Quantum Principal Component Analysis Only Achieves an Exponential Speedup Because of Its State Preparation Assumptions

Ewin Tang*
University of Washington, Seattle, Washington 98195, USA

(Received 21 November 2019; revised 3 June 2021; accepted 1 July 2021; published 4 August 2021)

A central roadblock to analyzing quantum algorithms on quantum states is the lack of a comparable input model for classical algorithms. Inspired by recent work of the author [E. Tang, *STOC 2019*.], we introduce such a model, where we assume we can efficiently perform ℓ^2-norm samples of input data, a natural analog to quantum algorithms that assume efficient state preparation of classical data. Though this model produces less practical algorithms than the (stronger) standard model of classical computation, it captures versions of many of the features and nuances of quantum linear algebra algorithms. With this model, we describe classical analogs to Lloyd, Mohseni, and Rebentrost's quantum algorithms for principal component analysis [S. Lloyd, M. Mohseni, and P. Rebentrost, Nat. Phys. **10**, 631 (2014).] and nearest-centroid clustering [S. Lloyd, M. Mohseni, and P. Rebentrost, Quantum algorithms for supervised and unsupervised machine learning]. Since they are only polynomially slower, these algorithms suggest that the exponential speedups of their quantum counterparts are simply an artifact of state preparation assumptions.

DOI: 10.1103/PhysRevLett.127.060503

Introduction.—Quantum machine learning (QML) has shown great promise toward yielding new exponential quantum speedups in machine learning ever since the pioneering linear systems algorithm of Harrow, Hassidim, and Lloyd [1]. Since machine-learning (ML) routines often push real-world limits of computing power, an exponential improvement to algorithm speed would allow for ML systems with vastly greater capabilities. While we have found many fast QML subroutines for ML problems since Harrow, Hassidim, and Lloyd [2–6], researchers have not been able to prove that these subroutines can be used to achieve an exponentially faster algorithm for a classical ML problem, even in the strongest input and output models [7,8]. A recent work of the author [9] suggests a surprising reason why: even our best QML algorithms, with issues with input and output models resolved, fail to achieve exponential speedups. This previous work constructs a classical algorithm matching, up to polynomial slowdown, a corresponding quantum algorithm for recommendation systems [10], which was previously believed to be one of the best candidates for an exponential speedup in machine learning [11]. In light of this result, we need to question our intuitions and reconsider one of the guiding questions of the field: when is quantum linear algebra exponentially faster than classical linear algebra?

The main challenge in answering this question is not in finding fast classical algorithms, as one might expect. Rather, it is that most QML algorithms are *incomparable* to classical algorithms since they take quantum states as input and output quantum states: we do not even know an analogous classical model of computation where we can search for similar classical algorithms [7]. The quantum recommendation system is unique in that it has a classical input, a data structure implementing quantum random access memory (QRAM), and classical output, a sample from a vector in the computational basis, allowing for rigorous comparisons to classical algorithms.

In our previous work, we suggest an idea for developing classical analogs to QML algorithms beyond this exceptional case [9]:

> *When QML algorithms are compared to classical ML algorithms in the context of finding speedups, any state preparation assumptions in the QML model should be matched with ℓ^2-norm sampling assumptions in the classical ML model.*

In this Letter, we implement this idea by introducing a new input model, "sample and query access" (SQ access), which is an ℓ^2-norm sampling assumption. We can get SQ access to data under typical state preparation assumptions, so fast classical algorithms in this model are strong barriers to their QML counterparts admitting exponential speedups. To support our contention that the resulting model is the right notion to consider, we use it to dequantize two seminal and well-known QML algorithms: quantum principal component analysis [12] and quantum supervised clustering [13]. That is, we give classical algorithms that, with classical SQ access assumptions replacing quantum state preparation assumptions, match the bounds and runtime of the corresponding quantum algorithms up to polynomial slowdown. Surprisingly, we do so using only

the classical toolkit originally applied to the recommendation systems problem, demonstrating the power of this model in analyzing QML algorithms.

From this work, we conclude that the exponential speedups of the quantum algorithms that we consider arise from strong input assumptions rather than from the "quantumness" of the algorithms since the speedups vanish when classical algorithms are given analogous assumptions. In other words, in a wide swath of settings, on *classical data*, these algorithms do not give exponential speedups. QML algorithms can still be useful for quantum data (say, states generated from a quantum system), though *a priori* it is not clear if they give a speedup in that case since the analogous "classical algorithm on quantum data" is not well-defined.

