Quantum SDP Solvers: Large Speed-Ups, Optimality, and Applications to Quantum Learning

Fernando G. S. L. Brandão
Institute of Quantum Information and Matter, California Institute of Technology, USA
fgslbrandao@gmail.com

Amir Kalev
Joint Center for Quantum Information and Computer Science, University of Maryland, USA
amirk@umd.edu

Tongyang Li
Joint Center for Quantum Information and Computer Science, University of Maryland, USA
tongyang@cs.umd.edu

Cedric Yen-Yu Lin
Joint Center for Quantum Information and Computer Science, University of Maryland, USA
cedriclin37@gmail.com

Krysta M. Svore
Station Q, Quantum Architectures and Computation Group, Microsoft Research, USA
ksvore@microsoft.com

Xiaodi Wu
Joint Center for Quantum Information and Computer Science, University of Maryland, USA
xwu@cs.umd.edu

Abstract

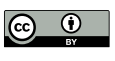
We give two new quantum algorithms for solving semidefinite programs (SDPs) providing quantum speed-ups. We consider SDP instances with m constraint matrices, each of dimension n, rank at most r, and sparsity s. The first algorithm assumes an input model where one is given access to an oracle to the entries of the matrices at unit cost. We show that it has run time $\tilde{O}(s^2(\sqrt{m}\epsilon^{-10} + \sqrt{n}\epsilon^{-12}))$, with ϵ the error of the solution. This gives an optimal dependence in terms of m, n and quadratic improvement over previous quantum algorithms (when $m \approx n$). The second algorithm assumes a fully quantum input model in which the input matrices are given as quantum states. We show that its run time is $\tilde{O}(\sqrt{m} + \mathrm{poly}(r)) \cdot \mathrm{poly}(\log m, \log n, B, \epsilon^{-1})$, with B an upper bound on the trace-norm of all input matrices. In particular the complexity depends only polylogarithmically in n and polynomially in r.

We apply the second SDP solver to learn a good description of a quantum state with respect to a set of measurements: Given m measurements and a supply of copies of an unknown state ρ with rank at most r, we show we can find in time $\sqrt{m} \cdot \mathrm{poly}(\log m, \log n, r, \epsilon^{-1})$ a description of the state as a quantum circuit preparing a density matrix which has the same expectation values as ρ on the m measurements, up to error ϵ. The density matrix obtained is an approximation to the maximum entropy state consistent with the measurement data considered in Jaynes' principle from statistical mechanics.

As in previous work, we obtain our algorithm by "quantizing" classical SDP solvers based on the matrix multiplicative weight update method. One of our main technical contributions is a quantum Gibbs state sampler for low-rank Hamiltonians, given quantum states encoding these Hamiltonians, with a poly-logarithmic dependence on its dimension, which is based on ideas developed in quantum principal component analysis. We also develop a "fast" quantum OR lemma with a quadratic improvement in gate complexity over the construction of Harrow et al. [14]. We believe both techniques might be of independent interest.

2012 ACM Subject Classification Theory of computation → Semidefinite programming; Theory of computation → Quantum query complexity; Theory of computation → Convex optimization

Keywords and phrases quantum algorithms, semidefinite program, convex optimization



46th International Colloquium on Automata, Languages, and Programming (ICALP 2019).
Editors: Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi;
Article No. 27; pp. 27:1–27:14

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Digital Object Identifier 10.4230/LIPIcs.ICALP.2019.27

Category Track A: Algorithms, Complexity and Games

Related Version A full version of the paper is available at `https://arxiv.org/abs/1710.02581`.

Acknowledgements We thank Scott Aaronson, Joran van Apeldoorn, András Gilyén, Cupjin Huang, and anonymous reviewers for helpful discussions. We are also grateful to Joran van Apeldoorn and András Gilyén for sharing a working draft of [4] with us. FB was supported by NSF. CYL and AK were supported by the Department of Defense. TL was supported by NSF CCF-1526380. XW was supported by NSF grants CCF-1755800 and CCF-1816695, and also by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Quantum Algorithms Teams program.

1 Introduction

Motivation. Semidefinite programming has been a central topic in the study of mathematical optimization, theoretical computer science, and operations research in the last decades. It has become an important tool for designing efficient optimization and approximation algorithms. The power of semidefinite programs (SDPs) lies in their generality (that extends the better-known linear programs (LPs)) and the fact that they admit polynomial-time solvers.

It is natural to ask whether quantum computers can have advantage in solving this important optimization problem. In Ref. [9], Brandão and Svore provided an affirmative answer, giving a quantum algorithm with worst-case running time $\tilde{O}(\sqrt{mn}s^2(R\tilde{R}/\varepsilon)^{32})$ [1], where n and s are the dimension and row sparsity of the input matrices, respectively, m the number of constraints, ϵ the accuracy of the solution, and $R, \tilde{R}$ upper bounds on the norm of the optimal primal and dual solutions. This is a *polynomial* speed-up in m and n comparing to the two state-of-the-art classical SDP-solvers [20, 7] (with complexity $\tilde{O}(m(m^2 + n^\omega + mns)\operatorname{poly}\log(R/\epsilon))$ [20], where ω is the exponent of matrix multiplication, and $\tilde{O}(mns(R\tilde{R}/\varepsilon)^4 + ns\left(R\tilde{R}/\varepsilon\right)^7)$ [7]), and beating the classical lower bound of $\Omega(m+n)$ [9]. The follow-up work by van Apeldoorn et al. [5] improved the running time giving a quantum SDP solver with complexity $\tilde{O}(\sqrt{mn}s^2(R\tilde{R}/\epsilon)^8)$. In terms of limitations, Ref. [9] proved a quantum lower bound $\Omega(\sqrt{m}+\sqrt{n})$ when $R, \tilde{R}, s, \epsilon$ are constants; stronger lower bounds can be proven if R and/or $\tilde{R}$ scale with m and n [5]. We note all these results are shown in an input model in which there is an oracle for the entry of each of the input matrices (see Oracle 1.1 below for a formal definition).

In this paper, we investigate quantum algorithms for SDPs (i.e., quantum SDP solvers) further in the following two perspectives: (1) the best dependence of parameters, especially the dimension n and the number of constraints m; (2) whether there is any reasonable alternative input model for quantum SDP solvers and what is its associated complexity. To that end, let us first formulate the precise SDP instance in our discussion.

