UNIVERSITY OF AMSTERDAM

UvA-DARE (Digital Academic Repository)

Convex optimization using quantum oracles

van Apeldoorn, J.; Gilyén, A.; Gribling, S.; de Wolf, R.

DOI
10.22331/q-2020-01-13-220

Publication date
2020

Document Version
Final published version

Published in
Quantum - the open journal for quantum science

License
CC BY

Link to publication

Citation for published version (APA):
van Apeldoorn, J., Gilyén, A., Gribling, S., & de Wolf, R. (2020). Convex optimization using quantum oracles. *Quantum - the open journal for quantum science*, *4*, Article 220. https://doi.org/10.22331/q-2020-01-13-220

General rights
It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: https://uba.uva.nl/en/contact, or a letter to: Library of the University of Amsterdam, Secretariat, P.O. Box 19185, 1000 GD Amsterdam, The Netherlands. You will be contacted as soon as possible.

UvA-DARE is a service provided by the library of the University of Amsterdam (https://dare.uva.nl)

Download date:15 Sept 2026

Convex optimization using quantum oracles

Joran van Apeldoorn[1], András Gilyén[2], Sander Gribling[3], and Ronald de Wolf[4]

[1]QuSoft, CWI, Amsterdam, the Netherlands. apeldoor@cwi.nl
[2]QuSoft, CWI, Amsterdam, the Netherlands. gilyen@cwi.nl
[3]QuSoft, CWI, Amsterdam, the Netherlands. gribling@cwi.nl
[4]QuSoft, CWI and University of Amsterdam, the Netherlands. rdewolf@cwi.nl

arXiv:1809.00643v4 [quant-ph] 20 Dec 2019

We study to what extent quantum algorithms can speed up solving convex optimization problems. Following the classical literature we assume access to a convex set via various oracles, and we examine the efficiency of reductions between the different oracles. In particular, we show how a separation oracle can be implemented using $\widetilde{\mathcal{O}}(1)$ quantum queries to a membership oracle, which is an exponential quantum speed-up over the $\Omega(n)$ membership queries that are needed classically. We show that a quantum computer can very efficiently compute an approximate subgradient of a convex Lipschitz function. Combining this with a simplification of recent classical work of Lee, Sidford, and Vempala gives our efficient separation oracle. This in turn implies, via a known algorithm, that $\widetilde{\mathcal{O}}(n)$ quantum queries to a membership oracle suffice to implement an optimization oracle (the best known classical upper bound on the number of membership queries is quadratic). We also prove several lower bounds: $\Omega(\sqrt{n})$ quantum separation (or membership) queries are needed for optimization if the algorithm knows an interior point of the convex set, and $\Omega(n)$ quantum separation queries are needed if it does not.

1 Introduction

Optimization is a fundamental problem in mathematics and computer science, with many real-world applications. As people try to solve larger and larger optimization problems, the efficiency of optimization becomes more and more important, motivating us to find the best possible algorithms. Recent experimental progress on building quantum computers draws attention to new approaches to the problem: can we solve optimization problems more efficiently by exploiting quantum effects such as superposition, interference, and entanglement? For many discrete optimization problems [Gro96, DH96, Szeg04, DHHM06, AŠ06] significant speed-ups have been shown, but less is known about *continuous* optimization problems.

One of the most successful continuous optimization paradigms is *convex* optimization, which optimizes a convex function over a convex set that is given explicitly (by a set of constraints) or implicitly (by an oracle). See Bubeck [Bub15] for a recent survey. Quantum algorithms for convex optimization have been considered before. In 2008, Jordan [Jor08] described a faster quantum algorithm for minimizing quadratic functions. Recently, for an important class of convex optimization problems (semidefinite optimization) quantum speed-ups were achieved using algorithms whose runtime scales polynomially with the desired precision and some geometric parameters [BS17, vAGGdW17, BKL+19, vAG19].

However, many convex optimization problems can be solved classically using algorithms whose runtime scales *logarithmically* with the desired precision and the relevant geometric parameters. We are aware of only one quantum speed-up which is partially in this regime, namely the very recent quantum interior point method of Kerenidis and Prakash [KP18]. In this paper we look at general convex optimization problems, considering algorithms that have such favorable logarithmic scaling with the precision.

The generic problem in convex optimization is minimizing a convex function $f : K \to \mathbb{R} \cup \{\infty\}$, where $K \subseteq \mathbb{R}^n$ is a convex set. We consider the setting where an interior point $x_0 \in \mathrm{int}(K)$ is given and radii $r, R > 0$ are known such that $B(x_0, r) \subseteq K \subseteq B(x_0, R)$, where $B(x_0, r)$ is the Euclidean ball of radius r centered at x_0.

It is well-known that if the convex function is bounded on K, then we can equivalently consider the problem of minimizing a *linear* function over a different convex set $K' \subseteq \mathbb{R}^{n+1}$, namely the epigraph $K' = \{(x, \mu) : x \in K, f(x) \geq \mu\}$ of f. Accessing K' is easy given access to K and f, and the parameters involved will be similar. Conversely, for any linear optimization problem over an unknown convex set K, there is an equivalent optimization problem over a known convex set (say, the ball), with an unknown bounded convex objective function f that can be evaluated easily given access to K. From now on we therefore focus on optimizing a known linear function over an unknown convex set.

We consider the setting where access to the convex set is given only in a black-box manner, through an oracle. The five basic problems (oracles) in convex optimization identified by Grötschel, Lovász, and Schrijver [GLS88] are: membership, separation, optimization, violation, and validity (see Section 2 for the definitions). They showed that all five basic problems are polynomial-time equivalent. That is, given an oracle O for one of these problems, one can implement an oracle for any of the other problems using a polynomial number of calls to O and polynomially many other elementary operations. Subsequent work made these polynomial-time reductions more efficient, reducing the degree of the polynomials. Recently Lee et al. [LSV18], in the classical setting, showed that with $\widetilde{\mathcal{O}}(n^2)$ calls[1] to a membership oracle (and $\widetilde{\mathcal{O}}(n^3)$ other elementary arithmetic operations) one can solve an optimization problem. They did so by showing that $\widetilde{\mathcal{O}}(n)$ calls to a membership oracle suffice to do separation, and then composing this with the known fact [LSW15] (see also [LSV18, Theorem 15]) that $\widetilde{\mathcal{O}}(n)$ calls to a separation oracle suffice for optimization.

Our main result (Section 4) shows that on a quantum computer, $\widetilde{\mathcal{O}}(1)$ calls to a membership oracle suffice to implement a separation oracle, and hence (by the known classical reduction from optimization to separation) $\widetilde{\mathcal{O}}(n)$ calls to a membership oracle suffice for optimization.[2] Lee et al. [LSV18] use a geometric idea to reduce separation to finding an approximate subgradient of a convex Lipschitz function. They then show that $\widetilde{\mathcal{O}}(n)$ evaluations of a convex Lipschitz function suffice to get an approximate subgradient. Our contributions here are twofold (Section 3 and 4). We use the same geometric idea, but we provide a simpler way to compute an approximate subgradient of a convex Lipschitz function (Section 3). We point out that this new algorithm is purely classical. Besides

[1] Here, and in the rest of the paper, the notation $\widetilde{\mathcal{O}}(\cdot)$ is used to hide polylogarithmic factors in n, r, R, ε.

[2] Although not stated explicitly in our results, we also use $\widetilde{\mathcal{O}}(n^3)$ additional operations for optimization using membership, like [LSV18]. This is because our quantum algorithm for separation uses only $\widetilde{\mathcal{O}}(n)$ gates in addition to the $\widetilde{\mathcal{O}}(1)$ membership queries, and we use the same reduction from optimization to separation as [LSV18]. If queries themselves have significant time complexity, then our algorithm does lead to a speedup in time complexity over the best known classical algorithm. For example, if each membership query (with the required precision) takes time $\widetilde{\mathcal{O}}(n^2)$ to implement, then our quantum algorithm for optimization has time complexity $\widetilde{\mathcal{O}}(n^3)$, while the classical algorithm will use time $\widetilde{\mathcal{O}}(n^4)$ because it uses $\widetilde{\mathcal{O}}(n^2)$ membership queries.

being simpler, the main advantage of our algorithm is that it is suitable for a quantum speed-up using known quantum algorithms (Jordan's algorithm) for computing approximate (sub)gradients [Jor05, GAW19], which we show in Section 4. To show our quantum speed-up, we have to extend Jordan's quantum algorithm for gradient-computation to the case of convex Lipschitz functions.

As a second set of results, in Section 5 we provide lower bounds on the number of membership or separation queries needed to implement several other oracles. We show that our quantum reduction from separation to membership indeed improves over the best possible classical reduction: $\Omega(n)$ classical membership queries are needed to do separation.[3] We only have partial results regarding the optimality of the reduction from optimization to separation. In the setting where we are not given an interior point of the set K, we can prove an essentially optimal $\Omega(n)$ lower bound on the number of quantum queries to a separation oracle needed to do optimization, using the general adversary bound. This lower bound implies that a quantum computer offers no query speed-up over a classical computer for the task of finding an interior point.

However, for the case of quantum algorithms that *do* know an interior point, we are only able to prove an $\Omega(\sqrt{n})$ lower bound. In the classical setting, regardless of whether or not we know an interior point, the reduction uses $\widetilde{\Theta}(n)$ queries. This raises the interesting question of whether knowing an interior point can lead to a better quantum algorithm. We therefore view closing the gap between upper and lower bound as an important direction for future work.

Finally, we briefly mention (Section 6) how to obtain upper and lower bounds for some of the other oracle reductions, using a convex polarity argument. As we show, in the setting where we are given an interior point, the relation between membership and separation is analogous to the relation between validity and optimization. In particular, our better quantum algorithm for separation using membership queries implies that on a quantum computer $\widetilde{\mathcal{O}}(1)$ queries to a validity oracle suffice to implement an optimization oracle. That is, on a quantum computer, finding the optimal value is equivalent to finding an optimizer. Also, the same polarity argument shows that algorithms for optimization using separation are essentially equivalent to algorithms for separation using optimization. In particular, this turns our lower bound on the number of separation queries needed to implement an optimization oracle into a lower bound on the reverse direction.

