

Sparse Approximate Solutions to Semidefinite Programs

Elad Hazan

IBM Almaden Research Center
650 Harry Road, San Jose, 95120 CA, USA
hazan@us.ibm.com

Abstract. We propose an algorithm for approximately maximizing a concave function over the bounded semi-definite cone, which produces **sparse** solutions. Sparsity for SDP corresponds to low rank matrices, and is an important property for both computational as well as learning theoretic reasons. As an application, building on Aaronson's recent work, we derive a linear time algorithm for Quantum State Tomography.

1 Introduction

In this paper we describe a simple algorithm for approximately solving a special case of Semi-Definite Programs (SDP), in which the goal is to maximize a concave function in the bounded semi-definite cone, which also produces a sparse solution. Our notion of sparsity is not the usual one, i.e. small number of non-zero entries in the solution matrix. Rather, the notion of sparsity in the SDP setting is low rank, coupled with a Cholesky composition representation of the solution. That is, our solutions will be of the form $X = VV^T \in \mathbb{R}^{n \times n}$, and the solution X is represented by the matrix $V \in \mathbb{R}^{n \times k}$. This notion of sparsity is computationally appealing: the time to compute vector-matrix products with the matrix X , as well as the space required to store it is $O(nk)$.

Our algorithm and its analysis are different from previous work on approximation algorithms for SDP. Unlike previous approximate SDP approaches, the algorithm is **not** based on the multiplicative weights method and its extensions. Rather, the main ingredient is an extension of the Frank-Wolfe [FW56] algorithm¹ for optimizing a single function over the bounded PSD cone. The analysis of this part crucially depends on the dual SDP, and goes beyond the use of online learning techniques which were prevalent in all previous approaches.

The algorithm has the appealing property of proceeding in iterations, such that in each iteration the solution is provably better than the preceding one, and has rank at most one larger. Thus after k iterations the algorithm attains a $\frac{1}{k}$ -approximate² solution with rank at most k . The previous algorithm of

¹ whose analysis was recently revisited, simplified and extended in [Cla]. The latter paper inspired this work.

² A ε -approximate solution to a SDP over the bounded cone is a PSD matrix with trace equal to one, which satisfies all constraints up to an additive term of ε

[AHK05], which was based on the multiplicative weights method, also produces sparse solutions, but requires $\Omega(k^2)$ iterations to achieve the same approximation guarantee. Besides elementary computation, in each iteration our algorithm performs at most one eigenvector computation (which can be replaced by two approximate eigenvector computations).

Next, we apply our method to the problem of Quantum State Tomography. In this problem the goal is to approximately learn a mixed quantum state given access to independent measurements. Recently, Aaronson [Aar] showed how a combination of techniques from learning theory and quantum computation can be used to learn an n -bit mixed quantum state with a linear number of measurements (an exponential improvement over previous techniques). We show how to approximately solve Aaronson’s proposed SDP in linear time.

This application is particularly suited for our method for the following reasons: First, Aaronson proved that an approximate solution to his SDP guarantees low generalization error. Thus solving the SDP approximately (rather than exactly) is part of the problem statement and suited for approximation techniques. Even more so, an exact solution corresponds to an exact match of the hypothesis to the data, and could lead to ”overfitting”. Second, a widely accepted principle known as ”Occam’s Razor” roughly states that simple hypothesis are preferable. Our algorithm generates a simple hypothesis, taking low-rank is a notion of simplicity. Third, the state matrix of an n -qubit state is of size 2^n . For even a small number of qubits the size of the matrices is too large to effectively deploy any algorithm with super-linear complexity. This rules out interior point methods. For this reason also, exploiting the sparsity of the input is absolutely necessary.

Finally, we show how to extend the optimization routine over the bounded semi-definite cone in order to approximately solve general SDPs. However, for general SDPs we do not improve the running time over existing techniques. Whereas for optimization over the bounded SDP cone our algorithm produces an ε -approximation in $O(\frac{1}{\varepsilon})$ iterations, which improves upon all existing approximation methods, for general SDPs we need $O(\frac{1}{\varepsilon^2})$ iterations.