The SDP approximate feasibility problem. We will work with the SDP approximate feasibility problem formulated as follows: Given an $\epsilon > 0$, m real numbers $a_1, \ldots, a_m \in \mathbb{R}$, and Hermitian $n \times n$ matrices $A_1, \ldots, A_m$ where $-I \preceq A_i \preceq I, \forall\, j \in [m]$, define the convex

[1] $\tilde{O}$ hides factors that are polynomial in $\log m$ and $\log n$.

region $\mathcal{S}_\epsilon$ as all X such that

$$\begin{aligned} \mathrm{Tr}(A_i X) &\leq a_i + \epsilon \quad \forall\, i \in [m]; \\ X &\succeq 0;\ \mathrm{Tr}[X] = 1. \end{aligned} \tag{1.1}$$

For approximate feasibility testing, it is required that either (1) If $\mathcal{S}_0 = \emptyset$, output fail; or (2) If $\mathcal{S}_\epsilon \neq \emptyset$, output an $X \in \mathcal{S}_\epsilon$. Throughout the paper, we denote by n the the dimension of the matrices, m the number of constraints, and ϵ the (additive) error of the solution. For Hermitian matrices A and B, we denote $A \preceq B$ if $B - A$ is positive semidefinite, and $A \succeq B$ if $A - B$ is positive semidefinite. We denote I_n to be the $n \times n$ identity matrix.

There are a few reasons that guarantee our choice of approximate SDP feasibility problem do not lose generality: (1) first, it is a routine[2] to reduce general optimization SDP problems to the feasibility problem; (2) second, for general feasible solution $X \succeq 0$ with width bound $\mathrm{Tr}(X) \leq R$, there is a procedure[3] to derive an equivalent SDP feasibility instance with variable $\hat{X}$ s.t. $\mathrm{Tr}(\hat{X}) = 1$. Note, however, the change of ϵ to ϵ/R in this conversion. Also note one can use an approximate feasibility solver to find a strictly feasible solution, by changing ϵ to $\epsilon/R\tilde{R}$ (see Lemma 18 of Ref. [9]). The benefit of our choice of (1.1) is its simplicity in presentation, which provides a better intuition behind our techniques and an easy adoption of our SDP solver in learning quantum states. In contrast to Ref. [5], we do not need to formulate the dual program of Eq. (1.1) since our techniques do not rely on it. We will elaborate more on these points in Section 1.4.

1.1 Quantum SDP solvers with optimal dependence on m and n

Existing quantum SDP solvers [9, 5] have close-to-optimal dependence on some key parameters but poor dependence on others. Seeking optimal parameter dependence has been an important problem in the development of classical SDP solvers and has inspired many new techniques. It is thus well motivated to investigate the optimal parameter dependence in the quantum setting. Our first contribution is the construction of a quantum SDP solver with the optimal dependence on m and n in the (plain) input model as used by [9, 5], given as follows:

▶ **Oracle 1.1** (Plain model for A_j)**.** *A quantum oracle, denoted $\mathcal{P}_A$, such that given the indices $j \in [m]$, $k \in [n]$ and $l \in [s]$, computes a bit string representation of the l-th non-zero element of the k-th row of A_j, i.e. the oracle performs the following map:*

$$|j, k, l, z\rangle \to |j, k, l, z \oplus (A_j)_{k f_{jk}(l)}\rangle, \tag{1.2}$$

with $f_{jk} : [r] \to [N]$ a function (parametrized by the matrix index j and the row index k) which given $l \in [s]$ computes the column index of the l-th nonzero entry.

Before we move on to our main result, we will define two primitives which will appear in our quantum SDP solvers. Our main result will also be written in terms of the cost for each primitive.

---

[2] To see why this is the case, for any general SDP problem, one can guess a candidate value (e.g., c_0) for the objective function (e.g., $\mathrm{Tr}(CX)$ and assume one wants to maximize $\mathrm{Tr}(CX)$) and convert it into a constraint (e.g., $\mathrm{Tr}(CX) \geq c_0$). Hence one ends up with a feasibility problem and the candidate value c_0 can then be found via binary search with $O(\log(1/\epsilon))$ overhead when $\mathrm{Tr}(CX) \in [-1, 1]$.

[3] The procedure goes as follows: (a) scale down every constraint by a factor R and let $X' = X/R$ (thus $\mathrm{Tr}(X') \leq 1$) (b) let $\hat{X} = \mathrm{diag}\{X, w\}$ be a block-diagonal matrix with X in the upper-left corner and a scaler w in the bottom-right corner. It is easy to see that $\mathrm{Tr}(\hat{X}) = 1 \iff \mathrm{Tr}(X) \leq 1$.

▶ **Definition 1** (trace estimation). *Assume that we have an s-sparse $n \times n$ Hermitian matrix H with $\|H\| \leq \Gamma$ and a density matrix ρ. Then we define $\mathcal{S}_{\mathrm{Tr}}(s, \Gamma, \epsilon)$ and $\mathcal{T}_{\mathrm{Tr}}(s, \Gamma, \epsilon)$ as the number of copies of ρ and the time complexity (in terms of oracle call and number of gates) of using the plain model (Oracle 1.1) for H, respectively, such that one can compute $\mathrm{Tr}[H\rho]$ with additive error ϵ with success probability at least $2/3$.*

▶ **Definition 2** (Gibbs sampling). *Assume that we have an s-sparse $n \times n$ Hermitian matrix H with $\|H\| \leq \Gamma$. Then we define $\mathcal{T}_{Gibbs}(s, \Gamma, \epsilon)$ as the complexity of preparing the Gibbs state $\frac{e^{-H}}{\mathrm{Tr}[e^{-H}]}$ with additive error ϵ using the plain model (Oracle 1.1) for H.*

Our main result is as follows.