Figure 1 gives an informal presentation of our results; the upper bounds arise from oracle reductions, the (change in) accuracy is ignored here for simplicity. The above-mentioned polarity manifests itself in the central symmetry of the figure.

Related independent work. In independent simultaneous work, Chakrabarti, Childs, Li, and Wu [CCLW18] discovered a similar upper bound as ours: combining the recent classical work of Lee et al. [LSV18] with a quantum algorithm for computing gradients, they show how to implement an optimization oracle via $\widetilde{\mathcal{O}}(n)$ quantum queries to a membership oracle and to an oracle for the objective function. Their proof stays quite close to [LSV18] while ours first simplifies some of the technical lemmas of [LSV18], giving us a slightly simpler presentation and a better error-dependence of the resulting algorithm. They also prove several lower bounds that are similar to the ones we prove here.

[3] We are not aware of an existing proof of this classical lower bound, but it may well be somewhere in the vast literature on convex optimization.

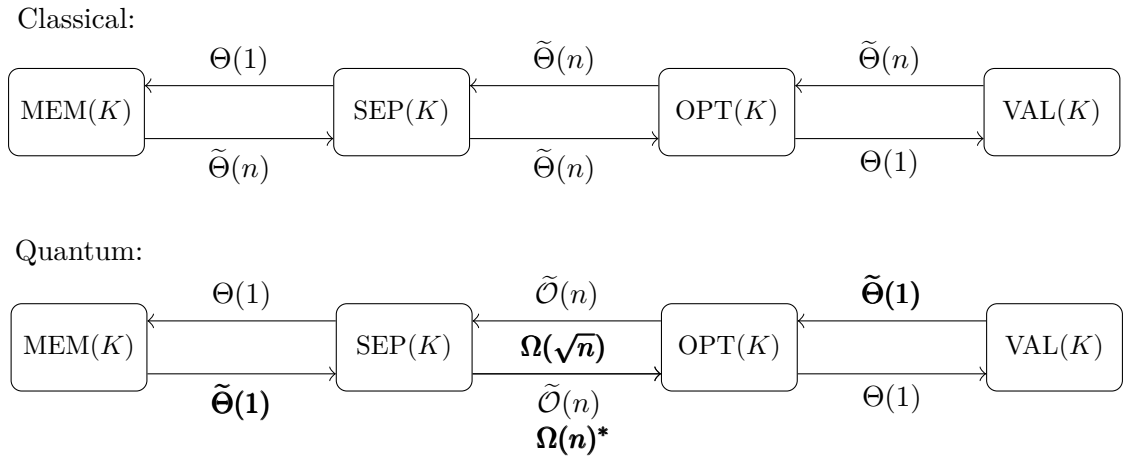

Figure 1: The top and bottom diagram illustrate the relations between the basic (weak) oracles for respectively classical and quantum queries, with boldface entries marking our new results. All upper and lower bounds hold in the setting where we know an interior point of K, except the $*$-marked $\Omega(n)$ lower bound on the number of separation queries needed for optimization. Notice the central symmetry of the diagrams, which is a consequence of polarity.

2 Preliminaries

We use $[n] := \{1, 2, \ldots, n\}$. For $p \geq 1$, $\varepsilon \geq 0$, and a set $C \subseteq \mathbb{R}^n$ we let

$$B_p(C, \varepsilon) = \{x \in \mathbb{R}^n : \exists y \in C \text{ such that } ||x - y||_p \leq \varepsilon\}$$

be the set of points of distance at most ε from C in the ℓ_p-norm. When $C = \{x\}$ is a singleton set we abuse notation and write $B_p(x, \varepsilon)$. We overload notation by setting

$$B_p(C, -\varepsilon) = \{x \in \mathbb{R}^n : B_p(x, \varepsilon) \subseteq C\}.$$

Whenever p is omitted it is assumed that $p = 2$.

Recall that a function $f : C \to \mathbb{R}$ is *Lipschitz* if there exists a constant $L > 0$ such that

$$|f(y') - f(y)| \leq L\|y' - y\|_2 \text{ for all } y, y' \in C.$$

We write that f is L-*Lipschitz*. The inner product between vectors $v, w \in \mathbb{R}^n$ is $\langle v, w\rangle = v^T w$.

Definition 1 (Subgradient)**.** *Let $C \subseteq \mathbb{R}^n$ be convex and let x be an element of the interior of C. For a convex function $f : C \to \mathbb{R}$ we denote by $\underline{\partial} f(x)$ the set of subgradients of f at x, i.e., those vectors g satisfying*

$$f(y) \geq f(x) + \langle g, y - x\rangle \text{ for all } y \in C.$$

Note that in the above definition $\underline{\partial} f(x) \neq \emptyset$ due to convexity.

If $f : C \to \mathbb{R}$ is L-Lipschitz, then for any x in the interior of C and any $g \in \underline{\partial} f(x)$ we have $\|g\| \leq L$, as follows. Consider a $y \in C$ such that $y - x = \alpha g$ for some $\alpha > 0$. Then since g is a subgradient of f at x we have

$$\alpha\|g\|^2 = \langle g, y - x\rangle \leq f(y) - f(x) \leq L\|y - x\| = \alpha L\|g\|, \tag{1}$$

and therefore $\|g\| \leq L$.

We will assume familiarity with quantum computing [NC00]. In particular, a standard quantum oracle corresponds to a unitary transformation that acts on two (finite-dimensional) registers, where the first register contains the query and the answer is added to the second register. For example, a function evaluation oracle for $f\colon X \to Y$ would map $|x, 0\rangle$ to $|x, f(x)\rangle$, where $|x\rangle$ and $|f(x)\rangle$ are basis states corresponding to binary representations of x and $f(x)$ respectively. Unlike classical algorithms, quantum computers can apply such an oracle to a *superposition* of different y's. They are also allowed to apply the inverse of a unitary oracle.

The standard quantum oracle described above models problems where there is a single correct answer to a query. When there are multiple good answers (for instance, different good approximations to the correct value) and the oracle is only required to give a correct answer with high probability, then we will work with the more liberal notion of *relational* quantum oracles.

Definition 2 (Relational quantum oracle)**.** *Let $\mathcal{F}\colon X \to \mathcal{P}(Y)$ be a function, such that for each $x \in X$ the subset $\mathcal{F}(x) \subseteq Y$ is the set of valid answers to an x query. A relational quantum oracle for $\mathcal{F}$ which answers queries with success probability $\geq 1-\rho$, is a unitary that for all $x \in X$ maps*

$$U\colon |x, 0, 0\rangle \mapsto \sum_{y \in Y} \alpha_{x,y} |x, y, \psi_{x,y}\rangle,$$

where $|\psi_{x,y}\rangle$ denotes some normalized quantum state and $\sum_{y\in\mathcal{F}(x)} |\alpha_{x,y}|^2 \geq 1-\rho$. Thus measuring the second register of $U|x, 0, 0\rangle$ gives a valid answer to the x query with probability at least $1-\rho$.

This definition is very natural for cases where the oracle is implemented by a quantum algorithm that produces a valid answer with probability $\geq 1-\rho$. In order to achieve our quantum speed-ups we will always assume access to the inverse $U^\dagger$ of the relational oracle as well, which is justified if U comes from an efficiently implementable quantum algorithm.

2.1 Oracles for convex sets

The five basic oracles for a convex set K that we consider are as follows (in contrast with the original [GLS88], we allow some error probability ρ in these oracles as in [LSV18]). Throughout we will assume that real vectors are represented with $\mathrm{polylog}(nR/(r\varepsilon))$ bits of precision per coordinate. In particular, we assume that the input / output of the following oracles is represented this way.[4]

Definition 3 (Membership oracle $\mathrm{MEM}_{\varepsilon,\rho}(K)$)**.** *Queried with a vector $y \in \mathbb{R}^n$, the oracle, with success probability $\geq 1-\rho$, correctly asserts one of the following*

- *$y \in B(K, \varepsilon)$, or*
- *$y \notin B(K, -\varepsilon)$.*

Definition 4 (Separation oracle $\mathrm{SEP}_{\varepsilon,\rho}(K)$)**.** *Queried with a vector $y \in \mathbb{R}^n$, the oracle, with success probability at least $\geq 1-\rho$, correctly asserts one of the following*

- *$y \in B(K, \varepsilon)$, or*

[4] Note that for weak oracles, where $\varepsilon > 0$, this is essentially without loss of generality, since the rounding errors can be incorporated into the error parameter of the oracle.

- $y \notin B(K, -\varepsilon)$,

and in the second case it returns a unit vector $g \in \mathbb{R}^n$ such that $\langle g, x\rangle \leq \langle g, y\rangle + \varepsilon$ for all $x \in B(K, -\varepsilon)$.

Definition 5 (Optimization oracle $\mathrm{OPT}_{\varepsilon,\rho}(K)$)**.** *Queried with a unit vector $c \in \mathbb{R}^n$, the oracle, with probability $\geq 1 - \rho$, does one of the following:*

- *it returns a vector $y \in \mathbb{R}^n$ such that $y \in B(K, \varepsilon)$ and $\langle c, x\rangle \leq \langle c, y\rangle + \varepsilon$ for all $x \in B(K, -\varepsilon)$,*
- *or it correctly asserts that $B(K, -\varepsilon)$ is empty.*

Note that the above optimization oracle corresponds to *maximizing* a linear function over a convex set; we could equally well state it for minimization.

