Improvements in Quantum SDP-Solving with Applications

Joran van Apeldoorn
QuSoft, CWI, The Netherlands
apeldoor@cwi.nl

András Gilyén
QuSoft, CWI, The Netherlands
gilyen@cwi.nl

Abstract

Following the first paper on quantum algorithms for SDP-solving by Brandão and Svore [13] in 2016, rapid developments have been made on quantum optimization algorithms. In this paper we improve and generalize all prior quantum algorithms for SDP-solving and give a simpler and unified framework.

We take a new perspective on quantum SDP-solvers and introduce several new techniques. One of these is the quantum operator input model, which generalizes the different input models used in previous work, and essentially any other reasonable input model. This new model assumes that the input matrices are embedded in a block of a unitary operator. In this model we give a $\widetilde{\mathcal{O}}\left(\left(\sqrt{m}+\sqrt{n}\gamma\right)\alpha\gamma^4\right)$ algorithm, where n is the size of the matrices, m is the number of constraints, γ is the reciprocal of the scale-invariant relative precision parameter, and α is a normalization factor of the input matrices. In particular for the standard sparse-matrix access, the above result gives a quantum algorithm where $\alpha = s$. We also improve on recent results of Brandão et al. [12], who consider the special case when the input matrices are proportional to mixed quantum states that one can query. For this model Brandão et al. [12] showed that the dependence on n can be replaced by a polynomial dependence on both the rank and the trace of the input matrices. We remove the dependence on the rank and hence require only a dependence on the trace of the input matrices.

After we obtain these results we apply them to a few different problems. The most notable of which is the problem of shadow tomography, recently introduced by Aaronson [2]. Here we simultaneously improve both the sample and computational complexity of the previous best results. Finally we prove a new $\widetilde{\Omega}(\sqrt{m}\alpha\gamma)$ lower bound for solving LPs and SDPs in the quantum operator model, which also implies a lower bound for the model of Brandão et al. [12].

2012 ACM Subject Classification Theory of computation → Quantum computation theory

Keywords and phrases quantum algorithms, semidefinite programming, shadow tomography

Digital Object Identifier 10.4230/LIPIcs.ICALP.2019.99

Category Track A: Algorithms, Complexity and Games

Related Version The full version of this paper is available at: `arXiv:1804.05058`.

Funding *Joran van Apeldoorn*: Supported by the Netherlands Organization for Scientific Research, grant number 617.001.351.
András Gilyén: Supported by ERC Consolidator Grant 615307-QPROGRESS and partially supported by QuantERA project QuantAlgo 680-91-034.

Acknowledgements We thank the authors of [12] for sending work-in-progress versions of their paper, and Fernando Brãndao, Tongyang Li and Xiaodi Wu for personal communication. A.G. thanks Robin Kothari and Nathan Wiebe for useful discussions. We thank Jamie Sikora for useful discussions about applications and for suggesting the state discrimination problem. We are grateful to Ronald de Wolf and Sander Gribling for useful discussions, and advice about the manuscript.

46th International Colloquium on Automata, Languages, and Programming (ICALP 2019).
Editors: Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi;
Article No. 99; pp. 99:1–99:15
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Semidefinite programs

In this paper we consider *semidefinite programs* (SDPs). SDPs have many applications in optimization, notable examples include approximation of NP-hard problems like MAXCUT [21] and polynomial optimization through the Sum-Of-Squares hierarchy [24, 29]. SDPs have also found applications in quantum information theory. Examples include POVM measurement design [18] and finding the winning probability of non-local games [16].

We consider the basic (primal) form of an SDP as follows:

$$\begin{aligned} \mathrm{OPT} = \max \quad & \operatorname{Tr}(CX) \\ \text{s.t.} \quad & \operatorname{Tr}(A_j X) \le b_j \qquad \text{for all } j \in [m], \\ & X \succeq 0, \end{aligned} \tag{1}$$

where $[m] := \{1, \ldots, m\}$. The input to the problem consists of $n \times n$ Hermitian constraint matrices $A_1, \ldots, A_m$, an objective matrix C, and reals $b_1, \ldots, b_m$. For normalization purposes we assume $\|C\|, \|A_j\| \le 1$. The number of constraints is m (we do not count the standard $X \succeq 0$ constraint for this). The variable X of this SDP is an $n \times n$ positive semidefinite (psd) matrix. We assume that $A_1 = I$ and $b_1 = R$, giving a known bound on the trace of a solution: $\operatorname{Tr}(X) \le R$. A primal SDP also has a *dual*. For a primal SDP of the above form (1) the dual SDP is

$$\begin{aligned} \mathrm{OPT} = \min \quad & b^T y \\ \text{s.t.} \quad & \sum_{j=1}^{m} y_j A_j - C \succeq 0, \\ & y \ge 0. \end{aligned} \tag{2}$$

We assume that the dual optimum is attained and that an explicit $r \ge 1$ is known such that at least one optimal dual solution y exists $\|y\|_1 \le r$. These assumptions imply that strong duality holds, justifying the use of OPT for both optimal values. The parameters r and R correspond to the scale of the SDP, without these bounds we could scale the SDP such that a larger error can be allowed. In some cases $r \cdot R$ may be quite large, but there are natural problems, where it is constant, cf. zero-sum games [4]. Finally, note that linear programs (LPs) correspond to the case where all constraint matrices are diagonal.

In this paper we build on the observation that a normalized psd matrix can be naturally represented as a quantum state. Since operations on quantum states can sometimes be cheaper to perform on a quantum computer than operations on classical descriptions of matrices, this can give rise to faster algorithms for solving SDPs on a quantum computer [13].

We say an algorithm is an ε-approximate *quantum SDP-solver* if for all input numbers $g \in \mathbb{R}$ and $\zeta \in (0,1)$, with success probability $1 - \zeta$, all of the following hold:

(i) The algorithm finds a vector $y' \in \mathbb{R}^{m+1}$ and a number $z \in \mathbb{R}$ defining its output

$$X := z \frac{e^{-\sum_{j=1}^{m} y'_j A_j + y'_0 C}}{\operatorname{Tr}\left(e^{-\sum_{j=1}^{m} y'_j A_j + y'_0 C}\right)}. \tag{3}$$

The output X is an *ε-feasible primal solution* with objective value at least $g - \varepsilon$, i.e.,

$$\forall j \in [m] \colon \operatorname{Tr}(XA_j) \le b_j + \varepsilon,$$

and $\operatorname{Tr}(XC) \ge g - \varepsilon$. If the algorithm cannot find such an output X, then it correctly concludes that no feasible solution exist (if we set $\varepsilon = 0$).

