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

Fernando G. S. L. Brandão,[1] Amir Kalev,[2] Tongyang Li,[2] Cedric Yen-Yu Lin,[2] Krysta M. Svore,[3] Xiaodi Wu[2]

[1]Institute of Quantum Information and Matter, California Institute of Technology, Pasadena, CA
[2]Department of Computer Science, Institute for Advanced Computer Studies, and Joint Center for Quantum Information and Computer Science, University of Maryland, MD
[3]Station Q, Quantum Architectures and Computation Group, Microsoft Research, Redmond, WA

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 the problem of learning 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, 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. [17]. We believe both techniques might be of independent interest.

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. [11], 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 [24, 8] (with complexity $\tilde{O}(m(m^2 + n^\omega + mns)\operatorname{poly}\log(R/\epsilon))$ [24], where ω is the exponent of matrix multiplication, and $\tilde{O}(mns(R\tilde{R}/\varepsilon)^4 + ns\left(R\tilde{R}/\varepsilon\right)^7)$ [8]), and beating the classical lower bound of $\Omega(m+n)$ [11]. The follow-up work by van Apeldoorn et al. [6] 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. [11] 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 n and m [6]. 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 ora:plain 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 (see Section 2 for details): 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 region $\mathcal{S}_\epsilon$ as all X such that

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

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

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

[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., $\operatorname{Tr}(CX)$ and assume one wants to maximize $\operatorname{Tr}(CX)$) and convert it into a constraint (e.g., $\operatorname{Tr}(CX) \geq c_0$). 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 $\operatorname{Tr}(CX) \in [-1, 1]$.

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. [11]). 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. [6], 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 n and m

Existing quantum SDP solvers [11, 6] 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 n and m in the (plain) input model as used by [11, 6], 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:

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

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.

Definition 1.1 (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 sample complexity 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 1.2 (Gibbs sampling)**.** Assume that we have an s-sparse $n \times n$ Hermitian matrix H with $\|H\| \leq \Gamma$. Then we define $\mathcal{T}_{\mathrm{Gibbs}}(s, \Gamma, \epsilon)$ 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 1.3 (informal; see Theorem 4.3)**.** *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 $A_j, j \in [m]$.*

[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 $\mathrm{Tr}(\hat{X}) = 1 \iff \mathrm{Tr}(X) \leq 1$.

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. [11], and results on $\mathcal{T}_{\mathrm{Gibbs}}(s,\Gamma,\epsilon)$ from Ref. [28]), we end up with the following concrete parameters:

Corollary 1.4 (informal; see Corollary 4.6)**.** *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 n and m, 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. [11].

1.2 Quantum SDP solvers with quantum inputs

Given the optimimality 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. [19, 15]) and quantum information (e.g., Refs. [1, 3]). 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. [11], 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: We imagine a specific setting in which the A_j's are "nice" so that the following oracles can be efficiently implemented. Assume A_j can be decomposed as $A_j = A_j^+ - A_j^-$, where $A_j^+, A_j^- \succeq 0$. A natural choice is to let A_j^+ (resp. A_j^-) be the positive (resp. negative) part of A although one has the freedom in choosing general A_j^+, A_j^-.

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 $j \in [m]$,

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

where the real values $\mathrm{Tr}[A_j^+]$ and $\mathrm{Tr}[A_j^-]$ 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 $j \in [m]$,

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

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

$$O_a|j\rangle\langle j| \otimes |0\rangle\langle 0|O_a^\dagger = |j\rangle\langle j| \otimes |a_j\rangle\langle a_j|, \tag{1.5}$$

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 $\mathrm{Tr}[A_j^+] + \mathrm{Tr}[A_j^-] \leq B$. 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 1.5 (trace estimation)**.** We define $\mathcal{S}_{\mathrm{Tr}}(B, \epsilon)$ and $\mathcal{T}_{\mathrm{Tr}}(B, \epsilon)$ as the sample complexity of a state $\rho \in \mathbb{C}^n$ and the gate complexity of using the quantum input oracle Oracle 1.2, Oracle 1.3, Oracle 1.4, respectively, for any quantum algorithm that distinguishes with success probability at least $1 - O(1/m)$ whether for a fixed $j \in [m]$, $\mathrm{Tr}(A_j\rho) > a_j + \epsilon$ or $\mathrm{Tr}(A_j\rho) \leq a_j$.

Definition 1.6 (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 $\mathcal{T}_{\mathrm{Gibbs}}(r_K, \Phi, B_K, \epsilon)$ 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 1.7 (informal; see Theorem 5.4)**.** *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.*

Contrary 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 in Lemma 5.6 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 in Lemma 5.8 with $\mathcal{T}_{\mathrm{Gibbs}}(r_K, \Phi, B_K, \epsilon) = O(\Phi \cdot \mathrm{poly}(\log n, r_K, B_K, \epsilon^{-1}))$. As a result,

Corollary 1.8 (informal; see Corollary 5.5)**.** *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.*

In the quantum input model, 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, the dependence on n needs to be $\Omega(\sqrt{n})$ [11].

The poly-logarithmic dependence on n is intriguing and suggests that quantum computers might offer exponential speed-ups for some SDP instances. However one has to careful as the input model we consider is inherently quantum, so it is incomparable with classical SDP solvers. As suggested to us by Aram Harrow (private 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 discussed in Section 1.5 to reduce the problem to a SDP of dimension $\mathrm{poly}(r)$). 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. In Ref. [5] more applications are given.

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

Theorem 1.9 (lower bound on Theorem 5.4)**.** *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.*

It is also worth mentioning that our quantum SDP solver does *not* assume the *sparsity* condition of A_i's which are crucial, e.g. in the SDP solvers of Refs. [11, 6]. This is because the assumption on Oracle 1.2, Oracle 1.3, and Oracle 1.4 provides 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).

1.3 Related works on quantum SDP solvers

Previous quantum SDP solvers [11, 6] 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 [11]. 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 (Lemma 3.2), which will be elaborated in Section 1.4.

The quantum input model was briefly mentioned in an earlier version of [11]. 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 (Lemma 5.8).

Subsequent to a previous version of this paper [10], an independent interesting result by van Apeldoorn and Gilyén [5] has improved the complexity of trace-estimation and Gibbs sampling. After a personal communication [14] introducing our fast version of the quantum OR lemma, the authors of Ref. [5] observed independently that the application of the quantum OR lemma [17] to decouple the dependence of m and n. As a result, Ref. [5] improved the complexity of Corollary 1.4 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. [5]. Using novel techniques, it also has improved the complexity of Corollary 1.8 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).

1.4 Techniques

At a high level, and in similarity to Refs. [11, 6], our quantum SDP solver can be seen as a "quantized" version of classical SDP solvers based on the matrix multiplicative weight update (MMWU) method [7]. In particular, we will leverage quantum Gibbs samplers as the main source of quantum speed-ups. In Refs. [11, 6], quantum Gibbs samplers with quadratic speed-ups (e.g., [28, 13]) have been exploited to replace the classical Gibbs state calculation step. 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 [8] which is the classical counterpart of previous quantum SDP solvers [11, 6]. Instead, we follow a zero-sum game framework to solve SDP feasibility problems, which is also based on the MMWU method (details in Section 2). This framework has appeared in the classical literature (e.g., [18]) and has already been used to in semidefinite programs of relevance in quantum complexity theory (e.g., [30, 15, 23]). 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)**.** Input a density matrix X, output an $i \in [m]$ such that Eq. (2.1) is violated. 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 [7].

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 [23] 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 [5] 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 Oracle 1.5 efficiently. Roughly speaking, the idea behind previous quantum SDP solvers [11, 6]

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 [17] 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 $\Lambda_1, \cdots, \Lambda_m$, 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. [17] 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 [26]. Overall, this leads to a complexity of $O(\sqrt{m} + \sqrt{n})$.

Lemma 1.10 (informal; see Lemma 3.2)**.** *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 1.11 (informal; see Theorem 7.4)**.** *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 $\Theta(\sqrt{n})$ [28, 13]. Our construction deviates significantly from [28, 13]. Because of the existence of copies of ρ^+ and ρ^-, we rely on efficient Hamiltonian simulation techniques developed in quantum principle component analysis (PCA) [25] and its follow-up work in [22]. 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 [28, 13].

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 [27, 16] requires n^2 number of copies of ρ, which is impractical already for relatively small ℓ.

An intereting 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 measurements $E_1, \cdots, E_m$, where $I \succeq E_i \succeq 0 \in C^{n\times n}, \forall i \in [m]$. This is an old problem, dating back at least to the work of Jaynes on statistical mechanics in the 50ies. Jaynes' principle [20] (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 n and m.

In fact, our algorithm also outputs each of the λ_i's (one can show that $\mathrm{poly}(\log(mn))/\varepsilon^2$ nonzero 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. Compare (1.6) and Eq. Algorithm 1.) In this sense our result can be seen as an *algorithmically* version of Jaynes' principle. We note that a similar idea was adopted by [23] 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 $E_1, \cdots, E_m$ 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 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 1.12 (informal; see Corollary 6.3)**.** *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 [2, 3], Aaronson asked whether one can predict information about a dimension-n quantum state with poly-log(n) many copies. In Ref. [2], 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. [3], 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 $\tilde{O}(\ell \cdot \log^4 m/\epsilon^5)$ [4] copies of ρ.

Noting that the shadow tomography problem is essentially the same problem considered by Jaynes [20], 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. [3], Aaronson obtained his result [3] 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 1.12 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 $r = O(\text{poly}\log n)$, the gate complexity of the entire procedure becomes $\tilde{O}(\sqrt{m}\,\text{poly}\log(n))$. 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. [2]

Although we have not worked out an explicit bound of the sample complexity of our procedure, the authors of [5] 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 [3]. 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 [4].

1.6 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 $\text{poly}\log(1/\epsilon)$? This probably implies that we have to consider a quantum version of the interior point method.

- Are there other class of measurements for which the quantum learning problem can solved in a computational 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.

- Can we design faster quantum algorithms for general optimization problems beyond SDP, e.g., more general convex optimization problems?

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

Organization. We will formulate the SDP feasibility problem and prove the correctness of the basic framework in Section 2. Our implementation of the fast quantum OR lemma is given in Section 3. We describe our main results the constructions of quantum SDP solvers in the plain input model and the quantum input model in Section 4, Section 5, respectively. The application to learning quantum states is illustrated in Section 6. In Section 7 (with full details in Appendix A) we demonstrate how to sample from the Gibbs state of low-rank Hamiltonians.

2 Feasibility of SDPs

In this section, we formulate the feasibility problem of SDPs. It is a standard fact that one can use binary search to reduce any optimization problem to a feasibility one. The high-level idea is to first guess a candidate value for the objective function, and add that as a constraint to the optimization problem. It converts the optimization problem into a feasibility problem. One can then use binary search on the candidate value to find a good approximation to the optimal one.

Definition 2.1 (Feasibility)**.** 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 region $\mathcal{S}_\epsilon$ as all X such that

$$\mathrm{Tr}(A_i X) \leq a_i + \epsilon \quad \forall\, i \in [m]; \tag{2.1}$$

$$X \succeq 0; \tag{2.2}$$

$$\mathrm{Tr}[X] = 1. \tag{2.3}$$

For approximate feasibility testing, it is required that:

- If $\mathcal{S}_0 = \emptyset$, output fail;
- If $\mathcal{S}_\epsilon \neq \emptyset$, output an $X \in \mathcal{S}_\epsilon$.

Zero-sum game approach for SDPs. We adopt the zero-sum game approach to solve SDPs. Note that it is different from [11, 6] which follow the primal-dual approach of [8] to solve SDPs. Instead of leveraging the dual program, we rely on the following oracle:

Oracle 2.1 (Search for violation)**.** Input a density matrix X, output an $i \in [m]$ such that Eq. (2.1) is violated. If no such i exists, output "FEASIBLE".

This oracle helps establish a game view to solve any SDP feasibility problem. Imagine Player 1 who 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. (This is exactly the function of Oracle 2.1.) 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 2.1). This actually refers to an *equilibrium* point of the zero-sum game, which can be approximated by the matrix multiplicative weight update method [7].

This game view of solving the SDP feasibility problem has appeared in the classical literature (e.g., [18]) and has already been used in solving semidefinite programs in the context of quantum complexity theory (e.g., [30, 15]). We observe that many techniques to quantize Arora-Kale's primal-dual approach [8] for solving SDPs in Refs. [11, 6] readily extends to the zero-sum game approach, e.g., using quantum Gibbs samplers to generate candidate solution states.

The main difference, however, lies in the way one make use of the matrix multiplicative weight update method [21], which is a meta algorithm behind both the Arora-Kale's primal-dual approach [8] and the game view approach (e.g., [18]). As we have elaborated in Section 1.4, there are a few advantages of adopting this game view approach.

Master algorithm. We present a master algorithm that solves the SDP feasibility problem with the help of Oracle 2.1. It should be understood that the master algorithm is *not* the final quantum algorithm, where a few steps will be replaced by their quantum counterparts. However, the master algorithm helps demonstrate the correctness of the algorithm and the number of oracle queries.

Our algorithm heavily relies on the matrix multiplicative weight method given in Algorithm 1.

Algorithm 1: Matrix multiplicative weights algorithm (Figure 3.1 of [21]).

Initialization: Fix a $\delta \leq 1/2$. Initialize the weight matrix $W^{(1)} = I_n$;
for $t = 1, 2, \ldots, T$ **do**
  Set the density matrix $\rho^{(t)} = \frac{W^{(t)}}{\mathrm{Tr}[W^{(t)}]}$;
  Observe the gain matrix $M^{(t)}$;
  Define the new weight matrix: $W^{(t+1)} = \exp[-\delta \sum_{\tau=1}^{t} M^{(\tau)}]$;

Proposition 2.2 (Corollary 4 of [21]). *Assume that for all $t \in [T]$, either $M^{(t)} \preceq 0$ or $M^{(t)} \succeq 0$. Then Algorithm 1 guarantees that after T rounds, for any density matrix ρ, we have*

$$(1-\delta) \sum_{t:\, M^{(t)} \preceq 0} \mathrm{Tr}(M^{(t)}\rho^{(t)}) + (1+\delta) \sum_{t:\, M^{(t)} \succeq 0} \mathrm{Tr}(M^{(t)}\rho^{(t)}) \geq \sum_{t=1}^{T} \mathrm{Tr}(M^{(t)}\rho) - \frac{\ln n}{\delta}. \tag{2.4}$$

We use Algorithm 1 and Proposition 2.2 to test the feasibility of SDPs.

Theorem 2.3 (Master Algorithm). *Assume we are given Oracle 2.1. Then for any $\epsilon > 0$, feasibility of the SDP in (2.1), (2.2), and (2.3) can be tested by Algorithm 2 with at most $\frac{16 \ln n}{\epsilon^2}$ queries to the oracle.*

Algorithm 2: Matrix multiplicative weights algorithm for testing the feasibility of SDPs.

Initialize the weight matrix $W^{(1)} = I_n$, and $T = \frac{16 \ln n}{\epsilon^2}$;
for $t = 1, 2, \ldots, T$ **do**
  Prepare the Gibbs state $\rho^{(t)} = \frac{W^{(t)}}{\mathrm{Tr}[W^{(t)}]}$;
  Find a $j^{(t)} \in \{1, 2, \ldots, m\}$ such that $\mathrm{Tr}(A_{j^{(t)}}\rho^{(t)}) > a_{j^{(t)}} + \epsilon$ by Oracle 2.1. Take $M^{(t)} = \frac{1}{2}(I_n - A_{j^{(t)}})$ if such $j^{(t)}$ can be found; otherwise, claim that $\mathcal{S}_\epsilon \neq \emptyset$, output $\rho^{(t)}$ as a feasible solution, and terminate the algorithm;
  Define the new weight matrix: $W^{(t+1)} = \exp[-\frac{\epsilon}{2} \sum_{\tau=1}^{t} M^{(\tau)}]$;
Claim that $\mathcal{S}_0 = \emptyset$ and terminate the algorithm;

Proof of Theorem 2.3. For all $j \in [m]$, denote $M_j = \frac{1}{2}(I_n - A_j)$; note that $0 \preceq M_j \preceq I \ \forall j \in [m]$. In round t, after computing the density matrix $\rho^{(t)}$, equivalently speaking, Oracle 2.1 checks whether there exists a $j \in [m]$ such that $\mathrm{Tr}(M_j\rho^{(t)}) < \frac{1}{2} - \frac{a_j+\epsilon}{2}$. If not, then $\mathrm{Tr}(M_j\rho^{(t)}) \geq \frac{1}{2} - \frac{a_j+\epsilon}{2}\ \forall j \in [m]$, $\mathrm{Tr}(A_j\rho^{(t)}) \leq a_j + \epsilon\ \forall j \in [m]$, and hence $\rho^{(t)} \in \mathcal{S}_\epsilon$.