▶ **Theorem 3.** *In the plain input model (Oracle 1.1), for any $0 < \epsilon < 1$, there is a quantum SDP solver for the feasibility problem (1.1) using $\frac{s}{\epsilon^4}\tilde{O}\big(\mathcal{S}_{\mathrm{Tr}}\big(\frac{s}{\epsilon^2}, \frac{1}{\epsilon}, \epsilon\big)\mathcal{T}_{Gibbs}\big(\frac{s}{\epsilon^2}, \frac{1}{\epsilon}, \epsilon\big) + \sqrt{m}\mathcal{T}_{\mathrm{Tr}}\big(\frac{s}{\epsilon^2}, \frac{1}{\epsilon}, \epsilon\big)\big)$ quantum gates and queries to Oracle 1.1, where s is the sparsity of $A_j, j \in [m]$.*

When combined with specific instantiation of these primitives (i.e., in our case, we directly make use of results on $\mathcal{S}_{\mathrm{Tr}}(s, \Gamma, \epsilon)$ and $\mathcal{T}_{\mathrm{Tr}}(s, \Gamma, \epsilon)$ from Ref. [9], and results on $\mathcal{T}_{\mathrm{Gibbs}}(s, \Gamma, \epsilon)$ from Ref. [24]), we end up with the following concrete parameters:

▶ **Corollary 4.** *In the plain input model (Oracle 1.1), for any $0 < \epsilon < 1$, there is a quantum SDP solver for the feasibility problem (1.1) using $\tilde{O}(s^2(\frac{\sqrt{m}}{\epsilon^{10}} + \frac{\sqrt{n}}{\epsilon^{12}}))$ quantum gates and queries to Oracle 1.1, where s is the sparsity of $A_j, j \in [m]$.*

Comparing to prior art, our main contribution is to decouple the dependence on m and n, which used to be $O(\sqrt{mn})$ and now becomes $O(\sqrt{m} + \sqrt{n})$. Note that the $(\sqrt{m} + \sqrt{n})$ dependence is optimal due to the quantum lower bound proven in Ref. [9].

▶ Remark 1.5. Even though our result achieves the optimal dependence on m and n, it is nontrivial to obtain quantum speed-ups by directly applying our quantum SDP solvers to SDP instances from classical combinatorial problems. The major obstacle is the poly-dependence on $1/\epsilon$, whereas, for interesting SDP instances such as Max-Cut, $1/\epsilon$ is linear in n. In fact, the general framework of the classical Arora-Kale SDP solver also suffers from the poly-dependence on $1/\epsilon$ and cannot be applied directly either. Instead, one needs to specialize the design of SDP solvers for each instance to achieve better time complexity.

Extending this idea to quantum seems challenging. One difficulty is that known classical approaches require explicit information of intermediate states, which requires $\Omega(n)$ time and space even to store. It is not clear how one can directly adapt classical approaches on intermediate states when stored as amplitudes in quantum states, which is the case for our current SDP solvers. It seems to us that a resolution of the problem might require an independent tool beyond the scope of this paper. We view this as an important direction for future work.

However, our quantum SDP solvers are sufficient for instances with mild $1/\epsilon$, which are natural in the context of quantum information, such as learnability of the quantum state problem (elaborated in Section 1.5) as well as examples in [4]. For those cases, we do establish a quantum speed-up as any classical algorithm needs at least linear time in n and/or m.

1.2 Quantum SDP solvers with quantum inputs

Given the optimality of the algorithm presented before (in terms of m and n), a natural question is to ask about the existence of alternative input models, *which can be justified for specific applications, and at the same time allows more efficient quantum SDP solvers.* This

is certainly a challenging question, but we can get inspiration from the application of SDPs in quantum complexity theory (e.g., Refs. [16, 12]) and quantum information (e.g., Refs. [1, 2]). In these settings, input matrices of SDP instances, with dimension 2^ℓ, are typically quantum states and/or measurements generated by poly(ℓ)-size circuits on ℓ qubits. For the sake of these applications, it might be reasonable to equip quantum SDP solvers with the ability to leverage these circuit information, rather than merely allowing access to the entries of the input matrices.

In this paper, we propose a *truly* quantum input model in which we can construct quantum SDP solvers with running time only *poly-logarithmic* in the dimension. We note that such proposal was mentioned in an earlier version of Ref. [9], whose precise mathematical form and construction of quantum SDP solvers were unfortunately incorrect, and later removed. Note that since we consider a non-standard input model in this section, our results are incomparable to those in the plain input model. We argue for the relevance of our quantum input model, by considering an applications of the framework to the problem of learning quantum states in Section 1.5.

Quantum input model. Consider a specific setting in which we are given decompositions of each A_j: $A_j = A_j^+ - A_j^-$, where $A_j^+, A_j^- \succeq 0$. (For instance, a natural choice is to let A_j^+ (resp. A_j^-) be the positive (resp. negative) part of A.)

▶ **Oracle 1.2** (Oracle for traces of A_j)**.** *A quantum oracle (unitary), denoted O_{Tr} (and its inverse $O_{\mathrm{Tr}}^\dagger$), such that for any $j \in [m]$,*

$$O_{\mathrm{Tr}}|j\rangle|0\rangle|0\rangle = |j\rangle|\operatorname{Tr}[A_j^+]\rangle|\operatorname{Tr}[A_j^-]\rangle, \tag{1.3}$$

where the real values $\operatorname{Tr}[A_j^+]$ and $\operatorname{Tr}[A_j^-]$ are encoded into their binary representations.

▶ **Oracle 1.3** (Oracle for preparing A_j)**.** *A quantum oracle (unitary), denoted O (and its inverse $O^\dagger$), which acts on $\mathbb{C}^m \otimes (\mathbb{C}^n \otimes \mathbb{C}^n) \otimes (\mathbb{C}^n \otimes \mathbb{C}^n)$ such that for any $j \in [m]$,*

$$O|j\rangle|0\rangle|0\rangle = |j\rangle|\psi_j^+\rangle|\psi_j^-\rangle, \tag{1.4}$$

where $|\psi_j^+\rangle, |\psi_j^-\rangle \in \mathbb{C}^n \otimes \mathbb{C}^n$ are any purifications of $\frac{A_j^+}{\operatorname{Tr}[A_j^+]}, \frac{A_j^-}{\operatorname{Tr}[A_j^-]}$, respectively.

▶ **Oracle 1.4** (Oracle for a_j)**.** *A quantum oracle (unitary), denoted O_a (and its inverse $O_a^\dagger$), such that for any $j \in [m]$,*

$$O_a|j\rangle|0\rangle = |j\rangle|a_j\rangle, \tag{1.5}$$

where the real value a_j is encoded into its binary representation.

Throughout the paper, let us assume that A_j has rank at most r for all $j \in [m]$ and $\operatorname{Tr}[A_j^+] + \operatorname{Tr}[A_j^-] \leq B$. The parameter B is therefore an upper bound to the trace-norm of all input matrices which we assume is given as an input of the problem. Similar to the plain input model, we will define the same two primitives and their associated costs in the quantum input model.

