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 et al. [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 recent 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

†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"; additional support was provided by the Institute for Quantum Information and Matter, an NSF Physics Frontiers Center (NSF Grant PHY-1733907). Email: agilyen@caltech.edu

‡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 relatively new 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 described a "dequantized" algorithm that solves the same problem as the quantum algorithm and only suffers from a polynomial slowdown. Tang's algorithm crucially exploits the structure of the input 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 [Tan18], low-rank linear system solving [GLT18, CLW18], low-rank semidefinite program solving [CLLW19], support vector machines [DBH19], nonnegative matrix factorization [CLS+19], and minimal conical hull [DHLT19]. These results show that the advertised exponential speedups of many QML algorithms disappear if the corresponding classical algorithms can use input assumptions analogous to the state preparation assumptions of the quantum algorithms. Previous papers [Tan18, GLT18, CLW18] have observed that these techniques can likely be used to dequantize all QML that operates on low-rank data. Apart from a few QML algorithms that assume sparse input data, such as Harrow, Hassidim, and Lloyd's pioneering algorithm (HHL) for solving sparse systems of linear equations in time poly-logarithmic in input size [HHL09], much of QML depends on some low-rank assumption. As a consequence, these dequantization results have drastically changed our understanding of the landscape of potential QML algorithm speedups, by either providing strong barriers for or completely disproving the existence of exponential quantum speedups for the corresponding QML problems.

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 effectively captures all known linear algebraic QML techniques, a natural question is what aspects of this framework can be dequantized. Understanding the quantum-inspired analogue of QSVT promises a unification of dequantization results and more intuition about potential quantum speedups, which helps to guide future quantum algorithms research.

1.2 Main results

Our work gives a *simple* framework of 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 with it we can dequantize any QSVT algorithms in the QRAM input model.

Sampling and query access model. Our framework assumes a specific input model called *sampling and query access*, which can be thought of as a classical analogue to quantum state preparation assumptions, i.e., the ability to prepare a state $|v\rangle$ proportional to some input vector v. If we have sampling and query access to a vector $v \in \mathbb{C}^n$, denoted $\mathrm{SQ}(v)$, we can efficiently make the following kinds of queries (Definition 2.6): (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 $\|v\|$. If we have sampling and query access to a matrix $A \in \mathbb{C}^{m\times n}$, denoted $\mathrm{SQ}(A)$, we have $\mathrm{SQ}(A(i,\cdot))$ for all rows i and also $\mathrm{SQ}(a)$ for a the vector of row norms (i.e., $a(i) := \|A(i,\cdot)\|$).

To motivate this definition, we make the following observations about this input model. First, this model naturally admits classical algorithms with similar properties to the corresponding QML algorithms. Second, 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 2.15). 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.

Matrix arithmetic. We make a conceptual contribution by defining the slightly more general notion of *oversampling and query access* to a vector or matrix (Definition 2.8). We have oversampling and query access to a vector 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 $|\tilde{v}(i)| \geq |v(i)|$ for all indices i; the definition for a matrix is analogous. We restrict our focus to when we don't relax too much, that is, when $\|\tilde{v}\|/\|v\|$ is independent of input size. With this definition comes the insight that *this input model is closed under arithmetic operations*. Though this closure property doesn't explicitly come into play much in our application of our framework to dequantizing QML, the essential power of quantum-inspired algorithms lies in its ability to use sampling and query access to input matrices to build oversampling and query access to increasingly complex arithmetic expressions on input, possibly with some approximation error, without paying the (at least) linear time necessary to compute such expressions in conventional ways.

Some simple closure properties of oversampling and query access follow easily. Given access to two vectors u and v, we have access to their outer product $uv^\dagger$ (Lemma 2.12). Given access to a constant number of vectors $v_1, \ldots, v_t$, we have access to linear combinations $\sum_{i=1}^t \lambda_i v_t$, and analogously with linear combinations of matrices (Lemmas 2.10 and 2.13). Our main results can be seen as approximate closure properties[2]: given access to two matrices A, B, we have access to a

[1] This assumption is important. When input data is quantum (say, it is gathered experimentally from a quantum system), a classical computer has little hope of performing linear algebra on it efficiently.

[2] 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 3.16).

matrix Z close to the product $A^\dagger B$ (Lemma 3.6 and Remark 3.7); given access to a matrix A and a Lipschitz function f, we have access to a matrix Z close to $f(A^\dagger A)$ (Theorem 3.1)[3]. So, if we have (over)sampling and query access to our input vectors and matrices, we can perform matrix arithmetic (including matrix functions) on them while remaining in the same access model. Therefore, one can think about oversampling and query access as a classical analogue to the quantum block-encodings in quantum singular value transformation [GSLW19], which support linear combinations, products, and low-degree polynomials (that is, approximations of Lipschitz functions) of input matrices.

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 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 will keep Z as an outer product $A'^\dagger B'$ for convenience. Nevertheless, our central tool will be an approximate matrix product protocol: see the key lemma in Section 1.4. In fact, we leverage this protocol to get our matrix function closure property.

Even singular value transformation. Our main result is that, given (over)sampling and query access to an input matrix $A \in \mathbb{C}^{m\times n}$, we can find a succinct and useful description of an *even singular value transformation* of A. This primitive is based on the even SVT used by Gilyén et al. [GSLW19]: given a function $f\colon [0,\infty) \to \mathbb{C}$, the even SVT is $f(\sqrt{A^\dagger A})$, applying f to the singular values of A and replacing left singular vectors with the corresponding right singular vectors (so if $A = \sum \sigma_i u_i v_i^\dagger$ is the singular value decomposition of A, then $f(\sqrt{A^\dagger A}) = \sum f(\sigma_i) v_i v_i^\dagger$).

However, the accesses assumed in our closure properties are in some sense the most natural choices and require the least overhead.

[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, $f(H) := \sum_{i=1}^n f(\lambda_i) v_i v_i^\dagger$, for λ_i and v_i the eigenvalues and eigenvectors of H.

Main theorem (informal version of Theorem 3.1). *Suppose we are given sampling and query access to a matrix $A \in \mathbb{C}^{m\times n}$ (that is, $\mathrm{SQ}(A)$) and a function $f\colon [0,\infty) \to \mathbb{C}$ such that f and $\bar{f}(x) := (f(x) - f(0))/x$ are L-Lipschitz and $\bar{L}$-Lipschitz, respectively. Then, for sufficiently small $\varepsilon, \delta > 0$, we can find a subset of (normalized) rows of A, $R \in \mathbb{C}^{r\times n}$, and a subset of (normalized) columns of R, $C \in \mathbb{C}^{r\times c}$ such that*

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

Let T be the time the sampling and query oracle takes to respond. Then finding R and C and computing $\bar{f}(CC^\dagger)$ takes $\mathcal{O}\big(r^2c + rcT\big)$ time, where

$$r = \tilde{\Theta}\Big(\frac{L^2\|A\|^2\|A\|_{\mathrm{F}}^2}{\varepsilon^2}\log\frac{1}{\delta}\Big) \qquad c = \tilde{\Theta}\Big(\frac{\bar{L}^2\|A\|^6\|A\|_{\mathrm{F}}^2}{\varepsilon^2}\log\frac{1}{\delta}\Big).$$

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 (R corresponds to the 'R' of the RUR decomposition, and $\bar{f}(CC^\dagger) \in \mathbb{C}^{r\times r}$ corresponds to the 'U'). More precisely, an RUR decomposition expresses a desired matrix as a linear combination of r^2 outer products of rows of the input matrix.[4] The matrix U encodes the coefficients in the linear combination. We want our output in the form 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 $\mathrm{SQ}(A)$ implies that we can exploit the RUR structure to gain oversampling and query access to the output matrix, enabling the evaluation of matrix-vector expressions. In particular, for an RUR decomposition, we can get oversampling and query access to approximations of $R^\dagger URb$ and $R^\dagger URMb$, for a matrix $M \in \mathbb{C}^{n\times n}$ and a vector $b \in \mathbb{C}^n$, in time independent of n.

More general results follow as corollaries of our main result on even SVT. For an arbitrary matrix A with $\mathrm{SQ}(A)$, we can perform generic (non-even) SVT (Theorem 3.3), where the output is given as an approximate *CUR decomposition* expressing the desired matrix as a linear combination of outer products of columns and rows of A. We can also perform eigenvalue transformation on Hermitian matrices (Theorem 3.4), where the output is given as an approximate RUR decomposition. Given an RUR (or CUR) decomposition, one can also approximately diagonalize the matrix U in order to recover an approximate eigenvalue decomposition (or SVD) of the desired matrix, see e.g. Theorem 3.4.

However, using only our main theorem about even SVT, we can directly recover most existing quantum-inspired machine learning algorithms without using the more advanced Theorems 3.3 and 3.4 discussed above, yielding faster dequantization for QML algorithms. In Section 1.3, we outline our results recovering such applications.

For some intuition on error bounds and time complexity, we consider how the parameters in our main theorem behave in a restricted setting: suppose that A has minimum singular value σ and $\|A\|_{\mathrm{F}}/\sigma$ is dimension-independent.[5] This condition simultaneously bounds the rank and condition number of A. Further suppose[6] that f's Lipschitz constant satisfies

$$L\|A\|^2 < C \max_{x,y\in[0,\|A\|^2]} |f(x) - f(y)|$$

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

[5] 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 $A \leftarrow \alpha A$.

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

for some dimension-independent C. C must be at least one, so we can think about such an f as being at most C-times "steeper" compared to the least possible "steepness". Under these assumptions, we can get an RUR decomposition to additive error $(\varepsilon \max_{x,y\in[0,\|A\|^2]}|f(x) - f(y)|)$ in runtime independent of dimensions (i.e., r, c are dimensionless). The precise runtime is

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

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.

Implications for quantum singular value transformation. The QSVT framework of Gilyén et al. [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 certain kinds of degree-d polynomials of A to an input quantum state, incurring only about d times the implementation cost of the input block-encoding. 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]. This latter normalization means that QRAM-based QSVT has an implicit dependence on the Frobenius norm $\|A\|_{\mathrm{F}}$. This dependence on $\|A\|_{\mathrm{F}}$ suggests lack of exponential speedup for QRAM-based QSVT, since $\|A\|_{\mathrm{F}}$ is the key parameter in the complexity of our corresponding classical algorithms. This is in contrast to sparsity-based QSVT, which instead has dependence on $\|A\|$ and the sparsity s, and generalizes algorithms like HHL that solve BQP-complete problems.

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. Indeed, if we are given matrices and vectors in the QRAM data structure, then by converting them to block-encodings, we can apply any function to the input that can be obtained by composing addition, scalar multiplication, matrix multiplication, and singular value transformation. Since this data structure gives us sampling and query access to input, we can classically approximately evaluate the same types of expressions.

In particular, we show that we can apply the singular value transform of a matrix $A \in \mathbb{C}^{m\times n}$ satisfying $\|A\|_{\mathrm{F}} = 1$ to $b \in \mathbb{C}^n$ in QRAM (Theorem 3.14). Our algorithm simulates sampling and query access to $v := p^{(\mathrm{QV})}(A)b$ up to $\varepsilon\|v\|$ error in $\mathrm{poly}(d, \frac{1}{\varepsilon}, \frac{\|b\|}{\|v\|}, \log mn)$ time, where $p(x)$ is a degree-d polynomial of the kind QSVT can apply and $p^{(\mathrm{QV})}(A)$ is the type of SVT that QSVT performs on A (Definition 3.13). This runtime is only polynomially slower than the corresponding quantum algorithm, except in the ε parameter.[7] Theorem 3.14 also dequantizes QSVT for block-encodings derived from (purifications of) density operators [GSLW19, Lemma 45] that come from some well-structured classical data. This gives evidence that QSVT with these kinds of block-encodings do not give inherent exponential speedups (though, if input preparation/output analysis

[7]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.

protocols have no classical analogues, they can play a part in an algorithm achieving an exponential speedup). QSVT using other types of block-encodings remains intact, since such types could admit block-encodings for matrices with large Frobenius norm.

1.3 Applications: dequantizing QML & more

With our main results, we can recover existing quantum-inspired algorithms for recommendation systems [Tan19], principal component analysis [Tan18], supervised clustering [Tan18], support vector machines [DBH19], low-rank matrix inversion [GLT18, CLW18], and semidefinite program solving [CLLW19]. We also propose new quantum-inspired algorithms for low-rank Hamiltonian simulation and discriminant analysis (dequantizing the quantum algorithm of Cong & Duan [CD16]). Our framework achieves these results with a conceptually simple analysis, and often admits faster and more general results.

For the following results, we assume our sampling and query access to the input takes $\mathcal{O}(1)$ time. There are data structures that can support such queries (Remark 2.15), and if the input is in QRAM, the runtime only increases by at most a factor of log of input size. We note here that, though our outputs are often in the form of oversampling and query access SQ_ϕ (Definition 2.8), via rejection sampling, one can think about this access as the same as sampling and query access, except one can only compute the norm up to some relative error (Lemma 2.9).

Recommendation systems (Section 4.1). Our framework gives a simpler and faster variant of Tang's dequantization [Tan19] of Kerenidis & Prakash's quantum recommendation systems [KP17]. This 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]. The task is as follows (Problem 4.1): given sampling and query access to a matrix $A \in \mathbb{R}^{m\times n}$, a row index $i \in [m]$, and a singular value threshold σ, sample from the i^{th} row of some $\hat{A} \in \mathbb{R}^{m\times n}$, where $\hat{A}$ is a σ-thresholded low-rank approximation of A. Specifically, $\hat{A}$ should be $\varepsilon\|A\|_{\mathrm{F}}$-close in additive Frobenius norm error to a singular value transform of A that is smoothly thresholded to keep only singular vectors with value at least σ.

We can rewrite our target low-rank approximation as $A \cdot t(A^\dagger A)$, where t is a step function that is zero for $x \le \frac{5}{6}\sigma^2$, one for $x \ge \frac{7}{6}\sigma^2$, and a linear interpolation between the two for $x \in [\frac{5}{6}\sigma^2, \frac{7}{6}\sigma^2]$. In other words, our low-rank approximation is A multiplied by a smoothened projector. We can use our main theorem Theorem 3.1 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 3.2, we have $\mathrm{SQ}_\phi(A(i,\cdot)R^\dagger U R)$, so we can get the sample from this row as desired. The runtime is dominated by $\widetilde{\mathcal{O}}\Big(\frac{\|A\|_{\mathrm{F}}^6\|A\|^{10}}{\sigma^{16}\varepsilon^6}\log^3\frac{1}{\delta}\Big)$ (Corollary 4.3), an improvement on the previous runtime $\widetilde{\mathcal{O}}\Big(\frac{\|A\|_{\mathrm{F}}^{24}}{\sigma^{24}\varepsilon^{12}}\log^3\frac{1}{\delta}\Big)$ of [Tan19].

Supervised clustering (Section 4.2). Because dequantizing Lloyd, Mohseni, and Rebentrost's supervised clustering algorithm [LMR13] only requires simple sampling subroutines (demonstrated by Tang [Tan18]), our algorithm trivially recovers this result. Given some dataset of points $q_1, \ldots, q_{n-1} \in \mathbb{R}^d$, our goal is to estimate the distance between their centroid and a new point

$p \in \mathbb{R}^d$, $\|p - \frac{1}{n-1}(q_1 + \cdots + q_{n-1})\|^2$. By rewriting this expression as $\|wM\|^2$ for

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

we reduce this problem to estimating $wM(wM)^\dagger$ to ε additive error (Problem 4.5). This can be done with a simple inner product estimation procedure in time $\mathcal{O}\left(\frac{Z^2}{\varepsilon^2}\log\frac{1}{\delta}\right)$, where $Z = \|M\|_{\mathrm{F}}^2\|w\|^2 = 4(\|p\|^2 + \frac{1}{n-1}\sum_{i=1}^{n-1}\|q_i\|^2)$ (Corollary 4.6).

Principal component analysis (Section 4.3). Our framework improves on Tang's dequantization [Tan18] of the quantum principal component analysis (qPCA) algorithm [LMR14]. Since the actual task being solved by the original quantum algorithm is underspecified, we describe the task as is performed in the dequantization. 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 $\{v_i\}_{i=1}^k$, the goal is to compute eigenvalue estimates $\{\hat{\lambda}_i\}_{i=1}^k$ such that $\sum|\hat{\lambda}_i - \lambda_i| \le \varepsilon \operatorname{Tr}(X^\dagger X)$ and eigenvector estimates $\{\mathrm{SQ}_\phi(\hat{v}_i)\}_{i=1}^k$ such that $\|\hat{v}_i - v_i\| \le \varepsilon$ (Problem 4.7). To avoid degeneracy conditions, we must have a gap assumption granting $|\lambda_i - \lambda_{i+1}| \ge \eta\|X\|^2$ for all $i \in [k]$.

Then, we can 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 3.9) 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. Altogether, this algorithm runs in time $\widetilde{\mathcal{O}}\left(\frac{\|X\|_{\mathrm{F}}^6}{\|X\|^2\lambda_k^2}\eta^{-6}\varepsilon^{-6}\log^3\frac{k}{\delta}\right)$ (Corollary 4.8), a major improvement over the original dequantization's runtime $\widetilde{\mathcal{O}}\left(\frac{\|X\|_{\mathrm{F}}^{36}}{\lambda_k^{12}\|X\|^{12}}\eta^{-6}\varepsilon^{-12}\log^3\frac{k}{\delta}\right)$.

Matrix inversion (Section 4.4). Our framework can generalize prior work on quantum-inspired versions of low-rank matrix inversion [GLT18, CLW18]. Given a matrix $\mathrm{SQ}(A) \in \mathbb{C}^{m \times n}$ and a vector $\mathrm{SQ}(b) \in \mathbb{C}^m$, the goal is to obtain $\mathrm{SQ}_\phi(A^+_{\sigma,\eta}b)$ where $A^+_{\sigma,\eta}$ is a pseudo-inverse of A smoothly thresholded to invert only the singular values that are at least σ.

We can rewrite $A^+_{\sigma,\eta}b = \iota(A^\dagger A)A^\dagger b$ for ι a function encoding a thresholded inverse. Namely, $\iota(x) = 1/x$ for $x \ge \sigma^2$, $\iota(x) = 0$ for $x \le (1-\eta)\sigma^2$, and is a linear interpolation between the endpoints for $x \in [(1-\eta)\sigma^2, \sigma^2]$. By our main theorem, we can find an RUR decomposition for $\iota(A^\dagger A)$, from which we can then get $\mathrm{SQ}(R^\dagger URA^\dagger b)$ via sampling techniques. Altogether, this algorithm takes $\widetilde{\mathcal{O}}\left(\frac{\|A\|_{\mathrm{F}}^6\|A\|^{22}}{\sigma^{28}\eta^6\varepsilon^6}\log^3\frac{1}{\delta}\right)$ time with no restriction on A, whereas the result of [GLT18] applies to strictly rank-k A and gets the incomparable runtime $\widetilde{\mathcal{O}}\left(\frac{\|A\|_{\mathrm{F}}^6 k^6\|A\|^{16}}{\sigma^{22}\eta^6\varepsilon^6}\log^3\frac{1}{\delta}\right)$.

Support vector machines (Section 4.5). 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 [DBH19]. The idea is to find a hyperplane best explaining m data points in a matrix $\mathrm{SQ}(X) \in \mathbb{R}^{m\times n}$ with labels $\mathrm{SQ}(y) \in \{\pm 1\}^m$. With regularization, this reduces to approximately solving the linear system

$$\begin{bmatrix} 0 & \vec{1}^\dagger \\ \vec{1} & XX^\dagger + \gamma^{-1} I \end{bmatrix} \begin{bmatrix} b \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 \\ y \end{bmatrix}.$$

Call the above matrix F, and let $\hat{F} := F/\operatorname{Tr}(F)$. The quantum algorithm approximately solves the linear system by applying $\hat{F}^+_{\lambda,\eta}$ to y. So, our goal is to output $\mathrm{SQ}_\phi(v)$ for $v \in \mathbb{R}^{m+1}$ satisfying $\|v - \hat{F}^+_{\lambda,\eta}[\begin{smallmatrix}0\\y\end{smallmatrix}]\| \le \varepsilon \|\hat{F}^+_{\lambda,\eta}[\begin{smallmatrix}0\\y\end{smallmatrix}]\|$ (Problem 4.12). To do this, we use our matrix arithmetic techniques in order to get oversampling and query access to $\mathrm{SQ}_\varphi(\hat{F})$ from $\mathrm{SQ}(X)$. Then, using $\mathrm{SQ}_\varphi(\hat{F})$, we run the quantum-inspired matrix inversion algorithm discussed above (Corollary 4.11), immediately giving us the desired v. This takes time $\widetilde{\mathcal{O}}\big(\lambda^{-28}\eta^{-6}\varepsilon^{-6}\log^3\frac{1}{\delta}\big)$ (Corollary 4.13). We solve the problem in the same generality as the original quantum algorithm, unlike the prior dequantization result [DBH19], which also lacks explicit error bounds or runtime bounds; the paper simply argues that the algorithm is polynomial time in the right parameters.