Otherwise, the oracle outputs an $M_{j^{(t)}} \in \{M_j\}_{j=1}^m$ such that $\mathrm{Tr}(M_{j^{(t)}}\rho^{(t)}) < \frac{1}{2} - \frac{a_{j^{(t)}}+\epsilon}{2}$. After $T = \frac{16\ln n}{\epsilon^2}$ iterations, by Proposition 2.2 (taking $\delta = \epsilon/4$ therein), this matrix multiplicative weights algorithm promises that for any density matrix ρ, we have

$$\Big(1+\frac{\epsilon}{4}\Big)\sum_{t=1}^{T}\mathrm{Tr}(M_{j^{(t)}}\rho^{(t)}) \geq \sum_{t=1}^{T}\mathrm{Tr}(M_{j^{(t)}}\rho) - \frac{4\ln n}{\epsilon}. \tag{2.5}$$

If $\mathcal{S}_0 \neq \emptyset$, there exists a $\rho^* \in \mathcal{S}_0$ such that $\mathrm{Tr}(M_{j^{(t)}}\rho^*) \geq \frac{1}{2} - \frac{a_{j^{(t)}}}{2}$ for all $t \in [T]$. On the other hand, $\mathrm{Tr}(M_{j^{(t)}}\rho^{(t)}) < \frac{1}{2} - \frac{a_{j^{(t)}}+\epsilon}{2}$ for all $t \in [T]$. Plugging these two inequalities into (2.5), we have

$$\Big(1+\frac{\epsilon}{4}\Big)\sum_{t=1}^{T}\Big(\frac{1}{2} - \frac{a_{j^{(t)}}+\epsilon}{2}\Big) > \sum_{t=1}^{T}\Big(\frac{1}{2} - \frac{a_{j^{(t)}}}{2}\Big) - \frac{4\ln n}{\epsilon}, \tag{2.6}$$

which is equivalent to

$$\frac{16\ln n}{\epsilon^2} > \frac{3+\epsilon}{2}T + \frac{1}{2}\sum_{t=1}^{T}a_{j^{(t)}}. \tag{2.7}$$

Furthermore, since $\frac{1}{2} - \frac{a_{j^{(t)}}}{2} \leq \mathrm{Tr}(M_{j^{(t)}}\rho^*) \leq 1$, we have $a_{j^{(t)}} \geq -1$ for all $t \in [T]$. Plugging this into (2.7), we have $\frac{16\ln n}{\epsilon^2} > (1+\frac{\epsilon}{2})T$, and hence

$$T < \frac{16\ln n}{\epsilon^2(1+\epsilon/2)} < \frac{16\ln n}{\epsilon^2}, \tag{2.8}$$

contradiction! Therefore, if $\mathrm{Tr}(M_{j^{(t)}}\rho^{(t)}) < \frac{1}{2} - \frac{a_{j^{(t)}}+\epsilon}{2}$ happens for at least $\frac{16\ln n}{\epsilon^2}$ times, it must be the case that $\mathcal{S}_0 = \emptyset$. □

3 Fast quantum OR lemma

To use our master algorithm (Algorithm 2), a key step is to implement Oracle 2.1 that finds a violated constraint in the SDP. This is basically to search among m measurements, which motivates us to use the quantum OR lemma from [17].

Lemma 3.1 (Corollary 11 of [17])**.** *Let $\Lambda_1, \ldots, \Lambda_m$ be projectors, and fix parameters $0 < \epsilon \leq 1/2$, $0 < \delta < 1/4m$. Let ρ be a state such that either $\exists\, j \in [m]$ such that $\mathrm{Tr}[\rho\Lambda_j] \geq 1-\epsilon$, or $\frac{1}{m}\sum_{j=1}^{m}\mathrm{Tr}[\rho\Lambda_j] \leq \delta$. Then there is a test that uses one copy of ρ and: in the former case, accepts with probability at least $(1-\epsilon)^2/7$; in the latter case, accepts with probability at most $4\delta m$.*

However, the focus of Lemma 3.1 was on the single copy of ρ and its proof in [17] leads to a poor gate complexity. As a result, we prove the "fast" quantum OR lemma below (Lemma 3.2).

This new version basically follows the analysis of the original quantum OR lemma; however, the projections are implemented with a quadratic speed-up in m by the fast amplification technique in [26]. This speed-up enables us to decouple the cost of $\sqrt{m} \cdot \sqrt{n}$ in [11, 6] to $(\sqrt{m} + \sqrt{n})$ (see Section 4 and Section 5 for more details); in particular, it leads to the optimal bound for solving SDPs when other parameters are constants.

Lemma 3.2. *Let $\Lambda_1, \ldots, \Lambda_m$ be projections, and fix parameters $0 < \varepsilon \leq 1/2$ and φ. Let ρ be a state such that either $\exists\, i \in [m]$ such that $\mathrm{Tr}[\rho\Lambda_i] \geq 1 - \varepsilon$, or $\frac{1}{m}\sum_{j=1}^{m} \mathrm{Tr}[\rho\Lambda_j] \leq \varphi$. Then there is a test that uses one copy of ρ and: in the former case, accepts with probability at least $(1-\varepsilon)^2/4 - \xi$; in the latter case, accepts with probability at most $3\varphi m + \xi$; here ξ satisfies $\xi > 0$ and $(1-\varepsilon)^2/4 - \xi > 3\varphi m + \xi$. Furthermore, as long as the controlled reflection $\mathrm{ctrl} - (I - 2\sum_{i=0}^{m-1} \Lambda_{i+1} \otimes |i\rangle\langle i|)$ can be performed in at most p operations, this test requires only $O(\xi^{-1}\sqrt{m}(p + \mathrm{poly}(\log m)))$ operations to complete.*

Proof. Similar to [17], we will reduce the task of distinguishing the two cases to estimating the eigenvalues of

$$\Lambda := \frac{1}{m}\sum_{i=1}^{m} \Lambda_i, \tag{3.1}$$

the average of these POVM operators. Write $P_{\geq\lambda}$ for the projector onto $\mathrm{span}\{|\lambda'\rangle : \Lambda|\lambda'\rangle = \lambda'|\lambda'\rangle, \lambda' \geq \lambda\}$. Then the following was shown in [17]:

Lemma 3.3 ([17, implicit in proof of Corollary 11])**.** *For any state ρ and $\lambda \leq \max_i \mathrm{Tr}(\Lambda_i\rho)/m$,*

$$\mathrm{Tr}(P_{\geq\lambda}\rho) \geq [\max_i \mathrm{Tr}(\Lambda_i\rho) - m\lambda]^2. \tag{3.2}$$

Choose $\lambda = (1-\varepsilon)/(2m)$. Then we want to distinguish between the following two cases:

Case 1 $\mathrm{Tr}(P_{\geq\lambda}\rho) \geq (1 - \varepsilon - m\lambda)^2 = (1-\varepsilon)^2/4$;

Case 2 $\mathrm{Tr}(\Lambda\rho) \leq \varphi$. This implies $\mathrm{Tr}(P_{\geq 0.8\lambda}\rho) \leq \varphi/(0.8\lambda) \leq 3m\varphi$.

We can explicitly decompose Λ as follows (see also [17, Section 2]): Let Q be the quantum Fourier transform on $\mathbb{Z}_m$, and define the projectors $\Pi = \sum_{i=0}^{m-1} \Lambda_{i+1} \otimes (Q|i\rangle\langle i|Q^\dagger)$, $\Delta = I \otimes |0\rangle\langle 0|$. Then

$$\Delta\Pi\Delta = \frac{1}{m}\sum_{i=1}^{m} \Lambda_i \otimes |0\rangle\langle 0| = \Lambda \otimes |0\rangle\langle 0|. \tag{3.3}$$

where $|0\rangle\langle 0|$ in the above equation is shorthand for $|0\rangle\langle 0|^\ell$ for $\ell = \lceil \log m \rceil$.

Let $a = \arccos(\sqrt{\lambda})$ and $b = \arccos(\sqrt{0.8\lambda})$. Consider the following algorithm, essentially based on the fast amplification algorithm of [26]:

Algorithm 3: The fast amplification algorithm in [26].

1. Create the state $\rho \otimes |0\rangle\langle 0|^{\otimes\ell}$.
2. Perform phase estimation of the rotation $(I - 2\Pi)(I - 2\Delta)$ on the state, with precision $(b-a)/2$ and error probability ξ. Let the measured eigenvalue be ϕ.
3. Accept iff $|\phi| \leq (a+b)/2$.

The following lemma in [26] follows from a direct application of Jordan's lemma:

Lemma 3.4. *[26, Section 2.1] If $|\psi\rangle \otimes |0\rangle^{\otimes \ell}$ is an eigenvector of $\Delta\Pi\Delta$ with eigenvalue $\cos^2\phi$, then*

$$|\psi\rangle \otimes |0\rangle^{\otimes \ell} = \frac{1}{\sqrt{2}}(|\phi\rangle + |-\phi\rangle) \tag{3.4}$$

where $|\phi\rangle$ and $|-\phi\rangle$ are some eigenvectors of $(I-2\Pi)(I-2\Delta)$ with eigenvalues ϕ and $-\phi$, respectively.

In Case 1, we have $\mathrm{Tr}(P_{\geq\lambda}\rho) \geq (1-\varepsilon)^2/4$, and therefore Algorithm 3 accepts with probability at least $(1-\varepsilon)^2/4 - \xi$. In Case 2, we have $\mathrm{Tr}(P_{\geq 0.8\lambda}\rho) \leq 3m\varphi$, and therefore Algorithm 3 accepts with probability at most $3m\varphi + \xi$.

Algorithm 3 requires applying the controlled version of the Grover iterate $(I-2\Pi)(I-2\Delta)$ $O(((b-a)\xi)^{-1}) = O(\sqrt{m}\xi^{-1})$ times. Furthermore, the controlled reflection ctrl-$(I-2\Delta)$ is implementable by $O(\log m)$ gates since $\Delta = I \otimes |0\rangle\langle 0|^{\otimes \lceil \log m \rceil}$, and the controlled reflection ctrl-$(I-2\Pi)$ is implementable using $O(p + \mathrm{poly}(\log m))$ gates by assumption. □

Remark 3.1. The gate complexity in Lemma 3.2 is optimal in $\sqrt{m}$, i.e., there exists projections $\Lambda_1, \ldots, \Lambda_m$ and a state ρ such that distinguishing whether $\exists\, i \in [m]\ \mathrm{Tr}[\rho\Lambda_i] \geq 2/3$ or $\frac{1}{m}\sum_{j=1}^{m}\mathrm{Tr}[\rho\Lambda_j] \leq 1/8m$ requires at least $\Omega(\sqrt{m})$ gates. In particular, assume that $\Lambda_i = |i\rangle\langle i|$ for all $i \in [m]$ and $\rho = |k\rangle\langle k|$ where $k \in [m+1]$. Then to distinguish whether $\exists\, i \in [m]\ \mathrm{Tr}[\rho\Lambda_i] \geq 2/3$ or $\frac{1}{m}\sum_{j=1}^{m}\mathrm{Tr}[\rho\Lambda_j] \leq 1/8m$, it is equivalent to searching whether $k \in [m]$ or not; deciding this requires at least $\Omega(\sqrt{m})$ gates due to the hardness of Grover search [9].

4 Quantum SDP solver in the plain model

Before we get into the quantum SDP solver in the plain model, we first modularize the cost of two important blocks as follows.

Definition 4.1 (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 sample complexity of ρ and the time complexity of using the plain model (Oracle 1.1) of H and two-qubit gates, respectively, such that one can compute $\mathrm{Tr}[H\rho]$ with additive error ϵ with success probability at least $2/3$.

Definition 4.2 (Gibbs sampling)**.** Assume that we have an s-sparse $n \times n$ Hermitian matrix H with $\|H\| \leq \Gamma$. Then we define $\mathcal{T}_{\mathrm{Gibbs}}(s,\Gamma,\epsilon)$ 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) of H and two-qubit gates.

As a subsequence of Lemma 3.2, Definition 4.1, and Definition 4.2, we prove the following theorem under the plain model:

Theorem 4.3. *Assume we are given Oracle 1.1. Furthermore, assume that A_j is s-sparse for all $j \in [m]$. Then for any $\epsilon > 0$, feasibility of the SDP in (2.1), (2.2), and (2.3) can be tested by Algorithm 4 with success probability at least 0.96 and $\frac{s}{\epsilon^4}\tilde{O}\big(\mathcal{S}_{\mathrm{Tr}}\big(\frac{s}{\epsilon^2},\frac{1}{\epsilon},\epsilon\big)\mathcal{T}_{Gibbs}\big(\frac{s}{\epsilon^2},\frac{1}{\epsilon},\epsilon\big) + \sqrt{m}\mathcal{T}_{\mathrm{Tr}}\big(\frac{s}{\epsilon^2},\frac{1}{\epsilon},\epsilon\big)\big)$ quantum gates and queries to Oracle 1.1.*

Algorithm 4: Efficiently testing the feasibility of SDPs: Plain model.

1 Initialize the weight matrix $W^{(1)} = I_n$, and $T = \frac{16 \ln n}{\epsilon^2}$;
2 **for** $t = 1, 2, \ldots, T$ **do**
3 Prepare $\log m \cdot \mathcal{S}_{\mathrm{Tr}}(\frac{s \log n}{\epsilon^2}, \frac{\log n}{\epsilon}, \epsilon)$ samples of Gibbs state $\rho^{(t)} = \frac{W^{(t)}}{\mathrm{Tr}[W^{(t)}]}$ by Definition 4.2;
4 Using these $\log m \cdot \mathcal{S}_{\mathrm{Tr}}(\frac{s \log n}{\epsilon^2}, \frac{\log n}{\epsilon}, \epsilon)$ copies of $\rho^{(t)}$, search for a $j^{(t)} \in [m]$ such that $\mathrm{Tr}[A_{j^{(t)}} \rho^{(t)}] > a_{j^{(t)}} + \epsilon$ by Lemma 3.2 (for each j, we use Definition 4.1 to compute $\mathrm{Tr}[A_j \rho]$). If such $j^{(t)}$ is found, take $M^{(t)} = \frac{1}{2}(I_n - A_{j^{(t)}})$; otherwise, claim that $\mathcal{S}_\epsilon \neq \emptyset$ (the SDP is feasible);
5 Define the new weight matrix: $W^{(t+1)} = \exp[-\frac{\epsilon}{4} \sum_{\tau=1}^{t} M^{(\tau)}]$;
6 Claim that $\mathcal{S}_0 = \emptyset$ and terminate the algorithm.

Proof of Theorem 4.3. The correctness of Algorithm 4 is automatically established by Theorem 2.3; it suffices to analyze the gate cost of Algorithm 4.

In Line 3 of Algorithm 4, we apply Definition 4.2 to compute the Gibbs state $\rho^{(t)}$. In round t, because $t \leq \frac{16 \ln n}{\epsilon^2}$, $\frac{\epsilon}{4} \sum_{\tau=1}^{t} M^{(\tau)}$ has sparsity at most $s' \leq t \cdot s = O(\frac{s \log n}{\epsilon^2})$, and $\|\frac{\epsilon}{4} \sum_{\tau=1}^{t} M^{(\tau)}\| \leq \frac{\epsilon}{4} \cdot t = O(\frac{\log n}{\epsilon})$. As a result, $\mathcal{T}_{\mathrm{Gibbs}}(\frac{s \log n}{\epsilon^2}, \frac{\log n}{\epsilon}, \epsilon)$ quantum gates and queries to Oracle 1.1 suffice to prepare a copy of the Gibbs state $\rho^{(t)}$. In addition, since to query an element of $\frac{\epsilon}{4} \sum_{\tau=1}^{t} M^{(\tau)}$ we need to query each of the $A_{j^{(\tau)}}$, we have an overhead of $s \cdot \frac{16 \ln n}{\epsilon^2}$ for constructing Oracle 1.1 for $\frac{\epsilon}{4} \sum_{\tau=1}^{t} M^{(\tau)}$ (in particular, Appendix D of the full version of [6] showed that this overhead $\Theta(\frac{s \ln n}{\epsilon^2})$ is necessary and sufficient for constructing the plain oracle for $\frac{\epsilon}{4} \sum_{\tau=1}^{t} M^{(\tau)}$). In total, Line 3 of Algorithm 4 costs

$$\frac{16 s \ln n}{\epsilon^2} \cdot \log m \cdot \mathcal{S}_{\mathrm{Tr}}\Big(\frac{s \log n}{\epsilon^2}, \frac{\log n}{\epsilon}, \epsilon\Big) \cdot \mathcal{T}_{\mathrm{Gibbs}}\Big(\frac{s \log n}{\epsilon^2}, \frac{\log n}{\epsilon}, \epsilon\Big) \tag{4.1}$$

quantum gates and queries to Oracle 1.1.

Next, using these $\log m \cdot \mathcal{S}_{\mathrm{Tr}}(\frac{s \log n}{\epsilon^2}, \frac{\log n}{\epsilon}, \epsilon)$ copies of $\rho^{(t)}$, we apply Definition 4.1 for $O(\log m)$ times to create two-outcome POVMs M_j for any $j \in [m]$ such that M_j decides whether $\mathrm{Tr}(A_j \rho) - a_j > \epsilon$ with success probability boosted to $1 - O(1/m)$. The gate complexity of each M_j is $\mathcal{T}_{\mathrm{Tr}}(\frac{s \log n}{\epsilon^2}, \frac{\log n}{\epsilon}, \epsilon)$ by Definition 4.1. Furthermore, because Oracle 1.1 is reversible, we can assume an explicit decomposition $M_j \otimes |0\rangle\langle 0|^{\otimes a} = P \Lambda_j P$, for some integer a, $P = I \otimes |0\rangle\langle 0|^{\otimes a}$, and some orthogonal projector Λ_j. Let $\widetilde{\rho} = \rho^{\otimes C} \otimes |0\rangle\langle 0|^a$ where $C = \log m \cdot \mathcal{S}_{\mathrm{Tr}}(\frac{s \log n}{\epsilon^2}, \frac{\log n}{\epsilon}, \epsilon)$ for a large enough constant in the big-O. We therefore need to decide between the cases

1. $\mathrm{Tr}[\Lambda_j \widetilde{\rho}] \geq 1 - \frac{0.01}{m}$ for some $j \in [m]$; or
2. $\mathrm{Tr}[\Lambda_j \widetilde{\rho}] \leq \frac{0.01}{m}$ for all $j \in [m]$.