▶ **Definition 6** (trace estimation)**.** *We define $\mathcal{S}_{\mathrm{Tr}}(B, \epsilon)$ and $\mathcal{T}_{\mathrm{Tr}}(B, \epsilon)$ as the sample complexity of a state $\rho \in \mathbb{C}^{n\times n}$ and the gate complexity of using the quantum input oracles (Oracle 1.2, Oracle 1.3, Oracle 1.4), respectively, for the fastest quantum algorithm that distinguishes with success probability at least $1 - O(1/m)$ whether for a fixed $j \in [m]$, $\operatorname{Tr}(A_j\rho) > a_j + \epsilon$ or $\operatorname{Tr}(A_j\rho) \leq a_j$.*

▶ **Definition 7** (Gibbs sampling). *Assume that $K = K^+ - K^-$, where $K^\pm = \sum_{j\in S} c_j A_j^\pm$, $c_j > 0$, $S \subseteq [m]$ and $|S| \leq \Phi$, and that K^+, K^- have rank at most r_K. Moreover, assume that $\mathrm{Tr}(K^+) + \mathrm{Tr}(K^-) \leq B_K$ for some B_K. Then we define $\mathcal{T}_{Gibbs}(r_K, \Phi, B_K, \epsilon)$ as the gate complexity of preparing the Gibbs state $\rho_G = \exp(-K)/\mathrm{Tr}(\exp(-K))$ to ϵ precision in trace distance using Oracle 1.2, Oracle 1.3, and Oracle 1.4.*

Our main result in the quantum input model is as follows.

▶ **Theorem 8.** *For any $\epsilon > 0$, there is a quantum algorithm for the approximate feasibility of the SDP using at most $\frac{1}{\epsilon^2}\tilde{O}\big(\mathcal{S}_{\mathrm{Tr}}(B,\epsilon)\mathcal{T}_{Gibbs}\big(\frac{r}{\epsilon^2}, \frac{1}{\epsilon^2}, \frac{B}{\epsilon}, \epsilon\big) + \sqrt{m}\mathcal{T}_{\mathrm{Tr}}(B,\epsilon)\big)$ quantum gates and queries to Oracle 1.2, Oracle 1.3, and Oracle 1.4.*

In contrast to the plain model setting, the quantum input model is a completely new setting so that we have to construct these two primitive by ourselves. In particular, we give a construction of trace estimation with $\mathcal{S}_{\mathrm{Tr}}(B,\epsilon) = \mathcal{T}_{\mathrm{Tr}}(B,\epsilon) = O(B^2 \log m/\epsilon^2)$ and a construction of Gibbs sampling $\mathcal{T}_{\mathrm{Gibbs}}(r_K, \Phi, B_K, \epsilon) = O(\Phi \cdot \mathrm{poly}(\log n, r_K, B_K, \epsilon^{-1}))$[4]. As a result,

▶ **Corollary 9.** *For any $\epsilon > 0$, there is a quantum algorithm for the feasibility of the SDP using at most $(\sqrt{m} + \mathrm{poly}(r)) \cdot \mathrm{poly}(\log m, \log n, B, \epsilon^{-1})$ quantum gates and queries to Oracle 1.2, Oracle 1.3, and Oracle 1.4.*

We also show the square-root dependence on m is also optimal by establishing the following result:

▶ **Theorem 10** (lower bound on Corollary 9). *There exists an SDP feasibility testing problem such that $B, r, \epsilon = \Theta(1)$, and solving the problem requires $\Omega(\sqrt{m})$ calls to Oracle 1.2, Oracle 1.3, and Oracle 1.4.*

Comparison between the plain model and the quantum input model. In the quantum input model (Oracle 1.2, Oracle 1.3, and Oracle 1.4), our quantum SDP solver has a *poly-logarithmic* dependence on n (but polynomial in r) and a *square-root* dependence on m, while in the plain input model (Oracle 1.1), the dependence on n needs to be $\Omega(\sqrt{n})$ [9]. It is also worth mentioning that our quantum SDP solver in Corollary 9 does *not* assume the *sparsity* of A_i's, which are crucial for the quantum SDP solvers with the plain model (such as Corollary 4 and Refs. [9, 5]). This is because the quantum input models provide an alternative way to address the technical difficulty that was resolved by the sparsity condition (namely efficient algorithms for Hamiltonian evolution associated with the input matrices of the SDP).

Comparison between quantum and classical input models. The poly-logarithmic dependence on n in Corollary 9 is intriguing and suggests that quantum computers might offer exponential speed-ups for some SDP instances. However one has to be cautious as the input model we consider is inherently quantum, so it is incomparable to classical SDP solvers. As suggested to us by Aram Harrow (personal communication), we could consider a classical setting in which we get as input all inner products between all eigenvectors of the input matrices. Then in that case one could solve the problem classically in time $\mathrm{poly}(r, m, 1/\epsilon)$ (essentially using Jaynes' principle which will be discussed in Section 1.5 to reduce the problem

[4] The construction details are given in Lemma 10 and 12 in the full version of our paper [8].

to a SDP of dimension poly(r)). We have not formalized this approach, and there seems to be some technical problems doing so when the input matrices have close-by eigenvalues. However Harrow's observation shows the importance of justifying the input model in terms of natural applications to argue for the relevance of the run time obtained. We present one application of it in Section 1.5; more applications are given in Ref. [4].

Furthermore, several quantum-inspired classical algorithms were recently proposed originated from Tang [25]. Such classical algorithms assume the following sampling access:

▶ **Definition 11** (Sampling access). *Let $A \in \mathbb{C}^{n\times n}$ be a matrix. We say that we have the* sampling access *to A if we can*

1. *sample a row index $i \in [n]$ of A where the probability of row i being chosen is $\frac{\|A_{i\cdot}\|^2}{\|A\|_F^2}$, and*[5]
2. *for all $i \in [n]$, sample an index $j \in [n]$ where the probability of j being chosen is $\frac{|A_{ij}|^2}{\|A_{i\cdot}\|^2}$*

with time and query complexity $O(\mathrm{poly}(\log n))$ for each sampling.

In particular, we notice that Ref. [10] recently gave a classical SDP solver for (1.1) with complexity $O(m\cdot\mathrm{poly}(\log n, r, \epsilon^{-1}))$, given the above sampling access to $A_1, \ldots, A_m$. We point out that this result is incomparable to Corollary 9 because the sampling access (Definition 11) and our quantum state model (Oracle 1.2, Oracle 1.3, and Oracle 1.4) are incomparable. Nevertheless, it reminds us that under various input models, the speedup of quantum SDP solvers (compared to their classical counterparts) can also vary.