Hamiltonian simulation (Section 4.6). Our framework can be used to give a Hamiltonian simulation algorithm for low-rank Hamiltonians. Given a Hermitian matrix $\mathrm{SQ}(H) \in \mathbb{C}^{n\times n}$ such that $\|H\| = t$ and $\|H^+\| \le 1/\sigma$ along with a unit vector $\mathrm{SQ}(b) \in \mathbb{C}^n$, the goal is to obtain $\mathrm{SQ}_\phi(v)$ where $\|v - e^{iH}b\|_{\mathrm{F}} \le \varepsilon$ (Problem 4.15).

In order to use our even SVT result, we split our desired transformation into even and odd parts: $e^{ix} = \cos(x) + i\sin(x) = \cos(x) + i\,\mathrm{sinc}(x)x$. We use even singular value transformation to apply the even functions cos and sinc; for an even function $g(x)$, let $f_g(x) := g(\sqrt{x})$, so that $g(H) = f_g(H^\dagger H)$ and we can rewrite

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

Then, using our main theorem, we can find RUR decompositions for both even SVTs, gaining sampling and query access to the matrix-vector products for the even and odd parts of the expression, from which sampling and query access to our estimate of $e^{iH}b$ follows. This takes $\widetilde{\mathcal{O}}\Big(\frac{\|H\|_{\mathrm{F}}^6 t^{10}}{\sigma^{16}} t^6 \varepsilon^{-6}\log^3\frac{1}{\delta}\Big)$ time (Corollary 4.16), which is dimension-independent if we think of the desired error as $t\varepsilon$, the natural choice for additive error. This algorithm also works if H is not strictly low-rank, in which case the output will be a version of e^{iH} where eigenvalues $\le \sigma$ are thresholded away. We also provide a version of this algorithm that works for all H without a dimension-independent runtime (Corollary 4.17). This version gets improved runtimes when $t = 1$.

Semidefinite program (SDP) solving (Section 4.7). We solve the problem of SDP-feasibility, improving on prior work of Chia et al. [CLLW19] dequantizing some versions of quantum SDP solvers [BKL+19, vAG19]. Given $m \in \mathbb{N}$, $b_1, \ldots, b_m \in \mathbb{R}$, and Hermitian matrices $A^{(1)}, \ldots, A^{(m)}$ such that $-I \preceq A^{(i)} \preceq I$ for all $i \in [m]$, let $\mathcal{S}_\varepsilon$ be the set of all X satisfying

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

The task is to differentiate whether $\mathcal{S}_0 \neq \varnothing$ (in which case the output should be an $X \in \mathcal{S}_\varepsilon$) or $\mathcal{S}_\varepsilon = \varnothing$ (in which case the output should be "infeasible"). Note that general SDPs can be reduced to this feasibility problem via a simple binary search.

By using the matrix multiplicative weights (MMW) method [AK16], SDP ε-feasibility reduces to estimating $\mathrm{Tr}[A^{(i)}X]$ up to $\varepsilon/4$ error given $\mathrm{SQ}(A^{(i)})$ for all $i \in [m]$ and X implicitly defined as a Gibbs state

$$X := \frac{\exp[-A]}{\mathrm{Tr}(\exp[-A])} \quad \text{where} \quad A := \frac{\varepsilon}{4} \sum_{\tau=1}^{\lesssim \ln(n)/\varepsilon^2} A^{(j_\tau)}.$$

To estimate $\mathrm{Tr}[A^{(i)}X]$, we first notice that we have $\mathrm{SQ}_\phi(A)$, since it is a linear combination of matrices that we have sampling and query access to (Lemma 2.13). Then, we can find approximations of the Gibbs state by applying eigenvalue transformation (Theorem 3.4) according to the exponential function to get $\exp[-A]$ as an RUR decomposition. Then the estimation of $\mathrm{Tr}[A^{(i)}X]$ can be performed by the usual SQ sampling techniques. This strategy solves the feasibility problem and when applicable outputs the ε-approximate solution of the SDP as an RUR decomposition (Corollary 4.21) in time[8] $\widetilde{\mathcal{O}}\Big(\frac{\|A^{(\cdot)}\|_{\mathrm{F}}^{22}}{\varepsilon^{46}} \ln^{23}(n) + m\frac{\|A^{(\cdot)}\|_{\mathrm{F}}^{14}}{\varepsilon^{28}} \ln^{13}(n)\Big)$.

For the same feasibility problem, the previous quantum-inspired SDP solver [CLLW19] proved a complexity bound $\widetilde{\mathcal{O}}\big(mr^{57}\varepsilon^{-92}\ln^{37}(n)\big)$, assuming that the constraint matrices have rank at most r. Since the rank constraint implies that $\big\|A^{(\cdot)}\big\|_{\mathrm{F}} \leq \sqrt{r}$, under this assumption our algorithm has complexity $\widetilde{\mathcal{O}}\big(r^{11}\varepsilon^{-46}\ln^{23}(n) + mr^7\varepsilon^{-28}\ln^{13}(n)\big)$. So, our new algorithm both solves a more general problem and also greatly improves the runtime.

Discriminant analysis (Section 4.8). We present a new dequantized algorithm, a classical analogue to Cong and Duan's quantum discriminant analysis algorithm [CD16]. The high-level idea is to find the vectors that best explain the way data points are classified. Cong and Duan reduces this idea to the following task: given matrices $\mathrm{SQ}(B)$ and $\mathrm{SQ}(W)$, find eigenvectors and eigenvalues of $\sqrt{W^\dagger W}(B^\dagger B)^{-1}\sqrt{W^\dagger W}$. They solve a version of this task (Problem 4.26) where one only needs to output approximate eigenvectors of $\mathrm{sqrt}(W^\dagger W)\,\mathrm{inv}(B^\dagger B)\,\mathrm{sqrt}(W^\dagger W)$, where sqrt and inv denote versions of square root and inverse thresholded at σ^2.

We achieve this goal by using Theorem 3.1 to approximate $\mathrm{sqrt}(W^\dagger W) \approx R_W^\dagger U_W R_W$ and $\mathrm{inv}(B^\dagger B) \approx R_B^\dagger U_B R_B$ by RUR decompositions. Then, we use Lemma 3.6 to approximate $R_W R_B^\dagger$ by small submatrices $R'_W R'^\dagger_B$. This yields an approximate RUR decomposition, $R_W^\dagger U R_W$ for $U = U_W R'_W R'^\dagger_B U_B R'_B R'^\dagger_W U_W$, of the matrix whose eigenvalues and vectors we want to find.

Finding eigenvectors from an RUR decomposition follows from an observation (Lemma 3.12): 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

[8] Here we use $\|A^{(\cdot)}\|_* := \max_{i\in[m]} \|A^{(i)}\|_*$. Note that this bound does not appear to be dimension-independent due to the normalizing assumption $\|A^{(\cdot)}\| \leq 1$. If we would relax this assumption, then we could get a dimension-independent bound corresponding to precision $\varepsilon\|A^{(\cdot)}\|$, by replacing $\|A^{(\cdot)}\|_{\mathrm{F}}$ with the "stable rank" $\|A^{(\cdot)}\|_{\mathrm{F}}/\|A^{(\cdot)}\|$. Then the resulting runtime bound is dimension-independent apart from the $\ln(n)$ factors, which come from MMW.

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. Our algorithm runs in $\widetilde{\mathcal{O}}\left(\left(\frac{\|B\|^4\|B\|_{\mathrm{F}}^6}{\varepsilon^6\sigma^{10}} + \frac{\|W\|^{10}\|W\|_{\mathrm{F}}^6}{\varepsilon^6\sigma^{16}}\right)\log^3\frac{1}{\delta}\right)$ time (Corollary 4.27).

What else is there? Though we have presented many dequantized versions of QML algorithms, the question remains of what QML algorithms don't have such versions. That is, what algorithms still have the potential to give exponential speedups?

Because QSVT generalizes essentially all known quantum linear algebra techniques, we restrict our focus to algorithms in that framework. As we noted previously, we only demonstrate lack of exponential speedup for QSVT with block-encodings coming from QRAM and density operators. Other kinds of block-encodings, such as those coming from sparsity assumptions, remain impervious to our techniques. The most well-known quantum linear algebra algorithms of this "dequantization-resistant" type are HHL [HHL09] and its derivatives. Sparse matrix inversion is BQP-complete, which explains why our techniques leave these speedups untouched. Nevertheless, HHL has serious caveats, as noted by Aaronson [Aar15]. In particular, HHL only gives an exponential speedup when the condition number of the input matrix is poly-logarithmic in dimension, which doesn't happen in typical datasets. This constraint hamstrings most attempts to apply HHL to practical problems, especially when combined with the typical QML constraints that quantum algorithms need quantum states as input and can only give quantum states as output. Work like Zhao et al. on Gaussian process regression [ZFF19] and Lloyd et al. on topological data analysis [LGZ16] attempt to address these issues to get a super-polynomial quantum speedup.

1.4 Techniques

Placing sampling and query access in the sketching context. The fundamental idea of quantum-inspired algorithms is to reduce dimensionality of input matrices to speed up linear algebra computations. So, using sketching techniques is natural here. Recall that the fundamental difference between quantum-inspired algorithms and traditional sketching algorithms is that we assume that we can perform 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. The kind of samples we get from sampling and query access is usually called *importance sampling* or *length-square sampling* in classical literature.

The quantum-inspired model is weaker than the standard sketching algorithm model (Remark 2.15): 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 $\mathcal{O}(\mathrm{nnz}(A) + T)$, where $\mathrm{nnz}(A)$ is the number of nonzero entries of A. So, we can also think about an $\mathcal{O}(T)$-time quantum-inspired algorithm as an $\mathcal{O}(\mathrm{nnz}(A) + T)$-time sketching algorithm, where the $\mathrm{nnz}(A)$ portion of the runtime can only be used to facilitate importance sampling. This viewpoint could be advantageous in some cases, for example in some streaming scenario [KP17]. Nevertheless, our primary motivation here is not to develop better generic sketching algorithms, but to better understand the scope of problems facilitating large quantum speed-ups.

A natural question is whether more modern types of sketches 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, Johnson-Lindenstrauss, and leverage score sampling all still take time linear in input-sparsity.

Furthermore, importance sampling is highly compatible with quantum-like algorithms: given the ability to query entries and obtain importance samples of the input, we can query entries and obtain importance samples of the output, analogously to the way quantum machine learning algorithms move from an input quantum state to an output quantum state. This insight unlocks surprising power in importance sampling. For example, it reveals that Frieze, Kannan, and Vempala's low-rank approximation algorithm (FKV) [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. Our goal is to develop a framework that demonstrates what can be done with importance sampling and establishes a classical frontier for quantum algorithms to push past.

Importance sampling to even singular value transformation. The fundamental property of importance sampling is its ability to efficiently approximate matrix products (and, by extension, vectors and higher-order tensors). This is our key lemma, which states that if we have sufficient access to two matrices, we can approximate their product by a product of matrices of smaller dimension:

Key lemma [DKM06] (informal version of Lemma 3.6). *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, Tan18, CLW18, CLLW19] indirectly used this lemma by using FKV, 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. By using FKV once, one can gain access to singular values and right singular vectors; by using it twice, one can gain access to a full SVD. Then, by applying functions to the approximate singular values, one can argue that the resulting expression is close to the desired expression. One could theoretically use this procedure to give a classical algorithm for singular value transformation, but we prove our main results without going through the full analysis of the low-rank approximation.

Instead, we use the key lemma twice to get an RUR decomposition of an even singular value transformation of the input (Theorem 3.1). Notice that, because we wish to run in time independent of dimension, the best we can do is to express the output based on the given input, as an RUR decomposition does. The proof of our main theorem is straightforward. Recall that, given $\mathrm{SQ}(A) \in \mathbb{C}^{m\times n}$, we wish to approximate $f(A^{\dagger}A)$ for f a function that, without loss of generality, satisfies $f(0) = 0$.

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

We then take $\bar{f}(CC^\dagger)$ to be the "U" of our RUR decomposition, finding it by naively computing the SVD of C in $\mathcal{O}(r^2c)$ time. The analysis is straightforward: we use that f and $\bar{f}$ are Lipschitz to argue that the error from approximating our matrix products propagates well (Lemma 5.5). We also use a variant of the key lemma (Lemma 3.8, from [KV17]) to give a spectral norm variant of the main theorem.

Though this analysis is much simpler than FKV, it gives improved results in our applications. Our approach has several advantages. The reduction first given by Tang to get an 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, FKV 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 FKV approximates the input by taking the same number of rows and columns.

The flexibility of singular value transformation also leads to easy generalization of results. For example, another important technical difference from previous work [CLW18, GLT18, CLLW19] is that our results do 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 only operate on large-enough singular values. That is, we implicitly take a low-rank approximation of the input before applying our singular value transformation.

General transformation results. We can bootstrap our algorithm for even SVT to get results for generic SVT (Theorem 3.3) and eigenvalue transformation (Theorem 3.4).

For generic SVT: consider a function $f : \mathbb{R} \to \mathbb{C}$ satisfying $f(0) = 0$ and a matrix $A \in \mathbb{C}^{m\times n}$. Given $\mathrm{SQ}(A)$ and $\mathrm{SQ}(A^\dagger)$, we give an algorithm to output a CUR decomposition approximating $f^{(\mathrm{SV})}(A)$. Our strategy is to apply our main result Theorem 3.1 to $g(A^\dagger A)$, for $g(x) := f(\sqrt{x})/\sqrt{x}$, and subsequently approximate matrix products with Lemma 3.6 to get an approximation of the form $A'R'^\dagger UR + g(0)A$:

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

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). The QML applications of even SVT in matrix inversion (Section 4.4) and Hamiltonian simulation in (Section 4.6) look similar to this, but we can use the additional structure in these problems to do this kind of approximation better.

As for eigenvalue transformation, consider a function $f : \mathbb{R} \to \mathbb{C}$ and a Hermitian matrix $H \in \mathbb{C}^{n\times n}$, given $\mathrm{SQ}(H)$. We wish to compute the eigenvalue transform $f(H)$. If f is even (so $f(x) = f(-x)$), then $f(H) = f(\sqrt{H^\dagger H})$, so the result follows from our main theorem for even SVT.

For non-even f, we use a different strategy, similar to the one used for quantum-inspired semidefinite programming [CLLW19]: first we find the eigenvectors and eigenvalues of H and then apply f to the eigenvalues. Let $\pi(x)$ be a (smoothened) step function that is a linear interpolation

between zero and one on $[0.5\varepsilon^2, \varepsilon^2]$. Then

$$\begin{aligned}
H &\approx \pi(HH^\dagger)H\pi(H^\dagger H) && \text{by definition of } \pi \\
&\approx R^\dagger\bar{\pi}(CC^\dagger)RHR^\dagger\bar{\pi}(CC^\dagger)R && \text{by main theorem (Theorem 3.1)} \\
&\approx R^\dagger\bar{\pi}(CC^\dagger)M\bar{\pi}(CC^\dagger)R && \text{by key lemma: } M \approx RHR^\dagger \\
&= R^\dagger(C_\sigma C_\sigma^+)^\dagger\bar{\pi}(CC^\dagger)M\bar{\pi}(CC^\dagger)C_\sigma C_\sigma^+ R, && \text{where } \sigma < \varepsilon
\end{aligned}$$

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. $\hat{U} := C_\sigma^+ R \in \mathbb{C}^{c\times n}$ is close to an isometry, which we argue by showing $(C_\sigma^+ R)(C_\sigma^+ R)^\dagger \approx I$. 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 H of size independent of n, we can directly compute its unitary eigendecomposition $UDU^\dagger$. This gives the approximate decomposition $H \approx (\hat{U}U)D(\hat{U}U)^\dagger$, with $\hat{U}U$ and D acting as approximate eigenvectors and eigenvalues of H, respectively. Some simple analysis shows that $f(H) \approx (\hat{U}U)f(D)(\hat{U}U)^\dagger$ in the desired sense. Therefore, our output approximation of $f(H)$ comes in the form of an RUR decomposition that can be rewritten in the form of an approximate eigendecomposition.

1.5 Related work

Our work bridges the fields of randomized algorithms and quantum algorithms for linear algebra. Thus, we interact with a diverse body of related work.

Randomized numerical linear algebra. Generally speaking, the techniques our framework uses belong to randomized linear algebra algorithms (see the surveys [Mah11, Woo14]). Our core primitive is importance sampling: see the survey by Kannan and Vempala [KV17] for algorithms using this type of sampling. 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].

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 2.10 and 3.10, but dequantized algorithms critically use low-rank assumptions on the input to "simulate" QML in a way that would not be possible were we only viewing such algorithms as large quantum circuits. The second is a paper by Rudi et al. [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 $\|H\|_F$, 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 et al. [ADBL19] implements and benchmarks quantum-inspired algorithms for regression and recommendation systems. This work makes various conclusions, and for example, suggests that the ε^2 scaling in the number of rows/columns taken in our recommendation systems algorithm is inherent. However, we are unsure of these results' implications for the broader question of whether QML algorithms can achieve practical speedups, for two reasons. First, our algorithms use a restricted model of computation in order to get a broad asymptotic result for generic applications of quantum machine learning. However, if we wish to compare QML to the best classical algorithm in practice, other sketching algorithms 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." For comparison, Arrazola et al. [ADBL19] claim that the running times of quantum-inspired algorithms are worse than directly computing the singular value decomposition for medium-sized matrices (e.g. $10^4 \times 10^4$). Second, the authors implement the quantum-inspired algorithms in a simple, non-optimized way in Python and then compare it to the well-optimized LAPACK library C implementation of singular value decomposition. These caveats make it difficult to draw definitive conclusions about the practicality of quantum-inspired algorithms as a whole from these experimental results.

Quantum machine learning. As mentioned in Section 1.3, our work has major implications for the landscape of quantum machine learning. In particular, our work suggests that the most promising way to get exponential speedups for algorithms fitting in the framework of quantum singular value transformation [GSLW19] is via algorithms that use sparse matrices as input (as opposed to those with input in QRAM), such as HHL [HHL09]. Such algorithms have other major caveats (mentioned by Aaronson [Aar15]) that make it difficult to find applications with the potential for practical super-polynomial speedups. Proposals for such applications include Gaussian process regression [ZFF19] and topological data analysis [LGZ16].

Related independent work. Independently from our work, Jethwani, Le Gall, and Singh simultaneously derived similar results [JLGS19]. 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) 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? Could this exponential advantage be exploited in a meaningful way?

(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)? Can we get even better runtimes by somehow avoiding SVD computation?

(c) 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)?

1.7 Organization

The paper proceeds as follows. Section 2.2 introduces the notion of (over)sampling and query access and some of its closure properties. Section 2.3 gives the fundamental idea of using sampling and query access to sketch matrices used for the main singular value transformation results in Section 3.1 and miscellaneous approximation results in Section 3.2. These results form the framework that is used to dequantize QSVT in Section 3.3 and recover all the quantum-inspired results in Section 4. 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. Proofs of the results in Section 3 are contained in Section 5.

2 Preliminaries

To begin with, we define notation to be used throughout this paper. For $n \in \mathbb{N}$, $[n] := \{1, \ldots, n\}$. 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 $\mathcal{O}(g \operatorname{poly}(\log g))$. 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 2.2) 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 $\|v\| := (\sum_{i=1}^n |v_i|^2)^{1/2}$). For a matrix $A \in \mathbb{C}^{m\times n}$, the *Frobenius norm* of A is $\|A\|_{\mathrm{F}} := \|\operatorname{vec}(A)\| = (\sum_{i=1}^m \sum_{j=1}^n |A(i,j)|^2)^{1/2}$ and the *spectral norm* of A is $\|A\| := \|A\|_{\mathrm{Op}} := \sup_{x\in\mathbb{C}^n, \|x\|=1} \|Ax\|$.

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

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

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 $A = \sum_{i=1}^{n} \lambda_i v_i v_i^\dagger$:

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

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

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 say that U is an isometry if $\|Ux\| = \|x\|$ for all x, or equivalently, if U is a subset of columns of a unitary. We define approximate isometry as follows[9]:

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 $\|U - V\| \leq 1 - \sqrt{1-\alpha}$, which is $\mathcal{O}(\alpha)$ for α bounded away from one.

2.2 Sampling and query access oracles

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

Definition 2.5 (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 $\mathbf{q}(v)$ (or $\mathbf{q}(A)$) denote the (time) cost of such a query.