Our dequantized algorithms in the SQ access model provide the first formal evidence supporting the crucial concern about strong input and output assumptions in QML. Based on these results, we recommend exercising care when analyzing quantum linear algebra algorithms since some algorithms with polylogarithmic run-times only admit polynomial speedups. QML problems that are bounded-error quantum polynomial time complete (BQP-complete), such as sparse matrix inversion [1] and quantum Boltzmann machine training [14], still cannot be dequantized in full unless the equality of complexity classes BQP = BPP holds. However, many QML problems that are not BQP-complete have strong input model assumptions (like QRAM) and low-rank-type assumptions (which makes sense for machine learning, where high-dimensional data often exhibits low-dimensional trends). This regime is precisely when the classical approaches we outline here work, so such problems are highly susceptible to dequantization. We believe continuing to explore the capabilities and limitations of this model is a fruitful direction for QML research.

Notation:—$[n] \coloneqq \{1, \ldots, n\}$. Consider a vector $x \in \mathbb{C}^n$ and matrix $A \in \mathbb{C}^{m\times n}$. $A_{i,*}$ and $A_{*,i}$ will refer to A's ith row and column, respectively. $\|x\|$, $\|A\|_F$, and $\|A\|$ will refer to ℓ^2, Frobenius, and spectral norm, respectively. $|x\rangle \coloneqq (1/\|x\|)\sum_{i=1}^{n} x_i|i\rangle$ and $|A\rangle \coloneqq (1/\|A\|_F)\sum_{i=1}^{m}\|A_{i,*}\||i\rangle|A_{i,*}\rangle$ (where, by the previous definition, $|A_{i,*}\rangle = (1/\|A_{i,*}\|)\sum_{j=1}^{n} A_{i,j}|j\rangle$). $A = \sum_{i=1}^{\min m,n} \sigma_i u_i v_i^\dagger$ is A's singular value decomposition, where $u_i \in \mathbb{C}^m$, $v_i \in \mathbb{C}^n$, and $\sigma_i \in \mathbb{R}$, $\{u_i\}$ and $\{v_i\}$ are sets of orthonormal vectors, and $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{\min m,n} \geq 0$. $A_\sigma \coloneqq \sum_{\sigma_i \geq \sigma} \sigma_i u_i v_i^\dagger$ and $A_k \coloneqq \sum_{i=1}^{k} \sigma_i u_i v_i^\dagger$ denote low-rank approximations to A. We assume basic arithmetic operations take unit time, and $\tilde{O}(f) \coloneqq O(f \log f)$.

Dequantization model.—A typical QML algorithm works in the model where state preparation of input is efficient and a quantum state is output for measurement and postprocessing. (Here, we assume an ideal and fault-tolerant quantum computer.) In particular, given a data point $x \in \mathbb{C}^n$ as input, we assume we can prepare copies of $|x\rangle$. For m input data points as a matrix $A \in \mathbb{C}^{m\times n}$, we additionally assume efficient preparation of $|A\rangle$ to preserve relative scale. We wish to compare QML and classical ML on classical data, so state preparation usually requires access to this data and its normalization factors. This informs the classical input model for our quantum-inspired algorithms, where we assume such access, and instead of preparing states, we can prepare measurements of these states.

Definition:—We have $O(T)$-time *sample and query access* to $x \in \mathbb{C}^n$ [notated $SQ(x)$] if, in $O(T)$ time, we can query an index $i \in [n]$ for its entry x_i, produce an independent measurement of $|x\rangle$ in the computational basis, and query for $\|x\|$. If we can only query for an estimate of the squared norm $\bar{x} \in (1 \pm \nu)\|x\|^2$, then we denote this by $\mathrm{SQ}^\nu(x)$. For $A \in \mathbb{C}^{m\times n}$, sample and query access to A [notated $\mathrm{SQ}(A)$] is $\mathrm{SQ}(A_{1,*}, \ldots, A_{n,*})$ along with $\mathrm{SQ}(\tilde{A})$, where $\tilde{A}$ is the vector of row norms, i.e., $\tilde{A}_i \coloneqq \|A_{i,*}\|$.

SQ access will be our classical analog to quantum state preparation. As we noted previously [9], we should be able to assume that classical analogs can efficiently measure input states: QML algorithms should not rely on fast state preparation as the "source" of an exponential speedup. The algorithm itself should create the speedup.

For typical instantiations of state preparation oracles on classical input, we can get efficient SQ access to input. For example, given input in QRAM [15], a strong proposed generalization of classical RAM that supports state preparation, we can get log-dimension-time SQ access to input [[9], Proposition 3.2]. Similarly, sparse and close-to-uniform vectors can be prepared efficiently and correspondingly admit efficient SQ access [16]. So, in usual QML settings, SQ assumptions are easier to satisfy than state preparation assumptions.