1.3 Related works on quantum SDP solvers

Previous quantum SDP solvers [9, 5] focus on the plain input model. A major contribution of ours is to improve the dependence $O(\sqrt{mn})$ to $O(\sqrt{m}+\sqrt{n})$ (ignoring dependence on other parameters) which is optimal given the lower bound $\Omega(\sqrt{m}+\sqrt{n})$ in [9]. To that end, we have also made a few technical contributions, including bringing in a new SDP solving framework and a fast version of quantum OR lemma, which will be elaborated in Section 1.4.

The quantum input model was briefly mentioned in an earlier version of [9]. The construction of quantum SDP solvers under the quantum input model therein was unfortunately incorrect. We provide the first rigorous mathematical formulation of the quantum input model and its justification in the context of learning quantum states (see Section 1.5). We also provide a construction of quantum SDP solvers in this model with a rigorous analysis. Moreover, we construct the first Gibbs state sampler with quantum inputs.

Subsequent to a previous version of this paper, an independent interesting result by van Apeldoorn and Gilyén [4] has improved the complexity of trace-estimation and Gibbs sampling. After a personal communication [26] introducing our fast version of the quantum OR lemma, the authors of Ref. [4] observed independently that the application of the quantum OR lemma [14] can be applied to decouple the dependence of m and n. As a result, Ref. [4] improved the complexity of Corollary 4 to $\tilde{O}(s(\frac{\sqrt{m}}{\epsilon^4}+\frac{\sqrt{n}}{\epsilon^5}))$ in the quantum operator model, a stronger input model than the plain one proposed by Ref. [4]. Using novel techniques, it also has improved the complexity of Corollary 9 to $\tilde{O}(\frac{B\sqrt{m}}{\epsilon^4}+\frac{B^{3.5}}{\epsilon^{7.5}})$ in the quantum input model. Note there is no explicit dependence on the rank r, which is an important advance (though it can be argued that rank r is implicitly included in the parameter B).

[5] Here $\|A\|_F$ is the Frobenius norm of A and $\|A_{i\cdot}\|$ is the ℓ_2 norm of the i^{th} row of A.

1.4 Techniques

At a high level, and in similarity to Refs. [9, 5], our quantum SDP solver can be seen as a "quantized" version of classical SDP solvers based on the matrix multiplicative weight update (MMWU) method [6]. In particular, we will leverage quantum Gibbs samplers as the main source of quantum speed-ups. In Refs. [9, 5], quantum Gibbs samplers with quadratic speed-ups (e.g., [24, 11]) have been exploited to replace the classical Gibbs state calculation step in [6]. Because the number of iterations in MMWU is poly-logarithmic in terms of the input size, the use of quantum Gibbs samplers, together with a few other tricks, leads to the overall quadratic quantum speed-up.

However, there are a few key differences (our major technical contributions) which are essential for our improvements.

Zero-sum game approach for MMW. Our quantum SDP solvers do not follow the primal-dual approach in Arora-Kale's SDP solver [7] which is the classical counterpart of previous quantum SDP solvers [9, 5]. Instead, we follow a zero-sum game framework to solve SDP feasibility problems, which is also based on the MMWU method. This framework has appeared in the classical literature (e.g., [15]) and has already been used to in semidefinite programs of relevance in quantum complexity theory (e.g., [27, 12, 19]). Let us briefly describe how the zero-sum game framework works when solving the SDP feasibility problem (1.1).

Assume there are two players. Player 1 wants to provide a feasible $X \in \mathcal{S}_\epsilon$. Player 2, on the other side, wants to find any violation of any proposed X, which can be formulated as follows.

▶ **Oracle 1.5** (Search for violation). *Inputs a density matrix X, outputs an $i \in [m]$ such that $\mathrm{Tr}(A_i X) > a_i + \epsilon$. If no such i exists, output "FEASIBLE".*

If the original problem is feasible, there exists a feasible point X_0 (provided by Player 1) such that there is no violation of X_0 that can be found by Player 2 (i.e., Oracle 1.5). This actually refers to an *equilibrium* point of the zero-sum game, which can also be approximated by the matrix multiplicative weight update method [6].

We argue that there are a few advantages of adopting this framework. One prominent example is its simplicity, which perhaps provides more intuition than the primal-dual approach. Together with our choice of the approximate feasibility problem, our presentation is simple both conceptually and technically (indeed, the simplicity of this framework has led to the development of the fast quantum OR lemma, another main technical contribution of ours.) Another example is that the zero-sum game approach does not make use of the dual program of SDPs and thus there is no dependence on the size of any dual solution. The game approach also admits an intuitive application of our SDP solvers to learning quantum states Section 1.5, which coincides with the approach adopted by [19] in a similar context.

One might wonder whether the simplicity of this framework will restrict the efficiency of SDP solvers. As indicated by the independent work of van Apeldoorn and Gilyén [4] which has achieved the same complexity of quantum SDP solvers following both the primal-dual approach and the zero-sum approach, we conclude that it is not the case at least up to our current knowledge.

Fast quantum OR lemma. We now outlines what is the main idea to find a solution to Oracle 1.5 efficiently. Roughly speaking, the idea behind previous quantum SDP solvers [9, 5] when applied to this context was to generate a new copy of a quantum state X for each time one would query the expectation value of one of the input matrices on it. The cost of

generating X (i.e., Gibbs sampling) is $O(\sqrt{n})$ (ignoring the dependence on other parameters) and one can use a Grover-search-like approach to test for m constraints with $O(\sqrt{m})$ iterations. The resultant cost is then $O(\sqrt{mn})$. Our key observation is to leverage the quantum OR lemma [14] to detect a single violation with only a single copy of X.

At a high level, given a single copy of any state ρ and m projections $\Lambda_1, \ldots, \Lambda_m$, the quantum OR lemma describes a procedure to distinguish between the case that $\exists\, i \in [m]$ s.t. $\mathrm{Tr}[\rho\Lambda_i]$ is very large, or $\frac{1}{m}\sum_{i=1}^{m} \mathrm{Tr}[\rho\Lambda_i]$ is very small. It is not hard to see that with some gap-amplification step and a search-to-decision reduction, the above procedure will output a violation i^* if any. By using quantum OR lemma, one can already decouple the cost of generating X and the number of iterations in violation-detection.