Definition 6 (Violation oracle $\mathrm{VIOL}_{\varepsilon,\rho}(K)$)**.** *Queried with a unit vector $c \in \mathbb{R}^n$ and a real number γ, the oracle, with probability $\geq 1 - \rho$, does one of the following:*

- *it asserts that $\langle c, x\rangle \leq \gamma + \varepsilon$ for all $x \in B(K, -\varepsilon)$,*
- *or it finds a vector $y \in B(K, \varepsilon)$ such that $\langle c, y\rangle \geq \gamma - \varepsilon$.*

Definition 7 (Validity oracle $\mathrm{VAL}_{\varepsilon,\rho}(K)$)**.** *Queried with a unit vector $c \in \mathbb{R}^n$ and a real number γ, the oracle, with probability $\geq 1 - \rho$, does one of the following:*

- *it asserts that $\langle c, x\rangle \leq \gamma + \varepsilon$ for all $x \in B(K, -\varepsilon)$,*
- *or it asserts that $\langle c, y\rangle \geq \gamma - \varepsilon$ for some $y \in B(K, \varepsilon)$.*

If in the above definitions both ε and ρ are equal to 0, then we call the oracle *strong*. If either is non-zero then we sometimes call it *weak*.

The above describes the classical oracles, and the quantum oracles are defined analogously, i.e., they are relational quantum oracles (see Definition 2), that use a binary representation for the input / output vectors.

When we discuss membership queries, we will always assume that we are given a small ball which lies inside the convex set. It is easy to see that without such a small ball one cannot obtain an optimization oracle using only $\mathrm{poly}(n)$ classical queries to a membership oracle (see, e.g., [GLS88, Sec. 4.1] or the example below). As the following example shows, the same holds for quantum queries. We will use a reduction from a version of the well-studied *search* problem:

Given $z \in \{0,1\}^N$ such that $|z| = 1$, find $b \in [N]$ such that $z_b = 1$.

It is not hard to see that if the access to z is given via classical queries $i \mapsto z_i$, then $\Omega(N)$ queries are needed. It is well known [BBBV97] that if we allow quantum queries, i.e., applications of the unitary $|i\rangle|b\rangle \mapsto |i\rangle|z_i \oplus b\rangle$, then $\Omega(\sqrt{N})$ queries are needed. Now let $N = 2^n$ and consider an input $z \in \{0,1\}^N$ to the search problem. Let $b \in \{0,1\}^n$ be the index such that $z_b = 1$. Consider maximizing the linear function $\langle e, z\rangle$ (where e is the all-1 vector) over the set $K_z = \prod_{i=1}^n [b_i - 1/2, b_i]$. Clearly the optimal solution to this convex optimization problem, even with a small constant additive error in the answer, gives the solution to the search problem. However, a membership query is essentially equivalent to querying a bit of z and therefore $\Omega(\sqrt{N}) = \Omega(2^{n/2})$ quantum queries to the membership oracle are needed for optimization.

3 Computing approximate subgradients of convex Lipschitz functions

Here we show how to compute an approximate subgradient (at 0) of a convex Lipschitz function. That is, given a convex set C such that $0 \in \mathrm{int}(C)$ and a convex function $f : C \to \mathbb{R}$, we show how to compute a vector $\tilde{g} \in \mathbb{R}^n$ such that $f(y) \geq f(0) + \langle \tilde{g}, y \rangle - a\|y\| - b$ for some real numbers $a, b > 0$ that will be defined later (see Lemma 12 and Lemma 18). The idea of the classical algorithm given in the next section is to pick a point $z \in B_\infty(0, r_1)$ uniformly at random and use the finite difference $\nabla^{(r_2)} f(z)$ (defined below) as an approximate subgradient of f at 0; the radii r_1 and r_2 need to be chosen small to make the approximation good. This results in a slightly simplified version of the algorithm of Lee et al. [LSV18]. In Section 3.2 we show how to improve on this classical algorithm on a quantum computer.

3.1 Classical approach

In the discussion that follows we will use the following approximation of the gradient.

Definition 8 (Finite-difference gradient approximation)**.** *For a function $f : C \to \mathbb{R}$, a real $r > 0$, and a point $x \in \mathbb{R}^n$ such that $B_1(x, r) \subseteq C$, and $i \in [n]$, we define*

$$\nabla_i^{(r)} f(x) := \frac{f(x + re_i) - f(x - re_i)}{2r},$$

where $e_i \in \{0, 1\}^n$ is the vector that has a 1 only in its ith coordinate. Similarly we define

$$\nabla^{(r)} f(x) := \left(\nabla_1^{(r)} f(x), \nabla_2^{(r)} f(x), \ldots, \nabla_n^{(r)} f(x)\right).$$

We will also consider a similar approximation of the Laplacian (the trace of the Hessian) of a function.

Definition 9 (Finite-difference Laplace approximation)**.** *For a function $f : C \to \mathbb{R}$, a real $r > 0$, and a point $x \in \mathbb{R}^n$ such that $B_1(x, r) \subseteq C$, and $i \in [n]$, we define*

$$\Delta_i^{(r)} f(x) := \frac{f(x + re_i) - 2f(x) + f(x - re_i)}{r^2}.$$

Similarly

$$\Delta^{(r)} f(x) := \sum_{i=1}^n \Delta_i^{(r)} f(x).$$

Note that for a convex function we have $\Delta_i^{(r)} f(x) \geq 0$ for all x such that $B_1(x, r) \subseteq C$.

The next two lemmas will be needed in the proof of the main result of this section, Lemma 12. In Lemma 10 we give an upper bound on the deviation $\left\|g - \nabla^{(r_2)} f(z)\right\|_1$ of a finite difference gradient approximation $\nabla^{(r_2)} f(z)$ from an actual subgradient g at the point z, in terms of the finite difference Laplace approximation $\Delta^{(r_2)} f(z)$. Then, in Lemma 11 we show that in expectation (over the points of a small ball around x), the finite difference Laplace approximation is small. Together with Markov's inequality this gives us good control over the quality of a finite difference gradient approximation.

Lemma 10. *If $r_2 > 0$, $z \in \mathbb{R}^n$, and $f : B_1(z, r_2) \to \mathbb{R}$ is convex, then*

$$\sup_{g \in \partial f(z)} \left\|g - \nabla^{(r_2)} f(z)\right\|_1 \leq \frac{r_2 \Delta^{(r_2)} f(z)}{2}.$$

Proof. Fix a $g \in \underline{\partial} f(z)$. For every $i \in [n]$, we have $f(z + r_2 e_i) \geq f(z) + \langle g, r_2 e_i \rangle = f(z) + r_2 g_i$, and, similarly, $f(z - r_2 e_i) \geq f(z) - r_2 g_i$. Rearranging gives

$$\underbrace{\frac{f(z) - f(z - r_2 e_i)}{r_2}}_{:=A} \leq g_i \leq \underbrace{\frac{f(z + r_2 e_i) - f(z)}{r_2}}_{:=B}.$$

Note that $|g_i - \frac{A+B}{2}| \leq \frac{B-A}{2}$ for any three real numbers $A \leq g_i \leq B$. Moreover, $\frac{A+B}{2} = \nabla_i^{(r_2)} f(z)$ and $B - A = r_2 \Delta_i^{(r_2)} f(z)$, thus $\left| g_i - \nabla_i^{(r_2)} f(z) \right| \leq \frac{r_2 \Delta_i^{(r_2)} f(z)}{2}$. Now we can finish the proof by summing this inequality over all $i \in [n]$. □

Lemma 11. *If $0 < r_2 \leq r_1$, and $f : B_\infty(x, r_1 + r_2) \to \mathbb{R}$ is convex and L-Lipschitz, then*

$$\mathop{\mathbb{E}}_{z \in B_\infty(x, r_1)} \Delta^{(r_2)} f(z) \leq \frac{nL}{r_1}.$$

Proof. Below we show that $\mathop{\mathbb{E}}_{z \in B_\infty(x, r_1)} \Delta_i^{(r_2)} f(z) \leq \frac{L}{r_1}$ for all $i \in [n]$, summing over i then proves the lemma.

Let $h_i(z) := f(z - r_2 e_i) - f(z)$; we have that

$$\begin{aligned}
\mathop{\mathbb{E}}_{z \in B_\infty(x, r_1)} \Delta_i^{(r_2)} f(z) &= \frac{1}{(2r_1)^n} \int_{z \in B_\infty(x, r_1)} \frac{f(z + r_2 e_i) - 2f(z) + f(z - r_2 e_i)}{r_2^2} \, dz \\
&= \frac{1}{(2r_1)^n} \int_{\substack{z_j \in [x_j - r_1, x_j + r_1], \\ j \in [n], j \neq i}} \int_{z_i \in [x_i - r_1, x_i + r_1]} \frac{f(z - r_2 e_i) - 2f(z) + f(z + r_2 e_i)}{r_2^2} \, dz \\
&= \frac{1}{(2r_1)^n} \int_{\substack{z_j \in [x_j - r_1, x_j + r_1], \\ j \in [n], j \neq i}} \Big(\int_{z_i \in [x_i - r_1, x_i + r_1]} \frac{h_i(z)}{r_2^2} \, dz \\
&\qquad - \int_{z_i \in [x_i - r_1, x_i + r_1]} \frac{h_i(z + r_2 e_i)}{r_2^2} \, dz \Big) \\
&= \frac{1}{(2r_1)^n} \int_{\substack{z_j \in [x_j - r_1, x_j + r_1], \\ j \in [n], j \neq i}} \Big(\int_{z_i \in [x_i - r_1, x_i + r_1]} \frac{h_i(z)}{r_2^2} \, dz \\
&\qquad - \int_{z_i \in [x_i - r_1 + r_2, x_i + r_1 + r_2]} \frac{h_i(z)}{r_2^2} \, dz \Big) \\
&= \frac{1}{(2r_1)^n} \int_{\substack{z_j \in [x_j - r_1, x_j + r_1], \\ j \in [n], j \neq i}} \Big(\int_{z_i \in [x_i - r_1, x_i - r_1 + r_2]} \frac{h_i(z)}{r_2^2} \, dz \\
&\qquad - \int_{z_i \in [x_i + r_1, x_i + r_1 + r_2]} \frac{h_i(z)}{r_2^2} \, dz \Big) \\
&\leq \frac{1}{(2r_1)^n} \int_{\substack{z_j \in [x_j - r_1, x_j + r_1], \\ j \in [n], j \neq i}} 2L \, dz \\
&= \frac{L}{r_1}.
\end{aligned}$$

The last inequality above follows from multiplying the upper bound $r_2 L$ on $|h_i|$ with the length r_2 of the integration intervals. □

Note that the above lemma is stated and proved for continuous random variables, but the same proof holds if we have a uniform hypergrid over the same hypercube, providing a discrete version of the above result. In the discrete case, in order to get the same

cancellations we need to assume that both r_1 and r_2 are integer multiples of the grid spacing.