This leads to a model based on SQ access that we codify with the informal definition of "dequantization." We say we "dequantize" a quantum protocol $\mathcal{S}$: $O(T)$-time state preparation of $|\phi_1\rangle, \ldots, |\phi_c\rangle \to |\psi\rangle$ if we describe a classical algorithm of the form $\mathcal{C}_\mathcal{S}$: $O(T)$-time $SQ(\phi_1, \ldots, \phi_c) \to SQ^\nu(\psi)$ with similar guarantees to $\mathcal{S}$ up to polynomial slowdown. This is the sense in which we dequantized the quantum recommendation system in prior work [10]. In the rest of this Letter, we will dequantize two quantum algorithms, giving detailed sketches of the classical algorithms and leaving proofs of correctness to the Supplemental Material [16]. These algorithms are applications of three protocols from our previous work [9] rephrased in our access model.

Nearest-centroid classification.—Lloyd, Mohseni, and Rebentrost's quantum algorithm for clustering estimates the distance of a data point to the centroid of a cluster of points [13]. The paper claims [28] that this quantum algorithm gives an exponential speedup over classical algorithms. We dequantize Lloyd *et al.*'s quantum supervised clustering algorithm [13] with only quadratic slowdown. Though classical algorithms by Aaronson [7] and

Wiebe *et al.* [[29], Sec. 7] dequantize this algorithm for close-to-uniform and sparse input data, respectively, we are the first to give a general classical algorithm for this problem.

Problem 1 (Centroid distance):—Suppose we are given access to $V \in \mathbb{C}^{n\times d}$ and $u \in \mathbb{C}^d$. Estimate $\|u - (1/n)\vec{1}V\|^2$ to ε additive error with probability $\geq 1-\delta$.

Note that we are treating vectors as rows, with $\vec{1}$ the vector of ones. Let $\bar{u} \coloneqq (u/\|u\|)$ and let $\bar{V}$ be V, normalized so all rows have unit norm. Both classical and quantum algorithms argue about $M \in \mathbb{R}^{(n+1)\times d}$ and $w \in \mathbb{R}^{n+1}$ instead of u and V, where

$$M \coloneqq \begin{bmatrix} \bar{u} \\ \frac{1}{\sqrt{n}}\bar{V} \end{bmatrix} \quad \text{and} \quad w \coloneqq \begin{bmatrix} \|u\| & -\frac{1}{\sqrt{n}}\tilde{V} \end{bmatrix}.$$

Because $wM = u - [(1/n)\vec{1}V]$, we wish to estimate $\|wM\|^2 = wMM^\dagger w^\dagger$. Let $Z \coloneqq \|w\|^2 = \|u\|^2 + (1/n)\|V\|_F^2$ be an "average norm" parameter appearing in our algorithms.

Theorem 1 (Quantum nearest centroid): Suppose that, in $O(T)$ time, we can (1) determine $\|u\|$ and $\|V\|_F$ or (2) prepare a state $|u\rangle, |V_1\rangle, \ldots, |V_n\rangle$, or $|\tilde{V}\rangle$. Then we can solve Problem 1 in $O\big(T(Z/\varepsilon)\log(1/\delta)\big)$ time.

The quantum algorithm proceeds by constructing the states $|M\rangle$ and $|w\rangle$ and then performing a swap test to get $|wM\rangle$. The swap test succeeds with probability $(1/Z)wMM^\dagger w^\dagger$, so we can run amplitude amplification to get an estimate up to ε error with $O\big((1/\varepsilon)\log(1/\delta)\big)$ overhead.

Dequantizing this algorithm is simply a matter of dequantizing the swap test, which is done in Algorithm 1. Here, $Q(y)$ is "query access" to y, which supports querying y's entries in $O(1)$ time but not querying samples or norms.

Algorithm 1. Inner product estimation.

Input: $O(T)$-time $SQ^\nu(x) \in \mathbb{C}^n$, $Q(y) \in \mathbb{C}^n$
Output: an estimate of $\langle x|y\rangle$
1: Let $s = 54(1/\varepsilon^2)\log(2/\delta)$.
2: Collect measurements $i_1, \ldots, i_s$ from $|x\rangle$.
3: Let $z_j = x_{i_j}^\dagger y_{i_j}(\|x\|^2/|x_{i_j}|^2)$ for all $j \in [s]$. $\triangleright \mathbb{E}[z_j] = \langle x|y\rangle$
4: Separate the z_j's into $6\log(2/\delta)$ buckets of size $9/\varepsilon^2$, and take the mean of each bucket.
5: Output the (component-wise) median of the means.

From a simple analysis of the random variable z_i's, we get the following result.

Proposition 1:—For $x, y \in \mathbb{C}^n$, given $SQ^\nu(x)$ and $Q(y)$, Algorithm 1 outputs an estimate of $\langle x|y\rangle$ to $(\varepsilon + \nu + \varepsilon\nu)\|x\|\|y\|$ error with probability $\geq 1-\delta$ in time $O\big((T/\varepsilon^2)\log(1/\delta)\big)$.

