|\|Y(k,\cdot)\|.$$

Now suppose that the index sequences $\vec{i}$ and $\vec{i}'$ only differ at the s-th position. Then by the triangle inequality,

$$|f(\vec{i}) - f(\vec{i}')| \le \frac{1}{r}\Big\|\frac{1}{p(i_s)}X(i_s,\cdot)^\dagger Y(i_s,\cdot) - \frac{1}{p(i'_s)}X(i'_s,\cdot)^\dagger Y(i'_s,\cdot)\Big\|_{\mathrm{F}}$$
$$\le \frac{2}{r}\max_{k\in[n]}\Big\|\frac{1}{p(k)}X(k,\cdot)^\dagger Y(k,\cdot)\Big\|_{\mathrm{F}} \le \frac{2\phi}{r}\sum_{k=1}^{n}\|X(k,\cdot)\|\|Y(k,\cdot)\|.$$

Now, by Lemma B.1, we conclude that

$$\Pr\left[|f - E[f]| \ge \sqrt{\frac{2\phi^2\log(2/\delta)}{r}}\sum_k \|X(\cdot,k)\|\|Y(k,\cdot)\|\right] \le \delta.$$

So, with probability $\ge 1-\delta$,

$$\|X^\dagger S^\dagger SY - X^\dagger Y\|_{\mathrm{F}} \le \mathbb{E}[\|X^\dagger S^\dagger SY - X^\dagger Y\|_{\mathrm{F}}] + \sqrt{\frac{2\phi^2\log(2/\delta)}{r}}\sum_k \|X(\cdot,k)\|\|Y(k,\cdot)\|$$
$$\le \Big(\sqrt{\frac{\phi}{r}} + \sqrt{\frac{2\phi^2\log(2/\delta)}{r}}\Big)\sum_k \|X(\cdot,k)\|\|Y(k,\cdot)\|$$
$$\le \sqrt{\frac{8\phi^2\log(2/\delta)}{r}}\sum_k \|X(\cdot,k)\|\|Y(k,\cdot)\|. \qquad \square$$

Lemma 4.12 (Inner product estimation, [Tan19, Proposition 4.2])**.** *Given* $\mathrm{SQ}_\phi(u), \mathrm{Q}(v) \in \mathbb{C}^n$*, we can output an estimate* $c \in \mathbb{C}$ *such that* $|c - \langle u, v\rangle| \le \varepsilon$ *with probability* $\ge 1-\delta$ *in time* $\mathcal{O}\big(\phi\|u\|^2\|v\|^2\frac{1}{\varepsilon^2}\log\frac{1}{\delta}(\mathbf{sq}_\phi(u) + \mathbf{q}(v))\big)$.

Proof. Define a random variable Z by sampling an index from the distribution p given by $\mathrm{SQ}_\phi(u)$, and setting $Z \coloneqq u(i)v(i)/p(i)$. Then

$$\mathbb{E}[Z] = \langle u, v\rangle \quad \text{and} \quad \mathbb{E}[|Z|^2] = \sum_{i=1}^{n} p(i)\frac{|u(i)v(i)|^2}{p(i)^2} \le \sum_{i=1}^{n}|u(i)v(i)|^2\frac{\phi\|u\|^2}{|u(i)|^2} = \phi\|u\|^2\|v\|^2.$$

So, we just need to boost the quality of this random variable. Consider taking $\bar{Z}$ to be the mean of $x \coloneqq 8\phi\|u\|^2\|v\|^2\frac{1}{\varepsilon^2}$ independent copies of Z. Then, by Chebyshev's inequality (stated here for complex-valued random variables),

$$\Pr[|\bar{Z} - \mathbb{E}[\bar{Z}]| \ge \varepsilon/\sqrt{2}] \le \frac{2\operatorname{Var}[Z]}{x\varepsilon^2} \le \frac{1}{4}.$$