Unfortunately, Ref. [14] has only been focusing on the use of a single copy of ρ, while its gate complexity is $O(m)$ for m projections. To optimize the gate complexity, we develop the following fast implementation of the quantum OR lemma with gate complexity $O(\sqrt{m})$, using ideas from the fast amplification technique in [22]. Overall, this leads to a complexity of $O(\sqrt{m} + \sqrt{n})$.

▶ **Lemma 12.** *Let $\Lambda_1, \ldots, \Lambda_m$ be projections, and fix parameters $0 < \varepsilon \leq 1/2$ and $\varphi, \xi > 0$. Let ρ be a state such that either $\exists\, j \in [m]$ $\mathrm{Tr}[\rho\Lambda_j] \geq 1 - \varepsilon$, or $\frac{1}{m}\sum_{j=1}^{m} \mathrm{Tr}[\rho\Lambda_j] \leq \varphi$. There is a test using one copy of ρ and $O(\xi^{-1}\sqrt{m}(p + \mathrm{poly}(\log m)))$ operations such that: in the former case, accepts with probability at least $(1-\varepsilon)^2/4 - \xi$; in the latter case, accepts with probability at most $3\varphi m + \xi$.*

The dependence on m is also tight, as one can easily embed Grover search into this problem.

Gibbs sampler with quantum inputs. To work with the quantum input model, as our main technical contribution, we construct the first quantum Gibbs sampler of low-rank Hamiltonians when given Oracles 1.2 and 1.3:

▶ **Theorem 13.** *Assume the $n \times n$ matrix $K = K^+ - K^-$ and K^+, K^- are PSD matrices with rank at most r_K and $\mathrm{Tr}[K^+] + \mathrm{Tr}[K^-] \leq B$. Given quantum oracles that prepare copies of $\rho^+ = K^+/\mathrm{Tr}(K^+)$, $\rho^- = K^-/\mathrm{Tr}(K^-)$ and estimates of $\mathrm{Tr}(K^+)$, $\mathrm{Tr}(K^-)$, there is a quantum Gibbs sampler that prepares the Gibbs state $\rho_G = \exp(-K)/\mathrm{Tr}(\exp(-K))$ to precision ϵ in trace distance, using $\mathrm{poly}(\log n, r_K, B, \epsilon^{-1})$ quantum gates.*

Our quantum Gibbs sampler has a poly-logarithmic dependence on n and polynomial dependence on the maximum rank of the input matrices, while in the plain input model the dependence of n is $\Theta(\sqrt{n})$ [24, 11]. Our construction deviates significantly from [24, 11]. Because of the existence of copies of ρ^+ and ρ^-, we rely on efficient Hamiltonian simulation techniques developed in quantum principle component analysis (PCA) [21] and its follow-up work in [18]. As a result, we can also get rid of the sparsity assumption which is crucial for evoking results about efficient Hamiltonian simulation into the Gibbs sampling used in [24, 11].

1.5 Application: Efficient learnability of quantum states

Problem description. Given many realizations of an experiment producing a quantum state with density matrix ρ, learning an approximate description of ρ is a fundamental task in quantum information and experimental physics. It refers to *quantum state tomography*, which has been widely used to identify quantum systems. However, to tomograph an ℓ-qubit state ρ (with dimension $n = 2^\ell$) , the optimal procedure [23, 13] requires n^2 number of copies of ρ, which is impractical already for relatively small ℓ.

An interesting alternative is to find a description of the unknown quantum state ρ which approximates $\mathrm{Tr}[\rho E_i]$ up to error ϵ for a specific collection of POVM elements $E_1, \ldots, E_m$, where $I \succeq E_i \succeq 0$ and $E_i \in \mathbb{C}^{n\times n}, \forall i \in [m]$. This is an old problem, dating back at least to the work of Jaynes on statistical mechanics in the 50ies. Jaynes' principle [17] (also known as the principle of maximum entropy) gives a general form for the solution of the problem above. It shows that there is always a state of the form

$$\frac{\exp\left(\sum_i \lambda_i E_i\right)}{\mathrm{Tr}\left(\exp\left(\sum_i \lambda_i E_i\right)\right)}, \tag{1.6}$$

which has the same expectation values on the E_i's as the original state ρ, where the λ_i's are real numbers. In words, there is always a Gibbs state with Hamiltonian given by a linear combination of the E_i's which gives the same expectation values as the state described by ρ. Therefore one can solve the learning problem by finding the right λ_i's (or finding a quantum circuit creating the state in Eq. (1.6)).

Applying quantum SDP solvers. By formulating the learning problem in terms of the SDP feasibility problem (with each A_i replaced by E_i) where one looks for a trace unit PSD σ matching the measurement statistics, i.e., $\mathrm{Tr}(\sigma E_i) \approx \mathrm{Tr}(\rho E_i), \forall i \in [m]$, we observe that our quantum SDP solvers actually provides a solution to the learning problem with associated speed-ups on m and n.

In fact, our algorithm also outputs each of the λ_i's (one can show that $\mathrm{poly}(\log(mn))/\varepsilon^2$ non-zero of them suffices for a solution with error ε), as well as a circuit description of the Gibbs state in Eq. (1.6) achieving the same expectation values as ρ up to error ε. (This is mainly because the similarity between the matrix multiplicative update method and Jaynes' principle.) In this sense our result can be seen as an *algorithmically* version of Jaynes' principle. We note that a similar idea was adopted by [19] in learning quantum states, although for a totally different purpose (namely proving lower bounds on the size of SDP approximations to constraint satisfaction problems).

It is worthwhile noting that our quantum SDP solvers when applied in this context will output a description of the state ρ in the form of Eq. (1.6) which has the same expectation values as ρ on measurements $E_1, \ldots, E_m$ up to error ϵ. This is slightly different from directly outputting estimates of $\mathrm{Tr}(E_i\rho)$ for each $i \in [m]$, which by itself will take $\Omega(m)$ time.

Relevance of quantum input model. More importantly, we argue that our quantum input model is *relevant* in this setting for low-rank measurements E_i's. Since all $E_i \succeq 0$ by definition, we can consider the following (slightly simplified version of) oracles:

Oracle 1.2 for traces of E_i: A unitary O_{Tr} such that for any $i \in [m]$, $O_{\mathrm{Tr}}|i\rangle|0\rangle = |i\rangle|\,\mathrm{Tr}[E_i]\rangle$.