For this protocol, quantum algorithms can achieve a quadratic speedup via amplitude estimation (but no more, by unstructured search lower bounds [30]). To apply this to nearest centroid, we write $wMM^\dagger w^\dagger$ as an inner product of tensors $\langle a|b\rangle$, where

$$a \coloneqq \sum_{i=1}^{d}\sum_{j=1}^{n+1}\sum_{k=1}^{n+1} M_{ji}\|M_{k,*}\||i\rangle|j\rangle|k\rangle = M \otimes \tilde{M};$$

$$b \coloneqq \sum_{i=1}^{d}\sum_{j=1}^{n+1}\sum_{k=1}^{n+1} \frac{w_j^\dagger w_k M_{ki}}{\|M_{k,*}\|}|i\rangle|j\rangle|k\rangle.$$

Then, we show we have SQ access to one of the tensors (a). With this, we see that the quadratic speedup from amplitude amplification is the only speedup that quantum nearest centroid achieves.

Theorem 2 (Classical nearest centroid): Suppose we are given $O(T)$-time $\mathrm{SQ}(V) \in \mathbb{C}^{n\times d}$ and $\mathrm{SQ}(u) \in \mathbb{C}^d$. Then one can output a solution to Problem 1 in $O\big(T(Z^2/\varepsilon^2)\log(1/\delta)\big)$ time.

Principal component analysis.—We now dequantize Lloyd, Mohseni, and Rebentrost's quantum principal component analysis (QPCA) algorithm [12], an influential early example of QML [31,32]. While the paper describes a more general strategy for Hamiltonian simulation of density matrices, their central claim is an exponential speedup in an immediate application: producing quantum states corresponding to the top principal components of a low-rank dataset [12].

The setup for the problem is as follows: suppose we are given a matrix $A \in \mathbb{R}^{n\times d}$ whose rows correspond to data in a dataset. We will find the principal eigenvectors and eigenvalues of $A^\dagger A$; when A is a mean zero dataset, this corresponds to the top principal components.

Problem 2 (Principal component analysis):—Suppose we are given access to $A \in \mathbb{C}^{n\times d}$ with singular values σ_i and right singular vectors v_i. Further suppose we are given σ, k, and η with the guarantee that, for all $i \in [k]$, $\sigma_i \geq \sigma$ and $\sigma_i^2 - \sigma_{i+1}^2 \geq \eta\|A\|_F^2$. With probability $\geq 1-\delta$, output estimates $\hat{\sigma}_1^2, \ldots, \hat{\sigma}_k^2$ and $\hat{v}_1, \ldots, \hat{v}_k$ satisfying $|\hat{\sigma}_i^2 - \sigma_i^2| \leq \varepsilon_\sigma\|A\|_F^2$ and $\|\hat{v}_i - v_i\| \leq \varepsilon_v$ for all $i \in [k]$.

Denote $\|A\|_F^2/\sigma^2$ by K. Lloyd *et al.* get the following.

Theorem 3: Given $\|A\|_F$ and the ability to prepare copies of $|A\rangle$ in $O(T)$ time, a quantum algorithm can output the desired estimates for Problem 2 $\hat{\sigma}_1^2, \ldots, \hat{\sigma}_k^2$ and $|\hat{v}_1\rangle, \ldots, |\hat{v}_k\rangle$ in $\tilde{O}(TK\min(\varepsilon_\sigma, \delta)^{-3})$ time.

Later results [[10,33], Theorems 5.2, 27] improve the run-time here to $\tilde{O}(TK\varepsilon_\sigma^{-1}\mathrm{polylog}(nd/\delta))$ when A is given in QRAM. We will compare to the original QPCA result.

To dequantize QPCA, we use a similar high-level idea to that of the quantum-inspired recommendation system [9]. We begin by using a low-rank approximation algorithm, Algorithm 2, to output a description of approximate top singular values and vectors.

Algorithm 2 finds the large singular vectors of A by reducing its dimension down to W, whose singular value decomposition we can compute quickly. Then, S, $\hat{U}$, $\hat{\Sigma}$

define approximate large singular vectors $\hat{V} \coloneqq S^\dagger \hat{U} \hat{\Sigma}^{-1}$. The full set of guarantees on the output of Algorithm 2 are in the Supplemental Material [16], but in brief, for the right setting of parameters, the columns of $\hat{V}$ and the diagonal entries of $\hat{\Sigma}$ satisfy the desired constraints for our $\hat{v}_i$'s and $\hat{\sigma}_i$'s in Problem 2. The $\hat{\sigma}_i$'s are output explicitly, but the $\hat{v}_i$'s are described implicitly: $\hat{v}_i = S^\dagger \hat{U}_{*,i}/\hat{\sigma}_i$. We have $O(T)$-time $\mathrm{SQ}(S)$ because all rows are normalized, and rows of S are simply rows of A. Thus, sampling from $\tilde{S}$ is a uniform sample from $[q]$ and sampling from $S_{i,*}$ is sampling from a row of A. $\hat{U}_{*,i}$ is an explicit vector, so in essence, we need SQ access to a linear combination of vectors, each of which we have SQ access to.