This corresponds to the two cases of Lemma 3.2, where $\varepsilon = \varphi = \frac{0.01}{m}$. Because each M_j can be implemented with $\mathcal{T}_{\mathrm{Tr}}(\frac{s \log n}{\epsilon^2}, \frac{\log n}{\epsilon}, \epsilon)$ two-qubit gates, the total gate complexity of implementing the reflection $I - 2 \sum_{j=0}^{m-1} \Lambda_j \otimes |j\rangle\langle j|$ is also $\mathcal{T}_{\mathrm{Tr}}(\frac{s \log n}{\epsilon^2}, \frac{\log n}{\epsilon}, \epsilon)$. As a result, the total cost of applying Lemma 3.2 is $\tilde{O}\big(\sqrt{m} \mathcal{T}_{\mathrm{Tr}}(\frac{s \log n}{\epsilon^2}, \frac{\log n}{\epsilon}, \epsilon)\big)$.

In Lemma 3.2, we choose $\xi = \frac{1}{3}(\frac{(1-\varepsilon)^2}{4} - 3m\varphi)$ – this is a positive constant. We can thus tell the two cases apart with constant probability. Then, we repeat the call of Lemma 3.2 for $L = \Theta(\log \frac{\log n}{\epsilon^2})$ times and accept if and only if Lemma 3.2 accepts for at least $\frac{L}{2} \cdot (\frac{(1-\epsilon)^2}{4} + 3m\varphi)$ times. By Chernoff's bound, this can enhance the success probability to at least $1 - \frac{\epsilon^2}{400 \ln n}$.

In all, we have a quantum algorithm that determines whether there exists a $j \in [m]$ such that $\mathrm{Tr}[A_j\rho] \geq a_j + \epsilon$ with success probability at least $1 - \frac{\epsilon^2}{400 \ln n}$, using $\tilde{O}\big(\sqrt{m}\mathcal{T}_{\mathrm{Tr}}(\frac{s \log n}{\epsilon^2}, \frac{\log n}{\epsilon}, \epsilon)\big)$ quantum gates and queries to Oracle 1.1. To find this j, we apply binary search on $j \in \{1, 2, \ldots, m\}$, i.e., apply the algorithm to $j \in \{1, \ldots, \lfloor m/2 \rfloor\}$ and $j \in \{\lceil m/2 \rceil, \ldots, m\}$ respectively, and if the output is yes then call the algorithm recursively. This gives an extra $\mathrm{poly}(\log m)$ overhead on the queries to Oracle 1.1, which is still $\tilde{O}\big(\sqrt{m}\mathcal{T}_{\mathrm{Tr}}(\frac{s \log n}{\epsilon^2}, \frac{\log n}{\epsilon}, \epsilon)\big)$. In addition, similar to the analysis of Line 3, there is an overhead of $s \cdot \frac{16 \ln n}{\epsilon^2}$ for constructing Oracle 1.1 of the Gibbs state using Oracle 1.1 of each of the $A_{j^{(\tau)}}$. Therefore, the total cost of executing Line 4 of Algorithm 4 is

$$\frac{s}{\epsilon^2}\tilde{O}\Big(\sqrt{m}\mathcal{T}_{\mathrm{Tr}}\Big(\frac{s \log n}{\epsilon^2}, \frac{\log n}{\epsilon}, \epsilon\Big)\Big). \tag{4.2}$$

Because Algorithm 4 has at most $\frac{16 \ln n}{\epsilon^2}$ iterations, with success probability at least $1 - \frac{16 \ln n}{\epsilon^2} \cdot \frac{\epsilon^2}{400 \ln n} = 0.96$ Algorithm 4 works correctly, and its execution takes

$$\begin{aligned}
&\frac{16s \ln n}{\epsilon^4} \cdot \Big(\log m \cdot \mathcal{S}_{\mathrm{Tr}}\Big(\frac{s \log n}{\epsilon^2}, \frac{\log n}{\epsilon}, \epsilon\Big) \cdot \mathcal{T}_{\mathrm{Gibbs}}\Big(\frac{s \log n}{\epsilon^2}, \frac{\log n}{\epsilon}, \epsilon\Big) + \tilde{O}\Big(\sqrt{m}\mathcal{T}_{\mathrm{Tr}}(\frac{s \log n}{\epsilon^2}, \frac{\log n}{\epsilon}, \epsilon)\Big)\Big) \\
&= \frac{s}{\epsilon^4}\tilde{O}\Big(\mathcal{S}_{\mathrm{Tr}}\Big(\frac{s}{\epsilon^2}, \frac{1}{\epsilon}, \epsilon\Big)\mathcal{T}_{\mathrm{Gibbs}}\Big(\frac{s}{\epsilon^2}, \frac{1}{\epsilon}, \epsilon\Big) + \sqrt{m}\mathcal{T}_{\mathrm{Tr}}\Big(\frac{s}{\epsilon^2}, \frac{1}{\epsilon}, \epsilon\Big)\Big).
\end{aligned} \tag{4.3}$$

two-qubit gates and queries to Oracle 1.1. □

To be more explicit, the complexities of $\mathcal{S}_{\mathrm{Tr}}$, $\mathcal{T}_{\mathrm{Tr}}$, and $\mathcal{T}_{\mathrm{Gibbs}}$ are given in previous literatures:

Lemma 4.4 (Lemma 12, [11])**.** *Given an s-sparse $n \times n$ Hermitian matrix H with $\|H\| \leq 1$ and a density matrix ρ, with probability larger than $1 - p_e$, one can compute $\mathrm{Tr}[H\rho]$ with additive error ϵ in time $O(s\epsilon^{-2}\log^4(ns/p_e\epsilon))$ using $O(\epsilon^{-2}\log(1/p_e))$ copies of ρ. In other words, $\mathcal{S}_{\mathrm{Tr}}(s, 1, \epsilon) = O(1/\epsilon^2)$ and $\mathcal{T}_{\mathrm{Tr}}(s, 1, \epsilon) = O(s/\epsilon^2)$.*

Lemma 4.5 ([28])**.** *Given an s'-sparse $n \times n$ Hermitian matrix H with $\|H\| \leq \beta$ for some $\beta > 0$, one can prepare the Gibbs state $\frac{e^{-H}}{\mathrm{Tr}[e^{-H}]}$ with additive error ϵ using $\tilde{O}(\frac{\sqrt{\dim(H)}\beta s'}{\epsilon})$ calls to Oracle 1.1 of H and two-qubit gates. In other words, $\mathcal{T}_{Gibbs}(s, \Gamma, \epsilon) = \tilde{O}(s\Gamma\sqrt{n}/\epsilon)$.*

As a consequence of Theorem 4.3, Lemma 4.4, and Lemma 4.5, we have the following complexity result for solving SDPs under the plain model:

Corollary 4.6. *Assume we are given Oracle 1.1. Furthermore, assume that A_j is s-sparse for all $j \in [m]$. Then for any $\epsilon > 0$, feasibility of the SDP in (2.1), (2.2), and (2.3) can be tested by Algorithm 4 with success probability at least 0.96 and $\tilde{O}(s^2(\frac{\sqrt{m}}{\epsilon^{10}} + \frac{\sqrt{n}}{\epsilon^{12}}))$ quantum gates and queries to Oracle 1.1.*

Proof. Note that $\mathcal{S}_{\mathrm{Tr}}(s, \Gamma, \epsilon) = \mathcal{S}_{\mathrm{Tr}}(s, 1, \frac{\epsilon}{\Gamma})$ and $\mathcal{T}_{\mathrm{Tr}}(s, \Gamma, \epsilon) = \mathcal{T}_{\mathrm{Tr}}(s, 1, \frac{\epsilon}{\Gamma})$ by renormalizing the Hamiltonian H to H/Γ. As a result, plugging Lemma 4.4 and Lemma 4.5 into Theorem 4.3, the complexity of solving the SDP becomes

$$\frac{s}{\epsilon^4} \cdot \tilde{O}\Big(\frac{1}{\epsilon^4} \cdot \frac{s\sqrt{n}}{\epsilon^4} + \frac{s\sqrt{m}}{\epsilon^6}\Big) = \tilde{O}\Big(s^2\Big(\frac{\sqrt{m}}{\epsilon^{10}} + \frac{\sqrt{n}}{\epsilon^{12}}\Big)\Big). \tag{4.4}$$

□

Remark 4.1. The $(\sqrt{m}+\sqrt{n})$ dependence is optimal compared to [11, 6].

Remark 4.2. Using more elaborated techniques and analyses, Ref. [5] improved the complexity of Corollary 4.6 to $\tilde{O}(s(\frac{\sqrt{m}}{\epsilon^4}+\frac{\sqrt{n}}{\epsilon^5}))$.

5 Quantum SDP solver with quantum inputs

In this section, we illustrate our quantum SDP solver in the quantum input model. To that end, we first provide a precise formulation of the quantum input model, and then demonstrate how to implement Oracle 2.1 in such scenario and how the actual quantum algorithm works.

5.1 The quantum input model

As mentioned in the introduction, we would like to equip the quantum SDP solver with some extra power beyond only accessing the entries of the input matrices (i.e., A_j, $j = 1, \cdots, m$, each of $n \times n$ size). We imagine the setting where these A_js are nice so that the following oracles, representing various means to access A_js, can be efficiently implemented.

Oracle 5.1 (Oracle for traces of A_j)**.** A quantum oracle (unitary), denoted O_{Tr} (and its inverse $O_{\mathrm{Tr}}^\dagger$), such that for any $j \in [m]$,

$$O_{\mathrm{Tr}}|j\rangle|0\rangle|0\rangle = |j\rangle|\operatorname{Tr}[A_j^+]\rangle|\operatorname{Tr}[A_j^-]\rangle, \tag{5.1}$$

where A_j^+ and A_j^- are two PSD matrices such that $A_j = A_j^+ - A_j^-$ (the real values $\operatorname{Tr}[A_j^+]$ and $\operatorname{Tr}[A_j^-]$ are encoded into their binary representations).

Oracle 5.2 (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 $j \in [m]$,

$$O|j\rangle\langle j| \otimes |0\rangle\langle 0| \otimes |0\rangle\langle 0|O^\dagger = |j\rangle\langle j| \otimes |\psi_j^+\rangle\langle\psi_j^+| \otimes |\psi_j^-\rangle\langle\psi_j^-|, \tag{5.2}$$

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.[5]

Oracle 5.3 (Oracle for a_j)**.** A quantum oracle (unitary), denoted O_a (and its inverse $O_a^\dagger$), such that for any $j \in [m]$,

$$O_a|j\rangle\langle j| \otimes |0\rangle\langle 0|O_a^\dagger = |j\rangle\langle j| \otimes |a_j\rangle\langle a_j|, \tag{5.3}$$

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

Similar to Section 4, we also modularize the cost of two important blocks as follows.

[5] By tracing out the extra space, one can easily obtain states $A_j^+ / \operatorname{Tr}[A_j^+]$, $A_j^- / \operatorname{Tr}[A_j^-]$.

Definition 5.1 (trace estimation)**.** Assume that $\mathrm{Tr}(A_j^+) + \mathrm{Tr}(A_j^-) \leq B$ for some bound B for all $j \in [m]$. Then we define $\mathcal{S}_{\mathrm{Tr}}(B,\epsilon)$ and $\mathcal{T}_{\mathrm{Tr}}(B,\epsilon)$ as the sample complexity of a state $\rho \in \mathbb{C}^n$ and the time complexity of using the quantum input oracle Oracle 5.1, Oracle 5.2, Oracle 5.3, and two-qubit gates, respectively, such that there exists a quantum algorithm which distinguishes with success probability at least $1 - O(1/m)$ whether for a fixed $j \in [m]$, $\mathrm{Tr}(A_j\rho) > a_j + \epsilon$ or $\mathrm{Tr}(A_j\rho) \leq a_j$.

Definition 5.2 (Gibbs sampling)**.** Assume that $K = K^+ - K^-$, where $K^\pm = \sum_{j\in S} c_j A_j^\pm$, $S \subseteq [m]$ and $|S| \leq \Phi$, $c_j > 0$, and $A_j^\pm$ refers to either A_j^+ or A_j^- for all $j \in [m]$. Moreover, assume that $\mathrm{Tr}(K^+) + \mathrm{Tr}(K^-) \leq B_K$ for some bound B_K, and that K^+, K^- have rank at most r_K. Then we define $\mathcal{T}_{\mathrm{Gibbs}}(r_K, \Phi, B_K, \epsilon)$ as the complexity of preparing the Gibbs state $\rho_G = \exp(-K)/\mathrm{Tr}(\exp(-K))$ to ϵ precision in trace distance using Oracle 5.1, Oracle 5.2, Oracle 5.3, and two-qubit gates.

5.2 Implementation of Oracle 2.1 – searching a violated constraint

Using Oracle 5.1, Oracle 5.2, and Oracle 5.3, Oracle 2.1 can be implemented by the following lemma, using our fast quantum OR lemma (Lemma 3.2):

Lemma 5.3. *Given $\epsilon, \delta \in (0,1)$. Assume we have Oracle 5.1, Oracle 5.2, Oracle 5.3, and $(\log 1/\delta) \cdot \tilde{O}(\mathcal{S}_{\mathrm{Tr}}(B,\epsilon))$ copies of a state ρ. Assume either $\exists\, j \in [m]$ such that $\mathrm{Tr}(A_j\rho) \geq a_j + \epsilon$, or $\mathrm{Tr}(A_j\rho) \leq a_j$ for all $j \in [m]$. Then there is an algorithm that in the former case, finds such a j; and in the latter case, returns "FEASIBLE". This algorithm has success probability $1 - \delta$ and uses in total $\log 1/\delta \cdot \tilde{O}(\sqrt{m}\mathcal{T}_{\mathrm{Tr}}(B,\epsilon))$ quantum gates and queries to Oracle 5.1, Oracle 5.2, and Oracle 5.3.*

Proof. First, we use Definition 5.1 to create two-outcome POVMs M_j, acting on ρ, $|\psi_j^+\rangle\langle\psi_j^+|$, and $|\psi_j^-\rangle\langle\psi_j^-|$ with $C = \mathcal{S}_{\mathrm{Tr}}(B,\epsilon)$ copies, such that M_j decides with probability $1 - O(1/\operatorname{poly}(m))$ whether $\mathrm{Tr}(A_j\rho) - a_j > \epsilon$.

Because we are given purifications of all A_j^+ and A_j^- in Oracle 5.2, for all $j \in \{1,\ldots,m\}$ we can assume an explicit decomposition $M_j \otimes |0\rangle\langle 0|^{\otimes a} = P\Lambda_j P$, for some integer a, $P = I \otimes |0\rangle\langle 0|^{\otimes a}$, and some orthogonal projector Λ_j. Let $\widetilde{\rho} = \rho^{\otimes C} \otimes (|\psi_j^+\rangle\langle\psi_j^+|)^{\otimes C} \otimes (|\psi_j^-\rangle\langle\psi_j^-|)^{\otimes C} \otimes |0\rangle\langle 0|^a$. We therefore need to decide between the cases

1. $\mathrm{Tr}[\Lambda_j\widetilde{\rho}] \geq 1 - O(1/\operatorname{poly}(m))$ for some j; or
2. $\mathrm{Tr}[\Lambda_j\widetilde{\rho}] \leq O(1/\operatorname{poly}(m))$ for all j.

This corresponds to the two cases of Lemma 3.2, where both ε and δ are $O(1/\operatorname{poly}(m))$. To implement the the projection $I - 2\sum_{j=1}^m \Lambda_j \otimes |j\rangle\langle j|$ in Lemma 3.2, we use Oracle 5.2 to obtain purifications $|\psi_j^+\rangle\langle\psi_j^+|$ and $|\psi_j^-\rangle\langle\psi_j^-|$ of $\frac{A_j^+}{\mathrm{Tr}[A_j^+]}$ and $\frac{A_j^-}{\mathrm{Tr}[A_j^-]}$, and apply the reflection with respect to $|\psi_j^+\rangle$ and $|\psi_j^-\rangle$; note that we can obtain the numbers $\mathrm{Tr}[A_j^+]$ and $\mathrm{Tr}[A_j^-]$ in superposition by Oracle 5.1. Including the controlling ancilla $|j\rangle\langle j|$, the p in Lemma 3.2 is at most $O(\log m)$.

In Lemma 3.2, choose $\xi = \frac{1}{3}(\frac{(1-\varepsilon)^2}{4} - 3m\varphi)$ – this is a positive constant. We can thus tell the two cases apart with constant probability, using $\mathcal{S}_{\mathrm{Tr}}(B,\epsilon)$ samples of ρ and $\tilde{O}(\sqrt{m}) \cdot \mathcal{T}_{\mathrm{Tr}}(B,\epsilon)$ other operations. Then, we repeat the call of Lemma 3.2 for $L = \Theta(\log \delta^{-1})$ times and accept if and only if Lemma 3.2 accepts for at least $\frac{L}{2} \cdot (\frac{(1-\varepsilon)^2}{4} + 3m\varphi)$ times. By Chernoff's bound, this enhances the success probability to at least $1 - \delta$.