Next, we take the (component-wise) median of $y \coloneqq 8\log\frac{1}{\delta}$ independent copies of $\bar{Z}$, which we call $\tilde{Z}$, to decrease failure probability. Consider the median of the real parts of $\bar{Z}$. The key observation is that if $\Re(\tilde{Z} - \mathbb{E}[Z]) \ge \varepsilon/\sqrt{2}$, then at least half of the $\bar{Z}$'s satisfy $\Re(\bar{Z} - \mathbb{E}[Z]) \ge \varepsilon/\sqrt{2}$. Let $E_i = \chi(\Re(\bar{Z}_i - \mathbb{E}[Z]) \ge \varepsilon/\sqrt{2})$ be the characteristic function for this event for a particular mean. The above argument implies that $\Pr[E_i] \le \frac{1}{4}$. So, by Hoeffding's inequality,

$$\Pr\left[\frac{1}{q}\sum_{i=1}^{q}E_i \ge \frac{1}{2}\right] \le \Pr\left[\frac{1}{q}\sum_{i=1}^{q}E_i \ge \frac{1}{4} + \Pr[E_i]\right] \le \exp(-q/8) \le \frac{\delta}{2}.$$

With this combined with our key observation, we can conclude that $\Pr[\Re(\tilde{Z} - \langle u, v\rangle) \geq \varepsilon/\sqrt{2}] \leq \delta/2$. From a union bound together with the analogous argument for the imaginary component, we have $\Pr[|\tilde{Z} - \langle u, v\rangle| \geq \varepsilon] \leq \delta$ as desired. The time complexity is the number of samples multiplied by the time to create one instance of the random variable Z, which is $\mathcal{O}(\mathbf{sq}(u) + \mathbf{q}(v))$. □

Lemma 6.3. *Consider $p(x)$ a degree-d polynomial of parity-d such that $|p(x)| \leq 1$ for $x \in [-1, 1]$. Recall that, for a function $f : \mathbb{C} \to \mathbb{C}$, we define $\bar{f}(x) \coloneqq (f(x) - f(0))/x$ (and $\bar{f}(0) = f'(0)$ when f is differentiable at zero).*

- *If p is even, then $\max_{x\in[0,1]} |q(x)| \leq 1$, $\max_{x\in[-1,1]} |q'(x)| \lesssim d^2$, $\max_{x\in[-1,1]} |\bar{q}(x)| \lesssim d^2$, and $\max_{x\in[-1,1]} |\bar{q}'(x)| \lesssim d^4$.*
- *If p is odd, then $\max_{x\in[-1,1]} |q(x)| \lesssim d$, $\max_{x\in[-1,1]} |q'(x)| \lesssim d^3$, $\max_{x\in[-1,1]} |\bar{q}(x)| \lesssim d^3$, and $\max_{x\in[-1,1]} |\bar{q}'(x)| \lesssim d^5$.*

Proof. We use the following Markov-Bernstein inequality [BE95, 5.1.E.17.f]. For every $p \in \mathbb{C}[x]$ of degree at most d

$$\max_{x\in[-1,1]} |p^{(k)}(x)| \lesssim \left(\min\left(d^2, \frac{d}{\sqrt{1-x^2}}\right)\right)^k \max_{x\in[-1,1]} |p(x)|, \tag{24}$$

where $\lesssim$ hides a constant depending on k. Note that by replacing x in the above equation with $2y-1$, we get that $\max_{y\in[0,1]} |p^{(k)}(y)| \lesssim d^{2k} \max_{y\in[0,1]} |p(y)|$ (paying an additional 2^k constant factor).

We make a couple observations about $\bar{r}(x)$ using Taylor expansions, where $r(x)$ is any degree-d polynomial. First,

$$\bar{r}(x) = \frac{r(x) - r(0)}{x} = \frac{r(x) - (r(x) - r'(y)x)}{x} = r'(y),$$

where $y \in [0, x]$ comes from the remainder term of the Taylor expansion of $r(x)$ at x. Similarly,

$$\begin{aligned}\bar{r}'(x) = \left(\frac{r(x) - r(0)}{x}\right)' &= \frac{1}{x^2}\left(xr'(x) - r(x) + r(0)\right) \\ &= \frac{1}{x^2}\left(xr'(x) - r(x) + r(x) - r'(x)x + r''(y)\frac{x^2}{2}\right) = \frac{1}{2}r''(y)\end{aligned}$$