Algorithm 2. Low-rank approximation [35].

Input: $O(T)$-time $\mathrm{SQ}(A) \in \mathbb{R}^{m\times n}$, σ, ε, δ
Output: $\mathrm{SQ}(S) \in \mathbb{C}^{\ell\times n}$, $Q(\hat{U}) \in \mathbb{C}^{q\times \ell}$, $Q(\hat{\Sigma}) \in \mathbb{C}^{\ell\times\ell}$
1: Set $K = \|A\|_F^2/\sigma^2$ and $q = \Theta((K^4/\varepsilon^2)\log(1/\delta))$.
2: Sample rows $i_1, \ldots, i_q$ from $\tilde{A}$ and define $S \in \mathbb{R}^{q\times n}$ such that $S_{r,*} \coloneqq A_{i_r,*}(\|A\|_F/\sqrt{q}\|A_{i_r,*}\|)$.
3: Sample columns $j_1, \ldots, j_q$ from $\mathcal{F}$, where $\mathcal{F}$ denotes the distribution given by sampling a uniform $r \sim [q]$, then sampling c from S_r.
4: Let $W \in \mathbb{C}^{q\times q}$ be the normalized submatrix $W_{*,c} \coloneqq (S_{*,j_c}/q\mathcal{F}(j_c))$.
5: Compute the left singular vectors of W $\hat{u}^{(1)}, \ldots, \hat{u}^{(\ell)}$ that correspond to singular values $\hat{\sigma}^{(1)}, \ldots, \hat{\sigma}^{(\ell)}$ larger than σ.
6: Output $\mathrm{SQ}(S)$, $\hat{U} \in \mathbb{R}^{q\times\ell}$ the matrix with columns $\hat{u}^{(i)}$, and $\hat{\Sigma} \in \mathbb{R}^{\ell\times\ell}$ the diagonal matrix with entries $\hat{\sigma}^{(i)}$.

Algorithm 3 does exactly this: it uses rejection sampling to dequantize the swap test over a subset of qubits [getting $|Vw\rangle$ via $\langle V|(|w\rangle \otimes I)$].

Algorithm 3. Matrix-vector SQ access.

Input: $O(T)$-time $\mathrm{SQ}(V^\dagger) \in \mathbb{C}^{k\times n}$, $Q(w) \in \mathbb{C}^k$
Output: $\mathrm{SQ}^\nu(Vw)$
1: **function** RejectionSample($SQ(V^\dagger), Q(w)$)
2: Sample $i \in [k]$ proportional to $|w_i|^2\|V_{*,i}\|^2$ by manually calculating all k probabilities.
3: Sample $s \in [n]$ from $V_{*,i}$ using $\mathrm{SQ}(V^\dagger)$.
4: Compute $r_s = (Vw)_s^2/(k\sum_{j=1}^k (V_{sj}w_j)^2)$ (after querying for w_j and V_{sj} for all $j \in [k]$).
5: Output s with probability r_s (success); otherwise, output $\varnothing$ (failure).
6: **end function**
7: Query: output $(Vw)_s$.
8: Sample: run RejectionSample until success (outputting s) or $kC(V,w)\log(1/\delta)$ failures (outputting $\varnothing$).
9: Norm(ν): Let p be the fraction of successes from running rejectionsample $(k/\nu^2)C(V,w)\log(1/\delta)$ times; output $pk\sum_{i=1}^k |w_i|^2\|V_{*,i}\|^2$.

Proposition 2:—For $V \in \mathbb{C}^{n\times k}$, $w \in \mathbb{C}^k$, given $\mathrm{SQ}(V^\dagger)$ and $Q(w)$, Algorithm 3 simulates $\mathrm{SQ}^\nu(Vw)$ where the time to query is $O(Tk)$, sample is $O(Tk^2C(V,w)\log(1/\delta))$, and query norm is $O(Tk^2C(V,w)(1/\nu^2)\log(1/\delta))$. Here, δ is the desired failure probability and $C(V,w) = \sum\|w_iV_{*,i}\|^2/\|Vw\|^2$.

In general, $C(V,w)$ may be arbitrarily large, but in this application it is $O(K)$. Quantum algorithms achieve a speedup here when k is large and $C(V,w)$ is small, such as when V is a high-dimensional unitary, confirming our intuition that unitary operations are hard to simulate classically.

Altogether, we get our desired result.