(ii) The algorithm finds a $y \in \mathbb{R}^{m+1}$ that is an ε-*feasible solution* to the dual problem with objective value at most $g + \varepsilon$, i.e.,

$$\sum_{j=1}^{m} y_j A_j - C \succeq -\varepsilon I, \tag{4}$$

and $b^T y \leq g + \varepsilon$, or it correctly concludes that no feasible y exists (if we set $\varepsilon = 0$).

(iii) The algorithm determines whether $\mathrm{OPT} \leq g - \varepsilon$ or $\mathrm{OPT} \geq g + \varepsilon$. If $\mathrm{OPT} \in (g - \varepsilon, g + \varepsilon)$ then it may output either. (Note that this essentially follows from (i)-(ii).)

Notice that this solves a decision version of the problem. However, we can easily find an approximation of OPT using binary search on g if we have an ε-approximate SDP-solver. Since ε is scale depended we actually care about the dependence on the scale invariant parameter $\gamma := Rr/\varepsilon$. An algorithm that only satisfies (i) will be called an ε-*approximate SDP primal oracle*. For such an algorithm the relevant scale invariant parameter is $\gamma := R/\varepsilon$. Due to the form of the objective value constraint in the first point, and to simplify statements like (3), we write $A_0 := -C$ and $b_0 := -g$.

Notation

We use the following definition for $\tilde{\mathcal{O}}$:

$$\widetilde{\mathcal{O}}_{d,e}\left(f(a,b,c)\right) := \mathcal{O}\left(f(a,b,c) \cdot \mathrm{polylog}(f(a,b,c), d, e)\right).$$

We define $\tilde{\Omega}$ in a similar way and $\tilde{\Theta}$ as the intersection of the two. We write δ_{ij} for the Kronecker delta function and e_j for the jth basis vector in the standard basis when the dimension of the space is clear from context. For a Hermitian matrix H we write $\mathrm{Spec}(H)$ for its spectrum (set of eigenvalues). For a function $f : \mathbb{R} \to \mathbb{R}$ we write $f(H)$ for the matrix we get by applying f to the eigenvalues of H, i.e.,

$$f(H) = U \begin{bmatrix} f(\lambda_1) & & \\ & \ddots & \\ & & f(\lambda_n) \end{bmatrix} U^{-1} \text{ where } H = U \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix} U^{-1}.$$

2 SDP-solving frameworks

In this section we present two frameworks for SDP-solving, providing the basis of our quantum algorithms. First we present an algorithm to implement a primal oracle, and then the Arora-Kale framework, which is used for finding a good approximation of the optimal value and an almost feasible solution to the dual. These together implement a full SDP-solver.

Both frameworks have a very similar iterative structure, with iteration count $\widetilde{\mathcal{O}}_n\left(\gamma^2\right)$, where $1/\gamma$ is the relevant scale-invariant precision parameter (as we will see the value of γ differs by a factor r in the two cases). The main difference is that the iterative step of the primal oracle framework requires only a simple search, whereas in the Arora-Kale framework one needs to solve a slightly more complex task. Both algorithms start with $y = 0$, and in each step only a constant number of indices of y are incremented, thus in both cases we will work with a y vector that is non-negative and $\widetilde{\mathcal{O}}_n\left(1/\theta^2\right)$-sparse.

2.1 An SDP primal oracle

For the primal oracle we use the same algorithm as Brandão et al. [12] following the proof of Lemma 4.6 of Lee, Raghavendra and Steurer [25]. A few small reductions are required to apply this technique. To be able to work with density operators instead of X, the b_js in the constraints $1 \dots m$ are scaled down by a factor R, such that every solution X' to the new SDP has trace at most 1. Then, we add one new variable denoted by ω such that

$$\rho := \begin{bmatrix} X' & 0 \\ 0 & \omega \end{bmatrix}.$$

Now $\mathrm{Tr}\left(\rho\right) = 1$ and $\rho \succeq 0$ imply that $\mathrm{Tr}\left(X'\right) \leq 1$, and we get a new SDP that is equivalent to the previous one. It can be shown that in our input models this reduction does not introduce more than a constant factor overhead in the complexity; note that this reduction also illustrates that for an SDP primal oracle $\frac{\varepsilon}{R}$ is the relevant scale-invariant precision parameter.

The following meta-algorithm for an SDP primal oracle assumes access to (some form of) a description of the SDP after the above-discussed reduction, and provides an output as in Eq. (3) of (i)

1. Let $y = 0 \in \mathbb{R}^{m+1}$ and $\theta = \frac{\varepsilon}{2R}$.
2. Repeat $\frac{\ln(n)}{\theta^2}$ times the following:
   a. Define $\rho := e^{-\sum_{j=0}^{m} y_j A_j} / \mathrm{Tr}\left(e^{-\sum_{j=0}^{m} y_j A_j}\right)$.
   b. Find an index j such that $\mathrm{Tr}\left(A_j \rho\right) \geq b_j$ or conclude correctly that for all j, $\mathrm{Tr}\left(A_j \rho\right) \leq b_j + \theta$.
   c. If no j is found, then we are done and output y and $z = R\mathrm{Tr}\left(X'\right)$, where $\mathrm{Tr}\left(X'\right)$ is the probability[1] of measuring ρ to be in the subspace corresponding to the variable X'.
   d. Otherwise update $y \leftarrow y + \theta e_j$.
3. Conclude that there is no solution for $\theta = 0$.