1.1 Existence of sparse solutions

In the first part of the paper we focus on the problem of maximizing a single concave function over the cone of semi-definite matrices with trace bounded by one. A natural question concerning sparse solutions is what is the minimal rank required of a solution which is ”close” to the optimum. Obviously, if the concave objective function happens to be linear, then the problem reduces to an eigenvector computation, and the optimal solution is of rank one. However, this is not the case in general, and even for quadratic functions the optimal solution can have full rank.

It follows from the work of Clarkson [Cla], that in order to attain an ε -additive approximation (in a sense that will be made precise below), the rank of the approximation needs to be at least $\Omega(\frac{1}{\varepsilon})$. However it does not follow from previous work that there even *exists* an ε -approximate solution with such rank.

The best bound we know of is from [AHK05], which implies the existence of an ε -approximate solution with rank $O(\frac{1}{\varepsilon^2})$.

In this paper we give a constructive proof that an ε -approximate solution with rank $O(\frac{1}{\varepsilon})$ exists by giving an efficient algorithm to find it (see Theorem 1 below).

2 Preliminaries

For two matrices $A, B \in \mathbb{C}^{n \times n}$ we denote by $A \bullet B$ the inner product of these matrices when thought of as vectors in \mathbb{C}^{n^2} , i.e. $A \bullet B = \sum_{ij} A_{ij} \cdot B_{ij} = \mathbf{Tr}(AB)$. All results in this paper apply to Hermitian matrices, and this generality is important for our application to Quantum State Tomography. However, for simplicity the reader may think of real matrices.

A matrix $A \in \mathbb{C}^{n \times n}$ is positive semi-definite (PSD), denoted by $A \succeq 0$, if and only if all its eigenvalues are non-negative real numbers. We write $A \succeq B$ if the matrix $A - B \succeq 0$ is PSD. In the analysis we require the following basic characterization of PSD matrices.

Fact 21 *A matrix X is PSD if and only if $X \bullet V \geq 0$ for all $V \succeq 0$.*

Consider a general SDP feasibility problem (as is standard, the optimization version is reduced to feasibility by binary search)

$$\begin{aligned} \forall i \in [m] . A_i \bullet X &\leq b_i \\ X &\in \mathcal{P} \end{aligned} \tag{1}$$

We say that a matrix $X \succeq 0$ is ε -approximate solution to (1) if it satisfies $\forall i . A_i \bullet X - b_i \leq \varepsilon$.

Quantum State Tomography In the problem of quantum state tomography (QST), the goal is to learn an approximate description of a certain quantum state given access to independent measurements of the state. Recently, Aaronson [Aar] showed that a small set of measurements (linear in the number of qubits) can be used to accurately learn a state in this model.

The problem of finding a mixed n -qubit state which approximately agrees with the measurements can be cast as the following optimization problem. The input is m Hermitian matrices $E_1, \dots, E_m \in \mathbb{C}^{N \times N}$ (for the QST problem we denote $N = 2^n$) with eigenvalues in $[0, 1]$, and m real numbers $b_1, \dots, b_m \in [0, 1]$. The goal is to find a Hermitian positive semidefinite matrix X that satisfies (for a given constant $\eta > 0$)

$$\begin{aligned} \forall i . |E_i \bullet X - b_i| &\leq \eta \\ X \succ 0 , \mathbf{Tr}(X) &= 1 \end{aligned} \tag{2}$$

As alternative (and equivalent up to the approximation parameter) formulation, which may even be more interesting for both theoretical reasons as well as applications, is

$$\begin{aligned} \sum_i (E_i \bullet X - b_i)^2 &\leq \eta \\ X \succeq 0, \text{Tr}(X) &= 1 \end{aligned} \tag{3}$$

As Aaronson notes, the above formulation is a convex program with semi-definite constraints, and thus solvable in polynomial time using interior point methods [NN94, Ali95] or the ellipsoid method [GLS88]. However, the exponential size of the measurement matrices quickly render any super-linear method impractical.