Theorem 4: Given $O(T)$-time $\mathrm{SQ}(A) \in \mathbb{C}^{n\times d}$, with $\varepsilon_\sigma, \varepsilon_v, \delta \in (0, 0.01)$, there is an algorithm that output the desired estimates for Problem 2 $\hat{\sigma}_1, \ldots, \hat{\sigma}_k$ and $O[T(K^9/\varepsilon^4)\log^3(k/\delta)]$-time $\mathrm{SQ}^{0.01}(\hat{v}_1, \ldots, \hat{v}_k)$ in $O((K^{12}/\varepsilon^6)\log^3(k/\delta) + T(K^8/\varepsilon^4)\log^2(k/\delta))$ time, where $\varepsilon = \min(0.1\varepsilon_\sigma K^{1.5}, \varepsilon_v^2\eta, \frac{1}{4}K^{-1/2})$.

Under the nondegeneracy condition $\eta \le \frac{1}{4}K^{-1/2}$, this run-time is $\tilde{O}(T(K^{12}/\varepsilon_\sigma^6\varepsilon_v^{12})\log^3(1/\delta))$. While the classical run-time depends on ε_v, note that a quantum algorithm must *also* incur this error term to learn about v_i from copies of $|v_i\rangle$. For example, computing entries or expectations of observables of v_i given copies of $|v_i\rangle$ requires $\mathrm{poly}(1/\varepsilon_v)$ or $\mathrm{poly}(n)$ time.

Discussion.—We have introduced the SQ access assumption as a classical analog to the QML state preparation assumption and demonstrated two examples where, in this classical model, we can dequantize QML algorithms with ease. We now discuss the implications of this work with respect to the related literature.

A natural question is of this work's relation to classical literature: does this work improve on classical algorithms for linear algebra in any regime? The answer may be no for a subtle but fundamental reason: recall that our main idea is to introduce an input model *strong enough* to give classical versions of QML while being *weak enough* to extend to settings like QRAM, where classical computers can only access the input in very limited ways. In particular, the SQ access model that we study is *weaker* than the typical input model used for classical sketching algorithms [36–38]. $O(T)$-time algorithms in the quantum-inspired access model are $\tilde{O}(\mathrm{nnz} + T)$-time algorithms in the usual RAM model (where "nnz" is the number of nonzero entries of the input), but not vice versa: typical sketching algorithms can exploit better data structures provided they only take $O(\mathrm{nnz})$ time (e.g., oblivious sketches), whereas the quantum-inspired model can only use the QRAM data structure. The crucial insight of this work is that some algorithms (such as Algorithm 2 of Frieze *et al.* [35]) generalize to the weaker quantum-inspired model. Our algorithms give exponential speedups in the quantum-inspired setting, but since the model is weaker, one might expect that they perform worse in typical settings for classical computation (see [34]). These model considerations also explain why we use Frieze *et al.* [35]: to our

knowledge, this algorithm is the only one from the classical literature that naturally generalizes to the SQ input model.

The closest analog to these results and techniques is a work by Van den Nest on probabilistic quantum simulation [39], which describes a notion of "computationally tractable" states that corresponds to our notion of SQ access for vectors. With this notion, the author describes special types of circuits on computationally tractable states where weak simulation is possible, using variants of Propositions 1 and 2. However, Van den Nest's work does not have a version of Algorithm 2, since this technique only runs quickly on low-rank matrices, making it ineffective on generic quantum circuits. We exploit this low-rank structure for efficient quantum simulation of a small but practically relevant class of circuits: quantum linear algebra on data with a low-rank structure. So, our techniques used for supervised clustering are within the scope of Van den Nest's work, whereas our techniques for principal component analysis are new to this line of work.

These techniques are not new to quantum simulation in general. Others have considered applying randomized numerical linear algebra to quantum simulation [40] but have not made the connection toward dequantizing quantum algorithms, especially in large generality. Low-rank approximation is crucial for tensor network simulations of quantum systems [41,42], where simulation can be done efficiently provided the input is, say, a matrix product state with a low tensor rank. In this context, low-rank approximation is often performed exactly and only on a subset of the space, instead of approximately done on the full state, as is done here. This reflects the fact that tensor network algorithms assume that the system is reasonably approximated by a tensor network and aims to work well in practice, whereas our "dequantized" algorithms must work on a broader class of input and prioritizes provable guarantees in an abstract computational model over real-world performance. Nevertheless, some of these dequantized algorithms might be able to be matched by tensor network contraction techniques, when the input has a low tensor rank. See the Supplemental Material for further discussion of this comparison [16].

Since this work, numerous follow-ups have cemented the significance of the SQ access model introduced here [43–46]. In particular, a recent work [45] essentially dequantizes the singular value transformation framework of Gilyen *et al.* [47] when input is given in QRAM. These works use fundamentally the same techniques to dequantize a wide swath of low-rank quantum machine learning—an exciting step forward in understanding QML.