2.2 The Arora-Kale framework

Similarly to previous work [13] we build our results on the Arora-Kale framework. For a detailed description see the original paper by Arora and Kale [7]. For our application, the following broad overview suffices.[2]

Now we describe the Arora-Kale meta-algorithm. This algorithm assumes access to (some form of) a description of the SDP, such that the first constraint is $\mathrm{Tr}\left(X\right) \leq R$, i.e., $A_1 = I$ and $b_1 = R$. It provides an output as in Eq. (4) of (ii). (Remember that we set $A_0 = -C$ and $b_0 = -g$.)

1. Let $y = 0 \in \mathbb{R}^{m+1}$ and set $\theta = \frac{\varepsilon}{6Rr}$.
2. Repeat $\frac{\ln(n)}{\theta^2}$ times the following:
   (a) Define $\rho := e^{-\sum_{j=0}^{m} y_j A_j} / \mathrm{Tr}\left(e^{-\sum_{j=0}^{m} y_j A_j}\right)$.

[1] Note that a θ-approximation of $\mathrm{Tr}\left(X'\right)$ is easy to compute by means of amplitude estimation if ρ can be efficiently prepared as a quantum state – which is the case in our algorithms.

[2] For more discussion on general (quantum) SDP-solvers along this line, see [5].

(b) Find a $\tilde{y}$ in the polytope

$$\begin{aligned}\mathcal{P}_\delta(\rho) := \Big\{\tilde{y} \in \mathbb{R}^{m+1} : {} & b^T\tilde{y} \leq 0, \\ & \sum_{j=0}^{m} \tilde{y}_j \mathrm{Tr}\left(A_j\rho\right) \geq -\delta, \\ & \tilde{y} \geq 0,\ \tilde{y}_0 = \frac{1}{2r},\ \|\tilde{y}\|_1 \leq 1\Big\}.\end{aligned}$$

for $\delta = \theta$; if cannot find such a $\tilde{y}$ then conclude that none exists for $\delta = 0$.

(c) If no such $\tilde{y}$ exists, then conclude that OPT $> g$ and stop.

(d) If such a $\tilde{y}$ exists, then update $y \leftarrow y + \theta\tilde{y}$.

3. Conclude OPT $\leq g + \varepsilon$ and output $\frac{2r\theta}{\ln(n)}y + \frac{\varepsilon}{R}e_1 - e_0$ as a dual solution.

Note that in the above meta-algorithm, up to a constant factor, the θ parameter is essentially $\gamma^{-1} = \frac{\varepsilon}{rRs}$, illustrating that γ^{-1} is the relevant scale-invariant precision parameter for SDP-solving, for a more detailed discussion see [5].

Brandão and Svore [13] observed that $\rho := e^{-\sum_j y_j A_j}/\mathrm{Tr}\left(e^{-\sum_j y_j A_j}\right)$ is a quantum Gibbs state and this state can be prepared efficiently on a quantum computer, allowing fast trace estimation, in particular resulting in a quadratic speedup in n compared to classical methods.

A procedure that solves step (b) is called a *θ-oracle*, not to be confused with the input oracles. In the rest of this paper we will assume that the cost of updating the y vector is no more than the cost of a θ-oracle call. This is justified, because we use the geometric approach of Apeldoorn et al. [5, Lemma 16] for implementing a 3-sparse γ^{-1}-oracle. Their oracle returns a 3-sparse vector, and thus requires updating only 3 entries of y, which can be done efficiently using an efficient data structure.

3 Input models & Subroutines

We consider three input models: the *sparse matrix model*, the *quantum state model*, and the *quantum operator model*. The first two models were already studied in previous work. The quantum operator model is a common generalization of the other two models, and in fact any other reasonable model for SDPs, as we show.

In all models we assume quantum oracle access to the numbers b_j via the input oracle O_b satisfying[3] for all $j \in [m]$:

$$O_b|j\rangle|0\rangle = |j\rangle|b_j\rangle.$$

For all input oracles we assume we can apply both the oracle and its inverse[4] in a controlled fashion.

Sparse matrix model

In the *sparse matrix model* the input matrices are assumed to be s-row sparse for a known bound $s \in [n]$, meaning that there are at most s non-zero elements per row. The model is close to the classical model for sparse matrices. Access to the A_j matrices is provided by

[3] For simplicity we assume the bitstring representation has at most $\mathcal{O}\left(\log(nmRr/\varepsilon)\right)$ bits.

[4] When we talk about samples, e.g. in Section 4.1 of the full version of this paper [3], then we do not assume we can apply the inverse operation.

two oracles, similar to previous work on Hamiltonian simulation in [9]. The first of the two oracles is a unitary O_{sparse}, which serves the purpose of sparse access. This oracle calculates the **index** : $[m] \times [n] \times [s] \to [n]$ function, which for input (j, k, ℓ) gives the column index of the ℓth non-zero element in the kth row of A_j. We assume this oracle computes the index "in place":

$$O_{\text{sparse}}|j, k, \ell\rangle = |j, k, \mathbf{index}(j, k, \ell)\rangle. \tag{5}$$

(In the degenerate case where the kth row has fewer than ℓ non-zero entries, $\mathbf{index}(j, k, \ell)$ is defined to be ℓ together with some special symbol indicating this case.)

We also need another oracle O_A, returning a bitstring[3] representation of $(A_j)_{ki}$ for every $j \in [m]$ and $k, i \in [n]$:

$$O_A|j, k, i, z\rangle = |j, k, i, z \oplus (A_j)_{ki}\rangle. \tag{6}$$

This model corresponds directly to a classical way of accessing sparse matrices.

Quantum state model

In contrast to the sparse matrix model, the *quantum state model* is inherently quantum and has no classical counterpart for SDPs. In this model we assume that each A_j has a fixed decomposition of the form

$$A_j = \mu_j^+ \varrho_j^+ - \mu_j^- \varrho_j^- + \mu_j^I I$$

for (subnormalized) density operators $\varrho_j^\pm$, non-negative reals $\mu_j^\pm$ and real number $\mu_j^I \in \mathbb{R}$.

▶ **Definition 1** (Subnormalized density operators & Purification). *A* subnormalized density operator *ϱ is a psd matrix of trace at most* 1*. A* purification *of a subnormalized density operator ϱ is a* 3*-register pure state such that tracing out the third register*[5] *and projecting on the subspace where the second register is $|0\rangle$ yields ϱ.*

We write "ϱ" and "ς" for subnormalized density operators to distinguish them from normalized density operators, for which we write "ρ" and "σ".