Oracle 1.3 for preparing E_i: A unitary O such that for any $i \in [m]$, $O|i\rangle\langle i| \otimes |0\rangle\langle 0|O^\dagger = |i\rangle\langle i| \otimes |\psi_i\rangle\langle\psi_i|$, where $|\psi_i\rangle\langle\psi_i|$ is any purification of $E_i/\,\mathrm{Tr}[E_i]$.

We now show how one can implement this oracle in the case where each E_i is a low rank projector and we have an efficient (with $\mathrm{poly}\log(n)$ many gates) implementation of the measurement. Let the rank of E_i's bounded by r and suppose the measurement operators E_i's are of the form

$$E_i = V_i P_i V_i^\dagger \tag{1.7}$$

for polynomial (in $\log(n)$) time circuits V_i, and projectors P_i of the form

$$P_i := \sum_{i=1}^{r_i} |i\rangle\langle i| \tag{1.8}$$

with $|i\rangle$ the computational basis and $r_i \leq r$. Then for Oracle 1.2 we just need to output the r_i's. Oracle 1.3 can be implemented efficiently (in time $r \operatorname{poly}\log(n)$) by first creating a maximally entangled state between the subspace spanned by P_i and a purification and applying V_i to one half of it. In more detail, consider the following purification of $E_i/\operatorname{Tr}(E_i)$:

$$|\psi_i\rangle := \frac{1}{\sqrt{r_i}} \sum_{i=1}^{r_i} (V_i \otimes I)|i,i\rangle \tag{1.9}$$

This can be constructed first by preparing the state $\frac{1}{\sqrt{r_i}} \sum_{i=1}^{r_i} |i,i\rangle$ in time r_i and then applying $V_i \otimes I$ to it (which can be done in time $\operatorname{poly}\log(n)$).

Efficient learning for low rank measurements. By applying our SDP solver in the quantum input model, we obtain that

▶ **Theorem 14.** *For any $\epsilon > 0$, there is a quantum procedure that outputs a description of the state ρ in the form of Eq. (1.6) (namely the λ_i's parameters) using at most $\operatorname{poly}(\log m, \log n, r, \epsilon^{-1})$ copies of ρ and at most $\sqrt{m} \cdot \operatorname{poly}(\log m, \log n, r, \epsilon^{-1})$ quantum gates and queries to Oracle 1.2 and Oracle 1.3.*

Let us briefly sketch how our SDP solver applies to this setting. Note first that we do not aim to estimate $\operatorname{Tr}(E_i\rho)$ for each $i \in [m]$, which helps us circumvent the $\Omega(m)$ lower bound. What we really want is to generate a state $\tilde{\rho}$ such that $\operatorname{Tr}(E_i\tilde{\rho}) \approx \operatorname{Tr}(E_i\rho)$ for each i. Our SDP solver will maintain and update a description of $\tilde{\rho}$ per iteration. In each iteration, given copies of $\tilde{\rho}$ and the actual unknown state ρ, we want to know whether $\operatorname{Tr}(E_i\tilde{\rho}) \approx \operatorname{Tr}(E_i\rho)$ $\forall i \in [m]$ or there is at least a violation i^*. To that end, we design for each i a projection for the following procedure: (1) perform multiple independent SWAP tests between $E_i/\operatorname{Tr}[E_i]$ (from Oracle 1.3) and ρ, $\tilde{\rho}$ respectively; (2) accept when the statistics of both SWAP tests (one with ρ, the other with $\tilde{\rho}$) are close. Hence, one can apply our fast quantum OR lemma on these projections to find such i^* if it exists.

Note that both the sample complexity and the gate complexity of the above procedure have a poly-log dependence on n (i.e., the dimension of the quantum state to learn).

Shadow tomography problem. In a sequence of works [1, 2], Aaronson asked whether one can predict information about a dimension-n quantum state with poly-log(n) many copies. In Ref. [1], he showed that a linear number of copies is sufficient to predict the outcomes of "most" measurements according to some (arbitrary) distribution over a class of measurements. Very recently, in Ref. [2], he referred the following problem as the "shadow tomography" problem: for any n-dimensional state ρ and two-outcome measurements $E_1, \ldots, E_m$, estimate $\operatorname{Tr}[\rho E_i]$ up to error ϵ, $\forall i \in [m]$. He has further designed a quantum procedure for the shadow tomography problem with $\tilde{O}(\ell \cdot \log^4 m/\epsilon^5)$ [6] copies of ρ.

[6] Here $\tilde{O}$ hides factors that are polynomial in $\log\log m$, $\log\log n$, and $\log 1/\epsilon$.

Noting that the shadow tomography problem is essentially the same problem considered by Jaynes [17], one can apply Jaynes' principle and its algorithmic version we discussed before. Although this can be used to give a version of the result of Ref. [2], Aaronson obtained his result [2] through a different route, based on a post-selection argument. A drawback of this approach is that its gate complexity is high, scaling linearly in m and as $n^{O(\log\log n)}$ (for fixed error).

Our Theorem 14 can be applied here to improve the time complexity. It gives a quantum procedure with a *square-root* dependence on m and $n^{O(1)}$ dependence on n for arbitrary E_i's.

When we assume r is small, say $r = O(\mathrm{poly}\log n)$, the gate complexity of the entire procedure becomes $\tilde{O}(\sqrt{m}\,\mathrm{poly}\log(n))$. This gives a class of measurement (namely any set of low-rank measurements which can be efficiently implemented) for which the learning problem is efficient both in the number of samples and the computational complexity. This solves an open problem proposed in Ref. [1]

Although we have not worked out an explicit bound of the sample complexity of our procedure, the authors of [4] followed our approach with more sophisticated techniques and obtained a sample complexity of $\tilde{O}(\ell \cdot \log^4 m/\epsilon^4)$, improving on the bound from [2]. We also note that very recently, Aaronson et al. claimed the same sample complexity (i.e., $\tilde{O}(\ell \cdot \log^4 m/\epsilon^4)$) in [3].

1.6 Overview of detailed results and proofs

In the full version of our paper [8], we formulate the SDP feasibility problem and prove the correctness of the basic framework in Appendix A. Our implementation of the fast quantum OR lemma is given in Appendix B. We describe our main results the constructions of quantum SDP solvers in the plain input model and the quantum input model in Appendix C and Appendix D, respectively. The application to learning quantum states is illustrated in Appendix E. In Appendix F (with full details in Appendix G) we demonstrate how to sample from the Gibbs state of low-rank Hamiltonians.