3 A sparse approximate SDP solver

Let $\mathcal{P} = \{X \succeq 0, \text{Tr}(X) = 1\}$ be the cone of SDP matrices with trace equals one. This convex set is a natural generalization of the simplex, and is the set of all quantum distributions. In this section we consider the the following optimization problem:

$$\begin{aligned} \max f(X) \\ X \in \mathcal{P} \end{aligned} \tag{4}$$

Besides an interesting problem by its own right, we show in the next section how to use the algorithm we develop in this section to solve general SDP. The following simple and efficient algorithm always maintains a sparse PSD matrix, rank at most k after k iterations. The algorithm can be viewed as a generalization of the Frank-Wolfe algorithm for optimizing over the simplex, which was recently revisited by Clarkson [Cla].

Algorithm 1 SparseApproxSDP

- 1: Let f be a given concave function, with curvature constant C_f as defined below. Initialize $X_1 = v_0 v_0^\top$ for an arbitrary rank one matrix $v_0 v_0^\top$ (with trace one).
 - 2: **for** $k = 1, ..\infty$ **do**
 - 3: Let $\varepsilon_k = \frac{C_f}{k^2}$. Compute $v_k \leftarrow \text{APPROXEV}(\nabla f(X_k), \varepsilon_k)$.
 - 4: Let $\alpha_k = \min\{1, \frac{\varepsilon_k}{k}\}$.
 - 5: Set $X_{k+1} = X_k + \alpha_k(v_k v_k^\top - X_k)$.
 - 6: **end for**
-

The procedure APPROXEV used in algorithm 1 is an approximate eigenvalue solver. It guarantees the following: for a negative semidefinite matrix M and any $\varepsilon > 0$, the vector $x = \text{APPROXEV}(M, \varepsilon)$ satisfies $x^\top M x \geq \lambda_{\max}(M) - \varepsilon$. At this point the reader may think of this procedure as an exact maximum eigenvalue computation. In the end of this section we prove that APPROXEV can be made to have running time which is linear in the number of non-zero entries of M .

A crucial property of the function f which effects the convergence rate is its curvature constant, defined by Clarkson [Cla], as follows.

Definition 1. Define the curvature constant of f as

$$C_f \triangleq \sup_{x, z \in \mathcal{P}} \sup_{y=x+\alpha(z-x)} \frac{1}{\alpha^2} [f(x) - f(y) + (y-x)^\top \nabla f(x)]$$

Note that C_f is upper bounded by the largest eigenvalue of the Hessian of $-f$.

Our main performance guarantee is given by the following Theorem. Henceforth let X^* denote the optimal solution to (4).

Theorem 1. Let C_f be defined as in the following section. Then the iterates X_k of Algorithm 1 satisfy for all $k > 1$

$$\forall X^* \in \mathcal{P} . f(X_k) \geq f(X^*) - \frac{8C_f}{k}$$

REMARK: A similar guarantee can be given for α_k chosen greedily by binary search.

Before proving this theorem, we define some notation.

– Denote by

$$z(X) = \max_{\|v\|=1} v^\top \nabla f(X) v = \lambda_{max}(\nabla f(X))$$

the largest eigenvalue of the gradient of f at X .

– Let $w(X) = z(X) + f(X) - X \bullet \nabla f(X)$. As we show in Lemma 1 below, $w(X)$ is the dual objective to optimization problem (4).

– We also denote by $h(X) = \frac{f(X^*) - f(X)}{4C_f}$ the normalized distance to the optimum at X (in value) and by $g(X) = \frac{w(X) - f(X)}{4C_f}$ the duality gap at X .