We assume access to an oracle O_μ that takes as input an index j and outputs binary representations[3] of μ_j^+, μ_j^- and μ_j^I.

Furthermore we assume access to a state-preparing oracle $O_{|\cdot\rangle}$ that prepares purifications[5] $|\psi_j^\pm\rangle$ of $\varrho_j^\pm$:

$$O_{|\cdot\rangle}|j\rangle|\pm\rangle|0\rangle = |j\rangle|\pm\rangle|\psi_j^\pm\rangle.$$

Finally we assume that a bound $B \in \mathbb{R}_+$ is known such that

$$\forall j : \mu_j^+ + \mu_j^- \leq B.$$

Note that a tight upper bound B can easily be found using $\mathcal{O}(\sqrt{m})$ quantum queries to O_μ by means of maximum finding [17].

[5] For simplicity we assume that for a d-dimensional density operator a purification has at most polylog(d) qubits.

Quantum operator model

Motivated by recent work [27, 20] we propose a new input model that we call the *quantum operator model*. In this model the input matrices are given by a unitary that implements a block-encoding:

▶ **Definition 2** (Block encoding). *Suppose that A is a w-qubit operator, $\alpha, \varepsilon \in \mathbb{R}_+$ and $k \in \mathbb{N}$, then we say that the $(a+w)$-qubit unitary U is an (α, a, ε)-block-encoding of A, if*

$$\left\| A - \alpha(\langle 0|^{\otimes a} \otimes I) U (|0\rangle^{\otimes a} \otimes I) \right\| \leq \varepsilon.$$

Roughly speaking this means that A is represented by a unitary

$$U \approx \begin{pmatrix} A/\alpha & \cdot \\ \cdot & \cdot \end{pmatrix}.$$

In the quantum operator model we assume access to an oracle O_U that acts as follows:

$$O_U |j\rangle|\psi\rangle = |j\rangle(U_j|\psi\rangle).$$

Where U_j is an $(\alpha, a, 0)$-block-encoding[6] of A_j, for some fixed[7] $\alpha \in \mathbb{R}$ and for $a = \mathcal{O}\left(\log(nmRr/\varepsilon)\right)$.

In Section 5 of the full version of this paper [3] we show that the sparse input model can be reduced to the quantum operator model with $\alpha = s$ and that the quantum state model can be reduced to it with $\alpha = B$. We also argue that if we can perform a measurement corresponding to $A_j \succeq 0$ using a ancilla qubits, i.e., accept a state ρ with probability $\mathrm{Tr}\left(A_j\rho\right)$, then we can implement a $(1, a+1, 0)$-block-encoding of A_j. In this way this input model is a common generalization of all reasonable input models for SDPs, since at the very least an input model should allow you to calculate $\mathrm{Tr}\left(A_jX\right)$.

Therefore if one can perform a POVM measurement on a quantum computer with a measurement operator M, one can also implement a block-encoding of M, and use it as an input matrix in our operator model. Similarly, being able to perform Hamiltonian simulation with a Hermitian matrix H gives access to H as a block-encoding, as shown by [28, 20]. Recent work [22] introduced QROM data structures that allow the efficient creation of a superposition over matrix elements of a matrix M. It turns out [15] that the corresponding block-encoding can be implemented with $\widetilde{\mathcal{O}}_{n,\gamma}\left(1\right)$ QROM calls, such that even for non-sparse matrices one has $\alpha \leq \sqrt{n}$.

Computational cost

We analyze the query complexity of algorithms and subroutines, i.e., the number of queries to controlled versions of the input oracles and their inverses. We denote the optimal quantum query complexity of an ε-approximate *quantum SDP-solver* with success probability $2/3$ by $T_{SDP}(\varepsilon)$ (this is a "meta quantity", which becomes concrete once the input model is specified). We only consider success probability $2/3$ to simplify the notation and proofs. However in all cases an ε-approximate SDP-solver with success probability $1-\zeta$ can easily be constructed using $\mathcal{O}\left(\log(1/\zeta)T_{SDP}(\varepsilon)\right)$ queries.

[6] If n is not a power of 2, then we simply define A_j to be zero on the additional $2^w - n$ dimensions.

[7] Having a single normalization parameter α is not a serious restriction as it is easy to make a block-encoding more subnormalized so that every A_j gets the same normalization, cf. Lemma 14 of the full version of this paper [3].

In our algorithms we assume access to a quantum-read/classical-write RAM (known as QCRAM), and assume one read/write operation has a constant gate complexity[8]; the size of the QCRAM that we use is typically $\widetilde{\mathcal{O}}_{n,m}\left(\left(\frac{Rr}{\varepsilon}\right)^2\right)$ bits. Most often in our results the number of non-query elementary operations, i.e., two-qubit gates and QCRAM calls, matches the query complexity up to polylog factors. In particular, if not otherwise stated, in our results a T-query quantum algorithm uses at most $\widetilde{\mathcal{O}}_{n,m}(T)$ elementary operations.

Subroutines

We work with two major subroutines which need to be implemented according to the specific input model. First, the algorithms require an implementation of a Gibbs-sampler.

▶ **Definition 3** (Gibbs-sampler). *A θ-precise* Gibbs-sampler *on bounded input vectors $y \in \mathbb{R}^{m+1}_{\geq 0}$ is a quantum circuit that works under the promise that the support of y has size at most d, and $\|y\|_1 \leq K$. It takes as input a data structure storing the vector y, and for any input satisfying the promise, it creates as output a purification of a θ-approximation of the Gibbs state*

$$e^{-\sum_{j=0}^m y_j A_j}/\mathrm{Tr}\left(e^{-\sum_{j=0}^m y_j A_j}\right).$$

The minimum cost of such a circuit is denoted by $T_{Gibbs}(K, d, 4\theta)$ (this is again a "meta quantity", which becomes concrete once the input model is specified).

For technical reasons we also allow Gibbs-samplers that require a random classical input seed $S \in \{0,1\}^a$ for some $a = \mathcal{O}(\log(1/\theta))$. In this case the output should be a θ-approximation of the Gibbs state with high probability ($\geq 4/5$) over a uniformly random input seed S.