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

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 write, for example, "$\mathrm{Q}(A) \in \mathbb{C}^{m \times n}$" instead of "$\mathrm{Q}(A)$ for $A \in \mathbb{C}^{m \times n}$". We will also sometimes abuse complexity notation like $\mathbf{q}$ to refer to known bounds on the complexity, instead of the complexity itself.

Definition 2.6 (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 $\mathcal{D}_v(i) := |v(i)|^2/\|v\|^2$;
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 $\mathbf{sq}(v) := \mathbf{q}(v) + \mathbf{s}(v) + \mathbf{n}(v)$.

We will refer to these samples as *importance samples from* v, though one can view them as measurements of the quantum state $|v\rangle := \frac{1}{\|v\|}\sum v_i|i\rangle$ 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.[10]

Definition 2.7. For $v \in \mathbb{C}^n$, $p \in \mathbb{R}^n_{\geq 0}$ is a *ϕ-oversampled importance sampling distribution* of v (for $\phi \geq 1$) if $\sum_{i=1}^n p(i) = 1$ and, for all $i \in [n]$, $p(i) \geq \mathcal{D}_v(i)/\phi = \frac{|v(i)|^2}{\phi\|v\|^2}$.

If p is a ϕ-oversampled importance sampling distribution of v, any given output $i \in [n]$ is no more than ϕ-times rarer in p compared to the desired distribution $\mathcal{D}_v$. As a result, intuitively, estimators that use $\mathcal{D}_v$ can also use p, with a factor ϕ increase in the number of samples necessary. For example, we can convert a sample from p to a sample from $\mathcal{D}_v$ with probability $1/\phi$ with rejection sampling: sample an i distributed as p, then accept the sample with probability $(\mathcal{D}_v(i)/p(i))/\phi$.

Definition 2.8 (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 $p(i) := \mathcal{D}_{\tilde{v}}(i)$, $\mathbf{s}_\phi(v) := \mathbf{s}(\tilde{v})$, $\mathbf{q}_\phi(v) := \mathbf{q}(\tilde{v})$, $\mathbf{n}_\phi(v) := \mathbf{n}(\tilde{v})$, and $\mathbf{sq}_\phi(v) := \mathbf{s}_\phi(v) + \mathbf{q}_\phi(v) + \mathbf{q}(v) + \mathbf{n}_\phi(v)$.

$\mathrm{SQ}_1(v)$ is the same as $\mathrm{SQ}(v)$, if we take $\tilde{v} = v$. Note that our algorithms 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). Finally, note that oversampling access implies an approximate version of the usual sampling access:

[10] 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.

Lemma 2.9. *Suppose we are given* $\mathrm{SQ}_\phi(v)$ *and some* $\delta \in (0,1]$. *Denote* $\widetilde{\mathbf{sq}}(v) := \phi\,\mathbf{sq}_\phi(v)\log\frac{1}{\delta}$. *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.*

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, $\widetilde{\mathbf{sq}}(v)$ 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, and then pay $\widetilde{\mathbf{sq}}(v)$ 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].

Lemma 2.10 (Linear combinations, Proposition 4.3 of [Tan19])**.** *Given* $\mathrm{SQ}_{\varphi_1}(v_1),\ldots,\mathrm{SQ}_{\varphi_k}(v_k) \in \mathbb{C}^n$ *and* $\lambda_1,\ldots,\lambda_k \in \mathbb{C}$, *we have* $\mathrm{SQ}_\phi(\sum\lambda_i v_i)$ *for* $\phi = k\frac{\sum\varphi_i\|\lambda_i v_i\|^2}{\|\sum\lambda_i v_i\|^2}$ *and* $\mathbf{sq}_\phi(\sum\lambda_i v_i) := \max_{i\in[k]}\mathbf{s}_{\varphi_i}(v_i) + \sum_{i=1}^k \mathbf{q}(v_i)$ *(after paying* $\mathcal{O}\Big(\sum_{i=1}^k \mathbf{n}_{\varphi_i}(v_i)\Big)$ *one-time pre-processing cost to query for norms).*

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 2.9, where we pay an additional "cancellation constant" factor of $k\frac{\sum\|\lambda_i v_i\|^2}{\|\sum\lambda_i v_i\|^2}$. This factor is only large when the linear combination has significantly smaller norm than the components v_i 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]. Assuming this cancellation is not too large, the resulting overhead won't affect 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 2.11 (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 $(a(i) := \|A(i,\cdot)\|)$.

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 $\mathbf{s}_\phi(A) := \max(\mathbf{s}(\tilde{A}(i,\cdot)),\mathbf{s}(\tilde{a}))$, $\mathbf{q}_\phi(A) := \max(\mathbf{q}(\tilde{A}(i,\cdot)),\mathbf{q}(\tilde{a}))$, $\mathbf{q}(A) := \max(\mathbf{q}(A(i,\cdot)))$, and $\mathbf{n}_\phi(A) := \mathbf{n}(\tilde{a})$ respectively. We also use the notation $\mathbf{sq}_\phi(A) := \max(\mathbf{s}_\phi(A),\mathbf{q}_\phi(A),\mathbf{q}(A),\mathbf{n}_\phi(A))$ We omit subscripts if $\phi = 1$.

Observe that $\mathrm{SQ}_\phi(A)$ implies $\mathrm{SQ}_\phi(\mathrm{vec}(A))$: 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, $\mathrm{SQ}_\phi(A)$ can be thought of as $\mathrm{SQ}_\phi(\mathrm{vec}(A))$, with the additional guarantees that we can compute marginals $\sum_{j=1}^n \mathcal{D}_{\mathrm{vec}(\tilde{A})}(i,j)$ and can sample from the resulting conditional distributions $\mathcal{D}_{\mathrm{vec}(\tilde{A})}(i,j)/\sum_{j=1}^n \mathcal{D}_{\mathrm{vec}(\tilde{A})}(i,j)$.

Lemma 2.12. *Given vectors $u \in \mathbb{C}^m, v \in \mathbb{C}^n$ with $\mathrm{SQ}_{\varphi_u}(u), \mathrm{SQ}_{\varphi_v}(v)$ access we have $\mathrm{SQ}_\phi(A)$ for their outer product $A := uv^\dagger$ 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 $\mathbf{n}_\phi(A) = \mathbf{n}_{\varphi_u}(u) + \mathbf{n}_{\varphi_v}(v)$,*

The above shows that Definition 2.11 is a faithful generalization of Definition 2.8, i.e., for a vector v we get back essentially the same definition if we think about it as a row / column matrix. Using the same ideas as in Lemma 2.10, we can extend sampling and query access of input matrices to linear combinations of those matrices.

Lemma 2.13. *Given $\mathrm{SQ}_{\varphi^{(1)}}(A^{(1)}), \ldots, \mathrm{SQ}_{\varphi^{(\tau)}}(A^{(\tau)}) \in \mathbb{C}^{m\times n}$, we have $\mathrm{SQ}_\phi(A) \in \mathbb{C}^{m\times n}$ for $A := \sum_{t=1}^\tau \lambda_t A^{(t)}$ with $\phi = \frac{\tau \sum_{t=1}^\tau \varphi^{(t)} \|\lambda_t A^{(t)}\|_{\mathrm{F}}^2}{\|A\|_{\mathrm{F}}^2}$ 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 $\mathcal{O}\left(\sum_{t=1}^\tau \mathbf{n}_{\varphi^{(t)}}(A^{(t)})\right)$ one-time pre-processing cost).*

Remark 2.14. From the lemmas we've introduced, we can already get oversampling and query access to some modest expressions. Suppose we have sampling and query access to $A^{(1)}, \ldots, A^{(\tau)} \in \mathbb{C}^{m\times n}$. Then

$$(A^{(1)})^\dagger A^{(1)} + \cdots + (A^{(\tau)})^\dagger A^{(\tau)} = A^\dagger A, \qquad \text{where } A := \begin{bmatrix} A^{(1)} \\ \vdots \\ A^{(\tau)} \end{bmatrix}.$$

One can verify that we can simulate $\mathrm{SQ}(A)$, only paying at most a factor of τ more for queries, giving a version of Lemma 2.12 for these "even" expressions $(A^{(t)})^\dagger A^{(t)}$. Moreover, if we are willing to pay factors of m as well, we can write

$$A^\dagger A = \sum_{i=1}^{m\tau} A(i,\cdot)^\dagger A(i,\cdot),$$

and get $\mathrm{SQ}_\phi(A^\dagger A)$ from $\mathrm{SQ}(A)$, Lemma 2.12, and Lemma 2.10. We can generalize this to RUR decompositions, a decomposition occurring 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 2.12 and Lemma 2.10, we have $\mathrm{SQ}_\phi(R^\dagger U R)$.

Quantum machine learning algorithms and their corresponding quantum-inspired algorithms have the potential to achieve exponential speedups when their state preparation procedures run in time $\mathrm{polylog}(n)$. So, the most interesting regime for us is when our sampling and query oracles take poly-logarithmic time. This assumption can be satisfied in various ways.

Remark 2.15. Below, we list various 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, their runtimes may be missing log factors that should be counted for a fair comparison between classical and quantum.

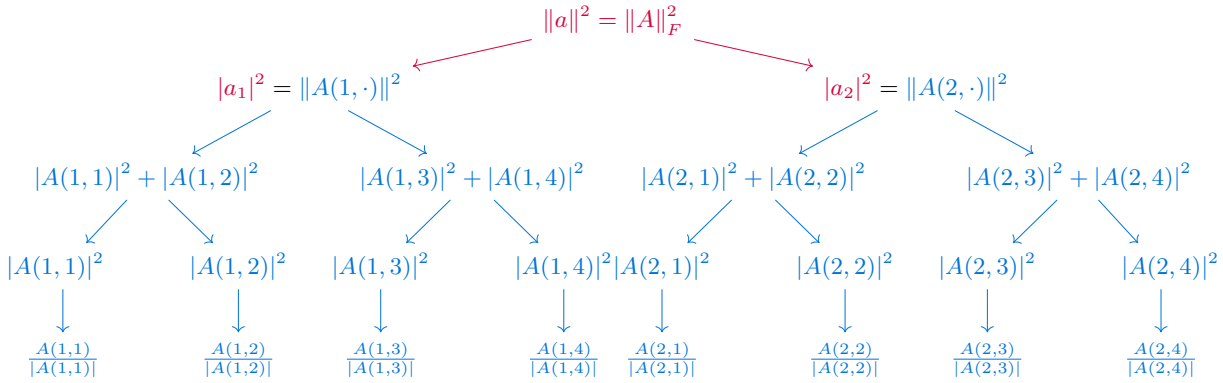

Figure 1: Dynamic data structure for $A \in \mathbb{C}^{2\times 4}$. We compose the data structure for a with the data structure for A's rows.

(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.

More precisely, the pre-processing time is linear in the number of non-zero entries of the input vector (resp. matrix), which we denote $\mathrm{nnz}(v)$ (resp. $\mathrm{nnz}(A)$). A direct consequence of this observation is that the quantum-inspired setting is more restrictive than the typical randomized numerical linear algebra algorithm setting. 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 QRAM data structure [Pra14, KP20, GLM08, WZP18, RSW+19, CGJ19], arguing that, with the right type of quantum access, this data structure allows for circuits preparing input states with linear gate count but polylog depth. Hardware might be able to parallelize these circuits enough so that they run in 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. 1) 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 can be useful for applications of QML where data can accumulate over time [KP17].

(c) (Integrability assumption) For $v \in \mathbb{C}^n$, suppose we can compute entries 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, a quantum state corresponding to v 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 an upper bound. 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 $\tilde{a}(i) := c\sqrt{s}$, 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).

If A is not much smaller than we expect, ϕ is 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 $\phi \leq (c/c')^2$. This kind of sparsity assumption is used in some QML and Hamiltonian simulation problems [HHL09].

(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.

2.3 Matrix sketches

Definition 2.16. For a distribution $p \in \mathbb{R}^m$, we say that a matrix $S \in \mathbb{R}^{s\times m}$ is *sampling according to* p if each row of S is independently chosen to be $e_i/\sqrt{sp(i)}$ with probability p_i.

We call such S's *importance sampling* sketches when p comes from $\mathrm{SQ}(A)$ for some $A \in \mathbb{C}^{m\times n}$, and we call them ϕ-oversampled importance sampling sketches if p comes from $\mathrm{SQ}_\phi(A)$ (or, more generally, from a ϕ-oversampled importance sampling distribution of a).

One should think of S as sketching A down to SA. First, some basic observations.

Remark 2.17. Let $S \in \mathbb{C}^{s\times m}$ be a ϕ-oversampled importance sampling sketch of $A \in \mathbb{C}^{m\times n}$. Then we can bound the row norms and Frobenius norm of SA. 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 $\|[SA](i,\cdot)\| = \frac{1}{\sqrt{s\cdot p(i)}}\|A(s_i,\cdot)\| \leq \sqrt{\phi/s}\|A\|_{\mathrm{F}}$ and, consequently, $\|SA\|_{\mathrm{F}} \leq \sqrt{\phi}\|A\|_{\mathrm{F}}$. When $\phi = 1$, these inequalities are equalities.

This remark describes unconditional bounds on norms, but as s increases, the norms of SA approach the norms of A. So, for example, $\|SA\|_{\mathrm{F}} = \Theta(\|A\|_{\mathrm{F}})$ and $\|SA\| = \Theta(\|A\|)$ with constant probability for $s = \Omega(\frac{1}{\phi^2})$ (Lemma 5.1) and $s = \tilde{\Omega}(\phi^2\|A\|_F^2/\|A\|^2)$ (Lemma 3.8), respectively. We will use the spectral norm bound extensively, but since we will usually be in the $\phi = \mathcal{O}(1)$ regime, the corresponding Frobenius norm bound won't see as much use.

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 $\mathrm{SQ}_\phi(A)$, 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, $(i_1,c_1),\ldots,(i_s,c_s)$.

The core technique of our quantum-inspired algorithms is to use these kinds of sketches to approximate matrix expressions. Further, we can chain them with a simple observation.

Lemma 2.18. *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* $r\geq 2\varphi^2\ln\frac{2}{\delta}$*, then with probability* $\geq 1-\delta$*, we have* $\mathrm{SQ}_\phi(SA)$ *and* $\mathrm{SQ}_\phi((SA)^\dagger)$ *for some* ϕ *satisfying* $\phi\leq 2\varphi$*. 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* $\mathbf{q}(SA)=\mathbf{q}(A)$*,* $\mathbf{s}_\phi(SA)=\mathbf{s}_\phi(A)$*,* $\mathbf{q}_\phi(SA)=\mathbf{q}_\phi(A)$*, and* $\mathbf{n}_\phi(SA)=\mathcal{O}(1)$*, after* $\mathcal{O}(\mathbf{n}_\phi(A))$ *pre-processing cost. The runtimes for* $\mathrm{SQ}_\phi((SA)^\dagger)$ *are* $\mathbf{q}((SA)^\dagger)=\mathbf{q}(A)$*,* $\mathbf{s}_\phi((SA)^\dagger)=\mathbf{s}_\phi(A)+r\,\mathbf{q}_\phi(A)$*,* $\mathbf{q}_\phi((SA)^\dagger)=r\,\mathbf{q}_\phi(A)$*, and* $\mathbf{n}_\phi((SA)^\dagger)=\mathbf{n}_\phi(A)$*.*

So, if we have a matrix A, along with $\mathrm{SQ}(A)$, we can find a sketch S of A, and then use the resulting $\mathrm{SQ}((SA)^\dagger)$ to find a sketch $T^\dagger$ of $(SA)^\dagger$. The same argument works for $\mathrm{SQ}_\varphi(A)$, just with a mild lower bound on the size of the sketches and δ failure probability. This fully sketched down version of A, SAT, will be used extensively, since it is small enough that we can compute functions of it in time independent of dimension. When we refer to sketching A down to SAT, we use the above observation for sampling T.

3 Main results and technical tools

3.1 Singular value transformation

First, our main result: given $\mathrm{SQ}_\phi(A)$ and a smooth function f, we can approximate $f(A^\dagger A)$ by the decomposition $R^\dagger ZR+f(0)I$.

Theorem 3.1 (Even singular value transformation)**.** *Let* $A\in\mathbb{C}^{m\times n}$ *and* $f\colon\mathbb{R}^+\to\mathbb{C}$ *be such that,* f *and* $\bar{f}(x):=(f(x)-f(0))/x$ *are* L*-Lipschitz and* $\bar{L}$*-Lipschitz, respectively, on* $\cup_{i=1}^n[\sigma_i^2-d,\sigma_i^2+d]$ *for some* $d>0$*. Take parameters* ε *and* δ *such that* $0<\varepsilon\lesssim L\|A\|_*^2$ *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* $\mathrm{SQ}_{\leq 2\phi}((SA)^\dagger)$ *(which we have by Lemma 2.18). Then, for* $R:=SA$ *and* $C:=SAT$*, we can achieve the bound*

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

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

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

By our discussion in Section 2.3, finding the sketches S and T for Theorem 3.1 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 $\mathcal{O}\big(rc\,\mathbf{sq}_\phi(A)\big)$.

We give intuition for this theorem in Section 1.2, so here we only discuss some technical remarks. The assumption that $\varepsilon \lesssim L\|A\|_*^2$ is for non-degeneracy: if $\varepsilon \geq L\|A\|^2$, then the naive approximation $f(0)I$ of $f(A^\dagger A)$ would suffice, since $\|f(0)I - f(A^\dagger A)\| \leq L\|A\|^2 \leq \varepsilon$ as desired. The bounds on r, c relative to d won't come into play often, since we can often design our singular value transforms such that they are smooth everywhere. For example, if our desired transform f becomes non-smooth outside the relevant interval $[0, \|A\|^2]$, we can apply Theorem 3.1 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 don't need to worry about the d parameter. Finally, we note that no additional log terms are necessary (i.e., $\tilde{\Omega}$ becomes Ω) when Frobenius norm is used.

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 4.

Lemma 3.2 (Norm bounds for even singular value transformation)**.** *Suppose the assumptions from Theorem 3.1 hold and the event in Eq. (3) occurs (that is, $R^\dagger \bar{f}(CC^\dagger)R \approx f(A^\dagger A) - f(0)I$). Then we can additionally assume that the following bounds also hold:*

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

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

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

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):

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

While we will primarily use the simple and fast primitive of even singular value transformation to recover "dequantized QML"-type results, we can also get generic singular value transformation and eigenvalue transformation results by bootstrapping Theorem 3.1.

Theorem 3.3 (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 $f : \mathbb{R} \to \mathbb{C}$ be a function such that $f(0) = 0$, $g(x) := f(\sqrt{x})/\sqrt{x}$ is L-Lipschitz, and $\bar{g}(x) := g(x)/x$ is $\bar{L}$-Lipschitz. Then, for $0 < \varepsilon \lesssim L\|A\|^3$, 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 $r = \tilde{\Omega}\Big(\phi^2 L^2 \|A\|^2 \|A\|_{\mathrm{F}}^4 \frac{1}{\varepsilon^2} \log\frac{1}{\delta}\Big)$ and $c = \tilde{\Omega}\Big(\phi^2 L^2 \|A\|^4 \|A\|_{\mathrm{F}}^2 \frac{1}{\varepsilon^2} \log\frac{1}{\delta}\Big)$. Finding S, M, and T

takes time

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

If we only wish to assume $\mathrm{SQ}_\phi(A)$, we can do so by using Lemma 3.5 instead of Lemma 3.6 in our proof, paying an additional factor of $\frac{1}{\delta}$.

Assuming that $\mathbf{sq}_\phi(A), \mathbf{sq}_\phi(A^\dagger) = \mathcal{O}(1)$, this runtime is dominated by the last line. For intuition, if A is strictly low-rank, with minimum singular value σ, or essentially equivalently, if $f(x) \equiv 0$ for $x \le \sigma$, then $L \le \ell/\sigma^2$ and $\bar{L} = \ell/\sigma^4$ for ℓ the Lipschitz constant of f. Estimating $f(A)$ takes time

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

Importantly, when $\varepsilon = \mathcal{O}(\ell\|A\|)$ (that is, if we want additive 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: for spectral norm bounds, one can always assume that A is 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.

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

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. (8), with $\sigma = \varepsilon/\ell$.

We do not use this theorem in our applications. Sometimes we implicitly use a similar strategy (e.g. in Section 4.4), but because we apply our matrix to a vector $(f(A^\dagger)b)$ we can use Lemma 3.10 instead of Lemma 3.6 when approximating. So, we get effectively the same result without needing a $\mathrm{SQ}(A^\dagger)$ oracle[11].