for some $y \in [0, x]$. Then, for p even, $\max_{x\in[0,1]} |q(x)| \leq 1$ by definition. We also have

$$\begin{aligned}\max_{x\in[0,1]} |q'(x)| &\lesssim d^2 \max_{x\in[0,1]} |q(x)| \leq d^2 \\ \max_{x\in[0,1]} |\bar{q}(x)| &\leq \max_{y\in[0,1]} |q'(y)| \lesssim d^2 \\ \max_{x\in[0,1]} |\bar{q}'(x)| &\leq \max_{y\in[0,1]} \frac{1}{2}|q''(y)| \lesssim d^4\end{aligned}$$

For p odd, the same argument applies provided we can show that $\max_{x\in[0,1]} |q(x)| \lesssim d$, which we do by splitting into two cases: $x \leq \frac{1}{2}$ and $x > \frac{1}{2}$.

$$\begin{aligned}\max_{x\in[0,\frac{1}{2}]} |q(x)| &= \max_{x\in[0,\frac{1}{2}]} \left|\frac{p(x)}{x}\right| = \max_{y\in[0,\frac{1}{2}]} |p'(y)| \lesssim \max_{x\in[0,\frac{1}{2}]} \frac{d}{\sqrt{1-x^2}} \max_{x\in[-1,1]} |p(x)| \lesssim d; \\ \max_{x\in(\frac{1}{2},1]} |q(x)| &= \max_{x\in(\frac{1}{2},1]} \left|\frac{p(x)}{x}\right| \leq \max_{x\in(\frac{1}{2},1]} |2p(x)| \leq 2.\end{aligned}$$

□

Corollary 6.18. *Given* $\mathrm{SQ}(X^T)$ *and* $\mathrm{SQ}(y)$*, with probability* $\geq 1-\delta$*, we can output a real number* $\hat{b}$ *such that* $|b-\hat{b}| \leq \varepsilon(1+b)$ *and* $\mathrm{SQ}_\phi(\hat{\alpha})$ *such that* $\|\hat{\alpha}-\alpha\| \leq \varepsilon\gamma\|y\|$*, where* α *and* b *come from Eq. (12). Our algorithm runs in* $\widetilde{\mathcal{O}}\big(\|X\|_{\mathrm{F}}^6\|X\|^{16}\gamma^{11}\varepsilon^{-6}\log^3\frac{1}{\delta}\big)$ *time, with* $\widetilde{\mathbf{sq}}_\phi(\hat{\alpha}) = \widetilde{\mathcal{O}}\Big(\|X\|_{\mathrm{F}}^4\|X\|^6\gamma^5\frac{\gamma^2 m}{\|\hat{\alpha}\|^2}\varepsilon^{-4}\log^2\frac{1}{\delta}\Big)$*. Note that when* $\gamma^{-1/2}$ *is chosen to be sufficiently large (e.g.* $O(\|X\|_{\mathrm{F}})$*) and* $\|\alpha\| = \Omega(\gamma\|y\|)$*, this runtime is dimension-independent.*

Proof. Denote $\sigma^2 \coloneqq \gamma^{-1}$, and redefine $X \leftarrow X^T$ (so we have $\mathrm{SQ}(X)$ instead of $\mathrm{SQ}(X^T)$). By the block matrix inversion formula[31] we know that

$$\begin{bmatrix} 0 & \vec{1}^T \\ \vec{1} & M \end{bmatrix}^{-1} = \begin{bmatrix} -\frac{1}{\vec{1}^T M^{-1}\vec{1}} & \frac{\vec{1}^T M^{-1}}{\vec{1}^T M^{-1}\vec{1}} \\ \frac{M^{-1}\vec{1}}{\vec{1}^T M^{-1}\vec{1}} & M^{-1} - \frac{M^{-1}\vec{1}\vec{1}^T M^{-1}}{\vec{1}^T M^{-1}\vec{1}} \end{bmatrix} \quad \Rightarrow \quad \begin{bmatrix} 0 & \vec{1}^T \\ \vec{1} & M \end{bmatrix}^{-1}\begin{bmatrix} 0 \\ y \end{bmatrix} = \begin{bmatrix} \frac{\vec{1}^T M^{-1} y}{\vec{1}^T M^{-1}\vec{1}} \\ M^{-1}\Big(y - \frac{\vec{1}^T M^{-1} y}{\vec{1}^T M^{-1}\vec{1}}\vec{1}\Big) \end{bmatrix}.$$