We use the approximate Gibbs states in order to estimate the quantity $\mathrm{Tr}(A_j\rho)$.

▶ **Definition 4** (Trace estimator). *A (θ,σ)-*trace estimator *is a quantum circuit that as input takes a quantum state ρ and index j. It outputs a sample from a random variable $Z_j \in \mathbb{R}$ which is an estimator of $\mathrm{Tr}(A_j\rho)$ with bias at most θ:*

$$|\mathrm{Tr}(A_j\rho) - \mathbb{E}[Z_j]| \leq \theta,$$

and the standard deviation of Z_j is at most σ. We write $T^{\sigma}_{Tr}(\theta)$ for the minimum cost of such a circuit (this is again a "meta quantity" as in the above definition).

4 Prior work

Classical SDP-solvers roughly fall into two categories: those with logarithmic dependence on R, r and $1/\varepsilon$, and those with polynomial dependence on these parameters but better dependence on m and n. In the first category the best known algorithm [26] at the time of writing has complexity

$$\widetilde{\mathcal{O}}_{Rr/\varepsilon}\left(m(m^2 + n^{\omega} + mns)\right).$$

where $\omega \in [2, 2.38]$ is the yet unknown exponent of matrix multiplication.

[8] Note that read/write operations of a QRAM or QCRAM of size S can be implemented using $\widetilde{\mathcal{O}}(S)$ two-qubit gates, so this assumption could hide a factor in the gate complexity which is at most $\widetilde{\mathcal{O}}(S)$.

In the second category Arora and Kale [7] gave an alternative framework for solving SDPs, using a matrix version of the "multiplicative weights update" method, see Section 2. Their framework can be tuned for specific types of SDPs, allowing for near linear-time algorithms in the case of for example the Goemans-Williamson SDP for the approximation of the maximum cut in a graph [21].

In 2016 Brandão and Svore [13] used the Arora-Kale framework to implement a general quantum SDP-solver in the sparse matrix model. They observed that the matrix

$$\rho := \frac{e^{-\sum_{j=0}^{m} y_j A_j}}{\operatorname{Tr}\left(e^{-\sum_{j=0}^{m} y_j A_j}\right)},$$

that is used for calculations in the Arora-Kale framework is in fact a $\log(n)$-qubit Gibbs state and can be efficiently prepared as a quantum state on a quantum computer. Using this they achieved a quantum speedup in terms of n. Combining this with a Grover-like speedup allowed for a speedup in terms of m as well, leading to an ε-approximate quantum SDP solver with complexity

$$\widetilde{\mathcal{O}}\left(\sqrt{mn}s^2\left(\frac{Rr}{\varepsilon}\right)^{32}\right).$$

They also showed an $\Omega(\sqrt{m}+\sqrt{n})$ quantum query lower bound for solving SDPs when all other parameters are constant. This left as open question whether a better lower bound, matching the $\sqrt{mn}$ upper bound, could be found. The upper bound for the sparse input model was subsequently improved by van Apeldoorn et al. [5] to

$$\widetilde{\mathcal{O}}\left(\sqrt{mn}s^2\left(\frac{Rr}{\varepsilon}\right)^{8}\right).$$

van Apeldoorn et al. also gave an $\Omega(\sqrt{\max(n,m)}\min(n,m)^{3/2})$ lower bound, albeit for non-constant parameters R and r. This bound implies that there is no general quantum SDP-solver that has a $o(nm)$ dependence on n and m and logarithmic dependence on R, r and $1/\varepsilon$. They also showed that every SDP-solver whose efficiency relies on outputting sparse dual solutions (including their algorithm and that of Brandão and Svore [13]) is limited, since problems with a lot of symmetry (like maxflow-mincut) in general require non-sparse dual solutions. Furthermore, they showed that for many combinatorial problems (like MAXCUT) R and r increase linearly with n and m.

Very recently Brandão et al. [11] gave an improved SDP-solver for the quantum state input model[9] that has a complexity bound with logarithmic dependence on n:

$$T_{SDP}(\varepsilon) = \widetilde{\mathcal{O}}_n\left(\sqrt{m}\ \mathrm{poly}\left(\frac{Rr}{\varepsilon}, B, \max_{j\in\{0,\ldots,m\}}[\mathrm{rank}(A_j)]\right)\right).$$

Brandão et al. also applied their algorithm to the problem of *shadow tomography*, giving the first non-trivial application of a quantum SDP-solver.

Subsequently these results where further improved by the introduction of the Fast Quantum OR lemma by the same authors [12]. Approaches prior to [12] searched for a violated constraint in the SDP using Grover-like techniques, resulting in a multiplicative

[9] This model was already introduced in the first version of [13] together with a similar complexity statement, but there were some unresolved issues in the proof, that were only fixed by the contributions of [11].

complexity of Gibbs-sampling and searching. The Fast Quantum OR lemma can be used to separate the search phase from the initial Gibbs-state preparation phase. This led to the improved complexity bound [12] of

$$\widetilde{\mathcal{O}}_n\left(\left(\sqrt{m}+\mathrm{poly}(\max_{j\in\{0\ldots m\}}[\mathrm{rank}(A_j)])\right)\ \mathrm{poly}\left(\frac{Rr}{\varepsilon},B\right)\right).$$

We thank the authors of [12] for sending us an early draft of [12] introducing the Fast Quantum OR Lemma, which enabled us to work on these improvements. During the correspondence the application of the Fast OR lemma to the sparse matrix model was independently suggested by Brandão et al. [30] and by us.

5 Our results

In this paper we present multiple results. The main contribution consists of multiple improvements to the algorithms for SDP solving, based on combining various recent quantum algorithmic developments. Although some of these improvements require quite technical proofs, they come from simple new perspectives and ideas, often combining previous works in new ways. We also apply the resulting algorithms to a few problems in convex optimization. Finally, we prove a new lower bound that fits the novel input models that we work with.