Thanks to Ronald de Wolf for giving the initial idea to look at QPCA. Thanks to Nathan Wiebe for helpful comments on this document. Thanks to Daniel Liang and Patrick Rall for their help fleshing out these ideas and reviewing a draft of this document. Thanks to Scott Aaronson for helpful discussions. This material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE-1762114.

*Corresponding author.
ewint@cs.washington.edu; ewintang.com

[1] A. W. Harrow, A. Hassidim, and S. Lloyd, Quantum Algorithm for Linear Systems of Equations, Phys. Rev. Lett. **103**, 150502 (2009).
[2] P. Rebentrost, M. Mohseni, and S. Lloyd, Quantum Support Vector Machine for Big Data Classification, Phys. Rev. Lett. **113**, 130503 (2014).
[3] N. Wiebe, D. Braun, and S. Lloyd, Quantum Algorithm for Data Fitting, Phys. Rev. Lett. **109**, 050505 (2012).
[4] S. Lloyd, S. Garnerone, and P. Zanardi, Quantum algorithms for topological and geometric analysis of data, Nat. Commun. **7**, 10138 (2016).
[5] Z. Zhao, J. K. Fitzsimons, and J. F. Fitzsimons, Quantum-assisted Gaussian process regression, Phys. Rev. A **99**, 052331 (2019).
[6] F. G. Brandao and K. M. Svore, Quantum speed-ups for solving semidefinite programs, in *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)* (IEEE, New York, 2017), https://doi.org/10.1109/focs.2017.45.
[7] S. Aaronson, Read the fine print, Nat. Phys. **11**, 291 (2015).
[8] A. M. Childs, Equation solving by simulation, Nat. Phys. **5**, 861 (2009).
[9] E. Tang, A quantum-inspired classical algorithm for recommendation systems, in *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing—STOC 2019* (ACM Press, New York, 2019), https://doi.org/10.1145/3313276.3316310.
[10] I. Kerenidis and A. Prakash, Quantum recommendation systems, in *8th Innovations in Theoretical Computer Science Conference (ITCS 2017), LIPIcs, Vol. 67* (Schloss Dagstuhl, Germany, 2017), pp. 49:1–49:21, https://doi.org/10.4230/LIPIcs.ITCS.2017.49.
[11] J. Preskill, Quantum computing in the NISQ era and beyond, Quantum **2**, 79 (2018).
[12] S. Lloyd, M. Mohseni, and P. Rebentrost, Quantum principal component analysis, Nat. Phys. **10**, 631 (2014).
[13] S. Lloyd, M. Mohseni, and P. Rebentrost, Quantum algorithms for supervised and unsupervised machine learning, arXiv:1307.0411.
[14] M. Kieferová and N. Wiebe, Tomography and generative training with quantum Boltzmann machines, Phys. Rev. A **96**, 062327 (2017).
[15] V. Giovannetti, S. Lloyd, and L. Maccone, Quantum Random Access Memory, Phys. Rev. Lett. **100**, 160501 (2008).
[16] See Supplemental Material, which includes Refs. [17–27], at http://link.aps.org/supplemental/10.1103/PhysRevLett.127.060503 for details on when sample and query access is possible, discussion on the relation of this work to the tensor networks literature, and full proofs for the results stated here.