So, we have reduced the problem of inverting the modified matrix to just inverting M^{-1} where $M = X^TX + \sigma^{-2}I$. M is invertible because $M \succeq \sigma^2 I$. Note that $M^{-1} = f(X^TX)$, where

$$f(\lambda) \coloneqq \frac{1}{\lambda+\sigma^2}.$$

So, by Theorem 5.1, we can find $R^\dagger \bar{f}(CC^\dagger)R$ such that $\|R^\dagger\bar{f}(CC^\dagger)R + \frac{1}{\sigma^2}I - f(X^TX)\| \leq \varepsilon\sigma^{-2}$, where (because $L = \sigma^{-4}$, $\bar{L} = \sigma^{-6}$)

$$r = \widetilde{\mathcal{O}}\left(\frac{L^2\|A\|^2\|A\|_{\mathrm{F}}^2\sigma^4}{\varepsilon^2}\log\frac{1}{\delta}\right) = \widetilde{\mathcal{O}}\left(\frac{K\kappa}{\varepsilon^2}\log\frac{1}{\delta}\right),$$
$$c = \widetilde{\mathcal{O}}\left(\frac{\bar{L}^2\|A\|^6\|A\|_{\mathrm{F}}^2\sigma^4}{\varepsilon^2}\log\frac{1}{\delta}\right) = \widetilde{\mathcal{O}}\left(\frac{K\kappa^3}{\varepsilon^2}\log\frac{1}{\delta}\right).$$

So, the runtime for estimating this is $\widetilde{\mathcal{O}}\Big(\frac{K^3\kappa^5}{\varepsilon^6}\log^3\frac{1}{\delta}\Big)$. We further approximate using Lemma 4.6: we find $r_1 \approx R^\dagger\vec{1}$, $r_y \approx R^\dagger\vec{y}$, and $\gamma \approx \vec{1}^\dagger y$ in $O(r\frac{K}{\varepsilon^2}\log\frac{1}{\delta})$ time (for the first two) and $O(\frac{1}{\varepsilon^2}\log\frac{1}{\delta})$ time (for the last one) such that the following bounds hold:

$$\|R^\dagger\vec{1} - r_1\| \leq \varepsilon\sqrt{m}\sigma \qquad \|R^\dagger y - r_y\| \leq \varepsilon\sqrt{m}\sigma \qquad |\vec{1}^\dagger y - \gamma| \leq \varepsilon m \tag{25}$$

Via Lemma 5.2, we observe the following additional bounds:

$$\|M^{-1}\| \leq \sigma^{-2} \qquad \|R^\dagger(\bar{f}(CC^\dagger))^{1/2}\| \leq (1+\varepsilon)\sigma^{-1} \qquad \|(\bar{f}(CC^\dagger))^{1/2}\| \leq \sigma^{-2} \tag{26}$$

Now, we compute what the subsequent errors are for replacing M^{-1} with $N \coloneqq R^\dagger ZR + \frac{1}{\sigma^2}I$,

[31] In a more general setting, we would use the Sherman-Morrison inversion formula, or the analogous formula for functions of matrices subject to rank-one perturbations.

where $Z \coloneqq \bar{f}(CC^\dagger)$.