5.1 Improvements to the quantum algorithms

In this paper we build on the Arora-Kale framework for SDP-solving in a similar fashion as [5, 13] and also use results from [11, 25] to construct a primal oracle. We improve on the previous results about quantum SDP-solving in three different ways:

Two-Phase Quantum Search and Minimum Finding

We give a computationally more efficient version of the Gentle Quantum Search Lemma [2] using the Fast Quantum OR Lemma from [12]. We also extend this to minimum finding to get our *Two-Phase Quantum Minimum Finding* (Lemma 7 of the full version of this paper [3]). As independently observed by the authors of [12] the Fast Quantum OR Lemma gives a speed-up for SDP primal oracles in general. Moreover, using Two-Phase Quantum Minimum Finding, we show how to improve the upper bound on the complexity of general SDP-solving from

$$T_{SDP}(\varepsilon)=\widetilde{\mathcal{O}}_n\left(\sqrt{m}T^{\sigma}_{Tr}((4\gamma)^{-1})T_{Gibbs}(\gamma,\gamma^2,\gamma^{-1})\gamma^3\sigma\right) \tag{7}$$

as implied in previous work [13, 5] to

$$T_{SDP}(\varepsilon)=\widetilde{\mathcal{O}}_n\left(\left(\sqrt{m}T^{\sigma}_{Tr}((4\gamma)^{-1})+T_{Gibbs}(\gamma,\gamma^2,\gamma^{-1})\right)\gamma^4\sigma^2\right), \tag{8}$$

where $\gamma=\Theta(Rr/\varepsilon)$. For the complexity of SDP primal oracles, the same upper bounds hold.

Quantum operator model and efficient data structures

We introduce the quantum operator input model unifying prior approaches. We show that both the sparse model and the quantum state model can be reduced to the quantum operator model with a constant overhead and with the choices of $\alpha=s$ and $\alpha=B$ respectively.

Moreover, we show that for $\sigma = \Theta(1)$, we have that

$$T^{\sigma}_{Tr}((4\gamma)^{-1}) = \widetilde{\mathcal{O}}_{\gamma}\left(\alpha\right),$$

in the quantum operator model.

The complexity of Gibbs-sampling in the sparse matrix input model previously was [5]:

$$T_{Gibbs}(K, d, \theta) = \widetilde{\mathcal{O}}_{\theta}\left(\sqrt{n}Ks^2d^2\right).$$

By considering the operator model, in which we can show how to simulate a linear combination of Hamiltonians efficiently, we can improve this to

$$T_{Gibbs}(K, d, \theta) = \widetilde{\mathcal{O}}_{\theta,d}\left(\sqrt{n}K\alpha\right).$$

This result is based on the idea of gradually building up an efficient data structure for state preparation, following ideas of [22]. This demonstrates that these data structures can be used efficiently even if one does not assume preprocessed data. Moreover, it shows that working in the operator model does not only unify prior approaches but also inspires more efficient quantum algorithms due to its conceptual clarity.

Table 1 Summary of the role of our various improvements, the theorem numbers refer to the numbers in the full version of the paper [3]. The main new results are on the bottom, the other complexity statements represent partial results following from only applying some of the improvements. We present the results for the sparse matrix and quantum state input models for comparison to prior work. However, note that our results presented for sparse input hold more generally for the quantum operator input model; to get the corresponding results one should just replace s by α in the table. Thereby similar bounds hold in the case of the quantum state input model too, after replacing s by B, which can be beneficial when $B^{2.5}\gamma^{2.5} \geq \sqrt{n}$. Notation: $\mathrm{rk} = \max_{j\in\{0,\ldots,m\}} \mathrm{rank}(A_j)$ and $\gamma = \frac{Rr}{\varepsilon}$.

	Without OR lemma / Two-Phase Search	
	Sparse input	Quantum state input
Previous Gibbs-sampling	$\widetilde{\mathcal{O}}\left(\sqrt{mn}s^2\gamma^8\right)$ [5]	$\widetilde{\mathcal{O}}_n\left(\sqrt{m}\mathrm{poly}\left(\gamma, B, \mathrm{rk}\right)\right)$ [11]
Improved Gibbs-sampling	$\widetilde{\mathcal{O}}\left(\sqrt{mn}s\gamma^4\right)$ Corollary 18	$\widetilde{\mathcal{O}}_n\left(\sqrt{m}B^{3.5}\gamma^{6.5}\right)$ Corollary 25
	With OR lemma / Two-Phase Search	
	Sparse input	Quantum state input
Previous Gibbs-sampling	$\widetilde{\mathcal{O}}\left(\left(\sqrt{m}+\sqrt{n}s\gamma^5\right)s\gamma^4\right)$ Theorem^{10} 8 + [5]	$\widetilde{\mathcal{O}}_n\left(\left(\sqrt{m}+\mathrm{poly}(\mathrm{rk})\right)\mathrm{poly}\left(\gamma, B\right)\right)$ [12]
Improved Gibbs-sampling	$\widetilde{\mathcal{O}}\left(\left(\sqrt{m}+\sqrt{n}\gamma\right)s\gamma^4\right)$ Theorem 17	$\widetilde{\mathcal{O}}_n\left((\sqrt{m}+B^{2.5}\gamma^{3.5})B\gamma^4\right)$ Theorem 24

Gibbs sampling for the quantum state model

We develop a new method for Gibbs-sampling in the quantum state model. As noted in [12] this model has the nice property that it is relatively easy to find the important eigenspaces of the input matrices. We introduce a new technique for finding these important eigenspaces that, in contrast to the approach in [12], does not introduce a dependence on the rank of the input matrices in the complexity. In particular we improve the complexity bound of [12]

$$T_{Gibbs}(K,d,\theta) = \mathcal{O}\left(\text{poly}(K,B,d,1/\theta, \max_{j\in\{0\ldots m\}}[\text{rank}(A_j)])\right),$$

to

$$T_{Gibbs}(K,d,\theta) = \widetilde{\mathcal{O}}_{d,\theta,n}\left((KB)^{3.5}\right),$$

both making the polynomial dependence explicit and improving it.