Theorem 3.4 (Eigenvalue transformation)**.** *Suppose we are given a Hermitian* $\mathrm{SQ}_\phi(A) \in \mathbb{C}^{n\times n}$*, 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* $\varepsilon \in (0, L\|A\|]$*. Then we can output matrices* $S \in \mathbb{C}^{s\times n}$*,* $N \in \mathbb{C}^{s'\times s}$*, and* $D \in \mathbb{C}^{s'\times s'}$*, with* $s = \widetilde{\mathcal{O}}\Big(\phi^2\|A\|^4\|A\|_{\mathrm{F}}^2\frac{L^6}{\varepsilon^6}\log\frac{1}{\delta}\Big)$ *and* $s' = \mathcal{O}\big(\|A\|_{\mathrm{F}}^2L^2/\varepsilon^2\big)$*, such that*

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

[11] While this assumption is not much more than $\mathrm{SQ}(A)$, a weaker input model is better since we want our algorithm to run whenever a corresponding QML algorithm can.

in time

$$\tilde{\mathcal{O}}\Big((L^{10}\varepsilon^{-10}\|A\|^8\|A\|_{\mathrm{F}}^2\phi^2\log\frac{1}{\delta}+L^6\varepsilon^{-6}\|A\|_{\mathrm{F}}^6\phi^3\log\frac{1}{\delta})(\mathbf{s}_\phi(A)+\mathbf{q}_\phi(A))$$
$$+(L^{16}\varepsilon^{-16}\|A\|^{12}\|A\|_{\mathrm{F}}^4\phi^4\log^2\frac{1}{\delta}+L^{18}\varepsilon^{-18}\|A\|^8\|A\|_{\mathrm{F}}^{10}\phi^7\log^3\frac{1}{\delta})\,\mathbf{q}(A)$$
$$+L^{22}\varepsilon^{-22}\|A\|^{16}\|A\|_{\mathrm{F}}^6\phi^6\log^3\frac{1}{\delta}\Big).$$

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

Under the reasonable assumptions[12] that $\mathbf{sq}(A)$ is small (like $\mathcal{O}(1)$) and $\varepsilon \le L\|A\|\frac{\|A\|}{\|A\|_{\mathrm{F}}}$, the complexity of this theorem is $\widetilde{\mathcal{O}}(L^{22}\varepsilon^{-22}\|A\|^{16}\|A\|_{\mathrm{F}}^6\log^3\frac{1}{\delta})$.

3.2 Technical tools

To prove the main theorems as well as the results in our applications, we need some technical tools that were developed or derived from previous results. We summarize them in this subsection.

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 3.5 (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 $S\in\mathbb{R}^{s\times m}$ to be sampled according to $p\in\mathbb{R}^m$ a ϕ-oversampled importance sampling distribution from X or Y. Then,*

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

If we have sampling and query access to both matrices in the product, the failure probability falls exponentially. This is the key lemma we use most in Section 4.

Lemma 3.6 (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 $S\in\mathbb{R}^{s\times m}$ to be sampled according to $r:=\frac{p+q}{2}$, where $p,q\in\mathbb{R}^m$ are ϕ_1,ϕ_2-oversampled importance sampling distributions from X,Y, respectively. Then S is a $2\phi_1,2\phi_2$-oversampled importance sampling sketch of X,Y, respectively. Further,*

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

Remark 3.7. Lemma 3.6 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 $\phi\le\phi_1\phi_2\frac{\|X\|_{\mathrm{F}}^2\|Y\|_{\mathrm{F}}^2}{\|M\|_{\mathrm{F}}^2}$. This is an approximate closure property for oversampling and query access under matrix products.

[12] 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 will appear later on in Section 4.

Given the above types of accesses, we can compute the sketch S necessary for Lemma 3.6 by taking $p = \mathcal{D}_{\tilde{x}}$ and $q = \mathcal{D}_{\tilde{y}}$), thereby finding a desired $M := X^\dagger S^\dagger S Y$. We can compute entries of M with only s 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 $|\tilde{M}(i,j)|^2 := s\sum_{\ell=1}^{s} |[S\tilde{X}](\ell,i)^\dagger [S\tilde{Y}](\ell,j)|^2$; showing that we have $\mathrm{SQ}(M)$ follows from the proofs of Lemmas 2.12 and 2.13, 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 $k_1, \ldots, k_s$, we have

$$\|\tilde{M}\|_{\mathrm{F}}^2 = s\sum_{\ell=1}^{s} \|[S\tilde{X}](\ell,\cdot)\|^2 \|[S\tilde{Y}](\ell,\cdot)\|^2 = s\sum_{\ell=1}^{s} \frac{4\|\tilde{X}(k_\ell,\cdot)\|^2\|\tilde{Y}(k_\ell,\cdot)\|^2}{s^2(\frac{\|\tilde{X}(k_\ell,\cdot)\|^2}{\|\tilde{X}\|_{\mathrm{F}}^2} + \frac{\|\tilde{Y}(k_\ell,\cdot)\|^2}{\|\tilde{Y}\|_{\mathrm{F}}^2})^2}$$
$$\leq \sum_{\ell=1}^{s} \frac{4\|\tilde{X}(k_\ell,\cdot)\|^2\|\tilde{Y}(k_\ell,\cdot)\|^2}{s(2\frac{\|\tilde{X}(k_\ell,\cdot)\|\|\tilde{Y}(k_\ell,\cdot)\|}{\|\tilde{X}\|_{\mathrm{F}}\|\tilde{Y}\|_{\mathrm{F}}})^2} = \|\tilde{X}\|_{\mathrm{F}}^2\|\tilde{Y}\|_{\mathrm{F}}^2. = \phi_1\phi_2\|X\|_{\mathrm{F}}^2\|Y\|_{\mathrm{F}}^2.$$

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

Lemma 3.8 (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 $S \in \mathbb{R}^{s\times n}$ a ϕ-oversampled importance sampling sketch of A. Then*

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

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 3.9 (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 $r = \tilde{\Omega}(\frac{\phi^2}{\varepsilon^2}\log\frac{1}{\delta})$ and $c = \tilde{\Omega}(\frac{\phi^2}{\varepsilon^2}\log\frac{1}{\delta})$. Then, if σ_i and $\hat{\sigma}_i$ are the singular values of A and SAT, respectively (where $\hat{\sigma}_i = 0$ for $i > \min(r,c)$), with probability $\geq 1-\delta$,*

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

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

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.

Lemma 3.10 (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 3.11. Lemma 3.10 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

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

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 3.10 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

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

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 3.12. *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 $\leq \phi$-oversampled importance sampling distribution of $(SA)^\dagger$, let $R := SA$ and $C := SAT$. 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 $\geq 1-\delta$, $((C_k)^+R)^\dagger$ 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 $k' \leq \min(k, \mathrm{rank}(A))$).*

One can 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 don't need it in our proofs, so we don't include this computation.

3.3 Dequantizing QSVT

We can use the above results to dequantize the quantum singular value transformation described by Gilyén et al. [GSLW19] for close-to-low-rank input.

Definition 3.13. 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 3.14. *Suppose we are given a matrix $A \in \mathbb{C}^{m\times n}$ satisfying $\|A\|_{\mathrm{F}} = 1$ via the oracles for $\mathrm{SQ}(A)$ with $\mathbf{sq}(A) = \mathcal{O}(\log(mn))$, 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 $|p(x)| \leq 1$ for all $x \in [-1, 1]$.*

Then with probability $\geq 1 - \delta$, for ε a sufficiently small constant, we can get $\mathrm{SQ}_\phi(v) \in \mathbb{C}^n$ such that $\|v - p^{(\mathrm{QV})}(A)b\| \leq \varepsilon\|p^{(\mathrm{QV})}(A)b\|$ 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 2.15, 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 isn't 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 3.14 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 3.15. *Consider $p(x)$ a degree-d polynomial of parity-d such that $|p(x)| \le 1$ 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 $\max_{x\in[0,1]} |q(x)| \le 1$, $\max_{x\in[-1,1]} |q'(x)| \lesssim d^2$, $\max_{x\in[-1,1]} |\bar{q}(x)| \lesssim d^2$, 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 3.14. Consider the even case: take $p(x) = q(x^2)$ for q a degree-$d/2$ polynomial, so $p^{(\mathrm{QV})}(A) = q(A^\dagger A)$, so we have the correct form to apply Theorem 3.1. q is uncontrolled outside of $[-1,1]$, so we instead apply the singular value transformation which is constant outside of $[-1,1]$.

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

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 3.15, f and $\bar{f}$ are Lipschitz with $L = \mathcal{O}(d^2), \bar{L} = \mathcal{O}(d^4)$. So, by Theorem 3.1, 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 3.6, 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 $\mathcal{O}\left(\|A\|_{\mathrm{F}}^2\|b\|^2 \frac{d^2}{\varepsilon^2} r\log(mn)\log\frac{1}{\delta}\right)$ time, using that $\|R\|_{\mathrm{F}} \lesssim \|A\|_{\mathrm{F}}$ (Eq. (5)) and $\mathbf{sq}(R^\dagger) = \mathcal{O}(r\,\mathbf{sq}(A))$ (Lemma 2.18). This suffices to maintain the error bound because (using Eqs. (6) and (7) and Lemma 3.15),

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

As a consequence, $v := R^\dagger \bar{f}(CC^\dagger)u + f(0)b$ satisfies $\|v - p^{(\mathrm{QV})}(A)b\| \le \varepsilon$. Via Lemma 2.10, we can get $\mathrm{SQ}_\phi(v)$ with

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

In the last step, we use that $\varepsilon \lesssim \|p^{(\mathrm{QV})}(A)b\|$; if we don't 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 $\mathcal{O}\big(r^2c\big)$ time, and estimating $R^\dagger b$ in $\mathcal{O}\Big(r\frac{d^2}{\varepsilon^2}\log(mn)\log\frac{1}{\delta}\Big)$ 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 3.3 here, but we will continue to use Theorem 3.1 here. Similarly to the even case, we take $g(x)$ to be $q(x) - q(0)$ 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 3.15, 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

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

We now use the approximating matrix product lemmas Lemmas 3.5 and 3.6 three times.

1. We use Lemma 3.5 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 $\mathcal{O}\big(\|A\|_{\mathrm{F}}^2\|R\|_{\mathrm{F}}^2d^4\varepsilon^{-2}\delta^{-1}\big) = \mathcal{O}\big(d^4\varepsilon^{-2}\delta^{-1}\big)$ samples, with each sample costing $\mathcal{O}(r\log(mn))$ time (by Lemma 2.18).

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

2. We use Lemma 3.6 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 3.5, 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 2.10, 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 3.16. 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 doesn't give an exponential speedup when A is in QRAM, but does when $A^{\dagger}$ is in QRAM. We sketch an argument why this isn't possible 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.

4 Applying the framework to dequantizing QML algorithms

Now, with our framework, we can recover previous dequantization results: recommendation systems (Section 4.1), supervised clustering (Section 4.2), principal component analysis (Section 4.3), low-rank matrix inversion (Section 4.4), support-vector machines (Section 4.5), and low-rank semidefinite programs (Section 4.7). We also propose new quantum-inspired algorithm for other applications, including Hamiltonian simulation (Section 4.6) and discriminant analysis (Section 4.8). 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 don't know the spectral norm, we can run Lemma 3.9 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.

4.1 Recommendation systems

Tang's dequantization [Tan19] of Kerenidis and Prakash's recommendation system [KP17] is the first dequantization in this line of work, leveraging techniques from randomized linear algebra.

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 4.1. 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 $A_r := \sum_{i=1}^{r} \sigma_i U(\cdot, i) V(\cdot, i)^\dagger$, which is the rank-$r$ matrix closest to A in spectral and Frobenius norm. Using singular value transformation, we define an analogous notion thresholding singular values instead of rank.

Definition 4.2 ($A_{\sigma,\eta}$)**.** We define $A_{\sigma,\eta}$ as a singular value transform of A satisfying:

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

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. We do this by noting that it suffices to find a good (smoothened) projector onto the top eigenvectors of $A^\dagger A$, so we can apply our even SVT result.

Corollary 4.3. *Suppose* $0 < \varepsilon \lesssim \|A\|/\|A\|_{\mathrm{F}}$ *and* $\eta \leq 0.99$*. A classical algorithm can solve Problem 4.1 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.

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

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

We will apply Theorem 3.1 with error parameter ε to get matrices R, C such that $AR^\dagger \bar{t}(CC^\dagger)R$ satisfies

$$\|A_{\sigma,1/6} - AR^\dagger \bar{t}(CC^\dagger)R\|_{\mathrm{F}} \leq \|A\|_{\mathrm{F}} \|t(A^\dagger A) - R^\dagger \bar{t}(CC^\dagger)R\| \leq \varepsilon \|A\|_{\mathrm{F}} \tag{9}$$

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 3.6, and if we were okay with paying factors of $\frac{1}{\delta}$, we could do this via Lemma 3.5. Here, we will instead implicitly define an approximation by approximating each row $[AR^\dagger](i,\cdot) = A(i,\cdot)R^\dagger$ via Lemma 3.6, 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[13]. Here, $A'(i,\cdot) := A(i,\cdot)S^\dagger S$ is our r'-sparse approximation, giving that

$$\|AR^\dagger - A'R^\dagger\|_{\mathrm{F}} = \sqrt{\textstyle\sum_{i=1}^m \|[AR^\dagger](i,\cdot) - [A'R^\dagger](i,\cdot)\|^2} \le \sqrt{\textstyle\sum_{i=1}^m \varepsilon^2\|A(i,\cdot)\|^2\sigma^2} = \varepsilon\sigma\|A\|_{\mathrm{F}}. \quad (10)$$

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

$$\begin{aligned}
\|\hat{A} - A_{\sigma,\eta}\|_{\mathrm{F}} &\le \|(A'R^\dagger - AR^\dagger)\bar{t}(CC^\dagger)R\|_{\mathrm{F}} + \|AR^\dagger\bar{t}(CC^\dagger)R - A_{\sigma,\eta}\|_{\mathrm{F}} && \text{by triangle inequality}\\
&\le \|A'R^\dagger - AR^\dagger\|_{\mathrm{F}}\left\|\sqrt{\bar{t}(CC^\dagger)}\right\|\left\|\sqrt{\bar{t}(CC^\dagger)}R\right\| + \varepsilon\|A\|_{\mathrm{F}} && \text{by Eq. (9)}\\
&\le \varepsilon\sigma\|A\|_{\mathrm{F}}\sigma^{-1}\sqrt{1+\varepsilon} + \varepsilon\|A\|_{\mathrm{F}} \lesssim \varepsilon\|A\|_{\mathrm{F}}. && \text{by Lemma 3.2 and Eq. (10)}
\end{aligned}$$

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 Lemma 2.10 and Lemma 2.9 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

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

[13]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 3.2, 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 sampling is exact, the only error in total variation distance is the probability of failure. □

Remark 4.4. 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 4.1), or more generally, any product of $\hat{A}$ with a vector, in time independent of dimension.

4.2 Supervised clustering

The 2013 paper of Lloyd et al. [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 [Tan18] (the second is an application of the quantum adiabatic algorithm with no proven runtime guarantees). Since the dequantization is very simple, only using the inner product protocol, 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 $q_1, \ldots, q_{n-1} \in \mathbb{R}^d$, $\|p - \frac{1}{n-1}(q_1 + \cdots + q_{n-1})\|^2$. This reduces to computing $\|wM\|$ for

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

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

Problem 4.5. 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 4.6. *There is a classical algorithm to solve Problem 4.5 in* $\mathcal{O}\big(\|M\|_{\mathrm{F}}^4\|w\|^4\frac{1}{\varepsilon^2}\log\frac{1}{\delta}\big)$ *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 also depends on the factor $\|w\|^2$ (the quantum-inspired algorithm is only quadratically slower).

Proof. Recall our notation for the vector of row norms $m := [\|M(1,\cdot)\|, \ldots, \|M(n,\cdot)\|]$ coming from Definition 2.11. 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 3.10 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. □

4.3 Principal component analysis

Principal component analysis (PCA) is an important data analysis tool, first proposed to be feasible via quantum computation by Lloyd et al. [LMR14]. Given $\widetilde{\mathcal{O}}\big(1/\varepsilon^3\big)$ copies of states with density matrix $\rho = X^\dagger X$, the qPCA 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 et al. 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 et al. [LMR14] suggested for quantum PCA.

Problem 4.7 (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. Note that, to robustly avoid degeneracy conditions, our runtime must depend on parameters for condition number and spectral gap:

$$K := \mathrm{Tr}(X^\dagger X)/\lambda_k \geq k \quad \text{and} \quad \eta := \min_{i\in[k]}|\lambda_i - \lambda_{i+1}|/\|X\|^2. \tag{11}$$

We also denote $\kappa := \|X\|^2/\lambda_k$. Dependence on K and η are necessary to reduce Problem 4.7 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, so learning v_k is also impossible. A straightforward coupon collector argument (given e.g. by Tang [Tan18]) shows that Problem 4.7 can be solved by a quantum algorithm performing spectral sampling[14], with runtime depending polynomially on K and $\frac{1}{\eta}$. We omit this argument for brevity. Classically, we can solve this PCA problem with quantum-inspired techniques, as first noted in [Tan18].

Corollary 4.8. *For $0 < \varepsilon \lesssim \eta\|X\|^2/\|X\|_{\mathrm{F}}^2$, we can solve Problem 4.7 in $\widetilde{\mathcal{O}}\left(\frac{\|X\|_{\mathrm{F}}^6}{\lambda_k^2\|X\|^2}\eta^{-6}\varepsilon^{-6}\log^3\frac{k}{\delta}\right)$ time to get $\mathrm{SQ}_\phi(\hat{v}_i)$ where $\widetilde{\mathbf{sq}}(\hat{v}_i) = \widetilde{\mathcal{O}}\left(\frac{\|X\|_{\mathrm{F}}^4}{\lambda_i\|X\|^2}\eta^{-2}\varepsilon^{-2}\log^2\frac{1}{\delta}\right)$.*

Note that $K > \kappa$, so these runtimes make sense. We briefly remark that the normalization of our η is not done in the typical way relative to qPCA procedures, which take gap relative to Frobenius norm (that is, using $\bar{\eta} = \eta\|X\|^2/\|X\|_{\mathrm{F}}^2$). With this re-normalization, our runtime becomes $\widetilde{\mathcal{O}}\left(\frac{\kappa^5}{K^3\varepsilon^6\eta^6}\log^3\frac{k}{\delta}\right)$, which roughly means that this algorithm gets better with larger $\|X\|_{\mathrm{F}}^2/\|X\|^2$, since the type of eigenvalue bound we need is weaker.

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 3.9).

Notice that $\eta\|X\|^2 \le \lambda_k$ follows from our definition of η. The algorithm will proceed as follows: first, consider $C := SXT \in \mathbb{C}^{c\times r}$ as described in Theorem 3.1, with parameters

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

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 3.9 with error parameter $\frac{\varepsilon\sqrt{\lambda_k}}{8\|X\|_{\mathrm{F}}}$, with probability $\ge 1-\delta$,

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

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

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

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

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

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

This is a function that is one when $|x-\hat{\lambda}_i| \leq \frac{1}{8}\eta\|X\|^2$, zero when $|x-\hat{\lambda}_i| \geq \frac{1}{4}\eta\|X\|^2$, and interpolates between them otherwise. From the eigenvalue gap and the aforementioned bound $|\hat{\lambda}_i - \lambda_i| \leq \frac{1}{8}\eta\|X\|^2$, we can conclude that $f_i(X^\dagger X) = v_i v_i^\dagger$ exactly. Further, by Theorem 3.1, 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 3.1 are satisfied because $\varepsilon \lesssim 8 \leq \frac{8}{\eta} = L\|X\|^2$ for L the Lipschitz constant of f_i. The values of r, c are chosen such that $\|R^\dagger \bar{f}_i(CC^\dagger)R - f_i(X^\dagger X)\| \leq \varepsilon/2$ (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: $\|\hat{v}_i\| \leq \sqrt{1+\varepsilon/2} \leq 1 + \varepsilon/4$, so

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

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

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 $\|\bar{v}_i^\dagger\| \leq \sqrt{\max_x \bar{f}_i(x)} \lesssim \lambda_i^{-\frac{1}{2}}$, using Lemmas 2.9 and 2.10, 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 $\widetilde{\mathbf{sq}}_\phi(\hat{v}_i) = \phi\,\mathbf{sq}_\phi(v)\log\frac{1}{\delta} \lesssim \frac{\|X\|_{\mathrm{F}}^2}{\lambda_i} r\log\frac{1}{\delta}$. □

4.4 Matrix inversion and principal component regression

The low-rank matrix inversion algorithm given by Gilyén et al. and Chia et al. [GLT18, CLW18] dequantizes 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 4.9 ($A^+_{\sigma,\eta}$)**.** We define $A^+_{\sigma,\eta}$ to be any singular value transform of A satisfying:

$$A^+_{\sigma,\eta} := \mathrm{tinv}^{(\mathrm{SV})}_{\sigma,\eta}(A) \qquad \mathrm{tinv}_{\sigma,\eta}(\lambda)\begin{cases} = 1/\lambda & \lambda \geq \sigma \\ = 0 & \lambda < \sigma(1-\eta) \\ \in [0, \sigma^{-1}] & \text{otherwise} \end{cases} \tag{12}$$

This definition is analogous to $A_{\sigma,\eta}$ in Section 4.1: 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 4.10. 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 $\|\hat{x} - x^*\| \leq \varepsilon\|x^*\|$, where $x^* := A^+_{\sigma,\eta} b$.