We are now ready to prove the main result of this section. Informally, the next lemma proves that an approximate subgradient of a convex Lipschitz function f at 0 can be obtained by an algorithm that outputs $\nabla^{(r_2)}\tilde{f}(z)$ for a random z close enough to 0, where $\tilde{f}$ is an approximate version of f. In other words, this lemma gives us a classical algorithm to compute an approximate subgradient of f using $2n$ classical queries to an approximate version of f.

Lemma 12. *Let $r_1 > 0$, $L > 0$, $\rho \in (0, 1/3]$, $\delta \in (0, r_1\sqrt{n}L/\rho]$, then $r_2 := \sqrt{\frac{\delta r_1 \rho}{\sqrt{n}L}} \leq r_1$. Suppose $f : C \to \mathbb{R}$ is a convex function that is L-Lipschitz on $B_\infty(0, 2r_1)$, and $\tilde{f} : B_\infty(0, 2r_1) \to \mathbb{R}$ is such that $\left\|\tilde{f} - f\right\|_\infty \leq \delta$. Then for a uniformly random $z \in B_\infty(0, r_1)$, with probability at least $1 - \rho$,*

$$f(y) \geq f(0) + \left\langle \nabla^{(r_2)}\tilde{f}(z), y \right\rangle - \frac{3n^{\frac{3}{4}}}{2}\sqrt{\frac{\delta L}{\rho r_1}}\|y\| - 2L\sqrt{n}r_1 \qquad \text{for all } y \in C.$$

Proof. Let $z \in B_\infty(0, r_1)$ and $g \in \underline{\partial} f(z)$. Recall $\|g\| \leq L$ by Equation (1). Then for all $y \in C$

$$\begin{aligned}
f(y) &\geq f(z) + \langle g, y - z\rangle \\
&= f(z) + \langle g, y - z\rangle + \left(\left\langle \nabla^{(r_2)}f(z), y\right\rangle - \left\langle \nabla^{(r_2)}f(z), y\right\rangle\right) + (f(0) - f(0)) \\
&= f(0) + \left\langle \nabla^{(r_2)}f(z), y\right\rangle + \langle g - \nabla^{(r_2)}f(z), y\rangle + (f(z) - f(0)) + \langle g, -z\rangle \\
&\geq f(0) + \left\langle \nabla^{(r_2)}f(z), y\right\rangle - \left\|g - \nabla^{(r_2)}f(z)\right\|_1\|y\|_\infty - L\|z\| - \|g\|\|z\| \\
&\geq f(0) + \left\langle \nabla^{(r_2)}f(z), y\right\rangle - \left\|g - \nabla^{(r_2)}f(z)\right\|_1\|y\|_\infty - L\sqrt{n}r_1 - L\sqrt{n}r_1 \\
&\geq f(0) + \left\langle \nabla^{(r_2)}\tilde{f}(z), y\right\rangle - \frac{\delta\sqrt{n}}{r_2}\|y\| - \left\|g - \nabla^{(r_2)}f(z)\right\|_1\|y\|_\infty - 2L\sqrt{n}r_1.
\end{aligned}$$

Note that in the last line we switched from f to $\tilde{f}$, using that $\nabla^{(r_2)}f(z)$ and $\nabla^{(r_2)}\tilde{f}(z)$ differ by at most δ/r_2 in each coordinate. Our choice of r_2 gives $\frac{\delta\sqrt{n}}{r_2} = n^{\frac{3}{4}}\sqrt{\frac{\delta L}{\rho r_1}}$ and by Lemma 10–11 we have

$$\mathop{\mathbb{E}}_{z \in B_\infty(x, r_1)}\left\|g - \nabla^{(r_2)}f(z)\right\|_1 \leq \frac{nLr_2}{2r_1} = \frac{n^{\frac{3}{4}}}{2}\sqrt{\frac{\delta L\rho}{r_1}}.$$

By Markov's inequality we get that $\left\|g - \nabla^{(r_2)}f(z)\right\|_1 \leq \frac{n^{\frac{3}{4}}}{2}\sqrt{\frac{\delta L}{\rho r_1}}$ with probability $\geq 1 - \rho$ over the choice of z. Plugging this bound on $\left\|g - \nabla^{(r_2)}f(z)\right\|_1$ into the above lower bound on $f(y)$ concludes the proof of the lemma. □

3.2 Quantum improvements

In this section we show how to improve subgradient computation of convex functions via Jordan's quantum algorithm for gradient computation [Jor05]. We use the formulation given by Gilyén et al. [GAW19, Lemma 20], for which we first introduce the following definition.

Definition 13 (Hyper-grid)**.** *For $k \in \mathbb{N}$ we define the following discretization of the interval $(-1/2, 1/2)$:*

$$G_k := \left\{ \frac{j}{2^k} - \frac{1}{2} + 2^{-k-1} : j \in \{0, \ldots, 2^k - 1\} \right\} \subset (-1/2, 1/2).$$

Similarly we define the n-dimensional hyper-grid G_k^n, which is the n-fold Cartesian product of G_k with itself.

Note that an element of G_k^n can be represented using $n \times k$ (qu)bits. Basically, Jordan's algorithm just sets up a uniform superposition over all grid points, applies a "phase query" to f, and then a quantum Fourier transform over each coordinate.

Lemma 14. (Jordan's quantum gradient computation algorithm [GAW19, Lemma 20]) *Let $m \in \mathbb{N}$, $c \in \mathbb{R}$ and $g \in \mathbb{R}^n$ such that $\|g\|_\infty \leq 1/3$. If $h : G_m^n \to \mathbb{R}$ is such that*

$$|h(x) - \langle g, x \rangle - c| \leq \frac{2^{-m}}{42\pi}, \tag{2}$$

for 99.9% of the points $x \in G_m^n$, then using a single query to a phase oracle $\mathrm{O}\colon |x\rangle \mapsto e^{2\pi i 2^m h(x)}|x\rangle$ Jordan's gradient computation algorithm outputs a vector $v \in \mathbb{R}^n$ such that:

$$\Pr\left[|v_i - g_i| > 2^{2-m}\right] \leq 1/3 \quad \text{for every } i \in [n].$$

We now show that the above algorithm allows us to compute an approximate subgradient of a function f, even if we are only given standard oracle access to a function $\tilde{f}$ which is sufficiently close to f. In particular, we will assume we are given access to a standard unitary oracle of a function $\tilde{f}\colon G_m^n \to \mathbb{R}$ which satisfies $|\tilde{f}(x) - f(x)| \leq \delta$ for all $x \in G_m^n$. That is, we assume we are given access to a unitary U acting as

$$U : |x\rangle|0\rangle \mapsto |x\rangle|\tilde{f}(x)\rangle \tag{3}$$

Note that if we can classically efficiently evaluate $\tilde{f}$, then it is well known that we can construct such a unitary as a small quantum circuit (see [NC00, Sec. 1.4.1]).

The main idea is that, using one application of U, a phase gate corresponding to the output register, and another application of $U^\dagger$ to uncompute the function value, we can implement a phase oracle for $\tilde{f}$. Moreover, Equation (4) below will also hold for $\tilde{f}$, with a slightly worse right-hand side, since f is close to $\tilde{f}$. A version of the following is proven in [GAW19, Theorem 21], for completeness we sketch a proof.

Corollary 15 (Gradient computation using approximate function evaluation)**.** *Let $\delta, B, r \in \mathbb{R}_+, c \in \mathbb{R}$, $\rho \in (0, 1/3]$. Let $x_0, g \in \mathbb{R}^n$ with $\|g\|_\infty \leq \frac{B}{r}$. Let $m := \left\lceil \log_2\left(\frac{B}{28\pi\delta}\right) \right\rceil$ and suppose $f : (x_0 + rG_m^n) \to \mathbb{R}$ is such that*

$$|f(x_0 + rx) - \langle g, rx \rangle - c| \leq \delta \tag{4}$$

for 99.9% of the points $x \in G_m^n$, and we have access to a standard unitary oracle U, providing $\mathcal{O}\left(\log\left(\frac{B}{\delta}\right)\right)$-bit fixed-point binary approximations $\tilde{f}(z)$ s.t. $|\tilde{f}(z) - f(z)| \leq \delta$ for all $z \in (x_0 + rG_m^n)$. Then we can compute a vector $\tilde{g} \in \mathbb{R}^n$ such that

$$\Pr\left[\|\tilde{g} - g\|_\infty > \frac{8 \cdot 42\pi\delta}{r} \right] \leq \rho,$$

with $\mathcal{O}\left(\log\left(\frac{n}{\rho}\right)\right)$ queries to U and $U^\dagger$ and gate complexity $\mathcal{O}\left(n \log\left(\frac{n}{\rho}\right) \log\left(\frac{B}{\delta}\right) \mathrm{loglog}\left(\frac{n}{\rho}\right) \mathrm{loglog}\left(\frac{B}{\delta}\right)\right)$.

Proof. As described above the corollary, we first implement a phase oracle for $\tilde{f}$ and then we apply Jordan's gradient computation algorithm (Lemma 14).