Lemma 1. Weak duality:

$$w(X) \geq w(X^*) \geq f(X^*) \geq f(X)$$

Proof. It suffices to prove that the formulation:

$$\min_{X \in \mathbb{S}} w(X)$$

(here \mathbb{S} is the set of all matrices in $\mathbb{R}^{n \times n}$) is the dual of (4). To see that, let's write the Lagrangian relaxation of (4), which is equivalent to $-\min_{X \in \mathcal{P}} -f(X)$ (for this formulation recall Fact 21):

$$-\max_{V \succeq 0, z \in \mathbb{R}} \min_{X \in \mathbb{S}} -f(X) - X \bullet V + z(X \bullet I - 1)$$

The optimum is obtained when all derivatives w.r.t x are zero, i.e.

$$-\nabla f(X) - V + zI = 0$$

which implies $V = zI - \nabla f(X)$. There is also the obvious constraint that $0 \preceq V = zI - \nabla f(X)$. Plugging back we get that the optimization problem becomes

$$\begin{aligned} & - \max_{z \in \mathbb{R}} \min_{X \in \mathbb{S}} -f(X) - X \bullet (zI - \nabla f(X)) + z(X \bullet I - 1) = \\ & - \max_{z \in \mathbb{R}} \min_{X \in \mathbb{S}} -f(X) + X \bullet \nabla f(X) - z = \\ & - \max_{z \in \mathbb{R}, X \in \mathbb{S}} -f(X) + X \bullet \nabla f(X) - z = \\ & \min_{z \in \mathbb{R}, X \in \mathbb{S}} f(X) - X \bullet \nabla f(X) + z \end{aligned}$$

Subject to $zI \succeq \nabla f(X)$, or $z \geq \lambda_{max}\{\nabla f(X)\}$. So w.l.o.g we have $z = z(X) = \lambda_{max}\{\nabla f(X)\}$ and the above becomes $\min_{X \in \mathbb{S}} w(X)$.

In particular, the above implies that $g(X) \geq h(X)$. We can now prove the theorem:

Proof (Proof of Theorem 1). By the definitions above we have

$$v_k^\top \nabla f(X_k) v_k \geq z(X_k) - \varepsilon_k = w(X_k) - f(X_k) + X_k \bullet \nabla f(X_k) - \varepsilon_k$$

Therefore

$$\begin{aligned} (v_k v_k^\top - X) \bullet \nabla f(X_k) &= v_k^\top \nabla f(X_k) v_k - X \bullet \nabla f(X_k) \\ &\geq w(X_k) - f(X_k) - \varepsilon_k \end{aligned} \tag{5}$$

Now,

$$\begin{aligned} f(X_{k+1}) &= f(X_k + \alpha_k(v_k v_k^\top - X_k)) \\ &\geq f(X_k) + \alpha_k(v_k v_k^\top - X_k) \bullet \nabla f(X_k) - \alpha_k^2 C_f \quad \text{by definition of } C_f \\ &\geq f(X_k) + \alpha_k(w(X_k) - f(X_k)) - \alpha_k^2 C_f - \varepsilon_k \quad \text{by (5)} \\ &= f(X_k) + 4C_f g(X_k) \alpha_k - C_f \alpha_k^2 - \varepsilon_k \\ &\geq f(X_k) + 4C_f h(X_k) \alpha_k - C_f \alpha_k^2 - \varepsilon_k \end{aligned}$$

By definition of $h(X_k)$ and Lemma 1 this implies

$$h(X_{k+1}) \leq h(X_k) - \alpha_k h(X_k) + \frac{1}{4} \alpha_k^2 + \frac{\varepsilon_k}{4C_f}$$

Let $\varepsilon_k = \frac{C_f}{k^2} \geq C_f \cdot h(X_k)$. By this choice, $h(X_{k+1}) \leq h(X_k) - \alpha_k h(X_k) + \frac{1}{2} \alpha_k^2$.

Now we prove inductively that $h(x_k) \leq \frac{2}{k}$. In the first iteration this is true since $\alpha_1 = 1$, and get that $h(X_2) \leq \frac{1}{2}$. So by taking $\alpha_k = \frac{2}{k}$ we have

$$\begin{aligned} h(X_{k+1}) &\leq h(X_k)(1 - \alpha_k) + \frac{1}{2} \alpha_k^2 \\ &\leq \frac{2}{k} \left(1 - \frac{2}{k}\right) + \frac{2}{k^2} \\ &= \frac{2}{k} \left(1 - \frac{1}{k}\right) \leq \frac{2}{k+1} \end{aligned}$$

3.1 Using approximate eigenvector computations

We now describe how to efficiently implement an eigenvector computation procedure using the Lanczos algorithm with a random start. This was first shown by [AHK05].