Corollary 4.11. *For $0 < \varepsilon \lesssim \frac{\|A\|^2}{\sigma^2}$ and $\eta \leq 0.99$, we can solve Problem 4.10 in $\widetilde{\mathcal{O}}\left(\frac{\varphi^6 K^3 \kappa^{11}}{\eta^6 \varepsilon^6} \log^3 \frac{1}{\delta}\right)$ time to give $\mathrm{SQ}_\phi(\hat{x})$ for $\widetilde{\mathbf{sq}}_\phi(\hat{x}) = \widetilde{\mathcal{O}}\left(\frac{\varphi^4 K^2 \kappa^5}{\eta^2 \varepsilon^2} \frac{\|x^*\|^2}{\|\hat{x}\|^2} \log^2 \frac{1}{\delta}\right)$.*

If we further assume that $\varepsilon < 0.99$, then $\widetilde{\mathbf{sq}}_\phi(\hat{x})$ can be simplified, since $\frac{\|x^*\|}{\|\hat{x}\|} \leq \frac{\|x^*\|}{\|x^*\| - \varepsilon\|x^*\|} \leq 100$. 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).

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

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

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

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

By Theorem 3.1 with $L = \frac{1}{\eta\sigma^4}$ and $\bar{L} = \frac{1}{\eta^2(1-\eta)^2\sigma^6}$, we can find such R and C with

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

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 3.2,

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

We use Remark 3.11 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 $\hat{x} := R^\dagger\bar{\iota}(CC^\dagger)u$ has the desired error and failure probability. Finally, we can use Lemmas 2.9 and 2.10 with matrix $R^\dagger$ and vector $\bar{\iota}(CC^\dagger)u$ to get $\mathrm{SQ}_\phi(\hat{x})$ for

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

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

4.5 Support vector machines

Support vector machine (SVM) is an important technique for classification with wide applications in supervised learning. A quantum algorithm for solving SVM was first introduced in [RML14]. In this paper, we present a dequantization of this quantum algorithm. 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 $e(j) \ge 0$ 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

$$\begin{bmatrix} 0 & \vec{1}^T \\ \vec{1} & XX^T + \gamma^{-1} I \end{bmatrix} \begin{bmatrix} b \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 \\ y \end{bmatrix}, \tag{13}$$

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 $\hat{F} := F/\operatorname{Tr}(F)$.

The quantum algorithm, given X and y in QRAM, outputs a quantum state $|\hat{F}^+_{\lambda,0.01}[\begin{smallmatrix} 0 \\ y \end{smallmatrix}]\rangle$ (Definition 4.9) in $\operatorname{poly}(\frac{1}{\lambda}, \frac{1}{\varepsilon}, \log mn)$ time. The quantum-inspired analogue is as follows.

Problem 4.12. 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 $\|\hat{x} - \hat{F}^+_{\lambda,\eta}[\begin{smallmatrix} 0 \\ y \end{smallmatrix}]\| \leq \varepsilon \|\hat{F}^+_{\lambda,\eta}[\begin{smallmatrix} 0 \\ y \end{smallmatrix}]\|$ with probability $\geq 1 - \delta$.

Note that we must assume $\|\hat{F}\| \leq 1$; the quantum algorithm makes the same assumption[15]. Another dequantization was reported in [DBH19], 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 3.1 to approximate $f(X^TX) \approx R^T Z R$ and use Lemma 3.5 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.

To solve Problem 4.12, though, we find it convenient to simply reduce to matrix inversion as described in Section 4.4: we first get $\mathrm{SQ}_\phi(\hat{F})$, and then we apply Corollary 4.11 to complete. Section VI.C of [DBH19] claims to dequantize this version, but give no correctness bound[16] or runtime bound (beyond arguing it is polynomial in the desired parameters).

Corollary 4.13. *For $0 < \varepsilon \lesssim 1$ and $\eta \leq 0.99$, we can solve Problem 4.12 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 3.10, 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 $\tilde{K} := xx^T$, 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 2.12. $\|\tilde{K}\|_{\mathrm{F}}^2 = (1+\varepsilon)^2\|X\|_{\mathrm{F}}^4$, 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 $L := \left[\begin{smallmatrix} 0 & \vec{1}^T \\ \vec{1} & \gamma^{-1}I \end{smallmatrix}\right]$

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

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

with $\mathbf{sq}(L) = \mathcal{O}(1)$. Our approximation to $\hat{F}$ is

$$M := \frac{1}{\mathrm{Tr}(F)}\Big(L + \begin{bmatrix} 0 & \vec{0}^T \\ \vec{0} & K \end{bmatrix}\Big); \qquad \|M - \hat{F}\| \leq \frac{1}{\mathrm{Tr}(F)}\|K - XX^T\|_{\mathrm{F}} \leq \frac{1}{\mathrm{Tr}(F)}\varepsilon\|X\|_{\mathrm{F}}^2 \leq \varepsilon.$$

Using Lemma 2.13, 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 $\|\hat{F}\| \leq 1$:

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

Note that we can compute $\mathrm{Tr}(F)$ given $\mathrm{SQ}(X)$. So, applying Corollary 4.11, we can get the desired $\mathrm{SQ}_\phi(v)$ in runtime

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

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 $\mathrm{Q}(M) = \mathcal{O}\big(\frac{1}{\varepsilon^2}\log\frac{q}{\delta}\big)$ doesn't 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:

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

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. (13) 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 4.14. *Given $\mathrm{SQ}(X^T)$ and $\mathrm{SQ}(y)$, with probability $\geq 1-\delta$, we can output a $\hat{b}$ 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. (13). 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. (13). We assume $\mathrm{SQ}(X^T)$ instead of $\mathrm{SQ}(X)$ for convenience, though both are possible via the observation that $f(XX^T) = X\bar{f}(X^TX)X^T$.

4.6 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

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

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 4.15. Consider a Hermitian matrix $H \in \mathbb{C}^{n\times n}$ satisfying $\|H\| = t$, 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 $\ge 1 - \delta$ for some $\hat{b} \in \mathbb{C}^n$ satisfying $\|\hat{b} - e^{iH}b\| \le \varepsilon$.

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 4.16. *Suppose H has minimum singular value σ and $\varepsilon < \min(0.5, \sigma)$. We can solve Problem 4.15 in $\widetilde{\mathcal{O}}\big(t^6 K^3 \kappa^5 \varepsilon^{-6} \log^3 \frac{1}{\delta}\big)$ time, giving $\mathrm{SQ}_\phi(\hat{b})$ with $\widetilde{\mathbf{sq}}_\phi(\hat{b}) = \widetilde{\mathcal{O}}\big(K^2\kappa^2 t^4 \varepsilon^{-4} \log^3 \frac{1}{\delta}\big)$.*

This runtime is dimensionless in a certain sense. The natural error bound to require is that $\|\hat{b} - e^{itH}b\| \le t\varepsilon$, since $|\frac{d}{dx}(e^{-itx})| = t$ (considering $x \in \mathbb{R}$). If we rescale ε up to $t\varepsilon$, the runtime is $\widetilde{\mathcal{O}}\big(K^3\kappa^5\varepsilon^{-6}\log^3\frac{1}{\delta}\big)$, which is dimensionless. The runtime of the algorithm in the following corollary does not have this property, so its scaling with t is worse, despite being faster for, say, $t = 1$.

Corollary 4.17. *Suppose $\varepsilon < \min(0.5, t^3)$. We can solve Problem 4.15 in $\widetilde{\mathcal{O}}\big(t^{16}\|H\|_{\mathrm{F}}^6 \varepsilon^{-6} \log^3 \frac{1}{\delta}\big)$ time, giving $\mathrm{SQ}_\phi(\hat{b})$ with $\widetilde{\mathbf{sq}}_\phi(\hat{b}) = \widetilde{\mathcal{O}}\big(t^8\|H\|_{\mathrm{F}}^4\varepsilon^{-4}\log^3\frac{1}{\delta}\big)$.*

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 3.1 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 3.1 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 $|a'(x)|, |b'(x)| \le |f'(x)|$, 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 3.1 on $f_{\cos}$ and f_{sinc},

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

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

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 4.17. Using the Lipschitz bounds above with Theorem 3.1, we can find $R_{\cos} \in \mathbb{C}^{r_{\cos} \times n}, C_{\cos} \in \mathbb{C}^{r_{\cos} \times c_{\cos}}, R_{\sinc} \in \mathbb{C}^{r_{\sinc} \times n}, C_{\sinc} \in \mathbb{C}^{r_{\sinc} \times c_{\sinc}}$ such that

$$\|R^\dagger_{\cos} \bar{f}_{\cos}(C_{\cos} C^\dagger_{\cos}) R_{\cos} + I - f_{\cos}(H^\dagger H)\| \le \varepsilon \tag{14}$$
$$\|R^\dagger_{\sinc} \bar{f}_{\sinc}(C_{\sinc} C^\dagger_{\sinc}) R_{\sinc} + i \cdot I - f_{\sinc}(H^\dagger H)\| \le \frac{\varepsilon}{t} \tag{15}$$

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

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

As a consequence,

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

Note that, by Lemma 3.2, $\|R_{\cos}\| \lesssim \|H\|$, $\|\bar{f}_{\cos}(C_{\cos} C^\dagger_{\cos})\| \lesssim 1$, and $\|R^\dagger_{\cos} \sqrt{\bar{f}_{\cos}(C_{\cos} C^\dagger_{\cos})}\| \lesssim 1$; the same bounds hold for the sinc analogues. We now approximate using Lemma 3.6 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 $R_{\sinc} H \approx WC$ 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

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

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

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

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 2.9 and 2.10, 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 $\text{SQ}_\phi(\hat{b})$ where

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

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

Proof of Corollary 4.16. Our approach is the same, though with different parameters. For Theorem 3.1, 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

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

all the conditions of Theorem 3.1 are satisfied: in particular, $\sigma^2/2 > \bar{\varepsilon}$ in both cases, up to rescaling ε by a constant factor:

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

Here, we used our initial assumption that $\varepsilon \le \sigma$. So, the bounds Eqs. (14) and (15) hold. Note that, by Lemma 3.2, $\|R_{\cos}\| \lesssim \|H\|$, $\|\bar{f}_{\cos}(C_{\cos}C_{\cos}^\dagger)\| \lesssim \sigma^{-2}$, 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 3.6 four times.

1. We approximate $R_{\cos}b \approx u$ to $\varepsilon\sigma\|b\|$ error, requiring $\mathcal{O}\big(\|H\|_{\text{F}}^2\sigma^{-2}\varepsilon^{-2}\log\frac{1}{\delta}\big)$ samples.

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

$$\begin{aligned}
&\Big\|\hat{b} - \Big(R_{\cos}^\dagger \bar{f}_{\cos}(C_{\cos}C_{\cos}^\dagger)R_{\cos}b + b + R_{\text{sinc}}^\dagger \bar{f}_{\text{sinc}}(C_{\text{sinc}}C_{\text{sinc}}^\dagger)R_{\text{sinc}}Hb + iHb\Big)\Big\| \\
&\quad\le \|R_{\cos}^\dagger \bar{f}_{\cos}(C_{\cos}C_{\cos}^\dagger)(u - R_{\cos}b)\| + \|R_{\text{sinc}}^\dagger \bar{f}_{\text{sinc}}(C_{\text{sinc}}C_{\text{sinc}}^\dagger)(R_{\text{sinc}}H - WC)b\| \\
&\qquad + \|R_{\text{sinc}}^\dagger \bar{f}_{\text{sinc}}(C_{\text{sinc}}C_{\text{sinc}}^\dagger)W(Cb - v)\| + \|iR^\dagger w - iHb\| \\
&\quad\lesssim \sigma^{-1}\|u - R_{\cos}b\| + \sigma^{-1}\|R_{\text{sinc}}H - WC\|\|b\| + \sigma^{-1}\|H\|_{\text{F}}\|Cb - v\| + \|R^\dagger w - Hb\| \le 4\varepsilon\|b\|
\end{aligned}$$

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 2.9 and 2.10, 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 $\text{SQ}_\phi(\hat{b})$ where

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

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

Remark 4.18. In the case where H is not low-rank, we could still run a modified version of Corollary 4.16 to compute a modified "$\exp_{\sigma,\eta}(iHt)$" where singular values below σ are smoothly thresholded away. Following the same logic as Definition 4.9, we could redefine $f_{\cos}$ such that $f_{\cos}(x) = 1$ for $x < \sigma^2(1-\eta)$, $f_{\cos}(x) = \cos(\sqrt{\lambda})$ for $x \ge \sigma^2$, and is a linear interpolation between the endpoints for the x in between (and f_{sinc} 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 4.19. 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.

4.7 Semidefinite program solving

A recent line of inquiry in quantum computing [BS17, vAGGdW17, 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 et al. [CLLW19] 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 4.20 (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[17]

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

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 4.21. *Let $F \geq \max_{j\in[m]}(\|A^{(j)}\|_{\mathrm{F}})$, and suppose $F = \Omega(1)$. Then we can solve Problem 4.20 with success probability $\geq 1 - \delta$ in cost*

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

providing sampling and query access to a solution.

Like prior work on quantum algorithms for SDP-solving, we use the matrix multiplicative weights (MMW) framework [AK16, Kal07] to solve Problem 4.20. Corollary 4.21 immediately follows from running the algorithm this framework admits (Algorithm 1), where we solve an instance of the problem described in Lemma 4.22 with precision $\theta = \varepsilon/4$ in each of the $\mathcal{O}\big(\ln(n)/\varepsilon^2\big)$ 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 $\mathcal{O}\big(\frac{\ln n}{\varepsilon^2}\big)$ iterations; a proof can be found, e.g., in the work of Brandão et al. [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.

[17] For simplicity, we assume here that X is normalized to have trace 1. This can be relaxed; for an example, see [vAGGdW17].

Algorithm 1: MMW based feasibility testing algorithm for SDPs

Set $X_1 := \frac{I_n}{n}$, and the number of iterations $T := \frac{16 \ln n}{\varepsilon^2}$;
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**
    $X_{t+1} := \exp[-\frac{\varepsilon}{4} \sum_{i=1}^{t} A^{(j_i)}] / \operatorname{Tr}[\exp[-\frac{\varepsilon}{4} \sum_{i=1}^{t} A^{(j_i)}]]$
  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 4.22 ("Efficient" trace estimation)**.** *Consider the setting described in Corollary 4.21. Given* $\theta \in (0, 1]$, $t \leq \frac{\ln(n)}{\theta^2}$ *and* $j_i \in [m]$ *for* $i \in [t]$, *defining* $H := \exp[-\theta \sum_{i=1}^{t} A^{(j_i)}]$, *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*

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

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.*

We will use Theorem 3.4 and Remark 3.11. In order to understand how precisely we need to approximate the matrix 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 4.23. *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 4.24 (Perturbations of the partition function)**.** *For all Hermitian matrices* $H, \tilde{H} \in \mathbb{C}^{n \times n}$,

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

The bound in the above lemma is tight, as shown by the example $\tilde{H} := H + \varepsilon I$. 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

$$\operatorname{Tr}(e^H) = n + \operatorname{Tr}(e^H - I) = n + \sum_i (e^{\lambda_i} - 1) \ge n + \sum_i \lambda_i = n + \operatorname{Tr}(H) \ge n - \sqrt{n}\|H\|_{\mathrm{F}} \ge n/2. \quad (16)$$

Lemma 4.25. *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*

$$|(\operatorname{Tr}(e^D) + n - r) - \operatorname{Tr}(e^H)| \le 2(e-1)\varepsilon \operatorname{Tr}(e^H), \quad (17)$$

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, notice that $\widetilde{U}$ is close to an isometry. The bound that $\|\widetilde{U}\widetilde{U}^\dagger - I\| \le \delta$ implies that the singular values σ_i of $\widetilde{U}$ satisfy $\sigma_i \in \sqrt{1 \pm \delta}$. Further, for $U := (\widetilde{U}\widetilde{U}^\dagger)^{-\frac{1}{2}}\widetilde{U}$, which is $\widetilde{U}$ with all singular values set to one, $UU^\dagger = I$ and

$$\|\widetilde{U} - U\| \le \max_{\sigma \in \sqrt{1\pm\delta}} |\sigma - 1| \le (2 - \sqrt{2})\delta.$$

Consequently,

$$\begin{aligned}\|H - U^\dagger D U\| &\le \|H - \widetilde{U}^\dagger D \widetilde{U}\| + \|\widetilde{U}^\dagger D\widetilde{U} - U^\dagger D \widetilde{U}\| + \|U^\dagger D\widetilde{U} - U^\dagger D U\| \le \varepsilon + \|\widetilde{U} - U\|(\|D\widetilde{U}\| + \|D\|) \\ &\le \varepsilon + (2-\sqrt{2})\delta(\|\widetilde{U}^+\|^{-1} + \|\widetilde{U}^+\|^{-2})\|\widetilde{U}^\dagger D\widetilde{U}\| \le \varepsilon + 4\delta(\|H\| + \varepsilon) \le 2\varepsilon. \quad (18)\end{aligned}$$

By Lemma 4.24 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,

$$\begin{aligned}&|\operatorname{Tr}(AU^\dagger(e^D - I)U) + \operatorname{Tr}(A) - \operatorname{Tr}(Ae^H)| \\ &\qquad = |\operatorname{Tr}(A(e^{U^\dagger D U} - e^H))| \le \|A\|\|e^{U^\dagger D U} - e^H\|_1 \le 2(e-1)\|A\|\varepsilon \operatorname{Tr}(e^H). \quad (19)\end{aligned}$$

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. (19) and triangle inequality.

$$
\begin{aligned}
&|\operatorname{Tr}(AU^\dagger(e^D - I)U) - \operatorname{Tr}(A\widetilde{U}^\dagger(e^D - I)\widetilde{U})| \\
&= |\operatorname{Tr}((UAU^\dagger - \widetilde{U}A\widetilde{U}^\dagger)(e^D - I))| \\
&\le \left\|UAU^\dagger - \widetilde{U}A\widetilde{U}^\dagger\right\| \left\|e^D - I\right\|_1 && \text{by trace-norm and Hölder's inequality} \\
&\le (\|U - \widetilde{U}\|\|AU^\dagger\| + \|\widetilde{U}A\|\|U^\dagger - \widetilde{U}^\dagger\|)\left\|e^D - I\right\|_1 && \text{analogously to Eq. (18)} \\
&\le \varepsilon\|A\|\left\|e^D - I\right\|_1 && \text{by assumption that } \delta \le \varepsilon/2 \\
&\le \varepsilon\|A\|\big(\operatorname{Tr}(e^D) + r\big) && \text{by triangle inequality} \\
&\le \varepsilon\|A\| \operatorname{Tr}(e^H). && \text{by Eqs. (16) and (17), } \operatorname{Tr}(e^D) \lesssim n \lesssim \operatorname{Tr}(e^H)
\end{aligned}
$$

□

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

Proof of Lemma 4.22. By Lemma 4.23, 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 $H := -\theta\sum_{i=1}^t A^{(j_i)}$. By triangle inequality, $\|H\|_{\mathrm{F}} \le \frac{F}{\theta}\ln(n)$. Because H is a linear combination of matrices, by Lemma 2.13, after paying $\frac{\ln(n)}{\theta^2}\,\mathbf{n}(A)$ cost, we can obtain $\mathrm{SQ}_\phi(H)$ for $\phi \le \frac{F^2\ln^2(n)}{\theta^2\|H\|_{\mathrm{F}}^2}$ with $\mathbf{q}(H) = \mathbf{q}_\phi(H) \le \frac{\ln(n)}{\theta^2}\,\mathbf{q}(A)$ and $\mathbf{s}_\phi(H) = \mathbf{s}(A)$.

If $\frac{F}{\theta}\ln(n) > \sqrt{n}/18$, 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 $\operatorname{Tr}(e^HA^{(m)})$ in time $\mathcal{O}\big(mn^2\big)$. The overall complexity is $\mathcal{O}\big(n^2(t\,\mathbf{q}(A) + n + m)\big) = \widetilde{\mathcal{O}}\Big(\frac{F^6}{\theta^6}\,\mathbf{q}(A)\ln^6(n) + m\frac{F^4}{\theta^4}\ln^4(n)\Big)$.