$$\begin{aligned}
\frac{\vec{1}^\dagger M^{-1} y}{\vec{1}^\dagger M^{-1}\vec{1}} &= \frac{\vec{1}^\dagger (R^\dagger Z R + \sigma^{-2} I) y \pm \|\vec{1}\|\|y\|\|R^\dagger Z R + \sigma^{-2} I - M^{-1}\|}{\vec{1}^\dagger (R^\dagger Z R + \sigma^{-2} I)\vec{1} \pm \|\vec{1}\|^2 \|R^\dagger Z R + \sigma^{-2} I - M^{-1}\|} \\
&= \frac{\vec{1}^\dagger R^\dagger Z R y + \sigma^{-2}\vec{1}^\dagger y \pm \varepsilon\sigma^{-2} m}{\vec{1}^\dagger R^\dagger Z R\vec{1} + \sigma^{-2}\vec{1}^\dagger\vec{1} \pm \varepsilon\sigma^{-2} m} && \text{by SVT bound} \\
&= \frac{\vec{1}^\dagger R^\dagger Z r_y \pm \|\vec{1}^\dagger R^\dagger Z\|\|Ry - r_y\| + \sigma^{-2}\gamma \pm \sigma^{-2}|\gamma - \vec{1}^\dagger y| \pm \varepsilon\sigma^{-2} m}{\vec{1}^\dagger R^\dagger Z r_1 \pm \|\vec{1}^\dagger R^\dagger Z\|\|R\vec{1} - r_1\| + (1 \pm \varepsilon)\sigma^{-2} m} \\
&= \frac{\vec{1}^\dagger R^\dagger Z r_y \pm (\sqrt{m}(1+\varepsilon)\sigma^{-3})(\varepsilon\sigma\sqrt{m}) + \sigma^{-2}\gamma \pm 2\varepsilon\sigma^{-2} m}{\vec{1}^\dagger R^\dagger Z r_1 \pm (\sqrt{m}(1+\varepsilon)\sigma^{-3})(\varepsilon\sigma\sqrt{m}) + \sigma^{-2} m \pm \varepsilon\sigma^{-2} m} && \text{by Eqs. (25) and (26)} \\
&= \frac{r_1^\dagger Z r_y \pm \|R\vec{1} - r_1\|\|Z r_y\| + \sigma^{-2}\gamma \pm \mathcal{O}\big(\varepsilon\sigma^{-2} m\big)}{r_1^\dagger Z r_1 \pm \|R\vec{1} - r_1\|\|Z r_1\| + \sigma^{-2} m \pm \mathcal{O}(\varepsilon\sigma^{-2} m)} \\
&= \frac{r_1^\dagger Z r_y \pm \varepsilon\sigma\sqrt{m}(\|ZRy\| + \|Z\|\|Ry - r_y\|) + \sigma^{-2}\gamma \pm \mathcal{O}\big(\varepsilon\sigma^{-2} m\big)}{r_1^\dagger Z r_1 \pm \varepsilon\sigma\sqrt{m}(\|ZR\vec{1}\| + \|Z\|\|R\vec{1} - r_1\|) + \sigma^{-2} m \pm \mathcal{O}(\varepsilon\sigma^{-2} m)} && \text{by Eq. (25)} \\
&= \frac{r_1^\dagger Z r_y + \sigma^{-2}\gamma \pm \mathcal{O}\big(\varepsilon\sigma^{-2} m\big)}{r_1^\dagger Z r_1 + \sigma^{-2} m \pm \mathcal{O}(\varepsilon\sigma^{-2} m)} && \text{by Eqs. (25) and (26)} \\
&= \frac{r_1^\dagger Z r_y + \sigma^{-2}\gamma}{r_1^\dagger Z r_1 + \sigma^{-2} m}(1 \pm \mathcal{O}(\varepsilon)) \pm \mathcal{O}(\varepsilon). && \text{by } r_1^\dagger Z r_1 \geq 0
\end{aligned}$$

We will approximate the output vector as

$$M^{-1} y - \frac{\vec{1}^\dagger M^{-1} y}{\vec{1}^\dagger M^{-1}\vec{1}} M^{-1}\vec{1} \approx R^\dagger Z r_y + \sigma^{-2} y - \frac{r_1^\dagger Z r_y + \sigma^{-2}\gamma}{r_1^\dagger Z r_1 + \sigma^{-2} m}(R^\dagger Z r_1 + \sigma^{-2}\vec{1}).$$