In all, we have a quantum algorithm that determines whether there exists a $j \in [m]$ such that $\mathrm{Tr}(A_j\rho) \geq a_j + \epsilon$ (or $\mathrm{Tr}(A_j\rho) \leq a_j$ for all $j \in [m]$) with success probability at least $1-\delta$, using $\log 1/\delta \cdot \tilde{O}(\sqrt{m}\mathcal{T}_{\mathrm{Tr}}(B,\epsilon))$ quantum gates and queries to Oracle 5.1, Oracle 5.2, and Oracle 5.3. To find this j, we take $\delta \leftarrow \delta/\log m$, and apply binary search on $j \in \{1,2,\ldots,m\}$, i.e., apply the algorithm to $j \in \{1,\ldots,\lfloor m/2 \rfloor\}$ and $j \in \{\lceil m/2 \rceil,\ldots,m\}$ respectively, and if the output is yes then call the algorithm recursively. This gives an extra $\mathrm{poly}(\log m)$ overhead on both sample complexity and gate complexity, which are still $(\log 1/\delta)\cdot\tilde{O}(\mathcal{S}_{\mathrm{Tr}}(B,\epsilon))$ and $\log 1/\delta \cdot \tilde{O}(\sqrt{m}\mathcal{T}_{\mathrm{Tr}}(B,\epsilon))$, respectively. □

5.3 Quantum SDP solvers with quantum inputs

We now instantiate Algorithm 2 to the fully quantum version (Algorithm 5). A key difference is that we use Definition 5.2 to generate (many copies) of the Gibbs state $\rho^{(t)}$ and rely on Lemma 5.3 to implement Oracle 2.1. At a high-level, the correctness of Algorithm 5 still roughly comes from Theorem 2.3, as well as Lemma 5.3. However, its gate complexity will be efficient because of the help of Oracle 5.1, Oracle 5.2, and Oracle 5.3.

Theorem 5.4. *Assume we are given Oracle 5.1, Oracle 5.2, and Oracle 5.3. Furthermore, assume* $\mathrm{Tr}[A_j^+] + \mathrm{Tr}[A_j^-] \leq B$ *for some bound* B*, and* A_j *have rank at most* r *for all* $j \in [m]$*. Then for any* $\epsilon > 0$*, feasibility of the SDP in (2.1), (2.2), and (2.3) can be tested by Algorithm 5 with success probability at least 0.96 and 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 5.1, Oracle 5.2, and Oracle 5.3.*

Algorithm 5: Efficiently testing the feasibility of SDPs: Quantum input model.

1 Initialize the weight matrix $W^{(1)} = I_n$, and $T = \frac{16\ln n}{\epsilon^2}$;
2 **for** $t = 1,2,\ldots,T$ **do**
3 Prepare $\tilde{O}(\mathcal{S}_{\mathrm{Tr}}(B,\epsilon))$ samples of the Gibbs state $\rho^{(t)} = \frac{W^{(t)}}{\mathrm{Tr}[W^{(t)}]}$ by Definition 5.2;
4 Using these $\tilde{O}(\mathcal{S}_{\mathrm{Tr}}(B,\epsilon))$ copies of $\rho^{(t)}$, search for a $j^{(t)} \in [m]$ such that $\mathrm{Tr}(A_{j^{(t)}}\rho^{(t)}) > a_{j^{(t)}} + \epsilon$ by Lemma 5.3 with $\delta = \frac{\epsilon^2}{400\ln n}$. Take $M^{(t)} = \frac{1}{2}(I_n - A_{j^{(t)}})$ if such $j^{(t)}$ is found; otherwise, claim that $\mathcal{S}_\epsilon \neq \emptyset$ (the SDP is feasible);
5 Define the new weight matrix: $W^{(t+1)} = \exp[-\frac{\epsilon}{2}\sum_{\tau=1}^{t} M^{(\tau)}]$;
6 Claim that $\mathcal{S}_0 = \emptyset$ and terminate the algorithm.

Proof of Theorem 5.4. The correctness of Algorithm 5 is automatically established by Theorem 2.3; it suffices to analyze the gate cost of Algorithm 5.

In Line 3 of Algorithm 5 we apply Definition 5.2 to compute the Gibbs state $\rho^{(t)}$. In round t, because $M_j = \frac{1}{2}[I_n - (A_j^+ - A_j^-)] = \frac{1}{2}I_n + \frac{1}{2}A_j^- - \frac{1}{2}A_j^+ \ \forall j \in [m]$, we take $K_t^+ = \frac{\epsilon}{2}\sum_{\tau=1}^{t}\frac{1}{2}A_{j^{(\tau)}}^+$ and $K_t^- = \frac{\epsilon}{2}\sum_{\tau=1}^{t}\frac{1}{2}A_{j^{(\tau)}}^-$. Because $t \leq \frac{16\ln n}{\epsilon^2}$, K_t^+, K_t^- have rank at most $t\cdot r = O(\log n \cdot r/\epsilon^2)$, and $\mathrm{Tr}[K_t^+]$, $\mathrm{Tr}[K_t^-]$ are at most $\frac{\epsilon t}{4}\cdot B = O(\log n \cdot B/\epsilon)$, Definition 5.2 guarantees that

$$\mathcal{T}_{\mathrm{Gibbs}}\Big(\frac{r\log n}{\epsilon^2},\frac{16\ln n}{\epsilon^2},\frac{B\log n}{\epsilon},\epsilon\Big) = \tilde{O}\Big(\mathcal{T}_{\mathrm{Gibbs}}\Big(\frac{r}{\epsilon^2},\frac{1}{\epsilon^2},\frac{B}{\epsilon},\epsilon\Big)\Big) \tag{5.4}$$

quantum gates and queries to Oracle 5.1, Oracle 5.2, and Oracle 5.3 suffice to prepare the Gibbs state $\rho^{(t)}$. Because there are at most $\frac{16\ln n}{\epsilon^2}$ iterations and in each iteration $\rho^{(t)}$ is prepared for $\tilde{O}(\mathcal{S}_{\mathrm{Tr}}(B,\epsilon))$ copies, in total the gate cost for Gibbs state preparation is

$$\frac{16\ln n}{\epsilon^2}\cdot\tilde{O}\Big(\mathcal{S}_{\mathrm{Tr}}(B,\epsilon)\mathcal{T}_{\mathrm{Gibbs}}\Big(\frac{r}{\epsilon^2},\frac{1}{\epsilon^2},\frac{B}{\epsilon},\epsilon\Big)\Big). \tag{5.5}$$

Furthermore, by Lemma 5.3, Line 4 finds a $j^{(t)}\in[m]$ such that $\mathrm{Tr}(A_{j^{(t)}}\rho^{(t)})>a_{j^{(t)}}+\epsilon$ with success probability at least $1-\frac{\epsilon^2}{400\ln n}$, using $\tilde{O}(\sqrt{m}\mathcal{T}_{\mathrm{Tr}}(B,\epsilon))$ quantum gates and queries to Oracle 5.1, Oracle 5.2, and Oracle 5.3. Because Algorithm 5 has at most $\frac{16\ln n}{\epsilon^2}$ iterations, with success probability at least $1-\frac{16\ln n}{\epsilon^2}\cdot\frac{\epsilon^2}{400\ln n}=0.96$ we can assume that Lemma 5.3 works correctly, and the total cost of running Line 4 is

$$\frac{16\ln n}{\epsilon^2}\cdot\tilde{O}\big(\sqrt{m}\mathcal{T}_{\mathrm{Tr}}(B,\epsilon)\big). \tag{5.6}$$

In total, by (5.5) and (5.6), the time complexity of executing Algorithm 5 is

$$\begin{aligned}&\frac{16\ln n}{\epsilon^2}\cdot\tilde{O}\Big(\mathcal{S}_{\mathrm{Tr}}(B,\epsilon)\mathcal{T}_{\mathrm{Gibbs}}\Big(\frac{r}{\epsilon^2},\frac{1}{\epsilon^2},\frac{B}{\epsilon},\epsilon\Big)\Big)+\frac{16\ln n}{\epsilon^2}\cdot\tilde{O}\big(\sqrt{m}\mathcal{T}_{\mathrm{Tr}}(B,\epsilon)\big)\\&=\frac{1}{\epsilon^2}\tilde{O}\Big(\mathcal{S}_{\mathrm{Tr}}(B,\epsilon)\mathcal{T}_{\mathrm{Gibbs}}\Big(\frac{r}{\epsilon^2},\frac{1}{\epsilon^2},\frac{B}{\epsilon},\epsilon\Big)+\sqrt{m}\mathcal{T}_{\mathrm{Tr}}(B,\epsilon)\Big).\end{aligned} \tag{5.7}$$

□

To be more explicit, in later sections we prove that:

- Lemma 5.6: $\mathcal{S}_{\mathrm{Tr}}(B,\epsilon)=\mathcal{T}_{\mathrm{Tr}}(B,\epsilon)=O(B^2\log m/\epsilon^2)$.
- Lemma 5.8: $\mathcal{T}_{\mathrm{Gibbs}}(r_K,\Phi,B_K,\epsilon)=O(\Phi\cdot\mathrm{poly}(\log n,r_K,B_K,\epsilon^{-1}))$.

As a consequence, we have the following complexity result for solving SDPs under the quantum input model:

Corollary 5.5. *Assume we are given Oracle 5.1, Oracle 5.2, and Oracle 5.3. Furthermore, assume* $\mathrm{Tr}[A_j^+]+\mathrm{Tr}[A_j^-]\le B$ *for some bound* B*, and* A_j *have rank at most* r *for all* $j\in[m]$*. Then for any* $\epsilon>0$*, feasibility of the SDP in (2.1), (2.2), and (2.3) can be tested by Algorithm 5 with success probability at least 0.96 and at most* $(\sqrt{m}+\mathrm{poly}(r))\cdot\mathrm{poly}(\log m,\log n,B,\epsilon^{-1})$ *quantum gates and queries to Oracle 5.1, Oracle 5.2, and Oracle 5.3.*

Proof. By Theorem 5.4, the complexity of solving the SDP is

$$\begin{aligned}&\frac{1}{\epsilon^2}\tilde{O}\Big(\frac{B^2\log m}{\epsilon^2}\cdot\frac{1}{\epsilon^2}\,\mathrm{poly}\Big(\log n,\frac{r}{\epsilon},\frac{B}{\epsilon},\frac{1}{\epsilon}\Big)+\sqrt{m}\cdot\frac{B^2\log m}{\epsilon^2}\Big)\\&=(\sqrt{m}+\mathrm{poly}(r))\cdot\mathrm{poly}(\log m,\log n,B,\epsilon^{-1}).\end{aligned} \tag{5.8}$$

□

Remark 5.1. When we use Definition 5.2 to prepare the Gibbs state $\rho^{(t)}$ in Line 3 of Algorithm 5, we have $W^{(t)} = -\frac{\epsilon t}{4} I_n + K_t^+ - K_t^-$ by Line 5 which actually has an extra $-\frac{\epsilon t}{4} I_n$ term. However, for any constant $c \in \mathbb{R}$ and Hermitian matrix H we have

$$\frac{e^{cI-H}}{\mathrm{Tr}[e^{cI-H}]} = \frac{e^c e^{-H}}{\mathrm{Tr}[e^c e^{-H}]} = \frac{e^{-H}}{\mathrm{Tr}[e^{-H}]}, \tag{5.9}$$

hence this $-\frac{\epsilon t}{4} I_n$ term does not change $\rho^{(t)}$.

Remark 5.2. In Corollary 5.5, the only restriction on the decomposition $A_j = A_j^+ - A_j^-$ for all $j \in [m]$ is that $\mathrm{Tr}[A_j^+] + \mathrm{Tr}[A_j^-] \le B$. If we assume this decomposition to be the eigen-decomposition, i.e., A_j^+ represents the subspace spanned by the eigenvectors of A_j with positive eigenvalues, and A_j^- represents the subspace spanned by the eigenvectors of A_j with negative eigenvalues, then by the low-rank assumption and $-I \preceq A_j \preceq I$, $\mathrm{Tr}[A_j^+] + \mathrm{Tr}[A_j^-] \le r$. In this case, Corollary 5.5 takes at most $\sqrt{m} \cdot \mathrm{poly}(\log m, \log n, r, \epsilon^{-1})$ quantum gates and queries to Oracle 5.1, Oracle 5.2, and Oracle 5.3.

Remark 5.3. The $\sqrt{m}$ dependence is optimal compared to Theorem 5.9 proved later.

Remark 5.4. Using more elaborated techniques and analyses, Ref. [5] explicitly computed the degrees of the parameters in (5.8) and improved the complexity of Corollary 5.5 to $\tilde{O}(\frac{B\sqrt{m}}{\epsilon^4} + \frac{B^{3.5}}{\epsilon^{7.5}})$ (the rank r is implicitly contained in B and hence this complexity is independent of r).

5.4 Trace estimation

In this subsection, we prove:

Lemma 5.6. *Assume we are given Oracle 5.1, Oracle 5.2, Oracle 5.3, and $O(B^2 \log m/\epsilon^2)$ copies of a state $\rho \in \mathbb{C}^n$, where $\mathrm{Tr}[A_j^+] + \mathrm{Tr}[A_j^-] \le B$ for some bound B for all $j \in [m]$. Then for any $\epsilon > 0$, Algorithm 6 distinguishes whether $\mathrm{Tr}(A_j\rho) > a_j + \epsilon$ or $\mathrm{Tr}(A_j\rho) \le a_j$ with success probability at least $1 - O(1/\mathrm{poly}(m))$. In other words, $\mathcal{S}_{\mathrm{Tr}}(B, \epsilon) = \mathcal{T}_{\mathrm{Tr}}(B, \epsilon) = O(B^2 \log m/\epsilon^2)$.*

Algorithm 6: Implementation of the POVM M_j.

1 Using Oracle 5.2, apply the SWAP test on ρ and $\frac{A_j^+}{\mathrm{Tr}[A_j^+]}$ for $\mathrm{poly}(\log m, \log n, B, \epsilon^{-1})$ times. Denote the frequency of getting 1 to be $\widetilde{p_{j,+}}$;

2 Using Oracle 5.2, apply the SWAP test on ρ and $\frac{A_j^-}{\mathrm{Tr}[A_j^-]}$ for $\mathrm{poly}(\log m, \log n, B, \epsilon^{-1})$ times. Denote the frequency of getting 1 to be $\widetilde{p_{j,-}}$;

3 Apply Oracle 5.1 to compute $\mathrm{Tr}[A_j^+]$ and $\mathrm{Tr}[A_j^-]$. Claim that $\mathrm{Tr}(A_j\rho) > a_j + \epsilon$ if $(2\widetilde{p_{j,+}} - 1)\,\mathrm{Tr}[A_j^+] - (2\widetilde{p_{j,-}} - 1)\,\mathrm{Tr}[A_j^-] > a_j + \epsilon/2$, and claim that $\mathrm{Tr}(A_j\rho) < a_j$ if $(2\widetilde{p_{j,+}} - 1)\,\mathrm{Tr}[A_j^+] - (2\widetilde{p_{j,-}} - 1)\,\mathrm{Tr}[A_j^-] \le a_j + \epsilon/2$;

Proof of Lemma 5.6. Recall that the SWAP test [12] on ρ and $\frac{A_j^+}{\operatorname{Tr}[A_j^+]}$ outputs 1 with probability $\frac{1}{2} + \frac{\operatorname{Tr}(A_j^+\rho)}{2\operatorname{Tr}[A_j^+]}$, and the SWAP test on ρ and $\frac{A_j^-}{\operatorname{Tr}[A_j^-]}$ outputs 1 with probability $\frac{1}{2} + \frac{\operatorname{Tr}(A_j^-\rho)}{2\operatorname{Tr}[A_j^-]}$. Therefore, by Chernoff's bound and the fact that $\operatorname{Tr}[A_j^+], \operatorname{Tr}[A_j^-] \le B$, we have

$$\Pr\Big[\Big|\widetilde{p_{j,+}} - \Big(\frac{1}{2} + \frac{\operatorname{Tr}(A_j^+\rho)}{2\operatorname{Tr}[A_j^+]}\Big)\Big| \ge \frac{\epsilon}{8\operatorname{Tr}[A_j^+]}\Big] \le \Pr\Big[\Big|\widetilde{p_{j,+}} - \Big(\frac{1}{2} + \frac{\operatorname{Tr}(A_j^+\rho)}{2\operatorname{Tr}[A_j^+]}\Big)\Big| \ge \frac{\epsilon}{8B}\Big] \tag{5.10}$$

$$\le 2e^{-\frac{O(B^2\log m/\epsilon^2)\cdot\epsilon^2}{64B^2\cdot 2}} \tag{5.11}$$

$$\le O\Big(\frac{1}{\operatorname{poly}(m)}\Big) \tag{5.12}$$

for a large constant in the big-O in (5.11). Similarly,

$$\Pr\Big[\Big|\widetilde{p_{j,-}} - \Big(\frac{1}{2} + \frac{\operatorname{Tr}(A_j^-\rho)}{2\operatorname{Tr}[A_j^-]}\Big)\Big| \ge \frac{\epsilon}{8\operatorname{Tr}[A_j^-]}\Big] \le O\Big(\frac{1}{\operatorname{poly}(m)}\Big). \tag{5.13}$$

In other words, with probability at least $1 - O(\frac{1}{\operatorname{poly}(m)})$,

$$\big|(2\widetilde{p_{j,+}} - 1)\operatorname{Tr}[A^+] - \operatorname{Tr}(A_j^+\rho)\big| \le \frac{\epsilon}{4}, \quad \big|(2\widetilde{p_{j,-}} - 1)\operatorname{Tr}[A_j^-] - \operatorname{Tr}(A_j^-\rho)\big| \le \frac{\epsilon}{4}. \tag{5.14}$$

Therefore, if $\operatorname{Tr}(A_j\rho) = \operatorname{Tr}(A_j^+\rho) - \operatorname{Tr}(A_j^-\rho) > a_j + \epsilon$, then with probability at least $1 - O(\frac{1}{\operatorname{poly}(m)})$,

$$(2\widetilde{p_{j,+}} - 1)\operatorname{Tr}[A_j^+] - (2\widetilde{p_{j,-}} - 1)\operatorname{Tr}[A_j^-] > a_j + \epsilon/2, \tag{5.15}$$

which is exactly the first part of Line 3. Similarly, we can use Chernoff's bound to prove that if $\operatorname{Tr}(A_j\rho) \le a_j$, then with probability at least $1 - O(\frac{1}{\operatorname{poly}(m)})$,

$$(2\widetilde{p_{j,+}} - 1)\operatorname{Tr}[A_j^+] - (2\widetilde{p_{j,-}} - 1)\operatorname{Tr}[A_j^-] \le a_j + \epsilon/2, \tag{5.16}$$

which is the second part of Line 3.