If $\frac{F}{\theta}\ln(n) \le \sqrt{n}/18$ we do the following. Note that if $\|H\| \le 1$, then $\operatorname{Tr}(e^H) \ge n/e$ and $\operatorname{Tr}(A^{(i)}e^H) \le \left\|A^{(i)}\right\|_{\mathrm{F}}\left\|e^H\right\|_{\mathrm{F}} \le Fe\sqrt{n}$, 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 3.4 (with $f(x) = x$, so that $L = 1$, and choosing $\varepsilon := \Theta(\theta)$) to find a diagonal matrix $D \in \mathbb{R}^{s\times s}$ with $s = \widetilde{\mathcal{O}}\Big(\phi^2\|H\|_{\mathrm{F}}^6/\varepsilon^6\log(1/\delta)\Big) = \widetilde{\mathcal{O}}\big(F^6\theta^{-6}\ln^6(n)\varepsilon^{-6}\log(1/\delta)\big)$ $= \widetilde{\mathcal{O}}\big(F^6\theta^{-12}\ln^6(n)\log(1/\delta)\big)$ together with an approximate isometry $\widetilde{U} = N(SH) \in \mathbb{C}^{s\times n}$ such that $\left\|H - \widetilde{U}^\dagger D\widetilde{U}\right\| \le \mathcal{O}(\varepsilon)$. 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 3.4 we have $\left\|\widetilde{U}\widetilde{U}^\dagger - I\right\| \lesssim \varepsilon^3\|H\|^{-3} \lesssim \frac{\varepsilon}{\|H\|+\varepsilon} + \varepsilon$ with probability at least $1 - \frac{\delta}{2}$. As per Theorem 3.4, the cost of this is $\log^3(1/\delta)$ times at most

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

By Lemma 4.25 we have[18] 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. (16) 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 3.11 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 4.25 and Eq. (16) we have

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

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

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

Therefore, Remark 3.11 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

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

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

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

4.8 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 $\{x_i \in \mathbb{R}^N : 1 \le i \le M\}$ 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

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

[18] In case applying Theorem 3.4 would result in $s > n$, we instead directly diagonalize H ensuring $s \le n$.

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

$$S_B^{\frac{1}{2}} S_W^{-1} S_B^{\frac{1}{2}}. \tag{20}$$

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. (20) (after truncating eigenvalues), leading to inexact eigenvectors. We construct a classical analogue of their quantum algorithm.[19] 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 4.4 and Section 4.5, so for brevity we will skip dequantizing this algorithm.

To formally analyze this algorithm, we could, as in Section 4.3, 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 4.26 (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 $S_W := W^\dagger W$ and $S_B := B^\dagger B$, 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.

Corollary 4.27. *For $\varepsilon < \sigma/\|B\|$, we can solve Problem 4.26 in $\widetilde{\mathcal{O}}\Big((\frac{\|B\|^4 \|B\|_{\mathrm{F}}^6}{\varepsilon^6 \sigma^{10}} + \frac{\|W\|^{10} \|W\|_{\mathrm{F}}^6}{\varepsilon^6 \sigma^{16}}) \log^3 \frac{1}{\delta} \Big)$ time, with $\widetilde{\mathbf{sq}}_\phi(U(\cdot, i)) = \widetilde{\mathcal{O}}\Big(\frac{\|B\|_{\mathrm{F}}^4}{\varepsilon^2 \sigma^4} \log^2 \frac{1}{\delta} \Big)$.*

Proof. By Theorem 3.1, we can find R_B, C_B, R_W, C_W such that

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

[19] 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.

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 $Z_B := \overline{\operatorname{sqrt}}(C_B C_B^\dagger)$ and $Z_W := \overline{\operatorname{inv}}(C_W C_W^\dagger)$. 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 $Z := Z_B R'_B R_W'^\dagger Z_W R'_W R_B'^\dagger Z_B$ 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 3.12, $([C_B]^+_{\frac{\sigma}{\sqrt2}} R_B)^\dagger$ is an $\varepsilon\sigma/\|B\|$-approximate projective isometry[20] 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, $U := R_B^\dagger([C_B]^+_{\frac{\sigma}{\sqrt2}})^\dagger V$ 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 $D := \Sigma$. This satisfies the desired bounds because

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

[20] 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:

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

$\tilde{U}D\tilde{U}^\dagger$ 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 5.4) 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 2.9 and 2.10, 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. □

5 Proofs

5.1 Sampling and query access

Lemma 2.9. *Suppose we are given $\mathrm{SQ}_\phi(v)$ and some $\delta \in (0, 1]$. Denote $\widetilde{\mathbf{sq}}(v) := \phi\,\mathbf{sq}_\phi(v)\log\frac{1}{\delta}$. We can sample from $\mathcal{D}_v$ with probability $\ge 1 - \delta$ in $\mathcal{O}(\widetilde{\mathbf{sq}}(v))$ time. We can also estimate $\|v\|$ to ν multiplicative error for $\nu \in (0, 1]$ with probability $\ge 1 - \delta$ in $\mathcal{O}\big(\frac{1}{\nu^2}\widetilde{\mathbf{sq}}(v)\big)$ time.*

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 $r(i) := \frac{|v(i)|^2}{|\tilde{v}(i)|^2}$. Otherwise, restart. We can perform this procedure: we can compute $r(i)$ given $\mathrm{SQ}_\phi(v)$ and $r(i) \le 1$ since $\tilde{v}$ bounds v.

The probability of accepting a sample in a round is $\sum_i \mathcal{D}_{\tilde{v}}(i)r(i) = \phi^{-1}$ 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.

Further, since the probability of accepting is ϕ^{-1} and we know $\phi\|v\|^2$, we can use this to estimate $\|v\|^2$. Suppose we ran $z = \mathcal{O}\big(\nu^{-2}\phi\log\frac{1}{\delta}\big)$ rounds, of which Z fraction of them lead to acceptance. Then, by a Chernoff bound,

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

so $Z(\phi\|v\|^2)$ is a good multiplicative approximation to $\|v\|^2$. □

Lemma 2.10 (Linear combinations, Proposition 4.3 of [Tan19])**.** *Given $\mathrm{SQ}_{\varphi_1}(v_1), \ldots, \mathrm{SQ}_{\varphi_k}(v_k) \in \mathbb{C}^n$ and $\lambda_1, \ldots, \lambda_k \in \mathbb{C}$, we have $\mathrm{SQ}_\phi(\sum\lambda_i v_i)$ for $\phi = k\frac{\sum\varphi_i\|\lambda_i v_i\|^2}{\|\sum\lambda_i v_i\|^2}$ and $\mathbf{sq}_\phi(\sum\lambda_i v_i) := \max_{i\in[k]}\mathbf{s}_{\varphi_i}(v_i) + \sum_{i=1}^k\mathbf{q}(v_i)$ (after paying $\mathcal{O}\Big(\sum_{i=1}^k\mathbf{n}_{\varphi_i}(v_i)\Big)$ one-time pre-processing cost to query for norms).*

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

$$\tilde{u}(s) = \sqrt{k \sum_{i=1}^{k} |\lambda_i \tilde{v}_i(s)|^2}.$$

That $|\tilde{u}(s)| \geq |u(s)|$ follows from Cauchy-Schwarz, and $\|\tilde{u}\|^2 = k \sum_{i=1}^{k} \|\lambda_i \tilde{v}_i\|^2 = k \sum_{i=1}^{k} \varphi_i \|\lambda_i v_i\|^2$, giving the desired value of ϕ.

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

$$\sum_{i=1}^{k} \frac{\|\lambda_i \tilde{v}_i\|^2}{\sum_\ell \|\lambda_\ell \tilde{v}_\ell\|^2} \frac{|\tilde{v}_i(j)|^2}{\|\tilde{v}_i\|^2} = \frac{\sum_{i=1}^{k} |\lambda_i \tilde{v}_i(j)|^2}{\sum_{\ell=1}^{k} \|\lambda_\ell \tilde{v}_\ell\|^2} = \frac{|\tilde{u}(j)|^2}{\|\tilde{u}\|^2}.$$

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 2.15), and we can sample from $\tilde{v}_i$ using our sampling and query access to it. □

Lemma 2.12. *Given vectors $u \in \mathbb{C}^m, v \in \mathbb{C}^n$ with $\mathrm{SQ}_{\varphi_u}(u), \mathrm{SQ}_{\varphi_v}(v)$ access we have $\mathrm{SQ}_\phi(A)$ for their outer product $A := uv^\dagger$ 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 $\|\tilde{v}\|$. □

Lemma 2.13. *Given $\mathrm{SQ}_{\varphi^{(1)}}(A^{(1)}), \ldots, \mathrm{SQ}_{\varphi^{(\tau)}}(A^{(\tau)}) \in \mathbb{C}^{m \times n}$, we have $\mathrm{SQ}_\phi(A) \in \mathbb{C}^{m \times n}$ for $A := \sum_{t=1}^{\tau} \lambda_t A^{(t)}$ with $\phi = \frac{\tau \sum_{t=1}^{\tau} \varphi^{(t)} \|\lambda_t A^{(t)}\|_{\mathrm{F}}^2}{\|A\|_{\mathrm{F}}^2}$ 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 $\mathcal{O}\big(\sum \mathbf{q}(A^{(t)})\big)$ 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 2.10, we can sample from $\tilde{a}$ by first sampling $s \in [\tau]$

with probability $\frac{\|\lambda_s \tilde{A}^{(s)}\|_F^2}{\sum_t \|\lambda_t \tilde{A}^{(t)}\|_F^2}$, then taking our sample to be $i \in [m]$ from $\tilde{a}^{(s)}$. If we pre-process by querying all the Frobenius norms $\|\tilde{A}^{(t)}\|_F$ 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 $i \in [m]$ from $\tilde{A}^{(s)}(i,\cdot)$. 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. □

5.2 Sketching matrices and technical tools

Here, we present various lemmas about approximating matrices by sketches that will be used to prove our results.

Lemma 5.1 (Frobenius norm bounds for matrix sketches)**.** *Given $A \in \mathbb{C}^{m\times n}$, let $S \in \mathbb{R}^{r\times m}$ be sampled according to $p \in \mathbb{R}^m$, a ϕ-oversampled importance sampling distribution from A. Then $\|SA\|_F^2 \le \phi\|A\|_F^2$ (always) and*

$$\Pr\left[\left|\|SA\|_F^2 - \|A\|_F^2\right| \le \sqrt{\frac{\phi^2 \ln(2/\delta)}{2r}}\|A\|_F^2\right] \le \delta.$$

Proof. $\|SA\|_F^2$ is the average of the norms of $r\|[SA](i,\cdot)\|^2$ for rows $i \in [r]$ and

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

Note that the second inequality implies that $\|SA\|_F^2 \le \phi\|A\|_F^2$. The other inequality in the theorem statement follows by Hoeffding's inequality. □

Lemma 2.18. *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 $(i_1,c_1),\dots,(i_r,c_r)$. If $r \ge 2\varphi^2 \ln\frac{2}{\delta}$, 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 $\mathbf{q}(SA) = \mathbf{q}(A)$, $\mathbf{s}_\phi(SA) = \mathbf{s}_\phi(A)$, $\mathbf{q}_\phi(SA) = \mathbf{q}_\phi(A)$, and $\mathbf{n}_\phi(SA) = \mathcal{O}(1)$, after $\mathcal{O}(\mathbf{n}_\phi(A))$ pre-processing cost. The runtimes for $\mathrm{SQ}_\phi((SA)^\dagger)$ are $\mathbf{q}((SA)^\dagger) = \mathbf{q}(A)$, $\mathbf{s}_\phi((SA)^\dagger) = \mathbf{s}_\phi(A) + r\,\mathbf{q}_\phi(A)$, $\mathbf{q}_\phi((SA)^\dagger) = r\,\mathbf{q}_\phi(A)$, and $\mathbf{n}_\phi((SA)^\dagger) = \mathbf{n}_\phi(A)$.

Proof. By Lemma 5.1, $\|SA\|_F^2 \ge \|A\|_F^2/2$ 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, $\|S\tilde{A}\|_F^2 = \|\tilde{A}\|_F^2$ by Remark 2.17, so $\phi = \|S\tilde{A}\|_F^2/\|SA\|_F^2 = \varphi\|A\|_F^2/\|SA\|_F^2 \le 2\varphi$. 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 $\mathrm{SQ}((S\tilde{A})^\dagger)$.

We have $\mathrm{SQ}(S\tilde{A})$. Because the rows of $S\tilde{A}$ are rescaled rows of A, we have SQ access to them from SQ access to A. Because $\|S\tilde{A}\|_F^2 = \|\tilde{A}\|_F^2$ and $\|[S\tilde{A}](i,\cdot)\|^2 = \|\tilde{A}\|_F^2/r$, after precomputing $\|\tilde{A}\|_F^2$, 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 $(SA)^\dagger$ 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 A). The distribution of the samples output by this procedure is correct: the probability of outputting j is

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

When $\varphi = 1$, the same argument as above works, except $\|SA\|_{\mathrm{F}}^2 = \|A\|_{\mathrm{F}}^2$ because $A = \tilde{A}$, so the oversampling constant remains the same, incurring no chance of failure. □

Lemma 3.5 (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 $S \in \mathbb{R}^{s\times m}$ to be sampled according to $p \in \mathbb{R}^m$ a ϕ-oversampled importance sampling distribution from X or Y. Then,*

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

Proof. It suffices to consider $s = 1$, because the variance of independent samples decreases as $1/s$. Since $E[S^\dagger S] = I$, we have $\mathbb{E}[X^\dagger S^\dagger SY] = X^\dagger Y$, so

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

The second other inequality follows similarly:

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

Drineas, Kannan, and Mahoney use a concentration inequality to prove tighter error bounds 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.

Lemma 5.2 (Matrix multiplication by subsampling [DKM06, Theorem 1])**.** *Suppose we are given $A \in \mathbb{C}^{n\times m}, B \in \mathbb{C}^{n\times p}, c \in \mathbb{Z}^+$ and a distribution $p \in \mathbb{R}^n$ satisfying the oversampling condition that, for some $\phi \ge 1$,*

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

Let $S \in \mathbb{R}^{c\times n}$ be sampled according to p. Then $A^\dagger S^\dagger SB$ is an unbiased estimator for $A^\dagger B$ and

$$\Pr\Big[\|A^\dagger S^\dagger SB - A^\dagger B\|_{\mathrm{F}} < \sqrt{\frac{8\phi^2\ln(2/\delta)}{c}}\underbrace{\sum\|A(k,\cdot)\|\|B(k,\cdot)\|}_{\le\|A\|_{\mathrm{F}}\|B\|_{\mathrm{F}}}\Big] > 1-\delta.$$

Our main technical tool follows as a simple application of Lemma 5.2.

Lemma 3.6 (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 $S \in \mathbb{R}^{s\times m}$ to be sampled according to $r := \frac{p+q}{2}$, where $p, q \in \mathbb{R}^m$ are ϕ_1, ϕ_2-oversampled importance sampling distributions from X, Y, respectively. Then S is a $2\phi_1, 2\phi_2$-oversampled importance sampling sketch of X, Y, respectively. Further,*

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

Proof. First, notice that $2r(i) \geq p(i)$ and $2r(i) \geq q(i)$, so r oversamples the importance sampling distributions for X and Y with constants $2\phi_1$ and $2\phi_2$, respectively. The bounds on $\|SX\|_{\mathrm{F}}$ and $\|SY\|_{\mathrm{F}}$ then follow from Lemma 5.1. We get the other bound by using Lemma 5.2; r 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:

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

□

We will use the above approximation results to prove that sketching preserves singular values well.

Lemma 3.9 (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 $r = \tilde{\Omega}(\frac{\phi^2}{\varepsilon^2}\log\frac{1}{\delta})$ and $c = \tilde{\Omega}(\frac{\phi^2}{\varepsilon^2}\log\frac{1}{\delta})$. Then, if σ_i and $\hat{\sigma}_i$ are the singular values of A and SAT, respectively (where $\hat{\sigma}_i = 0$ for $i > \min(r,c)$), with probability $\geq 1-\delta$,*

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

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

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 5.3 (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 5.4 (Weyl's inequality). *For $A, B \in \mathbb{C}^{m\times n}$, $|\sigma_k(A) - \sigma_k(B)| \leq \|A - B\|$.*

Proof of Lemma 3.9. We use known theorems, plugging in the values of r and c. Using Lemma 3.6 for the sketch S, we know that

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

by Lemma 2.18, $T^\dagger$ is an $\le 2\phi$-oversampled importance sampling sketch of $(SA)^\dagger$, so by Lemma 3.6 for $T^\dagger$,

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

and from Lemma 5.1,

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

By rescaling δ and union bounding, we can have all events happen with probability $\ge 1-\delta$. Then, from Lemma 5.3,

$$\sqrt{\sum |\sigma_i(SA)^2 - \sigma_i(A)^2|^2} \le \varepsilon \|A\|_{\mathrm{F}}^2$$
$$\sqrt{\sum |\sigma_i(SAT)^2 - \sigma_i(SA)^2|^2} \le \varepsilon \|A\|_{\mathrm{F}}^2$$

The result follows from the triangle inequality. The analogous result holds for spectral norm via Lemma 3.8 and Lemma 5.4; 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:

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

Lemma 3.12. *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 $R := SA$ and $C := SAT$. 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 $k' \le \min(k, \mathrm{rank}(A))$).*

Proof. The following occurs with probability $\ge 1-\delta$. By Lemma 3.9, $\|C_k^+\| \gtrsim \frac{1}{\sigma_k^2}$. By Lemma 3.8, $\|R^\dagger R - A^\dagger A\| = O(\|A\|^2)$, which implies that $\|R\| = O(\|A\|)$, and by Lemma 5.1, $\|R\|_{\mathrm{F}} = O(\|A\|_{\mathrm{F}})$. Further, $\|RR^\dagger - CC^\dagger\| \le \alpha\sigma_k^2 \frac{\|R\|\|R\|_{\mathrm{F}}}{\|A\|\|A\|_{\mathrm{F}}} = O(\alpha\sigma_k^2)$. 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:

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

5.3 Singular value transformation

We first introduce some lemmas which we will use in the section. The following three lemmas discuss *eigenvalue* transformation, but also imply similar results for singular value transformation.

Lemma 5.5 ([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 $\|f^{(\mathrm{EV})}(A) - f^{(\mathrm{EV})}(B)\|_{\mathrm{F}} \le L\|A-B\|_{\mathrm{F}}$.*

Lemma 5.6 ([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*

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

Lemma 5.7 ([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*

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

The log term in Lemmas 5.6 and 5.7 unfortunately cannot be removed (following from the fact that 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. We will only Lemma 5.7 because it makes subsequent analysis (Theorem 3.1) easier. Using the reduction from [GSLW19, Corollary 21], we can conclude that the same result holds for singular value transformation of general matrices.

Lemma 5.8. *Let $A, B \in \mathbb{C}^{m\times n}$ be matrices and let $f\colon \mathbb{R} \to \mathbb{C}$ be a singular value transformation that is L-Lipschitz continuous on the singular values of A and B. Then*

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

We now show how to perform singular value transformation on $A^\dagger A$ more efficiently.

Theorem 3.1 (Even singular value transformation)**.** *Let $A \in \mathbb{C}^{m\times n}$ and $f\colon \mathbb{R}^+ \to \mathbb{C}$ be such that, f and $\bar{f}(x) := (f(x) - f(0))/x$ are L-Lipschitz and $\bar{L}$-Lipschitz, respectively, on $\cup_{i=1}^n [\sigma_i^2 - d, \sigma_i^2 + d]$ for some $d > 0$. Take parameters ε and δ such that $0 < \varepsilon \lesssim L\|A\|_*^2$ 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 2.18). Then, for $R := SA$ and $C := SAT$, 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, $d > \bar{\varepsilon} := \|A\|\|A\|_{\mathrm{F}}(\frac{\phi^2\log(1/\delta)}{\min(r,c)})^{1/2}$) and

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

Lemma 3.2 (Norm bounds for even singular value transformation)**.** *Suppose the assumptions from Theorem 3.1 hold and the event in Eq. (3) occurs (that is, $R^\dagger \bar{f}(CC^\dagger)R \approx f(A^\dagger A) - f(0)I$). Then we can additionally assume that the following bounds also hold:*

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

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

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

Proof of Theorem 3.1 and Lemma 3.2. Since $f(A^\dagger A) = [f(x) - f(0)](A^\dagger A) + f(0)I$, it suffices to show for those f such that $f(0) = 0$. We choose r, c such that the following holds:

1. The ith singular value of $CC^\dagger$ does not differ from the ith singular value of A by more than $\bar{\varepsilon}$. This follows from Lemma 3.9 with error parameter $\varepsilon \|A\|_{\mathrm{F}}^{-1} \|A\|_*^{-1} \max(L^{-1}, \bar{L}^{-1}\|A\|^{-2})$.