Lemma 2. *Given a NSD matrix M with \tilde{N} non-zero entries and eigenvalues in the range $[-C, 0]$, there exists an algorithm which produces in time $\tilde{O}(\frac{\tilde{N}\sqrt{C}}{\sqrt{\varepsilon}})$ a vector x such that*

$$x^\top Mx \geq \lambda_{\max}(M) - \varepsilon$$

Proof. We need Theorem 3.2(a) of Kuczyński and Wozniakowski [KW92]:

Lemma 3. *Let $M \in \mathbb{R}^{n \times n}$ be a positive semidefinite matrix with \tilde{N} non-zero entries. Then with high probability, the Lanczos algorithm produces in $O(\frac{\log(n)}{\sqrt{\gamma}})$ iterations a unit vector x such that $\frac{x^\top Mx}{\lambda_{\max}(M)} \geq 1 - \gamma$.*

Note that each iteration of the Lanczos algorithm takes $\tilde{O}(\tilde{N})$ time. Now let $M_2 = CI + M$. Notice that M_2 is positive semidefinite, and $\lambda_i(M_2) = C + \lambda_i(M)$. We apply Lemma 3 with $\gamma = \frac{\varepsilon}{C}$ to obtain in time $\tilde{O}(\frac{\tilde{N}}{\sqrt{\gamma}})$ a unit vector x such that:

$$\frac{C + x^\top Mx}{C + \lambda_{\max}(M)} = \frac{x^\top M_2x}{\lambda_{\max}(M_2)} \geq 1 - \frac{\varepsilon}{C}$$

Simplifying this gives the lemma statement.

Combining the pieces together we obtain:

Theorem 2. *Let C_k be a bound on the absolute value of the largest eigenvalue of $\nabla f(X_k)$, $C = \max_k \{C_k\}$ and \tilde{N} be the maximal number of non-zero entries in $\nabla f(X)$. Algorithm 1 returns a δ -approximate solution in time*

$$\tilde{O}\left(\frac{C_f(n + T_{GD})}{\delta} + \frac{\tilde{N}\sqrt{C}C_f^{1.5}}{\delta^2}\right)$$

Where T_{GD} is the time to compute $\nabla f(X_k)$.

Proof. By Theorem 1, we have $f(X_k) \geq f(X^*) - \frac{4C_f}{k}$, hence it suffices to have $k = \frac{4C_f}{\delta}$ iterations. Besides the calls to APPROXEV, the computation required in each iteration is the computation of the gradient (in time T_{GD}) and other elementary computations which can be carried out in time $O(n)$. By lemma 2, the k 'th invocation of APPROXEV runs in time $\tilde{O}(\frac{\tilde{N}\sqrt{C_k}}{\sqrt{\varepsilon_k}}) = \tilde{O}(k\tilde{N}\sqrt{C_k/C_f})$. The total running time thus comes to

$$\frac{4C_f}{\delta} \cdot (n + T_{GD}) + \sum_{i=1}^k \tilde{O}\left(\frac{i \cdot \tilde{N}\sqrt{C_i}}{\sqrt{C_f}}\right)$$

4 Solving general SDPs

In this section we apply the sparse SDP solver to approximately solve general a SDP in the form 1. We note that the results in this section do not improve over [AHK05] in terms of running time (whereas for the case of optimizing a single function over the bounded semi-definite cone we do obtain an improvement in running time), and are given here only to illustrate how similar results can be obtained through a very different analysis. In fact, the algorithm below is almost identical to the one obtained in [AHK05] via Multiplicative Weights techniques.