With a single query to U and its inverse we can implement a phase oracle O that acts as $\mathrm{O} : |x\rangle \mapsto e^{2\pi i \frac{M}{3B}\tilde{f}(x_0+rx)}|x\rangle$, where $M := \frac{3B}{84\pi\delta}$, and[5] $m := \log_2(M)$. Let $h(x) := \frac{\tilde{f}(x_0+rx)}{3B}$, then by (4) 99.9% of the points $x \in G_m^n$ satisfy $\left|h(x) - \left\langle \frac{r}{3B}g, x\right\rangle - \frac{c}{3B}\right| \leq \frac{2\delta}{3B} = \frac{1}{42\pi M}$. Since $\left\|\frac{r}{3B}g\right\|_\infty \leq \frac{1}{3}$, by Lemma 14 we can compute a vector $v \in \mathbb{R}^n$ which is a coordinatewise $\frac{4}{M}$-approximator of $\frac{r}{3B}g$: for each $i \in [n]$ we have $\left|g_i - \frac{3B}{r}v_i\right| \leq \frac{12B}{rM} = \frac{8\cdot 42\pi\delta}{r}$ with probability at least $\frac{2}{3}$.

Note that the above success probability is per coordinate of g. However, repeating the whole procedure $\mathcal{O}(\log(\frac{n}{\rho}))$ times and taking the median of the resulting vectors coordinatewise gives a gradient approximator $\tilde{g}$ with the desired approximation quality with probability at least $1-\rho$. For the proof of the gate complexity we refer[6] to [GAW19, Theorem 21] where the complexity of Jordan's algorithm is analyzed in detail. □

Remark. With essentially the same approach, the above corollary of Jordan's quantum gradient computation algorithm can also be proven in the setting where our access to an approximation of f is not given by a standard quantum oracle but by a *relational* quantum oracle, see Appendix A for both the definition of this type of approximation to f and a proof of this corollary.

In terms of applications, we want to point out that if the membership oracle used in Section 4 comes from a deterministic algorithm, then we get a standard quantum oracle. Only when the membership oracle itself is relational (for example, when it is itself computed by a bounded-error quantum algorithm) do we need the more general setting of Appendix A.

We would like to apply the above corollary to compute gradients of a convex Lipschitz function. To that end, the function needs to be sufficiently close to a linear function on a small region. Fortunately convex Lipschitz functions have this property. The following two lemmas ensure that Equation (4) holds.

Lemma 16. *Let $S \subseteq \mathbb{R}^n$ be such that $S = -S$, and let $\mathrm{conv}(S)$ denote the convex hull of S. If $f : \mathrm{conv}(S) \to \mathbb{R}$ is a convex function, $f(0) = 0$, and $|f(s)| \leq \delta$ for all $s \in S$, then*

$$|f(s')| \leq \delta \text{ for all } s' \in \mathrm{conv}(S).$$

Proof. Since f is convex and $f(s) \leq \delta$ for all $s \in S$ we immediately get that $f(s') \leq \delta$ for all $s' \in \mathrm{conv}(S)$. Because $f(0) = 0$ and $S = -S$, due to convexity we get that $f(s') \geq -f(-s') \geq -\delta$. □

Lemma 17. *If $r_2 > 0$, $z \in \mathbb{R}^n$ and $f : B_1(z, r_2) \to \mathbb{R}$ is convex, then*

$$\sup_{y \in B_1(0,r_2)} \left| f(z+y) - f(z) - \left\langle y, \nabla^{(r_2)} f(z)\right\rangle \right| \leq \frac{r_2^2 \Delta^{(r_2)} f(z)}{2}.$$

Proof. Let $d(y) := f(z+y) - f(z) - \left\langle y, \nabla^{(r_2)} f(z)\right\rangle$ be the difference between $f(z+y)$ and its linear approximator. Let $S := \{\pm r_2 e_i : i \in [n]\}$. It is easy to see that $d(0) = 0$,

[5] We can assume without loss of generality that the upper bound B is such that M is a power of two.

[6] The correspondence with the parametrization of [GAW19, Theorem 21] is $\varepsilon \leftrightarrow \frac{8\cdot 42\pi\delta}{r}$, $M \leftrightarrow \frac{B}{r}$.

$S = -S$, and $\mathrm{conv}(S) = B_1(0, r_2)$. Also, for all $s \in S$ we have $|d(s)| \leq r_2^2 \Delta^{(r_2)} f(z)/2$:

$$\begin{aligned} d(\pm r_2 e_i) &= f(z \pm r_2 e_i) - f(z) - \left\langle \pm r_2 e_i, \nabla^{(r_2)} f(z) \right\rangle \\ &= f(z \pm r_2 e_i) - f(z) \mp r_2 \nabla_i^{(r_2)} f(z) \\ &= f(z \pm r_2 e_i) - f(z) \mp \frac{f(z + r_2 e_i) - f(z - r_2 e_i)}{2} \\ &= \frac{f(z + r_2 e_i) - 2f(z) + f(z - r_2 e_i)}{2} \\ &= r_2^2 \Delta_i^{(r_2)} f(z)/2 \quad \leq r_2^2 \Delta^{(r_2)} f(z)/2. \end{aligned}$$

Therefore Lemma 16 implies that $\sup_{y \in B_1(0,r_2)} |d(y)| \leq r_2^2 \Delta^{(r_2)} f(z)/2$. □

We can now state the main result of this section, the quantum analogue of Lemma 12.

Lemma 18. *Let $r_1 > 0$, $L > 0$, $\rho \in (0, 1/3]$, and suppose $\delta \in (0, r_1 n L/\rho]$. Suppose $f : C \to \mathbb{R}$ is a convex function that is L-Lipschitz on $B_\infty(0, 2r_1)$, and we have quantum query access[7] to $\tilde{f}$, which is a δ-approximate version of f, via a unitary U over a (fine-enough) hypergrid of $B_\infty(0, 2r_1)$. Then we can compute a $\tilde{g} \in \mathbb{R}^n$ using $\mathcal{O}(\log(n/\rho))$ queries to U and $U^\dagger$, such that with probability $\geq 1 - \rho$, we have*

$$f(y) \geq f(0) + \langle \tilde{g}, y \rangle - 23^2 \sqrt{\frac{\delta n^3 L}{\rho r_1}} \|y\|_1 - 2L\sqrt{n} r_1 \qquad \text{for all } y \in C$$

and hence (by Cauchy-Schwarz)

$$f(y) \geq f(0) + \langle \tilde{g}, y \rangle - (23n)^2 \sqrt{\frac{\delta L}{\rho r_1}} \|y\| - 2L\sqrt{n} r_1 \qquad \text{for all } y \in C.$$

Proof. Let $r_2 := \sqrt{\frac{\delta r_1 \rho}{nL}}$ and note that $r_2 \leq r_1$. The quantum algorithm works roughly as follows. It first picks a uniformly[8] random $z \in B_\infty(0, r_1)$. Then it uses Jordan's quantum algorithm to compute an approximate gradient at z by approximately evaluating f in superposition over a discrete hypergrid of $B_\infty(z, r_2/n)$. This then yields an approximate subgradient of f at 0.

We now work out this rough idea. Since $B_\infty(z, r_2/n) \subseteq B_1(z, r_2)$, Lemma 17 implies

$$\sup_{y \in B_\infty(0, r_2/n)} \left| f(z + y) - f(z) - \left\langle y, \nabla^{(r_2)} f(z) \right\rangle \right| \leq \frac{r_2^2 \Delta^{(r_2)} f(z)}{2}. \tag{5}$$

Also as shown by Lemma 11 and Markov's inequality we have

$$\Delta^{(r_2)} f(z) \leq \frac{2nL}{\rho r_1} \tag{6}$$

with probability $\geq 1 - \rho/2$ over the choice of z. If z is such that Equation (6) holds, then we get

$$\sup_{y \in B_\infty(0, r_2/n)} \left| f(z + y) - f(z) - \left\langle y, \nabla^{(r_2)} f(z) \right\rangle \right| \leq \frac{nL r_2^2}{\rho r_1} = \delta.$$

[7] Using Corollary 29 instead of Corollary 15 shows that a relational quantum oracle also suffices as input.

[8] A discrete quantum computer strictly speaking cannot do this, but (as noted after Lemma 11) a uniformly random point from a fine-enough hypergrid suffices.

Now apply the quantum algorithm of Corollary 15 with $r = 2r_2/n$, $c = f(z)$, $g = \nabla^{(r_2)} f(z)$, and $B = Lr$. This uses $\mathcal{O}(\log(n/\rho))$ queries to U and $U^\dagger$, and with probability $\geq 1 - \rho/2$ computes an approximate gradient $\tilde{g}$ such that

$$\left\|\nabla^{(r_2)} f(z) - \tilde{g}\right\|_\infty \leq \frac{8 \cdot 42\pi n}{2r_2} \cdot \delta = 4 \cdot 42 \cdot \pi \sqrt{\frac{\delta n^3 L}{\rho r_1}}. \tag{7}$$

Also, if z is such that Equation (6) holds, then by Lemma 10 we get that

$$\sup_{g \in \underline{\partial} f(z)} \left\|\nabla^{(r_2)} f(z) - g\right\|_1 \leq \frac{r_2 \Delta^{(r_2)} f(z)}{2} \leq \frac{nLr_2}{\rho r_1} = \sqrt{\frac{\delta nL}{\rho r_1}},$$

and therefore by the triangle inequality and Equation (7) we get that

$$\begin{aligned}
\sup_{g \in \underline{\partial} f(z)} \|g - \tilde{g}\|_\infty &\leq \sup_{g \in \underline{\partial} f(z)} \left\|g - \nabla^{(r_2)} f(z)\right\|_\infty + \left\|\nabla^{(r_2)} f(z) - \tilde{g}\right\|_\infty \\
&\leq \sup_{g \in \underline{\partial} f(z)} \left\|g - \nabla^{(r_2)} f(z)\right\|_1 + \left\|\nabla^{(r_2)} f(z) - \tilde{g}\right\|_\infty \\
&\leq \sqrt{\frac{\delta nL}{\rho r_1}} + 4 \cdot 42 \cdot \pi \sqrt{\frac{\delta n^3 L}{\rho r_1}} \quad < 23^2 \sqrt{\frac{\delta n^3 L}{\rho r_1}}.
\end{aligned}$$