Because Algorithm 6 only uses SWAP which only takes $O(1)$ quantum gates, in total we have $\mathcal{S}_{\operatorname{Tr}}(B,\epsilon) = \mathcal{T}_{\operatorname{Tr}}(B,\epsilon) = O(B^2\log m/\epsilon^2)$. □

5.5 Gibbs state preparation

With the access to Oracle 5.1 and Oracle 5.2, the following lemma shows how to prepare two normalized quantum states $K^\pm/\operatorname{Tr}[K^\pm]$ where $K^\pm = \sum_{j\in S} c_j A_j^\pm$, $c_j > 0$ and $A_j^\pm$ refers to either A_j^+ or A_j^-.

Lemma 5.7. *$K^{sgn}/\operatorname{Tr}[K^{sgn}]$ can be prepared by $|S|$ samples to Oracle 5.1 and one sample to Oracle 5.2, for both $sgn = +$ and $sgn = -$.*

Proof. Consider the following protocol, where we choose all $\pm$ to be $+$ when preparing $K^+/\operatorname{Tr}[K^+]$, and choose all $\pm$ to be $-$ when preparing $K^-/\operatorname{Tr}[K^-]$:

1. For all $j \in S$, sample Oracle 5.1 to obtain $\mathrm{Tr}[A_j^{\pm}]$;

2. To prepare $K^{\pm}/\mathrm{Tr}[K^{\pm}]$, toss a coin $i \in S$ such that $\Pr[i=j] = \frac{c_j \mathrm{Tr}[A_j^{\pm}]}{\sum_{k\in S} c_k \mathrm{Tr}[A_k^{\pm}]}$, take one sample of Oracle 5.2 to obtain $A_j^{\pm}/\mathrm{Tr}[A_j^{\pm}]$, and output this state.

By symmetry, we only consider the preparation of $K^{\pm}/\mathrm{Tr}[K^{\pm}]$. With probability $\frac{c_j \mathrm{Tr}[A_j^{\pm}]}{\sum_{k\in S} c_k \mathrm{Tr}[A_k^{\pm}]}$, the output state is $A_j^{\pm}/\mathrm{Tr}[A_j^{\pm}]$; therefore, in average the density matrix prepared is

$$\sum_{j\in S} \frac{c_j \mathrm{Tr}[A_j^{\pm}]}{\sum_{k\in S} c_k \mathrm{Tr}[A_k^{\pm}]} \cdot \frac{A_j^{\pm}}{\mathrm{Tr}[A_j^{\pm}]} = \frac{\sum_{j\in S} c_j A_j^{\pm}}{\sum_{k\in S} c_k \mathrm{Tr}[A_k^{\pm}]} = \frac{K^{\pm}}{\mathrm{Tr}[K^{\pm}]}. \tag{5.17}$$

Furthermore, Step 1 takes $|S|$ samples to Oracle 5.1, and Step 2 takes one sample to Oracle 5.2; this exactly matches the sample complexity claimed in Lemma 5.7. □

Combining Lemma 5.7 and Theorem 7.4 leads to a lemma that generates the Gibbs state in Line 3 of Algorithm 2:

Lemma 5.8. *Suppose $K = K^+ - K^-$, where $K^{\pm} = \sum_{j\in S} c_j A_j^{\pm}$, $c_j > 0$ and $A_j^{\pm}$ refers to either A_j^+ or A_j^-. Moreover, assume that $\mathrm{Tr}(K^+) + \mathrm{Tr}(K^-) \le B_K$ for some bound B_K, and that K^+, K^- have rank at most r_K. Then it is possible to prepare the Gibbs state $\rho_G = \exp(-K)/\mathrm{Tr}(\exp(-K))$ to ϵ precision in trace distance, with $|S| \cdot \mathrm{poly}(\log n, r_K, B_K, \epsilon^{-1})$ quantum gates and queries to Oracle 5.1 and Oracle 5.2. In other words, $\mathcal{T}_{Gibbs}(r_K, \Phi, B_K, \epsilon) = O(\Phi \cdot \mathrm{poly}(\log n, r_K, B_K, \epsilon^{-1}))$.*

5.6 Lower bound for quantum SDP solvers with quantum inputs

In this section, we prove quantum lower bounds in the quantum input setting.

Theorem 5.9 (Lower bound on Theorem 5.4)**.** *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 5.1 and Oracle 5.2.*

Proof. Consider the following two instances of the SDP feasibility testing problem:

1. For all $j \in [m]$, set $A_j^- = 0$. For a random $i^* \in [n]$, set $(A_j^+)_{i^*i^*} = 1$ for all $j \in [m]$. All other elements of matrices A_j^+ are set to zero. For a random $j^* \in [m]$, set $a_{j^*} = -1/2$. Set $a_j = 1/2$ for all $j \neq j^*$. Set $\epsilon = 1/4$.

2. For all $j \in [m]$, set $A_j^- = 0$. For a random $i^* \in [n]$, set $(A_j^+)_{i^*i^*} = 1$ for all $j \in [m]$. All other elements of matrices A_j^+ are set to zero. Set $a_j = 1/2$ for all $j \in [m]$. Set $\epsilon = 1/4$.

Note that the first problem is not feasible because there is no X such that $X \succeq 0$ and $\mathrm{Tr}[A_{j^*}X] \le -1/2 + 1/4 < 0$; the second problem is always feasible. For both problems, we have $B = r = 1$, and the state $\frac{A_j^+}{\mathrm{Tr}[A_j^+]}$ is always $|i^*\rangle\langle i^*|$ for all $j \in [m]$. Therefore, Oracle 5.2 provides no information for distinguishing between the two problems, and we should only rely on Oracle 5.1. But this is equivalent to searching for the j^* such that $a_{j^*} = -1/2$, and by reduction to the lower bound on Grover search it takes at least $\Omega(\sqrt{m})$ queries to Oracle 5.1 for distinguishing between the two problems. □

Combining Theorem 5.4 and Theorem 5.9, we obtain the optimal bound on SDP feasibility testing using Oracle 5.1 and Oracle 5.2, up to poly-logarithmic factors.

6 Application: efficient learnability of quantum states

We consider the following quantum state learning problem, also named "shadow tomography" in [3].

Question 6.1. *Let ρ be an unknown quantum state in an n-dimensional Hilbert space, $E_1, \ldots, E_m$ be known two-outcome POVMs, and $0 < \epsilon < 1$. Given independent copies of ρ, one wants to obtain an explicit quantum circuit for a state σ such that with probability at least $2/3$, $|\operatorname{Tr}[\sigma E_i] - \operatorname{Tr}[\rho E_i]| \leq \epsilon \; \forall \, i \in [m]$. What is the sample complexity (i.e., the number of required copies of ρ) and gate complexity (i.e., the total running time) of the best such procedure?*

Aaronson provides a solution with the sample complexity (i.e., the number of copies of ρ) of $\tilde{O}\big(\log^4 m \cdot \log n/\epsilon^5\big)$ in [3]. In this section we show that, for low rank matrices and small m, we can also make the learning process *computationally efficient* while keeping a comparable sample complexity, by using our previous result on speeding up solutions to SDPs.

6.1 Reduction of Question 6.1 to SDP feasibility

We start with a simple explanation of using the solution to SDP feasibility to address Question 6.1. Given (many copies of) any unknown quantum state ρ and two-outcome POVM $E_1, \ldots, E_m$, in order to estimate $\operatorname{Tr}[\rho E_i]$, it suffices to find a state σ that is the solution to the following SDP feasibility problem:

$$\operatorname{Tr}[\sigma E_i] \leq \operatorname{Tr}[\rho E_i] + \epsilon \quad \forall \, i \in [m]; \tag{6.1}$$

$$\operatorname{Tr}[\sigma E_i] \geq \operatorname{Tr}[\rho E_i] - \epsilon \quad \forall \, i \in [m]; \tag{6.2}$$

$$\operatorname{Tr}[\sigma] = 1; \tag{6.3}$$

$$\sigma \succeq 0. \tag{6.4}$$

Any feasible solution σ satisfies that $|\operatorname{Tr}[\sigma E_i] - \operatorname{Tr}[\rho E_i]| < \epsilon$ for all $i \in [m]$. Thus, our quantum SDP solver will generate a description of such σ. However, we do not know $\operatorname{Tr}[\rho E_i]$, and hence the constraints of the SDP feasibility problem, in advance. The *key* observation is that our SDP solver only relies on the implementation of Oracle 2.1, which does not need the knowledge of $\operatorname{Tr}[\rho E_i]$ for each i explicitly. It turns out that with the help of the fast quantum OR lemma, one only needs a few copies of ρ for the implementation of Oracle 2.1.

6.2 Finding the violated constraint using $\tilde{O}(\sqrt{m})$ gates

Similar to Section 5, we assume the existence of Oracle 5.1 and Oracle 5.2 to achieve efficient quantum circuits. Specifically, for the feasibility problem (6.1)-(6.4), we have:

Oracle 5.1 for traces of E_i: A unitary O_{Tr} such that for any $i \in [m]$, $O_{\mathrm{Tr}}|i\rangle|0\rangle = |i\rangle|\operatorname{Tr}[E_i]\rangle$.

Oracle 5.2 for preparing E_i: A unitary O (and its inverse $O^\dagger$) acting on $\mathbb{C}^m \otimes (\mathbb{C}^n \otimes \mathbb{C}^n)$ such that for any $i \in [m]$,

$$O|i\rangle\langle i| \otimes |0\rangle\langle 0|O^\dagger = |i\rangle\langle i| \otimes |\psi_i\rangle\langle\psi_i|, \tag{6.5}$$

where $|\psi_i\rangle\langle\psi_i|$ is any purification of $\frac{E_i}{\mathrm{Tr}[E_i]}$.

Furthermore, we assume that the POVM operator E_i has rank at most r, for all $i \in [m]$. Using Oracle 5.1 and Oracle 5.2, Oracle 2.1 (searching for violation) can be implemented by the following lemma:

Lemma 6.2. *Given $\epsilon, \delta \in (0,1)$. Assume we have Oracle 5.1, Oracle 5.2, and $\mathrm{poly}(\log m, \log n, r, \epsilon^{-1}, \log \delta^{-1})$ copies of two states $\rho, \sigma \in \mathbb{C}^n$. Assume either $\exists\, i \in [m]$ such that $|\mathrm{Tr}[\sigma E_i] - \mathrm{Tr}[\rho E_i]| \geq \epsilon$, or $|\mathrm{Tr}[\sigma E_i] - \mathrm{Tr}[\rho E_i]| \leq \epsilon/2$ for all $i \in [m]$. Then there is an algorithm that in the former case, finds such an i; and in the latter case, returns "FEASIBLE". This algorithm has success probability $1-\delta$ and uses in total $\sqrt{m} \cdot \mathrm{poly}(\log m, \log n, r, \epsilon^{-1}, \log \delta^{-1})$ quantum gates and queries to Oracle 5.1 and Oracle 5.2.*

Lemma 6.2 also follows from our fast quantum OR lemma (Lemma 3.2) by combining Lemma 5.6 and Lemma 5.3, where we replace ρ by $\rho^{\otimes C} \otimes \sigma^{\otimes C} \otimes |0\rangle\langle 0|^a$ for some $C = \mathrm{poly}(\log m, \log n, \epsilon^{-1})$; also notice that because $0 \leq E_i \leq I$ and $\mathrm{rank}(E_i) \leq r$, we have $B \leq r$. As a result, the detailed proof is omitted here.

6.3 Gate complexity of learning quantum states

Similar to Corollary 5.5, we solve Question 6.1 by using Lemma 5.8 to generate (copies) of the Gibbs state $\rho^{(t)}$ and relying on Lemma 6.2 to implement Oracle 2.1.

Corollary 6.3. *Assume we are given Oracle 5.1 and Oracle 5.2. Then for any $\epsilon > 0$, Question 6.1 can be solved by Algorithm 7 with success probability at least 0.96, using at most $\mathrm{poly}(\log m, \log n, r, \epsilon^{-1})$ copies of ρ, and at most $\sqrt{m} \cdot \mathrm{poly}(\log m, \log n, r, \epsilon^{-1})$ quantum gates and queries to Oracle 5.1 and Oracle 5.2.*

Algorithm 7: Efficiently learn a quantum state via measurements.

Initialize the weight matrix $W^{(1)} = I_n$, and $T = \frac{16 \ln n}{\epsilon^2}$;
for $t = 1, 2, \ldots, T$ **do**
  Prepare $\mathrm{poly}(\log m, \log n, r, \epsilon^{-1})$ samples of the Gibbs state $\rho^{(t)} = \frac{W^{(t)}}{\mathrm{Tr}[W^{(t)}]}$ by Lemma 5.8, and take $\mathrm{poly}(\log m, \log n, r, \epsilon^{-1})$ copies of ρ;
  Using these $\mathrm{poly}(\log m, \log n, r, \epsilon^{-1})$ copies of $\rho^{(t)}$ and ρ, apply Lemma 6.2 with $\delta = \frac{\epsilon^2}{400 \ln n}$ to search for an $i^{(t)} \in [m]$ such that $|\mathrm{Tr}[\rho E_{i^{(t)}}] - \mathrm{Tr}[\rho^{(t)} E_{i^{(t)}}]| \geq \epsilon$. **if** *such $i^{(t)}$ is found* **then**
    if $(\mathrm{Tr}[\rho E_{i^{(t)}}] - \mathrm{Tr}[\rho^{(t)} E_{i^{(t)}}] \geq \epsilon)$ **then**
      Take $M^{(t)} = \frac{1}{2}\big(I_n - (-E_{i^{(t)}} + \mathrm{Tr}[\rho E_{i^{(t)}}] I_n)\big)$;
    else $(\mathrm{Tr}[\rho E_{i^{(t)}}] - \mathrm{Tr}[\rho^{(t)} E_{i^{(t)}}] \leq -\epsilon)$
      Take $M^{(t)} = \frac{1}{2}\big(I_n - (E_{i^{(t)}} - \mathrm{Tr}[\rho E_{i^{(t)}}] I_n)\big)$;
  else (no such $i^{(t)}$ exists)
    Claim $\rho^{(t)}$ to be the solution, and terminate the algorithm;
  Define the new weight matrix: $W^{(t+1)} = \exp[-\frac{\epsilon}{2} \sum_{\tau=1}^{t} M^{(\tau)}]$;

Proof. Similar to Corollary 5.5, the correctness of Algorithm 7 is automatically established by Theorem 2.3; it suffices to analyze the gate cost of Algorithm 7.

In Line 3 of Algorithm 7 we apply Lemma 5.8 to compute the Gibbs state $\rho^{(t)}$. In round t, because either

$$M^{(t)} = \frac{1}{2}\big(I_n - (-E_{i^{(t)}} + \mathrm{Tr}[\rho E_{i^{(t)}}]I_n)\big) = \frac{1 - \mathrm{Tr}[\rho E_{i^{(t)}}]}{2} I_n + \frac{1}{2} E_{i^{(t)}} \tag{6.6}$$

when Line 6 executes, or

$$M^{(t)} = \frac{1}{2}\big(I_n - (E_{i^{(t)}} - \mathrm{Tr}[\rho E_{i^{(t)}}]I_n)\big) = \frac{1 + \mathrm{Tr}[\rho E_{i^{(t)}}]}{2} I_n - \frac{1}{2} E_{i^{(t)}} \tag{6.7}$$

when Line 8 executes, we can take $K_t^+ = \frac{\epsilon}{2}\sum_{\tau=1}^{t} \frac{1}{2} E_{i^{(\tau)}}^+$ and $K_t^- = \frac{\epsilon}{2}\sum_{\tau=1}^{t} \frac{1}{2} E_{i^{(\tau)}}^-$, where $E_{i^{(\tau)}}^+ = E_{i^{(\tau)}}$, $E_{i^{(\tau)}}^- = 0$ when (6.6) holds for round τ, and $E_{i^{(\tau)}}^+ = 0$, $E_{i^{(\tau)}}^- = E_{i^{(\tau)}}$ when (6.7) holds for round τ. Because $t \le \frac{16 \ln n}{\epsilon^2}$, K_t^+, K_t^- have rank at most $t \cdot r = O(\log n \cdot r/\epsilon^2)$ and $\mathrm{Tr}[K_t^+]$, $\mathrm{Tr}[K_t^-]$ are at most $\frac{\epsilon t}{4} \cdot r = O(\log n \cdot r/\epsilon)$, Lemma 5.8 guarantees that

$$\frac{16 \ln n}{\epsilon^2} \cdot \mathrm{poly}\left(\log n, \frac{r \log n}{\epsilon^2}, \frac{r \log n}{\epsilon}, \epsilon^{-1}\right) = \mathrm{poly}(\log n, r, \epsilon^{-1}) \tag{6.8}$$

quantum gates and queries to Oracle 5.1 and Oracle 5.2 suffice to prepare the Gibbs state $\rho^{(t)}$. Because there are at most $\frac{16 \ln n}{\epsilon^2} = \mathrm{poly}(\log n, \epsilon^{-1})$ iterations and in each iteration $\rho^{(t)}$ is prepared for $\mathrm{poly}(\log m, \log n, r, \epsilon^{-1})$ copies, in total the gate cost for Gibbs state preparation is $\mathrm{poly}(\log m, \log n, r, \epsilon^{-1})$.