   This immediately implies Eq. (6).

2. $\|R\|^2 = O(\|A\|^2)$. This is the spectral norm bound in Eq. (5) (the Frobenius norm bound follows from Lemma 5.1). We use Lemma 3.8:

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

   This is the only place the assumption that $\varepsilon \lesssim L\|A\|_*^2$ is used.

3. $\|f(R^\dagger R) - f(A^\dagger A)\|_* \le \frac{\varepsilon}{2}$. We need the polylog factors in our number of samples to deal with the $\log r$ that arises from Lemma 5.7 in the spectral norm case.

$$\begin{aligned}
&\|f(R^\dagger R) - f(A^\dagger A)\|_* \\
&\lesssim L\|R^\dagger R - A^\dagger A\|_* \log \operatorname{rank}(R^\dagger R) && \text{(Lemma 5.7 or Lemma 5.5)} \\
&\lesssim L\sqrt{\frac{\phi^2 \log r \log(1/\delta)}{r}} \|A\|_* \|A\|_{\mathrm{F}} \log r && \text{(Lemma 3.8 or Lemma 3.6)} \\
&\lesssim \varepsilon.
\end{aligned}$$

4. $\|f(CC^\dagger) - f(RR^\dagger)\|_* \le \frac{\varepsilon}{2\|R\|^2}$. This follows similarly to the above point, additionally using that $\|R\| = O(\|A\|)$.

Using the above points, we can conclude:

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

This immediately gives Eq. (3). When the event occurs (and $* = \mathrm{Op}$), we also have Eq. (7), since

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

Theorem 3.3 (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 $f : \mathbb{R} \to \mathbb{C}$ be a function such that $f(0) = 0$, $g(x) := f(\sqrt{x})/\sqrt{x}$ is L-Lipschitz, and $\bar{g}(x) := g(x)/x$ is $\bar{L}$-Lipschitz. Then, for $0 < \varepsilon \lesssim L\|A\|^3$, 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 $r = \tilde{\Omega}\Big(\phi^2 L^2\|A\|^2\|A\|_{\mathrm{F}}^4 \frac{1}{\varepsilon^2}\log\frac{1}{\delta}\Big)$ and $c = \tilde{\Omega}\Big(\phi^2 L^2\|A\|^4\|A\|_{\mathrm{F}}^2 \frac{1}{\varepsilon^2}\log\frac{1}{\delta}\Big)$. Finding S, M, and T takes time

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

Proof. Consider the the SVT $g(x) := f(\sqrt{x})/\sqrt{x}$, so that $f^{(\mathrm{SV})}(A) = Ag(A^\dagger A)$. First, use Theorem 3.1 to get $SA \in \mathbb{C}^{r\times n}$, $SAT \in \mathbb{C}^{r\times c}$ such that, with probability $\geq 1 - \delta/2$,

$$\|(SA)^\dagger \bar{g}((SAT)(SAT)^\dagger)SA - g(A^\dagger A)\| \leq \frac{\varepsilon}{\|A\|}.$$

Second, use Lemma 3.6 to get a sketch $T'^\dagger \in \mathbb{C}^{c'\times n}$ such that, with probability $\geq 1 - \delta/2$,

$$\|A(SA)^\dagger - AT'(SAT')^\dagger\| \leq \varepsilon(L\|A\|)^{-1}.$$

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 2.18):

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

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

$$\|A(SA)^\dagger \bar{g}((SAT)(SAT)^\dagger)SA - Ag(A^\dagger A)\| \leq \|A\|\|(SA)^\dagger \bar{g}((SAT)(SAT)^\dagger)SA - g(A^\dagger A)\| \leq \varepsilon$$

$$\begin{aligned}&\|(AT'(SAT')^\dagger - A(SA)^\dagger)\bar{g}((SAT)(SAT)^\dagger)SA\|\\ &\leq \|AT'(SAT')^\dagger - A(SA)^\dagger\|\|\bar{g}((SAT)(SAT)^\dagger)SA\|\\ &\leq \|AT'(SAT')^\dagger - A(SA)^\dagger\|\sqrt{L}\sqrt{\|g(A^\dagger A)\| + \frac{\varepsilon}{\|A\|}}\\ &\lesssim \|AT'(SAT')^\dagger - A(SA)^\dagger\|L\|A\| \leq \varepsilon\end{aligned}$$

This gives us a CUR decomposition of A, with AT' as C, $(SAT')^\dagger \bar{\pi}((SAT)(SAT)^\dagger)$ as U, and SA as R.

The time complexity of this procedure is

$$\mathcal{O}\big((r+c+c')\,\mathbf{s}(A) + (rc+rc')\,\mathbf{q}(A) + r^2c' + r^2c\big),$$

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

Theorem 3.4 (Eigenvalue transformation)**.** *Suppose we are given a Hermitian* $\mathrm{SQ}_\phi(A) \in \mathbb{C}^{n\times n}$*, 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* $\varepsilon \in (0, L\|A\|]$*. Then we can output matrices* $S \in \mathbb{C}^{s\times n}$*,* $N \in \mathbb{C}^{s'\times s}$*, and* $D \in \mathbb{C}^{s'\times s'}$*, with* $s = \widetilde{\mathcal{O}}\Big(\phi^2\|A\|^4\|A\|_{\mathrm{F}}^2 \frac{L^6}{\varepsilon^6} \log\frac{1}{\delta}\Big)$ *and* $s' = \mathcal{O}\big(\|A\|_{\mathrm{F}}^2 L^2/\varepsilon^2\big)$*, such that*

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

in time

$$\begin{aligned}\tilde{\mathcal{O}}\Big(&(L^{10}\varepsilon^{-10}\|A\|^8\|A\|_{\mathrm{F}}^2\phi^2 \log\frac{1}{\delta} + L^6\varepsilon^{-6}\|A\|_{\mathrm{F}}^6\phi^3\log\frac{1}{\delta})(\mathbf{s}_\phi(A) + \mathbf{q}_\phi(A)) \\ &+ (L^{16}\varepsilon^{-16}\|A\|^{12}\|A\|_{\mathrm{F}}^4\phi^4\log^2\frac{1}{\delta} + L^{18}\varepsilon^{-18}\|A\|^8\|A\|_{\mathrm{F}}^{10}\phi^7\log^3\frac{1}{\delta})\,\mathbf{q}(A) \\ &+ L^{22}\varepsilon^{-22}\|A\|^{16}\|A\|_{\mathrm{F}}^6\phi^6\log^3\frac{1}{\delta}\Big).\end{aligned}$$

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

Proof. Keep in mind throughout this proof that 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 singular vectors of A, we can add these projectors to our expression without incurring too much spectral norm error.

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

Second, use Theorem 3.1 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 2.18):

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

This approximation doesn't 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, use Remark 3.11(b) lazily: pull $t := \mathcal{O}\Big(\phi^3\|A\|_{\mathrm{F}}^6\frac{L^6}{\varepsilon^6}\log\frac{r^2}{\delta}\Big)$ samples from $\mathrm{SQ}_\phi(A)$ 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 $\frac{\varepsilon^3\|x\|\|y\|}{L^3\phi\|A\|_{\mathrm{F}}^2}$ additive error with *no additional queries to* $\mathrm{SQ}_\phi(A)$. Then, by union bound, with probability $\ge 1-\delta$, one can output an estimate of $R(i,\cdot)AR(j,\cdot)^\dagger$ up to $\frac{\varepsilon^3\|R(i,\cdot)\|\|R(j,\cdot)\|}{L^3 r\phi\|A\|_{\mathrm{F}}^2} \le \frac{\varepsilon^3}{L^3 r}$ error for all $i, j \in [s]$ such that $i \le j$. Let M be the matrix of these estimates, so $\|M - RAR^\dagger\|_{\mathrm{F}} \le \frac{\varepsilon^3}{L^3}$. From Eqs. (6) and (7),

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

We have shown that we can find an RUR approximation to A, with R as R and $\bar{\pi}(CC^\dagger)M\bar{\pi}(CC^\dagger)$ as U. However, if we wish to apply an eigenvalue transformation to A, we need to access the eigenvalues of A as well. Our goal is to modify our RUR decomposition to an approximate unitary eigendecomposition $\hat{U}\hat{D}\hat{U}^\dagger$, where $\hat{U}$ is an α-approximate isometry for $\alpha := \frac{\varepsilon}{L\|A\|}$. Recall that $\hat{U}$ being an α-approximate isometry means that $\|\hat{U}^\dagger\hat{U} - I\| \le \alpha$, or equivalently, there exists an isometry U such that $\|U - \hat{U}\| \le \alpha$. Then,

$$\begin{aligned}
&\|f(A) - \hat{U}f(\hat{D})\hat{U}^\dagger\| \\
&\le \|f(A) - Uf(\hat{D})U^\dagger\| + \|Uf(\hat{D})(U-\hat{U})^\dagger\| + \|(U-\hat{U})f(\hat{D})\hat{U}^\dagger\| \\
&\le \|f(A) - f(U\hat{D}U^\dagger)\| + \alpha\|Uf(\hat{D})\| + \alpha\|f(\hat{D})U^\dagger\| + \alpha\|f(\hat{D})(\hat{U}-U)^\dagger\| \\
&\le L\|A - U\hat{D}U^\dagger\|\log\operatorname{rank}(U\hat{D}U^\dagger) + 3\alpha\|f(\hat{D})\| \\
&\le L\Big(\|A - \hat{U}\hat{D}\hat{U}^\dagger\| + \|\hat{U}\hat{D}(U-\hat{U})^\dagger\| + \|(U-\hat{U})\hat{D}U^\dagger\|\Big)\log\operatorname{rank}(U\hat{D}U^\dagger) + 3\alpha\|f(\hat{D})\| \\
&\le L\Big(\frac{\varepsilon}{L} + 3\alpha\|\hat{D}\|\Big)\log\operatorname{rank}(U\hat{D}U^\dagger) + 3\alpha\|f(\hat{D})\| \\
&\lesssim \varepsilon.
\end{aligned}$$

The last line follows from noticing that $\|A\| \geq \|\hat{U}\hat{D}\hat{U}^\dagger\| + \frac{\varepsilon}{L} \leq (1+\alpha)^2\|\hat{D}\|\frac{\varepsilon}{L}$, so $\|\hat{D}\| = \mathcal{O}(\|A\|)$. Further, $\|f(\hat{D})\| \leq L\|\hat{D}\|$ since $f(x) = f(x) - f(0) \leq Lx$. The log rank is folded into the polylog term in r, c, c'.

To get α-approximate isometries, we use Lemma 3.12. Using that $\bar{\pi}$ zeroes out singular values that are smaller than $\frac{\varepsilon^2}{2L^2}$, we can rewrite our expression to get the desired approximate eigenvectors:

$$\begin{aligned} R^\dagger\bar{\pi}(CC^\dagger)M\bar{\pi}(CC^\dagger)R &= \big(C^+_{\frac{\varepsilon}{\sqrt{2}L}}R\big)^\dagger\big(C^\dagger_{\frac{\varepsilon}{\sqrt{2}L}}\bar{\pi}(CC^\dagger)M\bar{\pi}(CC^\dagger)C_{\frac{\varepsilon}{\sqrt{2}L}}\big)\big(C^+_{\frac{\varepsilon}{\sqrt{2}L}}R\big) \\ &= \underbrace{\big(R^\dagger U^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}}(D^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}})^+\big)}_{\hat{U}}\underbrace{\big(D^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}}(U^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}})^\dagger\bar{\pi}(CC^\dagger)M\bar{\pi}(CC^\dagger)U^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}}D^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}}\big)}_{\hat{D}}\underbrace{\big((D^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}})^+(U^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}})^\dagger R\big)}_{\hat{U}^\dagger} \end{aligned} \tag{21}$$

By Lemma 3.12 with our values of r and c, we get that $\hat{U} := R^\dagger U^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}}(D^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}})^+$ is an $\mathcal{O}\Big(\frac{\varepsilon^3}{L^3\|A\|^3}\Big)$-approximate isometry, which is better than what we need. Here, $\hat{U} \in \mathbb{C}^{n\times s'}$, where s' is the rank of $C_{\frac{\varepsilon}{\sqrt{2}L}}$, so we can deduce that $s' \lesssim \frac{\|A\|_F^2L^2}{\varepsilon^2}$ and $s' \leq \min(r, c, n)$.

Finally, we wish to show our bound on eigenvalue estimates. Notice that, while bounding $\|f(A) - \hat{U}f(\hat{D})\hat{U}^\dagger\|$, we inadvertently proved that $\|f(A) - Uf(\hat{D})U^\dagger\| \leq \varepsilon$ for U the isometry such that $\|U - \hat{U}\| \leq \alpha$. We can compute the eigenvalues of $Uf(\hat{D})U^\dagger$: they are precisely the eigenvalues of $f(\hat{D})$, where $\hat{D}$ is defined in Eq. (21). We can simply compute this eigendecomposition and output the eigenvalues; these have the desired bounds by Weyl's inequality. Though we have not as of yet assumed that $\hat{D}$ is diagonal, we can compute its unitary eigendecomposition $U^{(\hat{D})}D^{(\hat{D})}(U^{(\hat{D})})^\dagger$; so, going by the notation in the theorem statement, we can take $D := D$ and $N := (U^{(\hat{D})})^\dagger(D^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}})^+(U^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}})^\dagger$ to complete (including the isometry $(U^{(\hat{D})})^\dagger$ in our expression for $\hat{U}$ does not change the value of α).

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 denote $U^{(C)}D^{(C)}(V^{(C)})^\dagger$ with $U^{(C)} \in \mathbb{C}^{r\times r}, D^{(C)} \in \mathbb{C}^{r\times r}, V^{(C)} \in \mathbb{C}^{c\times r}$. 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

$$\hat{D} = U^{(\hat{D})}D^{(\hat{D})}(U^{(\hat{D})})^\dagger = D^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}}\bar{\pi}((D^{(C)})^2)(U^{(C)})^\dagger MU^{(C)}\bar{\pi}((D^{(C)})^2)D^{(C)}_{\frac{\varepsilon}{\sqrt{2}L}}.$$

Then, we compute D and N. By evaluating the expression for $\hat{D}$ from left-to-right, we only need to perform matrix multiplications that (naively) take $s'r^2$ time. The only cost of c we incur is in computing the SVD of C, and further, by expressing it in this way we can see that we can take $s := r$ for s as in the theorem statement. The runtime is

$$\widetilde{\mathcal{O}}\big((r+c+t)\,\mathbf{s}(A) + (rc + r^2t)\,\mathbf{q}(A) + s'^3 + r^2s' + r^2c\big). \qquad \square$$

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.

[ADBL19] Juan Miguel Arrazola, Alain Delgado, Bhaskar Roy Bardhan, and Seth Lloyd. Quantum-inspired algorithms in practice, 2019. 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

[vAGGdW17] Joran van Apeldoorn, András Gilyén, Sander Gribling, and Ronald de Wolf. Quantum SDP-solvers: Better upper and lower bounds. In *Proceedings of the 58th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 403–414, 2017. 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. ACM, 2003. 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, 1995.

[Bel97] R. Bellman. *Introduction to matrix analysis*. Society for Industrial and Applied Mathematics, 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

[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

[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.

[CLLW19] Nai-Hui Chia, Tongyang Li, Han-Hsuan Lin, and Chunhao Wang. Quantum-inspired classical sublinear-time algorithm for solving low-rank semidefinite programming via sampling approaches. arXiv: 1901.03254, 2019.

[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. AAAI Press, 2019. 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.

[DBH19] Chen Ding, Tian-Yi Bao, and He-Liang Huang. Quantum-inspired support vector machine, 2019. arXiv: 1906.08902

[DHLT19] Yuxuan Du, Min-Hsiu Hsieh, Tongliang Liu, and Dacheng Tao. A quantum-inspired algorithm for general minimum conical hull problems, 2019. 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, 2002.

[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. ACM, 2018.

[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. *Phys. Rev.*, 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.

[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

[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, 2011.

[JLGS19] Dhawal Jethwani, François Le Gall, and Sanjay K. Singh. Quantum-inspired classical algorithms for singular value transformation. arXiv: 1910.05699, 2019.

[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. *Phys. Rev.*, 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, 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.

[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, UC 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., 2016.

[Tan18] Ewin Tang. Quantum-inspired classical algorithms for principal component analysis and supervised clustering. arXiv: 1811.00414, 2018.

[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

[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

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

A Proof sketch for Remark 3.16

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 $\geq 1-\delta$. , so we recommend going through Section 4 if the reader

Following the argument from Remark 4.4, we can find a $B := AR^\dagger \bar{t}(CC^\dagger)R$ satisfying the above property in $\widetilde{\mathcal{O}}\left(\frac{\|A\|_{\mathrm{F}}^{28}}{\|A\|^{28}\varepsilon^{22}} \log^3 \frac{1}{\delta}\right)$ (rescaling ε appropriately). Here, R and $\bar{t}(CC^\dagger)$ come from an application of Theorem 3.1 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 3.12, $R^\dagger U_{\varepsilon\|A\|/2} D^+_{\varepsilon\|A\|/2}$ 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 $CC^\dagger \approx RR^\dagger$.

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

Consider one particular column $v := [R^\dagger U_{\varepsilon\|A\|/2} D^+_{\varepsilon\|A\|/2}](\cdot, \ell)$; 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 3.10 as we do in Corollary 4.3. 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 $\tilde{u}(i) := 2\|A(i,\cdot)\|$. 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 3.10).

B Deferred proofs

Lemma 3.10 (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 $Z := u(i)v(i)/p(i)$. 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 $x := 8\phi\|u\|^2\|v\|^2\frac{1}{\varepsilon^2}$ 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 $y := 8\log\frac{1}{\delta}$ 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 $\Pr[\Re(\tilde{Z} - \langle u, v\rangle) \ge \varepsilon/\sqrt{2}] \le \delta/2$. From a union bound together with the analogous argument for the imaginary component, we have $\Pr[|\tilde{Z} - \langle u, v\rangle| \ge \varepsilon] \le \delta$ 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 3.15. *Consider* $p(x)$ *a degree-*d *polynomial of parity-*d *such that* $|p(x)| \le 1$ *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* $\max_{x\in[0,1]}|q(x)| \le 1$, $\max_{x\in[-1,1]}|q'(x)| \lesssim d^2$, $\max_{x\in[-1,1]}|\bar{q}(x)| \lesssim d^2$, *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

$$\max_{x\in[-1,1]}|p^{(k)}(x)| \lesssim \Big(\min\{d^2, \frac{d}{\sqrt{1-x^2}}\}\Big)^k \max_{x\in[-1,1]}|p(x)|, \tag{22}$$

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 first 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,

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

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

$$\begin{aligned}\max_{x\in[0,1]} |q'(x)| &\lesssim d^2 \max_{x\in[0,1]} |q(x)| \le d^2 \\ \max_{x\in[0,1]} |\bar{q}(x)| &\le \max_{y\in[0,1]} |q'(y)| \lesssim d^2 \\ \max_{x\in[0,1]} |\bar{q}'(x)| &\le \max_{y\in[0,1]} \frac{1}{2}|q''(y)| \lesssim d^4\end{aligned}$$

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: $x \le \frac{1}{2}$ and $x > \frac{1}{2}$.

$$\begin{aligned}\max_{x\in[0,\frac{1}{2}]} |q(x)| &= \max_{x\in[0,\frac{1}{2}]} \Big|\frac{p(x)}{x}\Big| = \max_{y\in[0,\frac{1}{2}]} |p'(y)| \lesssim \max_{x\in[0,\frac{1}{2}]} \frac{d}{\sqrt{1-x^2}} \max_{x\in[-1,1]} |p(x)| \lesssim d \\ \max_{x\in(\frac{1}{2},1]} |q(x)| &= \max_{x\in(\frac{1}{2},1]} \Big|\frac{p(x)}{x}\Big| \le \max_{x\in(\frac{1}{2},1]} |2p(x)| \le 2\end{aligned}$$

□

Corollary 4.14. *Given* $\mathrm{SQ}(X^T)$ *and* $\mathrm{SQ}(y)$*, with probability* $\ge 1 - \delta$*, we can output a* $\hat{b}$ *such that* $|b - \hat{b}| \le \varepsilon(1 + b)$ *and* $\mathrm{SQ}_\phi(\hat{\alpha})$ *such that* $\|\hat{\alpha} - \alpha\| \le \varepsilon\gamma\|y\|$*, where* α *and* b *come from Eq. (13). 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 $\sigma^2 := \gamma^{-1}$, and redefine $X \leftarrow X^T$ (so we have $\mathrm{SQ}(X)$ instead of $\mathrm{SQ}(X^T)$). By the block matrix inversion formula[21] we know that

$$\begin{aligned}\begin{bmatrix} 0 & \vec{1}^T \\ \vec{1} & M \end{bmatrix}^{-1} &= \begin{bmatrix} -\frac{1}{\vec{1}^T M^{-1}\vec{1}} & \frac{\vec{1}^T M^{-1}}{\vec{1}^T M^{-1}\vec{1}} \\ \frac{M^{-1}\vec{1}}{\vec{1}^T M^{-1}\vec{1}} & M^{-1} - \frac{M^{-1}\vec{1}\vec{1}^T M^{-1}}{\vec{1}^T M^{-1}\vec{1}} \end{bmatrix} \\ \begin{bmatrix} 0 & \vec{1}^T \\ \vec{1} & M \end{bmatrix}^{-1}\begin{bmatrix} 0 \\ y \end{bmatrix} &= \begin{bmatrix} \frac{\vec{1}^T M^{-1} y}{\vec{1}^T M^{-1}\vec{1}} \\ M^{-1}\Big(y - \frac{\vec{1}^T M^{-1} y}{\vec{1}^T M^{-1}\vec{1}}\vec{1}\Big) \end{bmatrix}\end{aligned}$$

[21] 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.