An important consequence of this improvement is that we do not get a dependence on the rank of the input matrices in the complexity of SDP solving, unlike Brandão et al. [12]. Since the most natural use for the quantum state model is when the A_j matrices naturally correspond to quantum states, B is often just 1. However, if the states are highly mixed, then the rank is about n, eliminating the speedup over the sparse input model when a rank dependence is present. Finally note that this Gibbs-sampling method is only beneficial if $\sqrt{n} \leq (KB)^{2.5}$, otherwise the reduction to the quantum operator model with $\alpha = B$ gives a better algorithm.

For the quantum operator input model the above improvements lead to the complexity bound

$$T_{SDP}(\varepsilon) = \widetilde{\mathcal{O}}\left(\left(\sqrt{m} + \sqrt{n}\gamma\right)\alpha\gamma^4\right), \tag{9}$$

where $\gamma := \frac{Rr}{\varepsilon}$. Note that the $\Omega(\sqrt{n} + \sqrt{m})$ lower bound of [13] also applies to the quantum operator model due to our reductions, matching the above upper bound (9) up to polylog factors in n and m when γ and α are constant. For the quantum state input model our improved Gibbs-sampler yields the complexity bound

$$T_{SDP}(\varepsilon) = \widetilde{\mathcal{O}}_n\left(\left(\sqrt{m} + B^{2.5}\gamma^{3.5}\right)B\gamma^4\right).$$

In both cases, the same bound holds for an SDP primal oracle but with $\gamma := R/\varepsilon$.

5.2 Applications

In Section 4 of the full version of this paper [3] we give some applications of quantum SDP-solvers. Due to the large error dependence the last two of these are not of practical interest. However they do show that a theoretical improvement in one of the parameters is possible over classical computers and more specified quantum algorithms might improve the error dependence.

Shadow tomography of quantum states

We extend the idea of applying SDP-solving to the problem of shadow tomography: given an unknown, n-dimensional quantum state ρ, find ε-additive approximations of the expectation values $\text{Tr}\,(E_1\rho),\ldots,\text{Tr}\,(E_m\rho)$ of several binary measurement operators. This problem was introduced by Aaronson in [2], he gave an efficient algorithm in terms of the number of samples from ρ. In particular he proved that $\widetilde{\mathcal{O}}\left(\log^4(m)\log(n)/\varepsilon^5\right)$ samples suffice. Brandão et al. [12] applied their SDP-solver to get a more efficient algorithm in terms of computation time when the measurements E_i are given in the quantum state model, while keeping the sample complexity as low as $\text{poly}(\log(m),\log(n),1/\varepsilon,B)$. We simultaneously improve on both results, giving a sample bound of $\widetilde{\mathcal{O}}\left(\log^4(m)\log(n)/\varepsilon^4\right)$ while also improving the best known time complexity [2, 12] of the implementation for all input models. Finally we show that if we can efficiently implement the measurements $\mathrm{Tr}\left(E_1\rho\right), \ldots, \mathrm{Tr}\left(E_m\rho\right)$ on a quantum computer, then we can also efficiently represent $E_1, \ldots, E_m$ using the quantum operator input model, hence the computational complexity can be stated in terms of the number of measurements needed.

Quantum state discrimination

We apply the SDP-solvers to the problem of quantum state discrimination: given a set of quantum states, what is the best POVM for discriminating between the states? We consider the case of minimizing the total error in the measurements. In this case we get an algorithm with running time $\widetilde{\mathcal{O}}\left(\sqrt{k}\,\mathrm{poly}(d, 1/\varepsilon)\right)$ in the sparse input model, where k is the number of states and d is the dimension of the states. Due to the quantum state model for SDP-solving, we can also solve the problem when the states that need to be discriminated are actually given as quantum states, rather than classical descriptions of density operators.

Optimal design

We apply our sparse SDP-solver to the problem of E-optimal design: given a set of k experiments, find the optimal distribution of the experiments that minimizes the variance in our knowledge of a d-dimensional system. Our final bound is $\widetilde{\mathcal{O}}\left((\sqrt{k}+\sqrt{d})\mathrm{poly}(1/\varepsilon, P)\right)$, where P is a parameter that depends on the standard deviation of the experiments.

5.3 Lower bounds

We end the paper with proving new lower bounds. Lower bounds on the quantum query complexity of SDP-solving for the sparse input model were presented in previous works [13, 5]. We add to this by giving $\widetilde{\Omega}(\sqrt{m}B/\varepsilon)$ and $\widetilde{\Omega}(\sqrt{m}\alpha/\varepsilon)$ bounds for the quantum state model and quantum operator model respectively. These lower bounds show that the $\sqrt{m}$ factor and the polynomial dependence on the parameters B, α, and $1/\varepsilon$ are necessary.

Compared to problems with a discrete input, proving lower bounds on continuous-input quantum problems gives rise to extra challenges and often requires more involved techniques, see for example the work of Belovs [8] on generalizations of the adversary method. Due to these difficulties, fewer results are known in this regime. Examples of known continuous-input lower-bound results include phase-estimation related problems (cf. Bessen [10]) and the complexity-theoretic version of the no-cloning theorem due to Aaronson [1]. Recently, a new hybrid-method based approach was developed by Gilyén et al. [19] in order to handle continuous-input oracles, which they use for proving a lower bound for gradient computation. We use their techniques to prove our lower bounds, combined with efficient reductions between input models stemming from the smooth-functions of Hamitonians techniques developed in the work of van Apeldoorn et al. [5].

6 Subsequent work

A few months after the first version of our paper was posted on the arXiv, Kerenidis and Prakash [23] gave a quantum interior point algorithm for solving LPs and SDPs. They work in an input model where the input matrices are stored in QROM, which input model is also covered by our quantum operator input model. However, it is hard to compare their complexities to ours, because their final complexity statement depends polynomially on the condition number of the matrices that the interior point method encounters, and they do not give explicit bounds for these condition numbers. Also they have two accuracy parameters, while one accuracy parameter only appears as a logarithmic factor, their complexity depends polynomially on the other.

Very recently Apeldoorn et al. [6] and Chakrabarti et al. [14] developed improved quantum algorithms for general black-box convex optimization. Since we work in a model where we are given access directly to the constraints defining the problem, our results are incomparable.