[17] A. Prakash, Quantum algorithms for linear algebra and machine learning., Ph.D. thesis, UC Berkeley, 2014.
[18] L. Grover and T. Rudolph, Creating superpositions that correspond to efficiently integrable probability distributions, arXiv:quant-ph/0208112.
[19] F. Verstraete and J. I. Cirac, Matrix product states represent ground states faithfully, Phys. Rev. B **73**, 094423 (2006).
[20] S. R. White, Density Matrix Formulation for Quantum Renormalization Groups, Phys. Rev. Lett. **69**, 2863 (1992).
[21] G. Vidal, Efficient Classical Simulation of Slightly Entangled Quantum Computations, Phys. Rev. Lett. **91**, 147902 (2003).
[22] G. Vidal, Efficient Simulation of One-Dimensional Quantum Many-Body Systems, Phys. Rev. Lett. **93**, 040502 (2004).
[23] A. Cichocki, N. Lee, I. Oseledets, A.-H. Phan, Q. Zhao, and D. P. Mandic, Tensor networks for dimensionality reduction and large-scale optimization: part 1 low-rank tensor decompositions, Found. Trends Mach. Learn. **9**, 249 (2016).
[24] J. C. Bridgeman and C. T. Chubb, Hand-waving and interpretive dance: an introductory course on tensor networks, J. Phys. A **50**, 223001 (2017).
[25] J. Eisert, Computational Difficulty of Global Variations in the Density Matrix Renormalization Group, Phys. Rev. Lett. **97**, 260501 (2006).
[26] Z. Landau, U. Vazirani, and T. Vidick, A polynomial time algorithm for the ground state of one-dimensional gapped local Hamiltonians, Nat. Phys. **11**, 566 (2015).
[27] W. Hoeffding, Probability inequalities for sums of bounded random variables, in *The Collected Works of Wassily Hoeffding*, edited by N. I. Fisher and P. K. Sen (Springer, New York, NY, 1994), pp. 409–426.
[28] We were not able to verify the quantum algorithm (namely, the Hamiltonian simulation for preparing $|\phi\rangle$) as stated. For our purposes, we can make the minor additional assumption of efficient state preparation access to $|\phi\rangle$, which makes correctness obvious. When we refer to the quantum algorithm in this Letter, we mean this version of it.
[29] N. Wiebe, A. Kapoor, and K. M. Svore, Quantum algorithms for nearest-neighbor methods for supervised and unsupervised learning, Quantum Inf. Comput. **15**, 316 (2015).
[30] C. H. Bennett, E. Bernstein, G. Brassard, and U. Vazirani, Strengths and weaknesses of quantum computing, SIAM J. Comput. **26**, 1510 (1997).
[31] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, Quantum machine learning, Nature (London) **549**, 195 (2017).
[32] C. Ciliberto, M. Herbster, A. D. Ialongo, M. Pontil, A. Rocchetto, S. Severini, and L. Wossnig, Quantum machine learning: a classical perspective, Proc. R. Soc. A **474**, 20170551 (2018).
[33] S. Chakraborty, A. Gilyén, and S. Jeffery, The power of block-encoded matrix powers: improved regression techniques via faster Hamiltonian simulation, in *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019), LIPIcs* (Schloss Dagstuhl, Germany, 2019), https://doi.org/10.4230/LIPIcs.ICALP.2019.33.
[34] J. M. Arrazola, A. Delgado, B. R. Bardhan, and S. Lloyd, Quantum-inspired algorithms in practice, Quantum **4**, 307 (2020).
[35] A. Frieze, R. Kannan, and S. Vempala, Fast Monte-Carlo algorithms for finding low-rank approximations, J. ACM (JACM) **51**, 1025 (2004).
[36] M. W. Mahoney, Randomized algorithms for matrices and data, Found. Trends Mach. Learn. **3**, 123 (2011).
[37] D. P. Woodruff, Sketching as a tool for numerical linear algebra, Found. Trends Theor. Comput. Sci. **10**, 1 (2014).
[38] R. Kannan and S. Vempala, Randomized algorithms in numerical linear algebra, Acta Numer. **26**, 95 (2017).
[39] M. Van Den Nest, Simulating quantum computers with probabilistic methods, Quantum Inf. Comput. **11**, 784 (2011).
[40] A. Rudi, L. Wossnig, C. Ciliberto, A. Rocchetto, M. Pontil, and S. Severini, Approximating Hamiltonian dynamics with the Nyström method, Quantum **4**, 234 (2020).
[41] U. Schollwöck, The density-matrix renormalization group in the age of matrix product states, Ann. Phys. (Amsterdam) **326**, 96 (2011).
[42] R. Orús, A practical introduction to tensor networks: Matrix product states and projected entangled pair states, Ann. Phys. (Amsterdam) **349**, 117 (2014).
[43] N.-H. Chia, A. Gilyén, H.-H. Lin, S. Lloyd, E. Tang, and C. Wang, Quantum-inspired algorithms for solving low-rank linear equation systems with logarithmic dependence on the dimension, in *31st International Symposium on Algorithms and Computation (ISAAC 2020), LIPIcs* (Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2020), https://doi.org/10.4230/LIPIcs.ISAAC.2020.47.
[44] N.-H. Chia, T. Li, H.-H. Lin, and C. Wang, Quantum-inspired sublinear algorithm for solving low-rank semidefinite programming, in *45th International Symposium on Mathematical Foundations of Computer Science (MFCS 2020), LIPIcs* (Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2020), https://doi.org/10.4230/LIPIcs.MFCS.2020.23.
[45] N.-H. Chia, A. Gilyén, T. Li, H.-H. Lin, E. Tang, and C. Wang, Sampling-based sublinear low-rank matrix arithmetic framework for dequantizing quantum machine learning, in *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing—STOC 2020* (ACM Press, New York, 2020), https://doi.org/10.1145/3357713.3384314.
[46] D. Jethwani, F. L. Gall, and S. K. Singh, Quantum-inspired classical algorithms for singular value transformation, in *45th International Symposium on Mathematical Foundations of Computer Science (MFCS 2020), LIPIcs* (Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2020), https://doi.org/10.4230/LIPIcs.MFCS.2020.53.
[47] A. Gilyén, Y. Su, G. H. Low, and N. Wiebe, Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics, in *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing—STOC 2019* (ACM Press, New York, 2019), https://doi.org/10.1145/3313276.3316366.