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

$$f(\lambda) := \frac{1}{\lambda + \sigma^2}.$$

So, by Theorem 3.1, we can find $R^\dagger \bar{f}(CC^\dagger)R$ such that $\|R^\dagger \bar{f}(CC^\dagger)R + \frac{1}{\sigma^2}I - f(X^TX)\| \le \varepsilon\sigma^{-2}$, 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 $\widetilde{\mathcal{O}}\left(\frac{K^3\kappa^5}{\varepsilon^6}\log^3\frac{1}{\delta}\right)$. We further approximate using Lemma 3.6: 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:

$$\|R^\dagger\vec{1} - r_1\| \le \varepsilon\sqrt{m}\sigma \qquad \|R^\dagger y - r_y\| \le \varepsilon\sqrt{m}\sigma \qquad |\vec{1}^\dagger y - \gamma| \le \varepsilon m \tag{23}$$

Via Lemma 3.2, we observe the following additional bounds:

$$\|M^{-1}\| \le \sigma^{-2} \qquad \|R^\dagger(\bar{f}(CC^\dagger))^{1/2}\| \le (1+\varepsilon)\sigma^{-1} \qquad \|(\bar{f}(CC^\dagger))^{1/2}\| \le \sigma^{-2} \tag{24}$$

Now, we compute what the subsequent errors are for replacing M^{-1} with $N := R^\dagger ZR + \frac{1}{\sigma^2}I$, where $Z := \bar{f}(CC^\dagger)$.

$$\begin{aligned}
\frac{\vec{1}^\dagger M^{-1} y}{\vec{1}^\dagger M^{-1}\vec{1}} &= \frac{\vec{1}^\dagger(R^\dagger ZR + \sigma^{-2}I)y \pm \|\vec{1}\|\|y\|\|R^\dagger ZR + \sigma^{-2}I - M^{-1}\|}{\vec{1}^\dagger(R^\dagger ZR + \sigma^{-2}I)\vec{1} \pm \|\vec{1}\|^2\|R^\dagger ZR + \sigma^{-2}I - M^{-1}\|} \\
&= \frac{\vec{1}^\dagger R^\dagger ZRy + \sigma^{-2}\vec{1}^\dagger y \pm \varepsilon\sigma^{-2}m}{\vec{1}^\dagger R^\dagger ZR\vec{1} + \sigma^{-2}\vec{1}^\dagger\vec{1} \pm \varepsilon\sigma^{-2}m} && \text{by SVT bound} \\
&= \frac{\vec{1}^\dagger R^\dagger Zr_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 Zr_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 Zr_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 Zr_1 \pm (\sqrt{m}(1+\varepsilon)\sigma^{-3})(\varepsilon\sigma\sqrt{m}) + \sigma^{-2}m \pm \varepsilon\sigma^{-2}m} && \text{by Eqs. (23) and (24)} \\
&= \frac{r_1^\dagger Zr_y \pm \|R\vec{1} - r_1\|\|Zr_y\| + \sigma^{-2}\gamma \pm \mathcal{O}\big(\varepsilon\sigma^{-2}m\big)}{r_1^\dagger Zr_1 \pm \|R\vec{1} - r_1\|\|Zr_1\| + \sigma^{-2}m \pm \mathcal{O}(\varepsilon\sigma^{-2}m)} \\
&= \frac{r_1^\dagger Zr_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 Zr_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. (23)} \\
&= \frac{r_1^\dagger Zr_y + \sigma^{-2}\gamma \pm \mathcal{O}\big(\varepsilon\sigma^{-2}m\big)}{r_1^\dagger Zr_1 + \sigma^{-2}m \pm \mathcal{O}(\varepsilon\sigma^{-2}m)} && \text{by Eqs. (23) and (24)} \\
&= \frac{r_1^\dagger Zr_y + \sigma^{-2}\gamma}{r_1^\dagger Zr_1 + \sigma^{-2}m}(1 \pm \mathcal{O}(\varepsilon)) \pm \mathcal{O}(\varepsilon) && \text{by } r_1^\dagger Zr_1 \ge 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

$$\begin{aligned}\|M^{-1}y - R^\dagger Z r_y + \sigma^{-2}y\| &\le \|M^{-1} - R^\dagger Z R - \sigma^{-2}I\|\|y\| + \|R^\dagger Z\|\|Ry - r_y\| \\ &\le \varepsilon\sigma^{-2}\sqrt{m} + (1+\varepsilon)\sigma^{-3}\varepsilon\sigma\sqrt{m} \\ &\lesssim \varepsilon\sigma^{-2}\sqrt{m}\end{aligned}$$

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

$$\frac{\vec{1}^\dagger M^{-1}y}{\vec{1}^\dagger M^{-1}\vec{1}} \le \frac{\|\vec{1}M^{-1/2}\|\|M^{-1/2}y\|}{\|M^{-1/2}\vec{1}\|^2} = \frac{\|M^{-1/2}y\|}{\|M^{-1/2}\vec{1}\|} \le \frac{\|X\|}{\sigma}. \tag{25}$$

With these bounds, we can conclude that (continuing to use Eqs. (23) and (24))

$$\begin{aligned}&\left\|M^{-1}\Big(y - \frac{\vec{1}^\dagger M^{-1}y}{\vec{1}^\dagger M^{-1}\vec{1}}\vec{1}\Big) - \Big(R^\dagger Z r_y + \sigma^{-2}y - \frac{r_1^\dagger Z r_y + \sigma^{-2}\gamma}{r_1^\dagger Z r_1 + \sigma^{-2}m}(R^\dagger Z r_1 + \sigma^{-2}\vec{1})\Big)\right\| \\ &\le \|M^{-1}y - R^\dagger Z r_y + \sigma^{-2}y\| + \left\|\frac{\vec{1}^\dagger M^{-1}y}{\vec{1}^\dagger M^{-1}\vec{1}}M^{-1}\vec{1} - \frac{r_1^\dagger Z r_y + \sigma^{-2}\gamma}{r_1^\dagger Z r_1 + \sigma^{-2}m}(R^\dagger Z r_1 + \sigma^{-2}\vec{1})\Big)\right\| \\ &\le \varepsilon\sigma^{-2}\sqrt{m} + \frac{\vec{1}^\dagger M^{-1}y}{\vec{1}^\dagger M^{-1}\vec{1}}\|M^{-1}\vec{1} - R^\dagger Z r_1 - \sigma^{-2}\vec{1}\| + \left|\frac{\vec{1}^\dagger M^{-1}y}{\vec{1}^\dagger M^{-1}\vec{1}} - \frac{r_1^\dagger Z r_y + \sigma^{-2}\gamma}{r_1^\dagger Z r_1 + \sigma^{-2}m}\right|\|R^\dagger Z r_1 + \sigma^{-2}\vec{1}\| \\ &\lesssim \Big(1 + \frac{\vec{1}^\dagger M^{-1}y}{\vec{1}^\dagger M^{-1}\vec{1}}\Big)\varepsilon\sigma^{-2}\sqrt{m} + \varepsilon\Big(1 + \frac{\vec{1}^\dagger M^{-1}y}{\vec{1}^\dagger M^{-1}\vec{1}}\Big)\|R^\dagger Z r_1 + \sigma^{-2}\vec{1}\| \\ &= \varepsilon\Big(1 + \frac{\vec{1}^\dagger M^{-1}y}{\vec{1}^\dagger M^{-1}\vec{1}}\Big)\Big(\sigma^{-2}\sqrt{m} + \|R^\dagger Z r_1 + \sigma^{-2}\vec{1}\|\Big) \\ &\lesssim \varepsilon\frac{\|X\|}{\sigma}\Big(\sigma^{-2}\sqrt{m} + \|M^{-1}\vec{1}\| + \|(R^\dagger Z R + \sigma^{-2}I - M^{-1})\vec{1}\| + \|R^\dagger Z r_1 - R^\dagger Z R\vec{1}\|\Big) \qquad \text{by Eq. (25)} \\ &\lesssim \varepsilon\frac{\|X\|}{\sigma}\Big(\sigma^{-2}\sqrt{m} + \sigma^{-2}\sqrt{m} + \varepsilon\sigma^{-2}\sqrt{m} + \varepsilon\sigma^{-2}\sqrt{m}\Big) \\ &\lesssim \varepsilon\frac{\|X\|}{\sigma}\sigma^{-2}\sqrt{m}\end{aligned}$$

So, by rescaling ε down by $\frac{\|X\|}{\sigma}$, it suffices to sample from

$$\hat{\alpha} := R^\dagger Z\Big(r_y - \frac{r_1^\dagger Z r_y + \sigma^{-2}\gamma}{r_1^\dagger Z r_1 + \sigma^{-2}m}r_1\Big) - \sigma^{-2}\Big(y - \frac{r_1^\dagger Z r_y + \sigma^{-2}\gamma}{r_1^\dagger Z r_1 + \sigma^{-2}m}\vec{1}\Big).$$

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 2.9 and 2.10, we can get $\mathrm{SQ}_\phi(\hat{\alpha})$ for

$$\begin{aligned}\phi &= (r+2)\left(\frac{\|X\|_{\mathrm{F}}^2}{r}\left\|Z\left(r_y - \frac{r_1^\dagger Z r_y + \sigma^{-2}\gamma}{r_1^\dagger Z r_1 + \sigma^{-2}m} r_1\right)\right\|^2 + \sigma^{-4}\left(\|y\|^2 + \left(\frac{r_1^\dagger Z r_y + \sigma^{-2}\gamma}{r_1^\dagger Z r_1 + \sigma^{-2}m}\right)^2 \|\vec{1}\|^2\right)\right)\|\hat{\alpha}\|^{-2} \\ &\lesssim \left(\|X\|_{\mathrm{F}}^2 \frac{\|X\|^2}{\sigma^2}\sigma^{-6}m + r\sigma^{-4}\frac{\|X\|^2}{\sigma^2}m\right)\|\hat{\alpha}\|^{-2} \\ &\lesssim \left(\frac{\|X\|_{\mathrm{F}}^2}{\sigma^2} + r\right)\frac{\|X\|^2}{\sigma^2}\frac{\sigma^{-4}m}{\|\hat{\alpha}\|^2}\end{aligned}$$

so $\widetilde{\mathbf{sq}}_\phi(\hat{\alpha}) = \phi\,\mathbf{sq}_\phi(\hat{\alpha})\log\frac{1}{\delta} = \mathcal{O}\left(r(\frac{\|X\|_{\mathrm{F}}^2}{\sigma^2} + r)\frac{\|X\|^2}{\sigma^2}\frac{\sigma^{-4}m}{\|\hat{\alpha}\|^2}\log\frac{1}{\delta}\right)$. □

Lemma 4.24 (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]):

$$\frac{d}{dt}e^{M(t)} = \int_0^1 e^{yM(t)}\frac{dM(t)}{dt}e^{(1-y)M(t)}dy. \tag{26}$$

Let $A \in \mathbb{C}^{n\times n}$ with $\|A\| \le 1$, we define the function $g_A(t) := \mathrm{Tr}\left(Ae^{H+t(\tilde{H}-H)}\right)$, and observe that

$$\begin{aligned}
g_A'(t) &= \frac{d}{dt}\mathrm{Tr}\left(Ae^{H+t(\tilde{H}-H)}\right) && \text{by definiton} \\
&= \mathrm{Tr}\left(A\frac{d}{dt}e^{H+t(\tilde{H}-H)}\right) && \text{by linearity of trace} \\
&= \mathrm{Tr}\left(A\int_0^1 e^{y[H+t(\tilde{H}-H)]}(\tilde{H}-H)e^{(1-y)[H+t(\tilde{H}-H)]}dy\right) && \text{by Eq. (26)} \\
&= \int_0^1 \mathrm{Tr}\left(Ae^{y[H+t(\tilde{H}-H)]}(\tilde{H}-H)e^{(1-y)[H+t(\tilde{H}-H)]}\right)dy && \text{by linearity of trace}^{22} \\
&\le \int_0^1 \left\|Ae^{y[H+t(\tilde{H}-H)]}(\tilde{H}-H)e^{(1-y)[H+t(\tilde{H}-H)]}\right\|_1 dy && \text{by trace-norm inequality} \\
&\le \int_0^1 \left\|Ae^{y[H+t(\tilde{H}-H)]}\right\|_{\frac{1}{y}}\left\|(\tilde{H}-H)e^{(1-y)[H+t(\tilde{H}-H)]}\right\|_{\frac{1}{1-y}}dy && \text{by Hölder's inequality} \\
&\le \int_0^1 \|A\|\left\|e^{y[H+t(\tilde{H}-H)]}\right\|_{\frac{1}{y}}\left\|\tilde{H}-H\right\|\left\|e^{(1-y)[H+t(\tilde{H}-H)]}\right\|_{\frac{1}{1-y}}dy && \text{by Hölder's inequality} \\
&\le \left\|\tilde{H}-H\right\|\int_0^1 \left\|e^{y[H+t(\tilde{H}-H)]}\right\|_{\frac{1}{y}}\left\|e^{(1-y)[H+t(\tilde{H}-H)]}\right\|_{\frac{1}{1-y}}dy && \text{since } \|A\| \le 1 \\
&= \left\|\tilde{H}-H\right\|\left\|e^{H+t(\tilde{H}-H)}\right\|_1. && (27)
\end{aligned}$$

[22] Note that in case $A = I$, by the cyclicity of trace, this equation implies that $\frac{d}{dt}\mathrm{Tr}(e^{H(t)}) = \mathrm{Tr}(e^{H(t)}\frac{d}{dt}H(t))$.

Now we consider $z(t) := g_I(t) = \operatorname{Tr}\left(e^{H+t(\tilde{H}-H)}\right)$. From Eq. (27) we have $z'(t) \leq \|\tilde{H}-H\|z(t)$. Using Grönwall's differential inequality, we can conclude that $z(t) \leq z(0)e^{t\|\tilde{H}-H\|}$ 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

$$\int_0^1 g'_A(t)dt \overset{(27)}{\leq} \int_0^1 \|\tilde{H}-H\|z(t)dt \leq z(0)\int_0^1 \|\tilde{H}-H\|e^{t\|\tilde{H}-H\|}dt = \operatorname{Tr}(e^H)\left(e^{\|\tilde{H}-H\|}-1\right). \quad \square$$

Lemma 5.2 (Matrix multiplication by subsampling [DKM06, Theorem 1])**.** *Suppose we are given $A \in \mathbb{C}^{n\times m}, B \in \mathbb{C}^{n\times p}, c \in \mathbb{Z}^+$ and a distribution $p \in \mathbb{R}^n$ satisfying the oversampling condition that, for some $\phi \geq 1$,*

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

Let $S \in \mathbb{R}^{c\times n}$ be sampled according to p. Then $A^\dagger S^\dagger SB$ is an unbiased estimator for $A^\dagger B$ and

$$\Pr\left[\|A^\dagger S^\dagger SB - A^\dagger B\|_{\mathrm{F}} < \sqrt{\frac{8\phi^2\ln(2/\delta)}{c}}\underbrace{\sum \|A(k,\cdot)\|\|B(k,\cdot)\|}_{\leq \|A\|_{\mathrm{F}}\|B\|_{\mathrm{F}}}\right] > 1-\delta.$$

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

$$\begin{aligned}
\mathbb{E}[(SA)^\dagger(SB)] &= c\cdot\mathbb{E}[[SA](1,\cdot)^\dagger[SB](1,\cdot)] = c\sum_{i=1}^n p(i)\frac{A(i,\cdot)^\dagger B(i,\cdot)}{cp(i)} = A^\dagger B \\
\mathbb{E}[\|A^\dagger S^\dagger SB - A^\dagger B\|_{\mathrm{F}}^2] &= \sum_{i=1}^m\sum_{j=1}^p \mathbb{E}\left[\left|[A^\dagger S^\dagger SB - A^\dagger B](i,j)\right|^2\right] \\
&= c\sum_{i=1}^m\sum_{j=1}^p \mathbb{E}\left[\left|[SA](1,i)^\dagger[SB](1,j) - [A^\dagger B](i,j)\right|^2\right] \\
&\leq c\sum_{i=1}^m\sum_{j=1}^p \mathbb{E}\left[\left|[SA](1,i)^\dagger[SB](1,j)\right|^2\right] \\
&= c\sum_{i=1}^m\sum_{j=1}^p\sum_{k=1}^n p(k)\frac{|A(k,i)|^2}{cp(k)}\frac{|B(k,j)|^2}{cp(k)} \\
&= \frac{1}{c}\sum_k \frac{1}{p(k)}\|A(k,\cdot)\|^2\|B(k,\cdot)\|^2 \\
&\leq \frac{1}{c}\sum_k \frac{\phi\sum_\ell\|A(\cdot,\ell)\|\|B(\ell,\cdot)\|}{\|A(k,\cdot)\|\|B(k,\cdot)\|}\|A(\cdot,k)\|^2\|B(k,\cdot)\|^2 \\
&= \frac{\phi}{c}\Big(\sum_k \|A(k,\cdot)\|\|B(k,\cdot)\|\Big)^2.
\end{aligned}$$

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

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

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

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 $[n]^c \to \mathbb{R}$ defined to be

$$f(i_1, i_2, \ldots, i_c) := \Big\| A^\dagger B - \sum_{s=1}^{c} \frac{1}{cp(i_s)} A(i_s, \cdot)^\dagger B(i_s, \cdot) \Big\|_{\mathrm{F}},$$

Then, by Jensen's inequality, we have

$$\mathbb{E}[f] = \mathbb{E}[\|A^\dagger S^\dagger S B - A^\dagger B\|_{\mathrm{F}}] \leq \sqrt{\mathbb{E}[\|A^\dagger S^\dagger S B - AB\|_{\mathrm{F}}^2]} \leq \sqrt{\frac{\phi}{c}} \sum_k \|A(k, \cdot)\| \|B(k, \cdot)\|.$$

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

$$\begin{aligned}
|f(\vec{i}) - f(\vec{i'})| &\leq \frac{1}{c} \Big\| \frac{1}{p(i_s)} A(i_s, \cdot)^\dagger B(i_s, \cdot) - \frac{1}{p(i'_s)} A(i'_s, \cdot)^\dagger B(i'_s, \cdot) \Big\|_{\mathrm{F}} \\
&\leq \frac{2}{c} \max_{k \in [n]} \Big\| \frac{1}{p(k)} A(k, \cdot)^\dagger B(k, \cdot) \Big\|_{\mathrm{F}} \\
&\leq \frac{2\phi}{c} \sum_{k=1}^{n} \|A(k, \cdot)\| \|B(k, \cdot)\|.
\end{aligned}$$

Now, by Lemma B.1, we conclude that

$$\Pr\left[|f - E[f]| \geq \sqrt{\frac{2\phi^2 \ln(2/\delta)}{c}} \sum_k \|A(\cdot, k)\| \|B(k, \cdot)\| \right] \leq \delta.$$

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

$$\begin{aligned}
\|A^\dagger S^\dagger S B - A^\dagger B\|_{\mathrm{F}} &\leq \mathbb{E}[\|A^\dagger S^\dagger S B - A^\dagger B\|_{\mathrm{F}}] + \sqrt{\frac{2\phi^2 \ln(2/\delta)}{c}} \sum_k \|A(\cdot, k)\| \|B(k, \cdot)\| \\
&\leq \Big(\sqrt{\frac{\phi}{c}} + \sqrt{\frac{2\phi^2 \ln(2/\delta)}{c}} \Big) \sum_k \|A(\cdot, k)\| \|B(k, \cdot)\| \\
&\leq \sqrt{\frac{8\phi^2 \ln(2/\delta)}{c}} \sum_k \|A(\cdot, k)\| \|B(k, \cdot)\|
\end{aligned}$$

□