Algorithm 2 Fast SDP

- 1: Input: set of constraints given by $A_1, \dots, A_m \in \mathbb{C}^{N \times N}$ and $b_1, \dots, b_m \in \mathbb{R}$. Desired accuracy ε . Let $\omega = \max_i \{\lambda_{\max}(A_i)\}$.
- 2: Let $M = \frac{\log m}{\varepsilon}$
- 3: Apply algorithm 1 to the following function for k rounds, with $k = \frac{1}{\varepsilon}$.

$$f(X) = -\frac{1}{M} \log \left(\sum_{i=1}^m e^{M \cdot (A_i \bullet X - b_i)} \right)$$

- 4: if $f(X_k) < -\varepsilon$ return FAIL. Else, return X_k .
-

Lemma 4. *A matrix X for which $f(X) \geq -\varepsilon$ is a ε -approximate solution to QST.*

Proof. Consider the function (for $M = \frac{\log m}{\varepsilon}$ and $y \in \mathbb{R}^m$)

$$\Phi(y) = \frac{1}{M} \log \left(\sum_i e^{M \cdot y_i} \right)$$

It is a well known fact (see [GK94]) that for $M \geq 0$, we have

$$\lambda(x) \leq \Phi(x) \leq \lambda(x) + \frac{\log m}{M}$$

Where

$$\lambda(x) \triangleq \max_i y_i$$

Therefore, if X satisfies $f(X) \geq -\varepsilon$, we have $\Phi(X) \leq \varepsilon$ for $y_i = A_i \bullet X - b_i$. This implies that $\max_i y_i \leq \Phi(x) \leq \varepsilon$ and hence

$$\forall i . A_i \bullet X - b_i \leq \varepsilon$$

Lemma 5. *The function f above is concave, and its C_f value is bounded by $\frac{\omega^2 \log m}{\varepsilon}$. The value C is bounded by $C \leq \omega$.*

Proof. See [BV04] for a proof that that $-f$ is convex.

Let us now define $z(X) \in \mathbb{R}^{2m}$ as the vector such that $z(X)_i = e^{M \cdot (A_i \bullet X - b_i)}$ for $i \in [m]$. In the following we just say z when there's no ambiguity as for the X in question.

Differentiating $g(X) = -f(X)$ with respect to X we get

$$M \cdot \nabla g(X) = \frac{1}{1^\top z} \sum_{i=1}^m z_i \cdot M A_i \quad (6)$$

So we can bound the parameter C by

$$C \leq \lambda_{\max}(\nabla g) = \lambda_{\max}\left(\frac{1}{1^\top z} \sum_{i=1}^m z_i A_i\right) \leq \omega$$

And taking the second derivative we get

$$M \cdot \nabla^2 g(X) = -\frac{1}{(1^\top z)^2} \sum_{i,j=1}^m z_i z_j A_i \bullet A_j \cdot M^2 + \frac{1}{1^\top z} \sum_i z_i A_i \bullet A_i \cdot M^2$$

Hence,

$$\nabla^2 g(X) \preceq M \cdot \frac{1}{1^\top z} \sum_i z_i A_i \bullet A_i \preceq M \cdot \omega^2 I$$

and therefore $\lambda_{\max}(\nabla^2 g(X)) \leq M\omega^2$. Since C_f is bounded by the absolute value of the smallest eigenvalue of $\nabla^2 f$ on \mathcal{P} , this gives the Lemma.

Theorem 3. *If SDP (1) is feasible, then Algorithm 2 returns a ε -approximate solution, else it returns FAIL. The algorithm performs $\frac{\omega^2 \log m}{\varepsilon^2}$ approximate eigenvalue computations and has total running time of*

$$\tilde{O}\left(\frac{\omega^2 n}{\varepsilon^2} + \tilde{N} \cdot \frac{\omega^{3.5}}{\varepsilon^{3.5}}\right)$$

Where \tilde{N} is the total number of non-zero entries in A_1, \dots, A_m .

Proof. For the feasible solution X^* , we have $y_i^* = A_i \bullet X^* - b_i \leq 0$. Hence,

$$f(X^*) = -\frac{1}{M} \log\left(\sum_i e^{M y_i^*}\right) \geq -\frac{1}{M} \log(m) = -\varepsilon$$

Therefore, we get a $\delta = \varepsilon$ approximation after $\frac{C_f}{\varepsilon} \leq \frac{\omega^2 \log m}{\varepsilon^2}$ iterations, and have $f(X_k) \geq f(X^*) - \varepsilon \geq -2\varepsilon$, which according to Lemma 4 is a 2ε approximation solution to QST.