1.7 Open questions

This work leaves several natural open questions for future work. For example:

- Are there more examples of interesting SDPs where our form of input is meaningful? We have shown the example of learning quantum states. Intuitively, we are looking for SDP instances where the constraints are much "simpler" than the solution space. Is there any such example in the context of big data and/or machine learning?
- Our work has identified one setting where Gibbs sampling has a poly-log dependence on the dimension? Is there any other setting for the same purpose?
- For any reasonable quantum input setting, what is the effect of potential noises on quantum inputs in practice?
- Can we improve further on other parameters (e.g., the dependence on m and $1/\epsilon$)? In particular, is it possible to improve the error dependence to $\mathrm{poly}\log(1/\epsilon)$? This probably implies that we have to consider a quantum version of the interior point method.
- Are there other classes of measurements for which the quantum learning problem can solved in a computationally efficient way beyond the low-rank measurements we consider in this work? We note that most measurements of interest are not low rank (e.g. local measurements) and therefore the practical applicability of the present result is limited.

References

1 Scott Aaronson. The learnability of quantum states. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 463(2088):3089–3114, 2007. arXiv:quant-ph/0608142.

2 Scott Aaronson. Shadow Tomography of Quantum States. In *Proceedings of the Fiftieth Annual ACM Symposium on Theory of Computing*. ACM, 2018. arXiv:1711.01053.

3 Scott Aaronson, Xinyi Chen, Elad Hazan, and Ashwin Nayak. Online Learning of Quantum states, 2018. arXiv:1802.09025.

4 Joran van Apeldoorn and András Gilyén. Improvements in Quantum SDP-solving with Applications, 2018. arXiv:1804.05058.

5 Joran van Apeldoorn, András Gilyén, Sander Gribling, and Ronald de Wolf. Quantum SDP-Solvers: Better upper and lower bounds. In *58th Annual IEEE Symposium on Foundations of Computer Science*. IEEE, 2017. arXiv:1705.01843.

6 Sanjeev Arora, Elad Hazan, and Satyen Kale. The Multiplicative Weights Update Method: a Meta-Algorithm and Applications. *Theory of Computing*, 8(6):121–164, 2012. doi:10.4086/toc.2012.v008a006.

7 Sanjeev Arora and Satyen Kale. A combinatorial, primal-dual approach to semidefinite programs. In *Proceedings of the Thirty-ninth Annual ACM Symposium on Theory of Computing*, pages 227–236. ACM, 2007.

8 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, 2019. arXiv:1710.02581.

9 Fernando G. S. L. Brandão and Krysta Svore. Quantum speed-ups for semidefinite programming. In *58th Annual IEEE Symposium on Foundations of Computer Science*. IEEE, 2017. arXiv:1609.05537.

10 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, 2019. arXiv:1901.03254.

11 Anirban Narayan Chowdhury and Rolando D. Somma. Quantum algorithms for Gibbs sampling and hitting-time estimation. *Quantum Information & Computation*, 17(LA-UR-16-21218), 2017. arXiv:1603.02940.

12 Gus Gutoski and Xiaodi Wu. Parallel approximation of min-max problems with applications to classical and quantum zero-sum games. In *Proceedings of the Twenty-seventh Annual IEEE Symposium on Computational Complexity (CCC)*, pages 21–31. IEEE, 2012.

13 Jeongwan Haah, Aram W. Harrow, Zhengfeng Ji, Xiaodi Wu, and Nengkun Yu. Sample-optimal Tomography of Quantum States. In *Proceedings of the 48th Annual ACM Symposium on Theory of Computing*, pages 913–925. ACM, 2016. arXiv:1508.01797.

14 Aram W. Harrow, Cedric Yen-Yu Lin, and Ashley Montanaro. Sequential measurements, disturbance and property testing. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1598–1611. SIAM, 2017. arXiv:1607.03236.

15 Elad Hazan. *Efficient algorithms for online convex optimization and their applications*. PhD thesis, Princeton University, 2006.

16 Rahul Jain, Zhengfeng Ji, Sarvagya Upadhyay, and John Watrous. QIP=PSPACE. *Journal of the ACM (JACM)*, 58(6):30, 2011. arXiv:0907.4737.

17 Edwin T. Jaynes. Information theory and statistical mechanics. *Physical Review*, 106(4):620, 1957.

18 Shelby Kimmel, Cedric Yen-Yu Lin, Guang Hao Low, Maris Ozols, and Theodore J. Yoder. Hamiltonian simulation with optimal sample complexity. *npj Quantum Information*, 3(1):13, 2017. arXiv:1608.00281.

19 James R. Lee, Prasad Raghavendra, and David Steurer. Lower Bounds on the Size of Semidefinite Programming Relaxations. In *Proceedings of the Forty-seventh Annual ACM*

Symposium on Theory of Computing, pages 567–576, New York, NY, USA, 2015. ACM. doi:10.1145/2746539.2746599.

20 Yin Tat Lee, Aaron Sidford, and Sam Chiu-wai Wong. A faster cutting plane method and its implications for combinatorial and convex optimization. In *Proceedings of the Fifty-sixth Annual IEEE Symposium on Foundations of Computer Science*, pages 1049–1065. IEEE, 2015. arXiv:1508.04874.

21 Seth Lloyd, Masoud Mohseni, and Patrick Rebentrost. Quantum principal component analysis. *Nature Physics*, 10(9):631, 2014. arXiv:1307.0401.

22 Daniel Nagaj, Pawel Wocjan, and Yong Zhang. Fast amplification of QMA, 2009. arXiv:0904.1549.

23 Ryan O'Donnell and John Wright. Efficient Quantum Tomography. In *Proceedings of the 48th Annual ACM Symposium on Theory of Computing*, pages 899–912. ACM, 2016. arXiv:1508.01907.

24 David Poulin and Pawel Wocjan. Sampling from the thermal quantum Gibbs state and evaluating partition functions with a quantum computer. *Physical Review Letters*, 103(22):220502, 2009. arXiv:0905.2199.

25 Ewin Tang. A quantum-inspired classical algorithm for recommendation systems. In *Proceedings of the 51st Annual ACM Symposium on Theory of Computing*. ACM, 2019. arXiv:1807.04271.

26 Ronald de Wolf. Personal communication, 2017.

27 Xiaodi Wu. Parallelized Solution to Semidefinite Programmings in Quantum Complexity Theory, 2010. arXiv:1009.2211.