Furthermore, by Lemma 6.2, Algorithm 7 finds an $i^{(t)} \in [m]$ such that $|\mathrm{Tr}[\rho E_{i^{(t)}}] - \mathrm{Tr}[\rho^{(t)} E_{i^{(t)}}]| \ge \epsilon$ with success probability at least $1 - \frac{\epsilon^2}{400 \ln n}$, using $\mathrm{poly}(\log m, \log n, r, \epsilon^{-1})$ copies of ρ, and $\sqrt{m} \cdot \mathrm{poly}(\log m, \log n, r, \epsilon^{-1})$ quantum gates and queries to Oracle 5.1 and Oracle 5.2. Because Algorithm 7 has at most $\frac{16 \ln n}{\epsilon^2}$ iterations, with success probability at least $1 - \frac{16 \ln n}{\epsilon^2} \cdot \frac{\epsilon^2}{400 \ln n} = 0.96$ we can assume that the quantum search in Lemma 6.2 works correctly, and the total gate cost of calling Algorithm 7 is $\sqrt{m} \cdot \mathrm{poly}(\log m, \log n, r, \epsilon^{-1})$.

In conclusion, $\mathrm{poly}(\log m, \log n, r, \epsilon^{-1})$ is an upper bound on the number of copies of ρ, and $\sqrt{m} \cdot \mathrm{poly}(\log m, \log n, r, \epsilon^{-1})$ is an upper bound on the total number of quantum gates and queries to Oracle 5.1 and Oracle 5.2. □

Remark 6.1. Using the same idea as Theorem 5.9, we can prove that there exists a shadow tomography problem such that $r, \epsilon = \Theta(1)$, and solving the problem requires $\Omega(\sqrt{m})$ calls to Oracle 5.1 and Oracle 5.2. Therefore Corollary 6.3 is also optimal up to poly-logarithmic factors.

7 Gibbs sampling of low-rank Hamiltonians

In this section, we demonstrate how to sample from the Gibbs state of low-rank Hamiltonians given a quantum oracle generating desired states. We repeatedly use the following result of [25] (with a straightforward generalization in [22]):

Lemma 7.1 ([25, 22])**.** *Suppose we are given a quantum oracle that prepares copies of two unknown (normalized) l-qubit quantum states ρ^+ and ρ^-, and we wish to evolve under the Hamiltonian $H = a_+\rho^+ - a_-\rho^-$ for some nonnegative numbers $a_+, a_- \ge 0$. Then we can approximately implement the unitary $\exp(iHt)$ up to diamond-norm error δ, using $O(a^2t^2/\delta)$ copies of ρ^+ and ρ^- and $O(la^2t^2/\delta)$ other 1- or 2-qubit gates, where $a = a_+ + a_-$.*

By using phase estimation on the operator $\exp(iHt)$ with $t = O(1/a)$, we have

Lemma 7.2. *Under the same assumptions as Lemma 7.1, we can perform eigenvalue estimation of H: given an eigenstate of H, we can estimate its eigenvalue up to precision ϵ, with probability $1-\xi$, using $O(a^2\epsilon^{-2}\xi^{-2})$ copies of ρ^+ and ρ^- and $O(la^2\epsilon^{-2}\xi^{-2})$ other 1- or 2-qubit gates, where $a = a_+ + a_-$. This procedure disturbs the input state by at most a trace distance error of $O(\sqrt{\xi})$.*

In the following proof, we instead assume that eigenvalue estimation of H can be done exactly. This assumption is not true, but it helps to simplify the exposition; the assumption will be removed in Appendix A.

7.1 Computing the partition function

As a warm-up, we start with the following lemma:

Lemma 7.3. *Suppose $K = K^+ - K^-$, where K^+ and K^- are $n \times n$ PSD matrices, and there is a quantum oracle that prepares copies of the states $\rho^+ = K^+/\operatorname{Tr}(K^+)$, $\rho^- = K^-/\operatorname{Tr}(K^-)$, and an oracle for the numbers $\operatorname{Tr}(K^+)$, $\operatorname{Tr}(K^-)$. Moreover, assume that $\operatorname{Tr}(K^+) + \operatorname{Tr}(K^-) \le B$ for some bound B,[6] and that K^+, K^- have rank at most r_K. Then it is possible to estimate the partition function $Z = \operatorname{Tr}(\exp(-K))$ to multiplicative error ϵ with success probability at least $1-\xi$, with $\operatorname{poly}(\log n, r_K, B, \epsilon^{-1}, \xi^{-1})$ quantum gates.*

Proof Sketch. As mentioned above, we assume that we can implement the unitary evolution $\exp(iKt)$ as well as the phase estimation protocol, perfectly to infinite precision. This idealization is made here for the sake of flashing out the core ideas behind the proposed protocol. These assumptions will be lifted in Appendix A (Lemma A.3), where a careful error analysis of this scheme is presented.

Under these assumptions, let us first consider the estimation of

$$Z_{\text{supp}} \equiv \sum_{|\lambda_i| \ge \delta} e^{-\lambda_i}, \tag{7.1}$$

where $0 < \delta < 1$ is a small threshold and λ_i's are eigenvalues of K. Since $\delta > 0$ is a small parameter, Z_{supp} is the partition function when considering the approximated support of K.

The main idea in the estimation of Z_{supp} is to perform phase estimation of the unitary operator $e^{2\pi i K}$ on ρ^+ and ρ^-, after which we obtain

$$\rho^\pm = \frac{K^\pm}{\operatorname{Tr}(K^\pm)} \to \bar{\rho}^\pm = \frac{1}{\operatorname{Tr}(K^\pm)} \sum_\lambda \Pi_\lambda K^\pm \Pi_\lambda \otimes |\lambda\rangle\langle\lambda|, \tag{7.2}$$

where Π_λ is the projection onto the λ-eigenspace of K, and λ is any eigenvalue of K. Let us define

$$K^+_\lambda := \Pi_\lambda K^+ \Pi_\lambda, \quad K^-_\lambda := \Pi_\lambda K^- \Pi_\lambda. \tag{7.3}$$

Then,

$$K^+_\lambda - K^-_\lambda = \Pi_\lambda K \Pi_\lambda = \lambda \Pi_\lambda, \tag{7.4}$$

[6] The B here is denoted as B_K in Definition 5.2 and Lemma 5.8; for simplicity we make this abbreviation throughout Section 7 and Appendix A.

and therefore K_λ^+ and K_λ^- differ by a multiple of the identity in their support space (the λ-eigenspace of K). Hence K_λ^+ and K_λ^- are simultaneously diagonalizable, and their corresponding eigenvalues differ by exactly λ. In other words, there exists an eigenbasis of K, which we call $\{|v_i\rangle\}_i$ with corresponding eigenvalues λ_i, such that K_λ^+ and K_λ^- are diagonal in this eigenbasis for all λ. We can therefore write

$$K_\lambda^+ = \sum_{i:\lambda_i=\lambda} \lambda_i^+ |v_i\rangle\langle v_i|, \qquad K_\lambda^- = \sum_{i:\lambda_i=\lambda} \lambda_i^- |v_i\rangle\langle v_i|, \tag{7.5}$$

for some nonnegative numbers λ_i^+, λ_i^- satisfying $\lambda_i^+ - \lambda_i^- = \lambda_i$. Combining Eqs. (7.2) and (7.5), we obtain that $\bar{\rho}^+$ ($\bar{\rho}^-$) – the state after performing phase estimation of the unitary operator $e^{2\pi i K}$ on ρ^+ (ρ^-) is given by

$$\bar{\rho}^\pm = \frac{1}{\mathrm{Tr}(K^\pm)} \sum_{\lambda_i} \lambda_i^\pm |v_i\rangle\langle v_i| \otimes |\lambda_i\rangle\langle\lambda_i|. \tag{7.6}$$

Now consider the following procedure, and let its output be the random variable X:

Algorithm 8: Estimation of Z_{supp}

1. Let $\mathrm{sgn} = +$ with probability $\mathrm{Tr}(K^+)/[\mathrm{Tr}(K^+) + \mathrm{Tr}(K^-)]$, and $\mathrm{sgn} = -$ otherwise.
2. Perform phase estimation of the operator $e^{2\pi i K}$ on ρ^{sgn}; Let the output state be $\bar{\rho}^{\mathrm{sgn}} = \frac{1}{\mathrm{Tr}(K^{\mathrm{sgn}})} \sum_{\lambda_i} \lambda_i^{\mathrm{sgn}} |v_i\rangle\langle v_i| \otimes |\lambda_i\rangle\langle\lambda_i|$. Measure the second register and let the obtained eigenvalue of K be λ.
3. If $|\lambda| < \delta$ output 0; else if $\mathrm{sgn} = +$ output $\lambda^{-1} e^{-\lambda}$; else output $-\lambda^{-1} e^{-\lambda}$.

Then, under the assumption of perfect phase estimation, we have

$$\begin{aligned}
\mathbf{E}[X] &= \frac{\mathrm{Tr}(K^+)}{\mathrm{Tr}(K^+) + \mathrm{Tr}(K^-)} \sum_{|\lambda_i| \geq \delta} \frac{\lambda_i^+}{\mathrm{Tr}(K^+)} \frac{e^{-\lambda_i}}{\lambda_i} - \frac{\mathrm{Tr}(K^-)}{\mathrm{Tr}(K^+) + \mathrm{Tr}(K^-)} \sum_{|\lambda_i| \geq \delta} \frac{\lambda_i^-}{\mathrm{Tr}(K^-)} \frac{e^{-\lambda_i}}{\lambda_i} \\
&= \frac{1}{\mathrm{Tr}(K^+) + \mathrm{Tr}(K^-)} \sum_{|\lambda_i| \geq \delta} e^{-\lambda_i} = \frac{Z_{\mathrm{supp}}}{\mathrm{Tr}(K^+) + \mathrm{Tr}(K^-)},
\end{aligned} \tag{7.7}$$

where $\lambda_i^\pm$ are the eigenvalues of $K_{\lambda_i}^\pm$, satisfying $\lambda_i^+ - \lambda_i^- = \lambda_i$. Therefore $\mathbf{E}[X]$ is proportional to Z_{supp}, and obtaining a multiplicative estimate of $\mathbf{E}[X]$ gives us a multiplicative estimate of Z_{supp}.

The second moment of X reads

$$\mathbf{E}[X^2] = \frac{1}{\mathrm{Tr}(K^+) + \mathrm{Tr}(K^-)} \sum_{|\lambda_i| \geq \delta} (\lambda_i^+ + \lambda_i^-) \frac{e^{-2\lambda_i}}{\lambda_i^2} \leq \max_{|\lambda_i| \geq \delta} |\lambda_i|^{-2} e^{-2\lambda_i} \leq \delta^{-2} Z_{\mathrm{supp}}^2. \tag{7.8}$$

We see that $\mathbf{E}[X^2] \leq B^2 \delta^{-2} \mathbf{E}[X]^2$, and therefore by Chebyshev's inequality we can obtain, with constant probability, an ϵ-error multiplicative estimate of $\mathbf{E}[X]$, hence of Z_{supp}, by running the above procedure $O(B^2 \delta^{-2} \epsilon^{-2})$ times and taking the mean.

We still need to calculate Z, the full partition function including small eigenvalues of K. Let R denote the number of eigenvalues of K (including degeneracy) with absolute value at least δ, and note that $R \leq 2r_K$, where recall that r_K upper bounds the rank of K^+ and K^- . Define the following approximation of Z:

$$Z' \equiv Z_{\mathrm{supp}} + (n - R) = \sum_{|\lambda_i| \geq \delta} e^{-\lambda_i} + \sum_{|\lambda_i| < \delta} e^0. \tag{7.9}$$

Using $e^{\delta} \leq 1 + 2\delta$ and $e^{-\delta} \geq 1 - \delta$, we get that

$$|Z - Z'| \leq 2\delta(n - R). \tag{7.10}$$

Therefore if we make δ small enough, say $\delta = O(\epsilon)$, Z' gives a good multiplicative estimate for Z.

To compute Z', we need a good multiplicative estimate of $n - R$. This can essentially be done by estimating the probability of a random state having eigenvalue smaller than δ. Let the output of the following procedure be Y:

Algorithm 9: Estimation of $n - R$

1. Perform phase estimation of the operator $e^{2\pi i K}$ on the uniformly random state I/n; let the output eigenvalue be λ.
2. If $|\lambda| < \delta$ output 1; otherwise output 0.

Y is a Bernoulli random variable with mean $\mathbf{E}[Y] = (n - R)/n$ and variance $\mathrm{Var}[Y] = R(n - R)/n^2 \leq R\mathbf{E}[Y]^2$. By Chebyshev's inequality, $O(r_K \epsilon^{-2})$ repetitions of the above procedure gives us an ϵ-error multiplicative estimate of $\mathbf{E}[Y]$, and thus of $n - R$.

Putting everything together, we see that $O(B^2\epsilon^{-4} + r_K\epsilon^{-2})$ uses of (perfect) phase estimation of $e^{2\pi i K}$ suffices to get a $O(\epsilon)$-error multiplicative estimate of Z, completing the proof. □

7.2 Sampling from the Gibbs state

Theorem 7.4 (Full proof deferred to Appendix A)**.** *Suppose $K = K^+ - K^-$, where K^+ and K^- are $n \times n$ PSD matrices, and there is a quantum oracle that prepares copies of the states $\rho^+ = K^+/\operatorname{Tr}(K^+)$, $\rho^- = K^-/\operatorname{Tr}(K^-)$, and an oracle for the numbers $\operatorname{Tr}(K^+)$, $\operatorname{Tr}(K^-)$. Moreover, assume that $\operatorname{Tr}(K^+) + \operatorname{Tr}(K^-) \leq B$ for some bound B, and that K^+, K^- have rank at most r_K. Then it is possible to prepare the Gibbs state $\rho_G = \exp(-K)/\operatorname{Tr}(\exp(-K))$ to ϵ precision in trace distance, with $\operatorname{poly}(\log n, r_K, B, \epsilon^{-1})$ quantum gates.*

Proof sketch. Similar to the proof sketch of the partition function, here as well we assume an infinite precision implementation of the unitary evolution operator $\exp(iKt)$ as well as of the phase estimation protocol. In addition we assume that quantum principal component analysis can be implemented perfectly. These assumptions will be lifted in Appendix A (Theorem A.4), where a complete proof is presented.

The procedure is somewhat similar to that of calculating the partition function above. We pick $\delta = O(\epsilon)$, a small threshold, and first consider a procedure to sample from

$$\rho_{\mathrm{supp}} \equiv \sum_{|\lambda_i| \geq \delta} e^{-\lambda_i} |v_i\rangle\langle v_i| / Z_{\mathrm{supp}}, \tag{7.11}$$

where λ_i's and $|v_i\rangle$'s are eigenvalues and eigenstates of K. (In the case that ρ_{supp} is undefined, i.e. that all eigenvalues of K have magnitude less than δ, it is easy to see that the uniformly mixed state I/n is already an $O(\epsilon)$-trace distance error approximation to ρ_G. This is the case when $Z_{\mathrm{supp}} = 0$.) ρ_{supp} is the Gibbs state when considering only the (approximated) support of K. Consider the procedure in Algorithm 10.

Algorithm 10: Estimation of ρ_{supp}

1. Let $\text{sgn} = +$ with probability $\text{Tr}(K^+)/[\text{Tr}(K^+) + \text{Tr}(K^-)]$, and $\text{sgn} = -$ otherwise.
2. Perform phase estimation of the unitary operator $e^{2\pi i K}$ on ρ^{sgn}; let the output state be $\bar{\rho}^{\text{sgn}} = \frac{1}{\text{Tr}(K^{\text{sgn}})} \sum_{\lambda_i} \lambda_i^{\text{sgn}} |v_i\rangle\langle v_i| \otimes |\lambda_i\rangle\langle\lambda_i|$.
3. Project $\bar{\rho}^{\text{sgn}}$ onto $\bar{\varrho}^{\text{sgn}} = \frac{1}{\text{Tr}(K^{\text{sgn}})} \sum_{\lambda_i : |\lambda_i| \geq \delta} \lambda_i^{\text{sgn}} |v_i\rangle\langle v_i| \otimes |\lambda_i\rangle\langle\lambda_i|$.
4. The average state at this stage is

$$\bar{\varrho} = \frac{1}{\text{Tr}(K^+) + \text{Tr}(K^-)} \sum_{\lambda_i : |\lambda_i| \geq \delta} (\lambda_i^+ + \lambda_i^-) |v_i\rangle\langle v_i| \otimes |\lambda_i\rangle\langle\lambda_i|.$$

   Perform phase estimation of the operator $e^{2\pi i K}$ on $\bar{\varrho}$; let the measured eigenvalue be $\mu = \lambda^+ + \lambda^-$, and the resulting state be ϱ_μ.
5. Accept the state ϱ_μ with probability $\frac{\delta}{\mu} \frac{(1-\epsilon)e^{-\lambda_i}}{Z'_{\text{supp}}}$, for Z'_{supp} a ϵ-multiplicative error approximation of Z_{supp} by Algorithm 8.