To analyze this, we first note that

$$\begin{aligned}
\|M^{-1} y - R^\dagger Z r_y + \sigma^{-2} y\| &\leq \|M^{-1} - R^\dagger Z R - \sigma^{-2} I\|\|y\| + \|R^\dagger Z\|\|Ry - r_y\| \\
&\leq \varepsilon\sigma^{-2}\sqrt{m} + (1+\varepsilon)\sigma^{-3}\varepsilon\sigma\sqrt{m} \lesssim \varepsilon\sigma^{-2}\sqrt{m}
\end{aligned}$$

and analogously, $\|M^{-1}\vec{1} - R^\dagger Z r_1 + \sigma^{-2}\vec{1}\| \lesssim \varepsilon\sigma^{-2}\sqrt{m}$. We also use that

$$\frac{\vec{1}^\dagger M^{-1} y}{\vec{1}^\dagger M^{-1}\vec{1}} \leq \frac{\|\vec{1} M^{-1/2}\|\|M^{-1/2} y\|}{\|M^{-1/2}\vec{1}\|^2} = \frac{\|M^{-1/2} y\|}{\|M^{-1/2}\vec{1}\|} \leq \frac{\|X\|}{\sigma}. \tag{27}$$

With these bounds, we can conclude that (continuing to use Eqs. (25) and (26))

$$\begin{aligned}
&\left\| M^{-1}\Big(y - \frac{\vec{1}^\dagger M^{-1} y}{\vec{1}^\dagger M^{-1}\vec{1}}\vec{1}\Big) - \Big(R^\dagger Z r_y + \sigma^{-2} y - \frac{r_1^\dagger Z r_y + \sigma^{-2}\gamma}{r_1^\dagger Z r_1 + \sigma^{-2} m}(R^\dagger Z r_1 + \sigma^{-2}\vec{1})\Big)\right\| \\
&\le \|M^{-1}y - R^\dagger Z r_y + \sigma^{-2} y\| + \left\| \frac{\vec{1}^\dagger M^{-1} y}{\vec{1}^\dagger M^{-1}\vec{1}} M^{-1}\vec{1} - \frac{r_1^\dagger Z r_y + \sigma^{-2}\gamma}{r_1^\dagger Z r_1 + \sigma^{-2} m}(R^\dagger Z r_1 + \sigma^{-2}\vec{1})\Big)\right\| \\
&\le \varepsilon\sigma^{-2}\sqrt{m} + \frac{\vec{1}^\dagger M^{-1} y}{\vec{1}^\dagger M^{-1}\vec{1}}\|M^{-1}\vec{1} - R^\dagger Z r_1 - \sigma^{-2}\vec{1}\| + \left|\frac{\vec{1}^\dagger M^{-1} y}{\vec{1}^\dagger M^{-1}\vec{1}} - \frac{r_1^\dagger Z r_y + \sigma^{-2}\gamma}{r_1^\dagger Z r_1 + \sigma^{-2} m}\right| \|R^\dagger Z r_1 + \sigma^{-2}\vec{1}\| \\
&\lesssim \Big(1 + \frac{\vec{1}^\dagger M^{-1} y}{\vec{1}^\dagger M^{-1}\vec{1}}\Big)\varepsilon\sigma^{-2}\sqrt{m} + \varepsilon\Big(1 + \frac{\vec{1}^\dagger M^{-1} y}{\vec{1}^\dagger M^{-1}\vec{1}}\Big)\|R^\dagger Z r_1 + \sigma^{-2}\vec{1}\| \\
&= \varepsilon\Big(1 + \frac{\vec{1}^\dagger M^{-1} y}{\vec{1}^\dagger M^{-1}\vec{1}}\Big)\Big(\sigma^{-2}\sqrt{m} + \|R^\dagger Z r_1 + \sigma^{-2}\vec{1}\|\Big) \\
&\lesssim \varepsilon\frac{\|X\|}{\sigma}\Big(\sigma^{-2}\sqrt{m} + \|M^{-1}\vec{1}\| + \|(R^\dagger Z R + \sigma^{-2} I - M^{-1})\vec{1}\| + \|R^\dagger Z r_1 - R^\dagger Z R\vec{1}\|\Big) \qquad \text{by Eq. (27)} \\
&\lesssim \varepsilon\frac{\|X\|}{\sigma}\Big(\sigma^{-2}\sqrt{m} + \sigma^{-2}\sqrt{m} + \varepsilon\sigma^{-2}\sqrt{m} + \varepsilon\sigma^{-2}\sqrt{m}\Big) \\
&\lesssim \varepsilon\frac{\|X\|}{\sigma}\sigma^{-2}\sqrt{m}.
\end{aligned}$$