Thus with probability at least $1 - \rho$, for all $y \in C$ and for all $g \in \underline{\partial} f(z)$ we have that

$$\begin{aligned}
f(y) &\geq f(z) + \langle g, y - z \rangle \\
&= f(0) + \langle \tilde{g}, y \rangle + \langle g - \tilde{g}, y \rangle + (f(z) - f(0)) + \langle g, -z \rangle \\
&\geq f(0) + \langle \tilde{g}, y \rangle - |\langle g - \tilde{g}, y \rangle| - L\|z\| - \|g\|\|z\| \\
&\geq f(0) + \langle \tilde{g}, y \rangle - \|g - \tilde{g}\|_\infty \|y\|_1 - L\sqrt{n} r_1 - L\sqrt{n} r_1 \qquad \text{(by (1))} \\
&\geq f(0) + \langle \tilde{g}, y \rangle - 23^2 \sqrt{\frac{\delta n^3 L}{\rho r_1}} \|y\|_1 - 2L\sqrt{n} r_1 \\
&\geq f(0) + \langle \tilde{g}, y \rangle - (23n)^2 \sqrt{\frac{\delta L}{\rho r_1}} \|y\| - 2L\sqrt{n} r_1. \qquad \square
\end{aligned}$$

4 Algorithms for separation using membership queries

Let $K \subseteq \mathbb{R}^n$ be a convex set such that $B(0, r) \subseteq K \subseteq B(0, R)$. Given a membership oracle[9] $\mathrm{MEM}_{\varepsilon,0}(K)$ as in Definition 3, we will construct a separation oracle $\mathrm{SEP}_{\eta,\rho}(K)$ as in Definition 4. Let x be the point we want to separate from K. We first make a membership query to x itself, receiving answer $x \in B(K, \varepsilon)$ or $x \notin B(K, -\varepsilon)$. Suppose $x \notin B(K, -\varepsilon)$, then we need to find a hyperplane that approximately separates x from K.

[9]For simplicity we assume throughout this section that the membership oracle succeeds with certainty (i.e., its error probability is 0). This is easy to justify: suppose we have a classical T-query algorithm, which uses $\mathrm{MEM}_{\varepsilon,0}(K)$ queries and succeeds with probability at least $1-\rho$. If we are given access to a $\mathrm{MEM}_{\varepsilon,\frac{1}{3}}(K)$ oracle instead, then we can create a $\mathrm{MEM}_{\varepsilon,\frac{\rho}{T}}(K)$ oracle by $\mathcal{O}(\log(T/\rho))$ queries to $\mathrm{MEM}_{\varepsilon,\frac{1}{3}}(K)$ and taking the majority of the answers. Then running the original algorithm with $\mathrm{MEM}_{\varepsilon,\frac{\rho}{T}}(K)$ will fail with probability at most 2ρ. Therefore the assumption of a membership oracle with error probability 0 can be removed at the expense of only a small logarithmic overhead in the number of queries. A similar argument works for the quantum case.

Due to the rotational symmetry of the separation problem, for ease of notation we assume that $x = -\|x\|e_n$.[10] We define $h : \mathbb{R}^{n-1} \to \mathbb{R} \cup \{\infty\}$ as

$$h(y) := \inf_{(y,y_n)\in K} y_n,$$

see also Figure 2. Note that h implicitly depends on x, since we have rotated the space such that $x = -\|x\|e_n$.

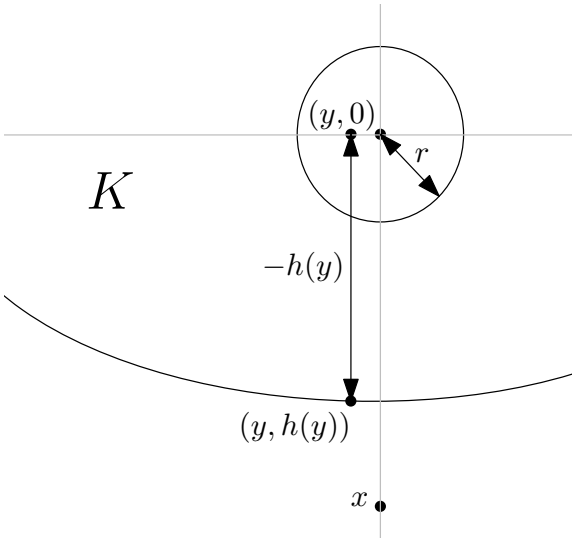

Figure 2: Graphical example of the relation between $h(y)$ and the distance from $(y, 0)$ to the border in the $-e_n$ direction.

Our h is a bit different from the one used in [LSV18], but we can show that it has many of the same properties. Since K is a convex set, h is a convex function over $\mathbb{R}^{n-1}$. As we show below, the function h is also Lipschitz (Lemma 19) and we can approximately compute its value using binary search with $\widetilde{\mathcal{O}}(1)$ classical queries to a membership oracle (Lemma 20). Furthermore, an approximate subgradient of h at 0 allows to construct a hyperplane approximately separating x from K (Lemma 21). Combined with the results of Section 3 this leads to the main results of this section, Theorems 22 and 23, which show how to efficiently construct a separation oracle using respectively classical and quantum queries to a membership oracle.

Analogously to [LSV18, Lemma 12] we first show that our h is Lipschitz.

Lemma 19. *For every $\delta \in (0, r)$, h is $\frac{R}{r-\delta}$-Lipschitz on $B(0, \delta) \subseteq \mathbb{R}^{n-1}$, that is, we have*

$$|h(y') - h(y)| \leq \frac{R}{r-\delta}\|y' - y\| \quad \textit{for all } y, y' \in B(0, \delta).$$

Proof. Observe that for all $y \in B(0, r)$ we have $-R \leq h(y) \leq 0$, because $B(0, r) \subseteq K \subseteq B(0, R)$. Let $y, y' \in B(0, \delta)$ be arbitrary but distinct points. Due to symmetry it will suffice to show that $h(y') - h(y) \leq \frac{R}{r-\delta}\|y' - y\|$.

[10] For the query complexity this is without loss of generality, since we can always apply a rotation to all the points such that this holds. If we instead consider the computational cost of our algorithm, then we have to take into account the cost of this rotation and its inverse. Note, however, that this rotation can always be written as the product of n rotations on only 2 coordinates, and hence can be applied in $\widetilde{\mathcal{O}}(n)$ additional steps. These rotations can also be found in $\widetilde{\mathcal{O}}(n)$ time via a greedy algorithm: first find a rotation on coordinates n and $n-1$ that leaves coordinate $n-1$ zero, then similarly for coordinates $n-2$ and n, and so on.

We will restrict our attention to the line through y and y', i.e., the line given by $y+\lambda z$ for $z := \frac{y'-y}{\|y'-y\|}$. Define the point

$$p := y + (\|y'-y\| + (r-\delta))z = y' + (r-\delta)z$$

on this line and note that $p \in B(0,r)$. Since y' lies between y and p on the line it is a convex combination of these two points. In particular, since $\|p-y'\| = r-\delta$, it is the convex combination

$$y' = \frac{\|y'-y\|}{\|y'-y\|+(r-\delta)}p + \frac{r-\delta}{\|y'-y\|+(r-\delta)}y.$$

Due to convexity we have

$$h(y') \le \frac{\|y'-y\|}{\|y'-y\|+(r-\delta)}h(p) + \frac{r-\delta}{\|y'-y\|+(r-\delta)}h(y),$$

which implies

$$h(y') - h(y) \le \frac{\|y'-y\|}{\|y'-y\|+(r-\delta)}(h(p)-h(y)) \le \frac{\|y'-y\|}{r-\delta}R. \qquad \square$$

Now we show how to compute the value of h using membership queries to K.

Lemma 20. *For all $y \in B(0,\frac{r}{2}) \subset \mathbb{R}^{n-1}$ we can compute a δ-approximation of $h(y)$ with $\mathcal{O}\left(\log\left(\frac{R}{\delta}\right)\right)$ queries to a $\mathrm{MEM}_{\varepsilon,0}(K)$ oracle, where $\varepsilon \le \frac{r}{3R}\delta$.*

Proof. Let $y \in B(0,\frac{r}{2})$, then $(y,h(y))$ is a boundary point of K by the definition of h. Note that $h(y) \in [-R,-r/2]$. Our goal is to perform binary search over this interval to find a good approximation of $h(y)$. If we had access to a perfect membership oracle, then this would be straightforward. However, since our membership oracle can give back a wrong answer when queried with a point that is ε-close to the boundary of K, a more careful analysis is needed.

Suppose $y_n \le -\frac{r}{2}$ is our current guess for $h(y)$. We first show that

(*a*) if $(y,y_n) \in B(K,\varepsilon)$, then $y_n \ge h(y) - \delta$, and

(*b*) if $(y,y_n) \notin B(K,-\varepsilon)$, then $y_n \le h(y) + \frac{2}{3}\delta$.