Note that $\frac{\delta}{\lambda^+ + \lambda^-} \frac{(1-\epsilon)e^{-\lambda}}{Z'_{\text{supp}}} \leq 1$ since $\lambda^\pm \geq 0$ and by assumption $|\lambda| = |\lambda^+ - \lambda^-| \geq \delta$, and $Z'_{\text{supp}} \leq (1+\epsilon) Z_{\text{supp}} \leq (1+\epsilon) \sum_{|\lambda_i| \geq \delta} e^{-\lambda_i}$ by (7.1). Moreover assuming that K has at least one eigenvalue with magnitude at least δ, the success probability in Line 5 of Algorithm 10 is at least

$$\frac{\delta}{\text{Tr}(K^+) + \text{Tr}(K^-)} \sum_{\lambda_i : |\lambda_i| \geq \delta} \frac{(1-\epsilon)e^{-\lambda_i}}{Z'_{\text{supp}}} \geq \frac{\delta(1-\epsilon)}{B(1+\epsilon)}, \tag{7.12}$$

and therefore we can output ρ_{supp} efficiently by repeating Algorithm 10 until success, which takes $O(B/\delta)$ trials in expectation.

Accounting for the randomness in Step 1, at the end of Step 3, we obtain the mixed state $\bar{\varrho}$. However, for Gibbs sampling, we should have factors of the form $e^{-\lambda_i} |v_i\rangle\langle v_i|$ instead of $(\lambda_i^+ + \lambda_i^-) |v_i\rangle\langle v_i|$ that appear in $\bar{\varrho}$. Therefore, at this stage of the protocol, to accept $|v_i\rangle\langle v_i|$ with probability proportional to $e^{-\lambda_i}/(\lambda_i^+ + \lambda_i^-)$, but for that we need to measure $\lambda_i^+ + \lambda_i^-$. This is done in steps 4 and 5 of the above procedure, which is equivalent to applying $\sum_{\lambda_i : |\lambda_i| \geq \delta} \frac{\delta}{\lambda_i^+ + \lambda_i^-} \frac{e^{-\lambda_i}}{Z_{\text{supp}}} |v_i\rangle\langle v_i| \otimes |\lambda_i\rangle\langle\lambda_i|$ to $\bar{\varrho}$. Upon keeping only the first register we obtain

$$\frac{\delta}{\text{Tr}(K^+) + \text{Tr}(K^-)} \sum_{|\lambda_i| \geq \delta} \frac{e^{-\lambda_i}}{Z_{\text{supp}}} |v_i\rangle\langle v_i| \propto \frac{1}{Z_{\text{supp}}} \sum_{|\lambda_i| \geq \delta} e^{-\lambda_i} |v_i\rangle\langle v_i| \equiv \rho_{\text{supp}}, \tag{7.13}$$

where ρ_{supp} is the Gibbs state when considering only the (approximated) support of K.

We still need to calculate ρ_G, the full Gibbs state including small eigenvalues of K. Recall that R denotes the number of eigenvalues (including degeneracy) of K with absolute value at least δ, and note that $R \leq 2r_K$, where r_K upper bounds the rank of K^+ and K^- . Define the following

approximation of ρ_G:

$$\rho'_G \equiv \frac{Z_{\text{supp}}}{Z'}\rho_{\text{supp}} + \frac{n-R}{Z'}\rho_{\text{ker}} = \frac{1}{Z'}\Big(\sum_{|\lambda_i|\geq\delta} e^{-\lambda_i}|v_i\rangle\langle v_i| + \sum_{|\lambda_i|<\delta}|v_i\rangle\langle v_i|\Big), \tag{7.14}$$

where $\rho_{\text{ker}} = \frac{1}{n-R}\sum_{|\lambda_i|<\delta}|v_i\rangle\langle v_i|$ is the uniformly random state on the orthogonal complement of the (approximate) support of K. Then

$$\|\rho_G - \rho'_G\|_{\text{Tr}} = \Big|\frac{1}{Z} - \frac{1}{Z'}\Big|\sum_{|\lambda_i|\geq\delta} e^{-\lambda_i} + \sum_{|\lambda_i|<\delta}\Big|\frac{e^{-\lambda_i}}{Z} - \frac{1}{Z'}\Big| \tag{7.15}$$

$$\leq \Big|\frac{1}{Z} - \frac{1}{Z'}\Big|\sum_{|\lambda_i|\geq\delta} e^{-\lambda_i} + \left[\sum_{|\lambda_i|<\delta}\Big|\frac{1}{Z} - \frac{1}{Z'}\Big| + \frac{2\delta}{Z}e^{-\lambda_i}\right] \tag{7.16}$$

$$\leq \Big|\frac{1}{Z} - \frac{1}{Z'}\Big|Z' + 2\delta \leq 4\delta. \tag{7.17}$$

Therefore if we make δ small enough, ρ'_G gives a good estimate (in trace distance) for ρ_G.

To estimate ρ_{ker}, we consider the output of Algorithm 11 below.

Algorithm 11: Estimation of ρ_{ker}

1. Perform phase estimation of the operator $e^{2\pi iK}$ on the uniformly random state I/n; let the output eigenvalue be λ and the resulting state be Π_λ.
2. If $|\lambda| \geq \delta$ abort; otherwise, accept the state.

Finally, ρ_G is generated by running the Algorithm 10 with probability $\frac{Z_{\text{supp}}}{Z} = \Omega(\frac{\delta}{B})$ (by (7.12)) until we accept ρ_{supp}, and running the Algorithm 11 with probability

$$\frac{n-R}{Z} \geq \frac{n-2r_K}{n} = \Omega(1), \tag{7.18}$$

until we accept ρ_{ker}. The detailed analysis is presented in Appendix A.3.

In the previous subsection we proved that, upon setting $\delta = O(\epsilon)$ we can obtain Z_{supp}, Z and $n-R$ up to an $O(\epsilon)$ multiplicative error with $\text{poly}(\log n, r_K, B, \epsilon^{-1})$ quantum gates. Therefore, Using Lemma 7 of [6] we obtain $\frac{Z_{\text{supp}}}{Z}$ and $\frac{n-R}{Z}$ to $O(\epsilon)$ multiplicative error. This, in turns, implies the with the above procedure we prepare the Gibbs state ρ_G up to error $O(\epsilon)$ in trace distance, with $\text{poly}(\log n, r_K, B, \epsilon^{-1})$ quantum gates. □

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 [5] with us. FB was supported by NSF. CYL and AK are supported by the Department of Defense. TL is supported by NSF CCF-1526380.

References

[1] Scott Aaronson. The learnability of quantum states. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 463(2088):3089–3114, 2007.

[2] Scott Aaronson. The learnability of quantum states. In *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, volume 463, pages 3089–3114. The Royal Society, 2007.

[3] Scott Aaronson. Shadow tomography of quantum states. 2017.

[4] Scott Aaronson, Xinyi Chen, Elad Hazan, and Ashwin Nayak. Online learning of quantum states. arXiv:1802.09025, 2018.

[5] Joran van Apeldoorn and András Gilyén. Improvements in quantum SDP-solving with applications. arXiv:1804.05058, 2018.

[6] 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.

[7] 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.

[8] 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.

[9] Charles H. Bennett, Ethan Bernstein, Gilles Brassard, and Umesh Vazirani. Strengths and weaknesses of quantum computing. *SIAM Journal on Computing*, 26(5):1510–1523, 1997.

[10] Fernando G. S. L. Brandão, Amir Kalev, Tongyang Li, Cedric Yen-Yu Lin, Krysta M. Svore, and Xiaodi Wu. Exponential quantum speed-ups for semidefinite programming with applications to quantum learning. arXiv:1710.02581v1, 2017.

[11] 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.

[12] Harry Buhrman, Richard Cleve, John Watrous, and Ronald de Wolf. Quantum fingerprinting. *Physical Review Letters*, 87(16):167902, 2001.

[13] Anirban Narayan Chowdhury and Rolando D. Somma. Quantum algorithms for Gibbs sampling and hitting-time estimation. *arXiv:1603.02940*, 2016.

[14] Ronald de Wolf. Personal communication, Nov, 2017.

[15] 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.

[16] 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.

[17] 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.

[18] Elad Hazan. Efficient algorithms for online convex optimization and their applications. Phd Thesis, Princeton University, 2006.

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

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

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

[22] 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.

[23] 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*, STOC '15, pages 567–576, New York, NY, USA, 2015. ACM.

[24] 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.

[25] Seth Lloyd, Masoud Mohseni, and Patrick Rebentrost. Quantum principal component analysis. *arXiv:1307.0401*, 2013.

[26] Daniel Nagaj, Pawel Wocjan, and Yong Zhang. Fast amplification of QMA. *arXiv:0904.1549*, 2009.

[27] 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.

[28] 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.

[29] Amnon Ta-Shma. Inverting well conditioned matrices in quantum logspace. In *Proceedings of the Forty-fifth Annual ACM Symposium on Theory of Computing*, pages 881–890. ACM, 2013.

[30] Xiaodi Wu. Parallelized solution to semidefinite programmings in quantum complexity theory. *arXiv:1009.2211*, 2010.

A Proof for Gibbs sampling of low-rank states

In this section we provide a complete proof of Lemma 7.3 and Theorem 7.4 given in Section 7.

A.1 Preliminaries

Before we start the proof, we first gather some preliminary facts that we need. First of all, the output of the phase estimation protocol is probabilistic and depends on the measurement. This is a problem since we often want the output of phase estimation to be consistent across multiple runs of the protocol. For example, when the eigenvalue is to our cutoff δ, the phase estimation protocol may not be able to consistently decide whether it was greater than or less than δ. To overcome this problem, in the proofs below we use Ta-Shma's *consistent phase estimation* algorithm instead:

Lemma A.1 ([29])**.** *Let U be a D-dimensional unitary matrix, and $\delta, \xi > 0$. There is a quantum algorithm that first chooses a random shift s, such that with probability at least $1 - D\xi$ the following holds for* all *eigenstates $|v_\lambda\rangle$ of U (where $U|v_\lambda\rangle = e^{2\pi i\lambda}|v_\lambda\rangle$) (in this case we call s a* good *shift):*

- *On input $|v_\lambda\rangle|\bar{0}\rangle$, where $|\bar{0}\rangle$ is a fixed reference state, the algorithm outputs a state $O(\sqrt{\xi})$-close to $|v_\lambda\rangle|f(s,\lambda)\rangle$ in trace distance.*
- *$f(s,\lambda)$ is a function only of s and λ, and $|f(s,\lambda) - \lambda| < \delta$.*

This algorithm requires $\mathrm{poly}(\xi^{-1}, \delta^{-1})$ *uses of the controlled-U operation and other quantum gates.*

The essential idea of this algorithm is to choose a random shift s, and perform phase estimation on $e^{is}U$ instead. If the precision δ is small enough, then with high probability over s, the eigenvalue $s + \lambda$ will always be far away from any half-multiple of δ (i.e. a number of the form $(z + 0.5)\delta$, $z \in \mathbb{Z}$), for all λ. The result of phase estimation will therefore (with high probability) depend only on λ, and not on the measurement.

Using the consistent phase estimation together with Lemma 7.1, we can straightforwardly derive the following lemma:

Lemma A.2. *Suppose we are given a quantum oracle that prepares copies of two unknown (normalized) n-qubit quantum states ρ^+ and ρ^-, and define the Hamiltonian $H = a_+\rho^+ - a_-\rho^-$. Also assume the ranks of ρ^+ and ρ^- are upper bounded by r. Then for $\delta, \xi > 0$, there is a quantum algorithm that first chooses a random shift s, such that with probability at least $1 - 2r\xi$ the following holds for all eigenstates $|v_i\rangle$ of H, where $H|v_i\rangle = \lambda_i|v_i\rangle$ (we call such a s a* good *shift):*

- *On input $|v_i\rangle|\bar{0}\rangle$, where $|\bar{0}\rangle$ is a fixed reference state, the algorithm outputs a state $O(\sqrt{\xi})$-close to $|v_\lambda\rangle|f(s,\lambda_i)\rangle$ in trace distance.*
- *$f(s,\lambda_i)$ is a function only of s and λ_i, and $|f(s,\lambda_i) - \lambda_i| < \delta$.*

This algorithm requires $\mathrm{poly}(a^+ + a^-, \xi^{-1}, \delta^{-1})$ *copies of ρ^+ and ρ^-, and* $\mathrm{poly}(n, a^+ + a^-, \xi^{-1}, \delta^{-1})$ *1- and 2-qubit quantum gates.*

This, in particular, allows us to consistently estimate eigenvalues of $K = K^+ - K^-$ using $\mathrm{poly}(\log n, B, \xi^{-1}, \delta^{-1})$ operations in total.

For technical reasons, we will also need an approximation of the minimum eigenvalue of K (possibly ignoring eigenvalues less than a threshold δ). We use the following procedure:

Algorithm 12: Estimation of minimum eigenvalue of K

1. **Input:** Quantum oracles for ρ^+, ρ^-. A random good shift s for eigenvalue estimation of K. Numbers $\delta, \gamma > 0$.
2. Use consistent phase estimation to estimate the eigenvalue of K on ρ^+ and ρ^-, with precision δ and error probability $\xi = O(B^{-1}\delta / \log \gamma^{-1})$. Discard the estimate if its absolute value is less than δ.
3. Repeat Step 2 $\Theta(B\delta^{-1} \log \gamma^{-1})$ times and output the minimum, denoted $\tilde{\lambda}_{\min}$.

Lemma A.2 implies that that with probability $1 - O(\gamma)$, the above algorithm outputs the minimum number $\tilde{\lambda}_{\min}$ such that $|\tilde{\lambda}_{\min}| \geq \delta$ and $\tilde{\lambda}_{\min} = f(s, \lambda)$ for some eigenvalue λ of K.

Finally, for operators A and B, we will use $A \approx_{O(\epsilon)} B$ to denote that A is $O(\epsilon)$-close to B in trace distance.

A.2 Computing the partition function

We will prove the following lemma, using consistent phase estimation protocol:

Lemma A.3. *Suppose $K = K^+ - K^-$, where K^+ and K^- are $n \times n$ PSD matrices, and there is a quantum oracle that prepares copies of the states $\rho^+ = K^+ / \operatorname{Tr}(K^+)$, $\rho^- = K^- / \operatorname{Tr}(K^-)$, and an oracle for the numbers $\operatorname{Tr}(K^+)$, $\operatorname{Tr}(K^-)$. Moreover, assume that $\operatorname{Tr}(K^+) + \operatorname{Tr}(K^-) \leq B$ for some bound B, and that K^+, K^- have rank at most r_K. Then it is possible to estimate the partition function $Z = \operatorname{Tr}(\exp(-K))$ to multiplicative error ϵ with success probability at least $1 - \xi$, with $\operatorname{poly}(\log n, r_K, B, \epsilon^{-1}, \xi^{-1})$ quantum gates.*

Proof. As stated previously, we are using consistent phase estimation to unambiguously decide whether to keep an eigenvector in our approximate support. To be precise, choose $\delta = O(\epsilon)$, $\xi = O(\epsilon^2 \delta^2 B^{-2} r_K^{-1})$, and pick a random shift s – assume that this s is a good shift (this happens with probability $1 - O(\epsilon^2 \delta^2 B^{-2})$). Define $\widetilde{Z}_{\text{supp}} = \sum_{|f(s,\lambda_i)| \geq \delta} e^{-f(s,\lambda_i)}$, and consider Algorithm 13 for estimating $\widetilde{Z}_{\text{supp}}$ (let its output be $\widetilde{X}$). In Step 3 we need to discard eigenvalues smaller than the approximate minimum eigenvalue $\tilde{\lambda}_{\min}$ to keep the expectation of $\tilde{X}$ well-bounded. This is one consequence of possible error due to the application of the phase estimation procedure.