So, by rescaling ε down by $\frac{\|X\|}{\sigma}$, it suffices to sample from

$$\hat{\alpha} \coloneqq R^\dagger Z\Big(r_y - \frac{r_1^\dagger Z r_y + \sigma^{-2}\gamma}{r_1^\dagger Z r_1 + \sigma^{-2} m} r_1\Big) - \sigma^{-2}\Big(y - \frac{r_1^\dagger Z r_y + \sigma^{-2}\gamma}{r_1^\dagger Z r_1 + \sigma^{-2} m}\vec{1}\Big).$$

To gain sampling and query access to the output, we consider this as a matrix-vector product, where the matrix is $(R^\dagger \mid y \mid \vec{1})$ and the vector is the corresponding coefficients in the linear combination. Then, by Lemmas 3.5 and 3.6, we can get $\mathrm{SQ}_\phi(\hat{\alpha})$ for

$$\begin{aligned}
\phi &= (r+2)\left(\frac{\|X\|_{\mathrm{F}}^2}{r}\left\|Z\Big(r_y - \frac{r_1^\dagger Z r_y + \sigma^{-2}\gamma}{r_1^\dagger Z r_1 + \sigma^{-2} m} r_1\Big)\right\|^2 + \sigma^{-4}\Big(\|y\|^2 + \Big(\frac{r_1^\dagger Z r_y + \sigma^{-2}\gamma}{r_1^\dagger Z r_1 + \sigma^{-2} m}\Big)^2 \|\vec{1}\|^2\Big)\right)\|\hat{\alpha}\|^{-2} \\
&\lesssim \left(\|X\|_{\mathrm{F}}^2 \frac{\|X\|^2}{\sigma^2}\sigma^{-6} m + r\sigma^{-4}\frac{\|X\|^2}{\sigma^2} m\right)\|\hat{\alpha}\|^{-2} \lesssim \Big(\frac{\|X\|_{\mathrm{F}}^2}{\sigma^2} + r\Big)\frac{\|X\|^2}{\sigma^2}\frac{\sigma^{-4} m}{\|\hat{\alpha}\|^2}
\end{aligned}$$

so $\widetilde{\mathbf{sq}}_\phi(\hat{\alpha}) = \phi\,\mathbf{sq}_\phi(\hat{\alpha})\log\frac{1}{\delta} = \mathcal{O}\Big(r(\frac{\|X\|_{\mathrm{F}}^2}{\sigma^2} + r)\frac{\|X\|^2}{\sigma^2}\frac{\sigma^{-4} m}{\|\hat{\alpha}\|^2}\log\frac{1}{\delta}\Big)$. □

Lemma 6.28 (Perturbations of the partition function)**.** *For all Hermitian matrices $H, \tilde{H} \in \mathbb{C}^{n\times n}$,*

$$\left|\mathrm{Tr}(e^{\tilde{H}}) - \mathrm{Tr}(e^H)\right| \le \left\|e^{\tilde{H}} - e^H\right\|_1 \le \left(e^{\|\tilde{H}-H\|} - 1\right)\mathrm{Tr}(e^H).$$

Proof. We will use the following formula introduced by [KS48, Fey51] (see also [Bel97, Page 181]):

$$\frac{d}{dt}e^{M(t)} = \int_0^1 e^{yM(t)}\frac{dM(t)}{dt}e^{(1-y)M(t)}dy. \tag{28}$$