For the proof of (*a*) consider a $g \in \underline{\partial} h(y)$. Since g is a subgradient we have that $h(z) \ge h(y) + \langle g, z-y\rangle$ for all $z \in \mathbb{R}^{n-1}$. Hence, for all $z \in \mathbb{R}^{n-1}$ and z_n such that $(z,z_n) \in K$ we have

$$\left\langle \begin{pmatrix} -g \\ 1 \end{pmatrix}, \begin{pmatrix} y \\ h(y) \end{pmatrix} \right\rangle \le \left\langle \begin{pmatrix} -g \\ 1 \end{pmatrix}, \begin{pmatrix} z \\ h(z) \end{pmatrix} \right\rangle \le \left\langle \begin{pmatrix} -g \\ 1 \end{pmatrix}, \begin{pmatrix} z \\ z_n \end{pmatrix} \right\rangle$$

where the first inequality is a rewriting of the subgradient inequality and the second inequality uses that $z_n \ge h(z)$ since $(z,z_n) \in K$. Since $(y,y_n) \in B(K,\varepsilon)$ it follows from the above inequality that

$$\left\langle \begin{pmatrix} -g \\ 1 \end{pmatrix}, \begin{pmatrix} y \\ y_n \end{pmatrix} \right\rangle \ge \left\langle \begin{pmatrix} -g \\ 1 \end{pmatrix}, \begin{pmatrix} y \\ h(y) \end{pmatrix} \right\rangle - \varepsilon \left\| \begin{pmatrix} -g \\ 1 \end{pmatrix} \right\| \ge \left\langle \begin{pmatrix} -g \\ 1 \end{pmatrix}, \begin{pmatrix} y \\ h(y) \end{pmatrix} \right\rangle - \varepsilon(\|g\|+1).$$

Lemma 19 together with the argument of Equation (1) implies that $\|g\| \le \frac{2R}{r}$. Since

$$\varepsilon(\|g\|+1) \le \varepsilon\left(\frac{2R}{r}+1\right) \le \varepsilon\frac{3R}{r} \le \delta,$$

we obtain the inequality of (*a*).

For (*b*), consider the convex set C which is the convex hull of $B((y,0),r/2)$ and $(y,h(y))$. Note that $B(C,-\varepsilon)$ is the convex hull of $B((y,0),r/2-\varepsilon)$ and $\left(y,h(y)\left(1-\frac{2\varepsilon}{r}\right)\right)$. Since $C \subseteq K$, we have $B(C,-\varepsilon) \subseteq B(K,-\varepsilon)$. Therefore $(y,y_n) \notin B(K,-\varepsilon)$ implies $(y,y_n) \notin B(C,-\varepsilon)$, and

$$y_n \leq h(y)\left(1-\frac{2\varepsilon}{r}\right) = h(y) - \varepsilon\frac{2h(y)}{r} \leq h(y) + \varepsilon\frac{2R}{r} \leq h(y) + \frac{2}{3}\delta.$$

Now we can analyze the binary search algorithm. By making $\mathcal{O}\left(\log\left(\frac{R}{\delta}\right)\right)$ $\mathrm{MEM}_{\varepsilon,0}(K)$ queries to points of the form (y,z), we can find a value $y_n \in [-R,-\frac{r}{2}]$ such that $(y,y_n) \in B(K,\varepsilon)$ but $(y,y_n-\frac{\delta}{3}) \notin B(K,-\varepsilon)$. By (*a*)-(*b*) we get that $|h(y)-y_n| \leq \delta$. □

The following lemma shows how to convert an approximate subgradient of h to a hyperplane that approximately separates x from K.

Lemma 21. *Suppose $-\|x\|e_n = x \notin B(K,-\varepsilon)$, and $\tilde{g} \in \mathbb{R}^{n-1}$ is an approximate subgradient of h at 0, meaning that for some $a,b \in \mathbb{R}$ and for all $y \in \mathbb{R}^{n-1}$*

$$h(y) \geq h(0) + \langle \tilde{g}, y\rangle - a\|y\| - b,$$

then $s := \frac{(-\tilde{g},1)}{\|(-\tilde{g},1)\|}$ satisfies $\langle s,z\rangle \geq \langle s,x\rangle - \frac{aR+b}{\|(-\tilde{g},1)\|} - \frac{2R}{r}\frac{\varepsilon}{\|(-\tilde{g},1)\|}$ for all $z \in K$.

Proof. Let us introduce the notation $z=(y,z_n)$ and $s' := (-\tilde{g},1) = \|(-\tilde{g},1)\|s$, then

$$\begin{aligned}
\langle s',z\rangle &= z_n - \langle \tilde{g},y\rangle \\
&\geq h(y) - \langle \tilde{g},y\rangle \\
&\geq h(0) - a\|y\| - b \\
&\geq -\|x\| - \frac{2R}{r}\varepsilon - aR - b \\
&= \langle s',x\rangle - aR - b - \frac{2R}{r}\varepsilon,
\end{aligned}$$

where the last inequality used claim (*b*) from the proof of Lemma 20 with the point $(0,-\|x\|)$ and $\delta = \frac{3R}{r}\varepsilon$. □

We now construct a separation oracle using $\widetilde{\mathcal{O}}(n)$ classical queries to a membership oracle. In particular, to construct an η-precise separation oracle, we require an ε-precise membership oracle with

$$\varepsilon = \frac{\eta}{676}n^{-2}\left(\frac{r}{R}\right)^3\left(\frac{\eta}{R}\right)^2\rho$$

The analogous result in [LSV18, Theorem 14] uses the stronger assumption[11]

$$\varepsilon \approx \frac{\eta}{8\cdot 10^6}n^{-\frac{7}{2}}\left(\frac{r}{R}\right)^6\left(\frac{\eta}{R}\right)^2\rho^3.$$

Compared to this, our result scales better in terms of n, $\frac{r}{R}$ and ρ.

[11] It seems that Lee et al. [LSV18, Algorithm 1] did not take into account the change in precision analogous to our Lemma 20, therefore one would probably need to worsen their exponent of $\frac{r}{R}$ from 6 to 7.

Theorem 22. *Let K be a convex set satisfying $B(0,r) \subseteq K \subseteq B(0,R)$. For any $\eta \in (0,R]$ and $\rho \in (0,1/3]$, we can implement the oracle $\mathrm{SEP}_{\eta,\rho}(K)$ using $\mathcal{O}\Big(n \log\Big(\frac{n}{\rho}\frac{R}{\eta}\frac{R}{r}\Big)\Big)$ classical queries to a $\mathrm{MEM}_{\varepsilon,0}(K)$ oracle, where $\varepsilon \leq \eta(26n)^{-2}\big(\frac{r}{R}\big)^3\big(\frac{\eta}{R}\big)^2\rho$.*

Proof. Let $x \notin B(K,-\varepsilon)$ be the point we want to separate from K. Let $\delta := \eta\frac{n^{-2}}{9\cdot 24}\big(\frac{r}{R}\cdot\frac{\eta}{R}\big)^2\rho$, then $\varepsilon \leq \frac{r}{3R}\delta$. By Lemma 19 we know that h is $\frac{2R}{r}$-Lipschitz on $B(0,r/2)$. By Lemma 20 we can evaluate h to within error δ using $\mathcal{O}\Big(\log\Big(\frac{R}{\delta}\Big)\Big)$ queries to a $\mathrm{MEM}_{\varepsilon,0}(K)$ oracle. Let us choose $r_1 := \frac{r}{12\sqrt{n}}\frac{\eta}{R}$, then $r_1\sqrt{n} \leq \frac{r}{4}$, therefore $B_\infty(0,2r_1) \subseteq B(0,r/2)$. Also note that $\delta \leq \frac{\eta}{6\rho} = \frac{2r_1\sqrt{n}R}{\rho r}$. Hence by Lemma 12, using $\mathcal{O}\Big(n\log\Big(\frac{R}{\delta}\Big)\Big)$ queries to a $\mathrm{MEM}_{\varepsilon,0}(K)$ oracle, we can compute an approximate subgradient $\tilde{g}$ such that with probability at least $1-\rho$ we have

$$h(y) \geq h(0) + \langle \tilde{g}, y\rangle - \frac{3n^{\frac{3}{4}}}{2}\sqrt{\frac{\delta 2R}{\rho r_1 r}}\|y\| - \frac{4R}{r}\sqrt{n}r_1 \qquad \text{for all } y \in \mathbb{R}^{n-1}.$$

Substituting the value of r_1 and δ we get $h(y) \geq h(0) + \langle \tilde{g}, y\rangle - \frac{\eta}{2R}\|y\| - \frac{\eta}{3}$, which by Lemma 21 gives an s such that $\langle s,z\rangle \geq \langle s,x\rangle - \frac{5}{6}\eta - \frac{2R}{r}\varepsilon \geq \langle s,x\rangle - \eta$ for all $z \in K$ □

Finally, we give a proof of our main result: we construct a separation oracle using $\widetilde{\mathcal{O}}(1)$ quantum queries to a membership oracle.

Theorem 23. *Let K be a convex set satisfying $B(0,r) \subseteq K \subseteq B(0,R)$. For any $\eta \in (0,R]$ and $\rho \in (0,1/3]$, we can implement the oracle $\mathrm{SEP}_{\eta,\rho}(K)$ using $\mathcal{O}\Big(\log\big(\frac{n}{\rho}\big)\log\Big(\frac{n}{\rho}\frac{R}{\eta}\frac{R}{r}\Big)\Big)$ quantum queries to a $\mathrm{MEM}_{\varepsilon,0}(K)$ oracle (and its inverse), where $\varepsilon \leq \eta(58n)^{-\frac{9}{2}}\big(\frac{r}{R}\big)^3\big(\frac{\eta}{R}\big)^2\rho$.*