References

1 Scott Aaronson. Quantum Copy-Protection and Quantum Money. In *Proceedings of the 24th IEEE Conference on Computational Complexity (CCC)*, pages 229–242, 2009. arXiv: 1110.5353 doi:10.1109/CCC.2009.42.

2 Scott Aaronson. Shadow Tomography of Quantum States. In *Proceedings of the 50th ACM Symposium on Theory of Computing (STOC)*, pages 325–338, 2018. arXiv: 1711.01053 doi:10.1145/3188745.3188802.

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

4 Joran van Apeldoorn and András Gilyén. Quantum algorithms for zero-sum games. arXiv: 1904.03180, 2019.

5 Joran van Apeldoorn, András Gilyén, Sander Gribling, and Ronald de Wolf. Quantum SDP-Solvers: Better upper and lower bounds. In *Proceedings of the 58th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 403–414, 2017. arXiv: 1705.01843 doi:10.1109/FOCS.2017.44.

6 Joran van Apeldoorn, András Gilyén, Sander Gribling, and Ronald de Wolf. Convex optimization using quantum oracles. arXiv: 1809.00643, 2018.

7 Sanjeev Arora and Satyen Kale. A Combinatorial, Primal-Dual Approach to Semidefinite Programs. *Journal of the ACM*, 63(2):12:1–12:35, 2016. Earlier version in STOC'07. doi:10.1145/2837020.

8 Aleksandrs Belovs. Variations on Quantum Adversary. arXiv: 1504.06943, 2015.

9 Dominic W. Berry, Andrew M. Childs, and Robin Kothari. Hamiltonian Simulation with Nearly Optimal Dependence on all Parameters. In *Proceedings of the 56th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 792–809, 2015. arXiv: 1501.01715 doi:10.1109/FOCS.2015.54.

10 Arvid J. Bessen. Lower bound for quantum phase estimation. *Physical Review A*, 71(4):042313, 2005. arXiv: quant-ph/0412008 doi:10.1103/PhysRevA.71.042313.

11 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, 2017. First arXiv version. arXiv:1710.02581v1.

12 Fernando G. S. L. Brandão, Amir Kalev, Tongyang Li, Cedric Yen-Yu Lin, Krysta M. Svore, and Xiaodi Wu. Quantum SDP Solvers: Large Speed-ups, Optimality, and Applications to Quantum Learning, 2018. arXiv:1710.02581v2.

13 Fernando G. S. L. Brandão and Krysta M. Svore. Quantum Speed-ups for Solving Semidefinite Programs. In *Proceedings of the 58th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 415–426, 2017. arXiv: 1609.05537 doi:10.1109/FOCS.2017.45.

14 Shouvanik Chakrabarti, Andrew M. Childs, Tongyang Li, and Xiaodi Wu. Quantum algorithms and lower bounds for convex optimization. arXiv: 1809.01731, 2018.

15 Shantanav Chakraborty, András Gilyén, and Stacey Jeffery. The power of block-encoded matrix powers: improved regression techniques via faster Hamiltonian simulation. arXiv: 1804.01973, 2018.

16 Richard Cleve, Peter Høyer, Benjamin Toner, and John Watrous. Consequences and limits of nonlocal strategies. In *Proceedings of the 19th IEEE Conference on Computational Complexity (CCC)*, pages 236–249, 2004. arXiv: quant-ph/0404076 doi:10.1109/CCC.2004.1313847.

17 Christoph Dürr and Peter Høyer. A Quantum Algorithm for Finding the Minimum. arXiv: quant-ph/9607014, 1996.

18 Yonina C. Eldar. A Semidefinite Programming Approach to Optimal Unambiguous Discrimination of Quantum States. *IEEE Transactions on Information Theory*, 49:446–456, 2003. arXiv: quant-ph/0206093 doi:10.1109/TIT.2002.807291.

19 András Gilyén, Srinivasan Arunachalam, and Nathan Wiebe. Optimizing quantum optimization algorithms via faster quantum gradient computation. In *Proceedings of the 30th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1425–1444, 2019. arXiv: 1711.00465 doi:10.1137/1.9781611975482.87.

20 András Gilyén, Yuan Su, Guang Hao Low, and Nathan Wiebe. Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics. In *Proceedings of the 51st ACM Symposium on Theory of Computing (STOC)*, 2019. (to appear) arXiv: 1806.01838 doi:10.1145/3313276.3316366.

21 Michel X. Goemans and David P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, 42(6):1115–1145, 1995. Earlier version in STOC'94. doi:10.1145/227683.227684.

22 Iordanis Kerenidis and Anupam Prakash. Quantum Recommendation Systems. In *Proceedings of the 8th Innovations in Theoretical Computer Science Conference (ITCS)*, pages 49:1–49:21, 2017. arXiv: 1603.08675 doi:10.4230/LIPIcs.ITCS.2017.49.

23 Iordanis Kerenidis and Anupam Prakash. A Quantum Interior Point Method for LPs and SDPs. arXiv: 1808.09266, 2018.

24 Jean Lasserre. Global Optimization with Polynomials and the Problem of Moments. *SIAM Journal on Optimization*, 11(3):796–817, 2001. doi:10.1137/S1052623400366802.

25 James R. Lee, Prasad Raghavendra, and David Steurer. Lower Bounds on the Size of Semidefinite Programming Relaxations. In *Proceedings of the 47th ACM Symposium on Theory of Computing (STOC)*, pages 567–576, 2015. arXiv: 1411.6317 doi:10.1145/2746539.2746599.

26 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 56th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 1049–1065, 2015. arXiv: 1508.04874 doi:10.1109/FOCS.2015.68.

27 Guang Hao Low and Isaac L. Chuang. Hamiltonian Simulation by Qubitization. arXiv: 1610.06546, 2016.

28 Guang Hao Low and Isaac L. Chuang. Hamiltonian Simulation by Uniform Spectral Amplification. arXiv: 1707.05391, 2017.

29 Pablo A. Parrilo. *Structured Semidefinite Programs and Semialgebraic Geometry Methods in Robustness and Optimization*. PhD thesis, California Institute of Technology, 2000. URL: https://thesis.library.caltech.edu/1647/.

30 Xiaodi Wu. Personal communication. Email, November 2017.