According to Theorem 2 the total running time is bounded by

$$\tilde{O}\left(\frac{C_f(n + T_{GD})}{\delta} + \frac{\tilde{N} \sqrt{C} C_f^{1.5}}{\delta^2}\right)$$

The gradient of f is (see equation (6))

$$\nabla f(X_k) = -\frac{1}{1^\top z} \sum_{i=1}^m z_i \cdot A_i$$

This is a convex combination of the matrices $\{-A_i\}$ according to the distribution z . Note that the distribution z at iteration k , denoted z_k , can be obtained from z_{k-1} in time $\tilde{O}(\tilde{N})$ since $X_k = (1 - \alpha_k)X_{k-1} + \alpha_k v_k v_k^\top$. Therefore $T_{GD} = \tilde{O}(\tilde{N})$. Also, in our case $\delta = \varepsilon$, and C_f, C are bounded as in lemma 5.

Plugging these bounds the lemma is obtained.

5 QST in linear time

The application to QST is straightforward. To solve formulation (3), apply Theorem 1 to obtain the corollary below. We note that the curvature constant C_f is bounded by the largest eigenvalue of the Hessian, which is bounded by $C_f \leq 2 \sum_{i=1}^m \lambda_{\max}(E_i)^2$.

Corollary 1. *Algorithm 1 returns a ε -approximate solution to an instance of QST in $\frac{4C_f}{\varepsilon}$ iterations.*

As for formulation (2), we have $2m$ constraints of the form $E_i \bullet X - b_i \leq \eta$ and $b_i - E_i \bullet X \leq \eta$. The parameter ω is bounded by the maximum eigenvalue of E_i , which is just 1. Also recall that for QST the dimension of the matrices E_i is $N = 2^n$. By Theorem 3 we conclude

Corollary 2. *Algorithm 2 returns a ε -approximate solution to an instance of QST in $\frac{\log m}{\varepsilon^2}$ iterations and has total running time of*

$$\tilde{O}\left(\frac{N}{\varepsilon^2} + \frac{\tilde{N}}{\varepsilon^{3.5}}\right)$$

Where \tilde{N} is the total number of non-zero entries in the matrices E_1, \dots, E_m .

Using (standard) clever implementation techniques, one can remove the dependence on $N = 2^n$ completely, and obtain a running time of

$$\tilde{O}\left(\frac{\tilde{N}}{\varepsilon^{3.5}}\right)$$

6 Acknowledgements

Thanks to Prateek Jain for correcting some mistakes in earlier versions.

References

- [Aar] Scott Aaronson. The learnability of quantum states. *arXiv:quant-ph/0608142v3*.
- [AHK05] Sanjeev Arora, Elad Hazan, and Satyen Kale. Fast algorithms for approximate semidefinite programming using the multiplicative weights update method. In *46th IEEE FOCS*, pages 339–348, 2005.
- [Ali95] F. Alizadeh. Interior point methods in semidefinite programming with applications to combinatorial optimization. *SIAM J. Optim.*, 5(1):13–51, 1995.
- [BV04] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004.
- [Cla] Kenneth L. Clarkson. Coresets, sparse greedy approximation, and the Frank-Wolfe algorithm. In *SODA 2008: Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*.
- [FW56] M. Frank and P. Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3:149–154, 1956.
- [GK94] Michael D. Grigoriadis and Leonid G. Khachiyan. Fast approximation schemes for convex programs with many block and coupling constraints. *SIAM Journal on Optimization*, 4:86–107, 1994.
- [GLS88] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer Verlag, 1988.
- [KW92] J. Kuczyński and H. Woźniakowski. Estimating the largest eigenvalue by the power and lanczos algorithms with a random start. *SIAM Journal on Matrix Analysis and Applications*, 13(4):1094–1122, 1992.
- [NN94] Y. Nesterov and A. Nemirovskii. *Interior Point Polynomial Methods in Convex Programming: Theory and Applications*. Society for Industrial and Applied Mathematics, Philadelphia, 1994.