Another consequence, is that if some eigenvalues of K are close enough, they could be mapped to the same approximation $\tilde{\lambda}$, and are therefore treated as degenerate. We will redo the analysis to illustrate this fact: Recall that Π_λ be the projection onto the λ-eigenspace of K. Define the projector

$$\widetilde{\Pi}_{\tilde{\lambda}} = \sum_{\lambda : f(s,\lambda) = \tilde{\lambda}} \Pi_\lambda \tag{A.1}$$

to be the projector that projects onto the set of eigenvectors of K, with eigenvalues that get mapped to $\tilde{\lambda}$ under our consistent phase estimation procedure. We note that if we define the unnormalized states (here id stands for "ideal")

$$\widetilde{K}^+_{\tilde{\lambda},\text{id}} = \widetilde{\Pi}_{\tilde{\lambda}} K^+ \widetilde{\Pi}_{\tilde{\lambda}}, \qquad \widetilde{K}^-_{\tilde{\lambda},\text{id}} = \widetilde{\Pi}_{\tilde{\lambda}} K^- \widetilde{\Pi}_{\tilde{\lambda}} \tag{A.2}$$

Algorithm 13: Estimation of $\widetilde{Z}_{\text{supp}}$

1. Let $\text{sgn} = +$ with probability $\text{Tr}(K^+)/[\text{Tr}(K^+) + \text{Tr}(K^-)]$, and $\text{sgn} = -$ otherwise.
2. Use consistent phase estimation to perform eigenvalue estimation of K on ρ^{sgn}; let the output be the normalized state $\tilde{\rho}^{\text{sgn}} = \frac{1}{\text{Tr}[K^{\text{sgn}}]} \sum_{\tilde{\lambda}} \widetilde{K}^{\text{sgn}}_{\tilde{\lambda}} \otimes |\tilde{\lambda}\rangle\langle\tilde{\lambda}|$ for some unnormalized states $\widetilde{K}^{\text{sgn}}_{\tilde{\lambda}}$. Measure the obtained eigenvalue to obtain some $\tilde{\lambda}$. With probability at least $1 - O(\xi)$, $\tilde{\lambda} = f(s, \lambda)$ for some eigenvalue λ of K.
3. Compute $\tilde{\lambda}_{\min}$ by Algorithm 12. If $|\tilde{\lambda}| < \delta$ or $\tilde{\lambda} < \tilde{\lambda}_{\min}$ output 0; else if $\text{sgn} = +$ output $\tilde{\lambda}^{-1} e^{-\tilde{\lambda}}$; else output $-\tilde{\lambda}^{-1} e^{-\tilde{\lambda}}$.

then

$$\widetilde{K}^+_{\tilde{\lambda},\text{id}} - \widetilde{K}^-_{\tilde{\lambda},\text{id}} = \widetilde{\Pi}_{\tilde{\lambda}} K \widetilde{\Pi}_{\tilde{\lambda}} = \sum_{i:f(s,\lambda_i)=\tilde{\lambda}} \lambda_i |v_i\rangle\langle v_i| \approx_{\xi} \tilde{\lambda} \sum_{i:f(s,\lambda_i)=\tilde{\lambda}} |v_i\rangle\langle v_i| = \tilde{\lambda}\widetilde{\Pi}_{\tilde{\lambda}}. \tag{A.3}$$

Note that consistent phase estimation implements an operation $O(\sqrt{\xi})$-close to the operation $\sum_{\tilde{\lambda}} \widetilde{\Pi_{\lambda}} \otimes |\tilde{\lambda}\rangle\langle\tilde{\lambda}|$, and therefore

$$\tilde{\rho}^{\text{sgn}} \equiv \frac{1}{\text{Tr}[K^{\text{sgn}}]} \sum_{\tilde{\lambda}} \widetilde{K}^{\text{sgn}}_{\tilde{\lambda}} \otimes |\tilde{\lambda}\rangle\langle\tilde{\lambda}| \approx_{O(\sqrt{\xi})} \frac{1}{\text{Tr}[K^{\text{sgn}}]} \sum_{\tilde{\lambda}} \widetilde{\Pi}_{\tilde{\lambda}} K^{\text{sgn}} \widetilde{\Pi}_{\tilde{\lambda}} \otimes |\tilde{\lambda}\rangle\langle\tilde{\lambda}| = \frac{1}{\text{Tr}[K^{\text{sgn}}]} \sum_{\tilde{\lambda}} \widetilde{K}^{\text{sgn}}_{\tilde{\lambda},\text{id}} \otimes |\tilde{\lambda}\rangle\langle\tilde{\lambda}|. \tag{A.4}$$

Thus $\tilde{K}^{\text{sgn}}_{\tilde{\lambda}} \approx_{O(B\sqrt{\xi})} \tilde{K}^{\text{sgn}}_{\tilde{\lambda},\text{id}}$, and hence

$$\tilde{K}^+_{\tilde{\lambda}} - \tilde{K}^-_{\tilde{\lambda}} \approx_{O(B\sqrt{\xi})} \tilde{\lambda}\widetilde{\Pi}_{\tilde{\lambda}}. \tag{A.5}$$

We see that the consistent phase estimation of Step 2 serves to approximately project ρ^+ or ρ^- onto the span of eigenvectors of K with eigenvalue approximately equal to some $\tilde{\lambda}$; and on this space, the unnormalized output states at Step 2 approximately differ only on by a multiple of the identity on their support. There is therefore a basis of vectors $\{|\tilde{v}_i\rangle\}$ where $\widetilde{K}^+_{\tilde{\lambda}}$ and $\widetilde{K}^-_{\tilde{\lambda}}$ are approximately diagonal for all $\tilde{\lambda}$. These vectors are approximate eigenvectors of K, i.e.

$$\|K|\tilde{v}_i\rangle - \tilde{\lambda}_i|\tilde{v}_i\rangle\| = O(\xi) \tag{A.6}$$

for some numbers $\tilde{\lambda}_i$.[7] Working in the approximate eigenbasis basis, we can write

$$\widetilde{K}^+_{\tilde{\lambda}} \approx_{O(B\sqrt{\xi})} \sum_{i:\tilde{\lambda}_i=\tilde{\lambda}} \tilde{\lambda}^+_i |\tilde{v}_i\rangle\langle\tilde{v}_i|, \qquad \widetilde{K}^-_{\tilde{\lambda}} \approx_{O(B\sqrt{\xi})} \sum_{i:\tilde{\lambda}_i=\tilde{\lambda}} \tilde{\lambda}^-_i |\tilde{v}_i\rangle\langle\tilde{v}_i| \tag{A.7}$$

for nonnegative numbers $\tilde{\lambda}^+_i - \tilde{\lambda}^-_i = \tilde{\lambda}$. This gives the following approximation for $\tilde{\rho}^{\text{sgn}}$:

$$\tilde{\rho}^{\text{sgn}} \approx_{O(\sqrt{\xi})} \frac{1}{\text{Tr}[K^{\text{sgn}}]} \sum_i \tilde{\lambda}^{\text{sgn}}_i |\tilde{v}_i\rangle\langle\tilde{v}_i| \otimes |\tilde{\lambda}_i\rangle\langle\tilde{\lambda}_i|. \tag{A.8}$$

[7] Note that the basis $\{|\tilde{v}_i\rangle\}$ and exact eigenbasis of K, $\{|v_i\rangle\}$, are not necessarily equivalent, because the vectors in the former are only approximate eigenvectors of K.

Therefore the expectation of $\widetilde{X}$ is upper bounded by

$$\mathbf{E}[\widetilde{X}] \leq \frac{\sum_i e^{-\tilde{\lambda}_i}}{\mathrm{Tr}[K^+] + \mathrm{Tr}[K^-]} + O(\sqrt{\xi}) \max_{i:\tilde{\lambda}_i \geq \tilde{\lambda}_{\min}} \tilde{\lambda}_i^{-1} e^{-\tilde{\lambda}} \tag{A.9}$$

$$\leq (1+\delta)\frac{\widetilde{Z}_{\mathrm{supp}}}{\mathrm{Tr}[K^+] + \mathrm{Tr}[K^-]} + O(\sqrt{\xi}\delta^{-1})\widetilde{Z}_{\mathrm{supp}} \tag{A.10}$$

$$\leq (1+\delta+\sqrt{\xi}\delta^{-1}B)\frac{\tilde{Z}_{\mathrm{supp}}}{\mathrm{Tr}[K^+] + \mathrm{Tr}[K^-]} \tag{A.11}$$

$$= (1+O(\epsilon))\frac{\tilde{Z}_{\mathrm{supp}}}{\mathrm{Tr}[K^+] + \mathrm{Tr}[K^-]}. \tag{A.12}$$

A similar bound holds for lower bounding $\mathbf{E}[\widetilde{X}]$, showing that knowing $\mathbf{E}[\widetilde{X}]$ would give a $O(\epsilon)$-multiplicative error approximation to $\widetilde{Z}_{\mathrm{supp}}$. Just as in the ideal case, we can simply repeat our procedure $O(\eta^{-1}B^2\delta^{-2}\epsilon^{-2})$ times and take the mean to obtain a $O(\epsilon)$-multiplicative error approximation of $\mathbf{E}[\widetilde{X}]$, and hence of $\tilde{Z}_{\mathrm{supp}}$.

As before, we also need to estimate the number of eigenvalues λ (including degeneracy) with $|f(s,\lambda)| < \delta$, i.e. the number of i's with $|\tilde{\lambda}_i| < \delta$. Let this number be $n - \tilde{R}$. Let the output of the following procedure be $\tilde{Y}$:

Algorithm 14: Estimation of $n - \tilde{R}$

1. Perform consistent phase estimation to estimate eigenvalues of K on the uniformly random state I/n; let the output eigenvalue be $\tilde{\lambda}$.
2. If $|\tilde{\lambda}| < \delta$ output 1; otherwise output 0.

It is clear that $n\mathbf{E}[\tilde{Y}]$ is an $O(r_K\sqrt{\xi})$-multiplicative error approximation of $n - \tilde{R}$, and it can be proven as before that $O(r_K\epsilon^{-2})$ repetitions of the above procedure suffice to give an $O(\epsilon)$-error multiplicative stimate of $(n - \tilde{R})/n$. It can again be argued that $O(r_K\epsilon^{-2})$ repetitions suffice to estimate $\mathbf{E}[\widetilde{Y}]$, and thus $n - \widetilde{R}$, to $O(\epsilon)$-multiplicative error.

Finally, to estimate the full partition function we merely note that $\widetilde{Z}_{\mathrm{supp}} + (n - \widetilde{R})$ is an $O(\delta + \epsilon) = O(\epsilon)$-multiplicative error estimate of the partition function Z; we can therefore estimate Z by estimating both terms separately and taking the sum. □

A.3 Computing the Gibbs function

In this section we prove the following result:

Theorem A.4. *Suppose $K = K^+ - K^-$, where K^+ and K^- are $n \times n$ PSD matrices, and there is a quantum oracle that prepares copies of the states $\rho^+ = K^+/\mathrm{Tr}(K^+)$, $\rho^- = K^-/\mathrm{Tr}(K^-)$, and an oracle for the numbers $\mathrm{Tr}(K^+)$, $\mathrm{Tr}(K^-)$. Moreover, assume that $\mathrm{Tr}(K^+) + \mathrm{Tr}(K^-) \leq B$ for some bound B, and that K^+, K^- have rank at most r_K. Then it is possible to prepare the Gibbs state $\rho_G = \exp(-K)/\mathrm{Tr}(\exp(-K))$ up to error ϵ in trace distance, with $\mathrm{poly}(\log n, r_K, B, \epsilon^{-1})$ quantum gates.*

Proof. The procedure will the sketch given in Section 7.2, but using consistent phase estimation rather than the naïve protocol. We again assume we chose a good shift s for the operator K, and

first consider a procedure to sample from

$$\tilde{\rho}_{\text{supp}} \equiv \sum_{i:|\tilde{\lambda}_i|\geq\delta} e^{-\tilde{\lambda}_i}|\tilde{v}_i\rangle\langle\tilde{v}_i|/\widetilde{Z}_{\text{supp}}, \tag{A.13}$$

where $\delta > 0$ is a small threshold and $\tilde{\lambda}_i$, $|\tilde{v}_i\rangle$ were defined previous in (A.6). ρ_{supp} is the Gibbs state when considering only the space spanned by approximate eigenvectors of K whose eigenvalues estimates (under consistent phase estimation) are at least δ in aboslute value. Again choose $\delta = O(\epsilon)$, $\xi = O(\epsilon^2\delta^2 B^{-2} r_K^{-1})$, and pick a good random shift s – assume that this s (this happens with probability $1 - O(\epsilon^2\delta^2 B^{-2})$). Consider Algorithm 15 below.

Algorithm 15: Estimation of $\tilde{\rho}_{\text{supp}}$

---

1. **Input:** A good random shift s, an $O(\epsilon)$-multiplicative error estimate $\widetilde{Z}'_{\text{supp}}$ of $\widetilde{Z}_{\text{supp}}$, quantum oracles for ρ^+, ρ^-.

2. Let $\text{sgn} = +$ with probability $\text{Tr}(K^+)/[\text{Tr}(K^+) + \text{Tr}(K^-)]$, and $\text{sgn} = -$ otherwise.

3. Use consistent phase estimation to perform eigenvalue estimation of K on ρ^{sgn}; let the output be the normalized state $\tilde{\rho}^{\text{sgn}} = \frac{1}{\text{Tr}[K^{\text{sgn}}]}\sum_{\tilde{\lambda}} \widetilde{K}^{\text{sgn}}_{\tilde{\lambda}} \otimes |\tilde{\lambda}\rangle\langle\tilde{\lambda}|$ for unnormalized states $\widetilde{K}^{\text{sgn}}_{\tilde{\lambda}}$. Including the randomness on choosing sgn, we have the state

$$\tilde{\rho} \equiv \frac{\text{Tr}[K^+]}{\text{Tr}[K^+] + \text{Tr}[K^-]}\tilde{\rho}^+ + \frac{\text{Tr}[K^-]}{\text{Tr}[K^+] + \text{Tr}[K^-]}\tilde{\rho}^- \tag{A.14}$$

$$\approx_{O(\sqrt{\xi})} \frac{1}{\text{Tr}[K^+] + \text{Tr}[K^-]}\sum_i (\tilde{\lambda}_i^+ + \tilde{\lambda}_i^-)|\tilde{v}_i\rangle\langle\tilde{v}_i| \otimes |\tilde{\lambda}_i\rangle\langle\tilde{\lambda}_i|. \tag{A.15}$$

   Here in the second line we used the approximation (A.8).

4. Apply the projection $I \otimes \sum_{|\tilde{\lambda}|<\delta, \tilde{\lambda}\geq\tilde{\lambda}_{\min}} |\tilde{\lambda}\rangle\langle\tilde{\lambda}|$, where $\tilde{\lambda}_{\min}$ is the output of Algorithm 12. In other words, measure the second register to make sure that $|\tilde{\lambda}| \geq \delta$ and $\tilde{\lambda} \geq \tilde{\lambda}_{\min}$, and reject otherwise.

5. Apply the measurement operator

$$\sum_{i:|\tilde{\lambda}_i|\geq\delta, \tilde{\lambda}_i\geq\tilde{\lambda}_{\min}} \frac{\delta}{\tilde{\lambda}^+ + \tilde{\lambda}^-}\frac{e^{-\tilde{\lambda}}}{2Z'_{\text{supp}}}|\tilde{v}_i\rangle\langle\tilde{v}_i| \otimes |\tilde{\lambda}_i\rangle\langle\tilde{\lambda}_i|, \tag{A.16}$$

   up to $O(\sqrt{\xi})$ error, to the state. We can do this by first estimating $\tilde{\lambda}^+ + \tilde{\lambda}^-$ by quantum principal analysis (i.e. phase estimation of $\tilde{\rho}$), to precision $O(\sqrt{\xi}\delta)$ and error probability $O(\sqrt{\xi})$; then accept the resulting state with probability approximately $\frac{\delta}{\tilde{\lambda}^+ + \tilde{\lambda}^-}\frac{e^{-\tilde{\lambda}}}{2Z'_{\text{supp}}}$.

Algorithm 15 will give us a good approximation for $\tilde{\rho}_{\text{supp}}$, the Gibbs state on the approximate support of K (ignoring small eigenvalues). As before, we will need to approximate the Gibb state

on the approximate kernel of K as well, which we define as

$$\tilde{\rho}_{\text{ker}} = \frac{1}{n - \widetilde{R}} \sum_{i:\tilde{\lambda}_i} |v_i\rangle\langle v_i|. \tag{A.17}$$

This state can easily be approximated by starting with the completely mixed state I/n and performing consistent phase estimation to estimate eigenvalues of K, postselecting on the case that the measured estimate is less than δ in magnitude.

To complete our estimation for the full Gibbs state, we see that $\rho_G = \exp(-K)/\operatorname{Tr}(\exp(-K))$ can be approximated by

$$\rho_G \approx_{O(\delta)} \frac{1}{Z} \sum_i e^{-\tilde{\lambda}_i} |\tilde{v}_i\rangle\langle\tilde{v}_i| \tag{A.18}$$

$$= \frac{\widetilde{Z}_{\text{supp}}}{Z} \frac{1}{\widetilde{Z}_{\text{supp}}} \sum_{i:|\tilde{\lambda}_i| \geq \delta} e^{-\tilde{\lambda}_i} |\tilde{v}_i\rangle\langle\tilde{v}_i| + \frac{n - \widetilde{R}}{Z} \frac{1}{n - \widetilde{R}} \sum_{i:|\tilde{\lambda}_i| < \delta} e^{-\tilde{\lambda}_i} |\tilde{v}_i\rangle\langle\tilde{v}_i| \tag{A.19}$$

$$\approx_{O(\epsilon)} \frac{\widetilde{Z}_{\text{supp}}}{Z} \tilde{\rho}_{\text{supp}} + \frac{n - \widetilde{R}}{Z} \tilde{\rho}_{\text{ker}}. \tag{A.20}$$

Thus by Lemma 7 of [6] , it suffices to have $O(\epsilon)$-multiplicative error estimates for $\widetilde{Z}_{\text{supp}}$, $n - \widetilde{R}$, and Z, and $O(\epsilon)$-trace distance error approximations for $\tilde{\rho}_{\text{supp}}$ and $\tilde{\rho}_{\text{ker}}$. We have already shown how to achieve all of this. □