Let $A \in \mathbb{C}^{n\times n}$ with $\|A\| \le 1$, we define the function $g_A(t) \coloneqq \operatorname{Tr}\left(Ae^{H+t(\tilde{H}-H)}\right)$, and observe that

$$\begin{aligned}
g'_A(t) =& \frac{d}{dt}\operatorname{Tr}\Big(Ae^{H+t(\tilde{H}-H)}\Big) && \text{by definiton}\\
=& \operatorname{Tr}\left(A\frac{d}{dt}e^{H+t(\tilde{H}-H)}\right) && \text{by linearity of trace}\\
=& \operatorname{Tr}\left(A\int_0^1 e^{y[H+t(\tilde{H}-H)]}(\tilde{H}-H)e^{(1-y)[H+t(\tilde{H}-H)]}dy\right) && \text{by Eq. (28)}\\
=& \int_0^1 \operatorname{Tr}\Big(Ae^{y[H+t(\tilde{H}-H)]}(\tilde{H}-H)e^{(1-y)[H+t(\tilde{H}-H)]}\Big)dy && \text{by linearity of trace}^{32}\\
\le& \int_0^1 \left\|Ae^{y[H+t(\tilde{H}-H)]}(\tilde{H}-H)e^{(1-y)[H+t(\tilde{H}-H)]}\right\|_1 dy && \text{by trace-norm inequality}\\
\le& \int_0^1 \left\|Ae^{y[H+t(\tilde{H}-H)]}\right\|_{\frac{1}{y}}\left\|(\tilde{H}-H)e^{(1-y)[H+t(\tilde{H}-H)]}\right\|_{\frac{1}{1-y}} dy && \text{by Hölder's inequality}\\
\le& \int_0^1 \|A\|\left\|e^{y[H+t(\tilde{H}-H)]}\right\|_{\frac{1}{y}}\left\|\tilde{H}-H\right\|\left\|e^{(1-y)[H+t(\tilde{H}-H)]}\right\|_{\frac{1}{1-y}} dy && \text{by Hölder's inequality}\\
\le& \left\|\tilde{H}-H\right\|\int_0^1 \left\|e^{y[H+t(\tilde{H}-H)]}\right\|_{\frac{1}{y}}\left\|e^{(1-y)[H+t(\tilde{H}-H)]}\right\|_{\frac{1}{1-y}} dy && \text{since } \|A\| \le 1\\
=& \left\|\tilde{H}-H\right\|\left\|e^{H+t(\tilde{H}-H)}\right\|_1. && (29)
\end{aligned}$$

Now we consider $z(t) \coloneqq g_I(t) = \operatorname{Tr}\left(e^{H+t(\tilde{H}-H)}\right)$. From Eq. (29) we have $z'(t) \le \|\tilde{H}-H\|z(t)$. Using Grönwall's differential inequality, we can conclude that $z(t) \le z(0)e^{t\|\tilde{H}-H\|}$ for every $t \in [0,\infty)$.

Finally, we use the fact that there exists a matrix A of operator norm at most 1 such that $\left\|e^{\tilde{H}}-e^H\right\|_1 = \operatorname{Tr}(A(e^{\tilde{H}}-e^H))$ (take, e.g., $\operatorname{sgn}(e^{\tilde{H}}-e^H)$). We finish the proof by observing that for such an A, $\left\|e^{\tilde{H}}-e^H\right\|_1 = \operatorname{Tr}(Ae^{\tilde{H}}) - \operatorname{Tr}(Ae^H) = g_A(1) - g_A(0) = \int_0^1 g'_A(t)dt$ and

$$\int_0^1 g'_A(t)dt \overset{(29)}{\le} \int_0^1 \|\tilde{H}-H\|z(t)dt \le z(0)\int_0^1 \|\tilde{H}-H\|e^{t\|\tilde{H}-H\|}dt = \operatorname{Tr}(e^H)\Big(e^{\|\tilde{H}-H\|}-1\Big). \qquad \square$$

[32] Note that in case $A = I$, by the cyclicity of trace, this equation implies that $\frac{d}{dt}\operatorname{Tr}(e^{H(t)}) = \operatorname{Tr}(e^{H(t)}\frac{d}{dt}H(t))$.