Proof. Let $x \notin B(K,-\varepsilon)$ be the point that we want to separate from K. Let us define $\delta := \eta\frac{23^{-4}}{4\cdot 24}n^{-\frac{9}{2}}\big(\frac{r}{R}\cdot\frac{\eta}{R}\big)^2\rho$, then $\varepsilon \leq \frac{r}{3R}\delta$. By Lemma 19 we know that h is $\frac{2R}{r}$-Lipschitz on $B(0,r/2)$. By Lemma 20 we can evaluate h to within error δ using $\mathcal{O}\Big(\log\Big(\frac{R}{\delta}\Big)\Big)$ queries to a $\mathrm{MEM}_{\varepsilon,0}(K)$ oracle. Let us choose $r_1 := \frac{r}{12\sqrt{n}}\frac{\eta}{R}$, then $r_1\sqrt{n} \leq \frac{r}{4}$, therefore $B_\infty(0,2r_1) \subseteq B(0,r/2)$. Also note that $\delta \leq \frac{\eta}{6\rho} = \frac{2r_1 nR}{\rho r}$. Hence by Lemma 18, using $\mathcal{O}\Big(\log\big(\frac{n}{\rho}\big)\log\Big(\frac{R}{\delta}\Big)\Big)$ queries to a $\mathrm{MEM}_{\varepsilon,0}(K)$ oracle, we can compute an approximate subgradient $\tilde{g}$ such that with probability at least $1-\rho$ we have

$$h(y) \geq h(0) + \langle \tilde{g}, y\rangle - (23n)^2\sqrt{\frac{2\delta R}{\rho r_1 r}}\|y\| - \frac{4R}{r}\sqrt{n}r_1 \qquad \text{for all } y \in \mathbb{R}^{n-1}.$$

Substituting the value of r_1 and δ we get $h(y) \geq h(0) + \langle \tilde{g}, y\rangle - \frac{\eta}{2R}\|y\| - \frac{\eta}{3}$, which by Lemma 21 gives an s such that $\langle s,z\rangle \geq \langle s,x\rangle - \frac{5}{6}\eta - \frac{2R}{r}\varepsilon \geq \langle s,x\rangle - \eta$ for all $z \in K$. □

5 Lower bounds

For a convex set K satisfying $B(0,r) \subseteq K \subseteq B(0,R)$, we have shown in Theorem 23 that one can implement a $\mathrm{SEP}(K)$ oracle with $\widetilde{\mathcal{O}}(1)$ quantum queries to a $\mathrm{MEM}(K)$ oracle if the membership oracle is sufficiently precise. In this section we first show that this is exponentially better than what can be achieved using classical access to a membership oracle. We also investigate how many queries to a membership/separation oracle are needed in order to implement an optimization oracle. Our results are as follows.

- We show that $\Omega(n)$ classical queries to a membership oracle are needed to implement a weak separation oracle.
- We show that $\Omega(n)$ classical (resp. $\Omega(\sqrt{n})$ quantum) queries to a separation oracle are needed to implement a weak optimization oracle; even when we *know an interior point* in the set.
- We show an $\Omega(n)$ lower bound on the number of classical and/or quantum queries to a separation oracle needed to optimize over the set when we *do not know an interior point*.

In this section we will always assume that the input oracle is a strong oracle but the output oracle is allowed to be a weak oracle with error ε. Furthermore, we will make sure that R, $1/r$, and $1/\varepsilon$ are all upper bounded by a polynomial in n. This guarantees that the lower bound is based on the dimension of the problem, not the required precision.

5.1 Classical lower bound on the number of MEM queries needed for SEP

Here we show that a separation query can provide $\Omega(n)$ bits of information about the underlying convex set K; since a classical membership query returns a 0 or a 1 and hence can give at most 1 bit of information,[12] this theorem immediately implies a lower bound of $\Omega(n)$ on the number of classical membership queries needed to implement one separation query.

Theorem 24. *Let $\varepsilon \leq \frac{1}{48}$. There exist a set of $m = 2^{\Omega(n)}$ convex sets $K_1, \ldots, K_m$ and points $y, x_0 \in \mathbb{R}^n$ such that $B(x_0, 1/3) \subseteq K_i \subseteq B(x_0, 2\sqrt{n})$ for all $i \in [m]$, and such that the result of a classical query to $\mathrm{SEP}_{\varepsilon,0}(K_i)$ with the point y correctly identifies i.*

Proof. Let $h_1, \ldots, h_m \in \mathbb{R}^n$ be a set of $m = 2^{\Omega(n)}$ entrywise non-negative unit vectors such that $\langle h_i, h_j \rangle \leq 0.51$ for all distinct $i, j \in [m]$.[13]

Now pick an $i \in [m]$ and define $\hat{K}_i := \{x : \langle h_i, x \rangle \leq 0\} \cap B(0, \sqrt{n})$ and $K_i := B(\hat{K}_i, \varepsilon)$. Then $\hat{K}_i = B(K_i, -\varepsilon)$. Note that for $x_0 = -e/3$ we have $B(x_0, 1/3) \subseteq K_i \subseteq B(x_0, 2\sqrt{n})$. We claim that a query to $\mathrm{SEP}_{\varepsilon,0}(K_i)$ with the point $y = 3\varepsilon e \in \mathbb{R}^n$ will identify h_i. First note that $y \notin B(K_i, \varepsilon)$, since $\hat{K}_i$ does not contain any entrywise positive vectors and y has distance at least 3ε from all vectors that have at least one non-positive entry. Hence a separation query with y must return a unit vector g that describes a valid separating hyperplane for K_i.

On the other, if g describes a valid separating hyperplane for K_j, then

$$\forall x \in \hat{K}_j \colon \langle g, x \rangle \leq \langle g, y \rangle + \varepsilon \leq \|g\| \cdot \|y\| + \varepsilon \leq (3\sqrt{n} + 1)\varepsilon \leq 4\sqrt{n}\varepsilon. \tag{8}$$

Now consider the specific point x that is the projection of g onto $h_j^{\perp}$ (the hyperplane orthogonal to h_j) scaled by a factor $\sqrt{n}$, i.e., $x = \sqrt{n}(g - \langle g, h_j \rangle h_j)$. Since $\langle h_j, x \rangle = 0$ and

[12] This is not true for *quantum* membership queries!

[13] We can show that such a set of vectors exists as follows. Let $n = ck$ for sufficiently large constant c. Choose $m = 2^k$ (which is $2^{\Omega(n)}$) uniformly random vectors $v_1, \ldots, v_m$ in $\{0,1\}^n$. Note that the expected Hamming weight of one such vector is $n/2$, and the expected inner product between two vectors is $n/4$ (the inner product just counts for how many of the n bit-positions both vectors have a 1). By a standard calculation (Chernoff bound plus a union bound), one can show that with high probability these 2^k vectors each have Hamming weight $\geq 0.495n$, and the inner product between any two of them is $\leq 0.252n$. Fix 2^k such vectors with these properties, and define $h_i := v_i/\|v_i\|$. These are unit vectors with non-negative entries, and pairwise inner products $\langle h_i, h_j \rangle = \langle v_i, v_j \rangle / (\|v_i\| \|v_j\|) \leq 0.252n/(0.495n) < 0.51$.

$\|x\| \leq \sqrt{n}$, we have $x \in \hat{K}_j$. Choosing this x in (8) gives the following inequality

$$\sqrt{n}(1 - \langle g, h_j \rangle^2) = \langle g, x \rangle \leq 4\sqrt{n}\varepsilon.$$

Hence (8) implies $|\langle g, h_j \rangle| \geq \sqrt{1 - 4\varepsilon} \geq \sqrt{\frac{11}{12}} \geq \frac{19}{20}$.

Since (8) holds for $j = i$, it follows that at least one of the two vectors $g - h_i$ and $g + h_i$ has length at most $\sqrt{2(1 - |\langle g, h_i \rangle|^2)} \leq \sqrt{8\varepsilon}$; assume the former for simplicity. If (8) would also hold for $j \neq i$, then we would get a contradiction:

$$\frac{19}{20} \leq |\langle g, h_j \rangle| \leq |\langle g - h_i, h_j \rangle| + |\langle h_i, h_j \rangle| \leq \sqrt{8\varepsilon} + 0.51 < \frac{19}{20}.$$

Hence g uniquely identifies h_i. □

5.2 Lower bound on number of SEP queries for OPT (given an interior point)

We now consider lower bounding the number of quantum queries to a separation oracle needed to do optimization. In fact, we prove a lower bound on the number of separation queries needed for validity, which implies the same bound on optimization. We will use a reduction from a version[14] of the well-studied *search* problem:

Given $z \in \{0,1\}^n$ such that either $|z| = 0$ or $|z| = 1$, decide which of the two holds.

It is not hard to see that if the access to z is given via classical queries, then $\Omega(n)$ queries are needed. It is well known [BBBV97] that if we allow quantum queries, then $\Omega(\sqrt{n})$ queries are needed (i.e., Grover's quantum search algorithm [Gro96] is optimal). We use this problem to show that there exist convex sets for which it is hard to construct a weak validity oracle, given a strong separation oracle. Since a separation oracle can be used as a membership oracle, this gives the same hardness result for constructing a weak validity oracle from a strong membership oracle.

Theorem 25. *Let $0 < \rho \leq 1/3$. Let $\mathcal{A}$ be an algorithm that implements a $\mathrm{VAL}_{(5n)^{-1},\rho}(K)$ oracle for every convex set K (with $B(x_0, r) \subseteq K \subseteq B(x_0, R)$) using only queries to a $\mathrm{SEP}_{0,0}(K)$ oracle, and unitaries that are independent of K. Then the following statements are true, even when we restrict to convex sets K with $r = 1/3$ and $R = 2\sqrt{n}$:*

- *if the queries to $\mathrm{SEP}_{0,0}(K)$ are classical, then the algorithm uses $\Omega(n)$ queries.*
- *if the queries to $\mathrm{SEP}_{0,0}(K)$ are quantum, then the algorithm uses $\Omega(\sqrt{n})$ queries.*

Proof. Let $z \in \{0,1\}^n$ have Hamming weight $|z| = 0$ or $|z| = 1$. We construct a set K_z in such a way that solving the weak validity problem solves the search problem for z, while separation queries for K_z can be answered using a single query to z. The known classical and quantum lower bounds on the search problem then imply the two claims of the theorem, respectively.

Define $K_z := \prod_{i=1}^n [-1, z_i]$. Observe that if we set $x_0 = (-1/2, \ldots, -1/2)$, then $B(x_0, \frac{1}{3}) \subseteq K_z \subseteq B(x_0, 2\sqrt{n})$.

We first show how to implement a strong separation oracle using a single query to z. Suppose the input is the point y. The strong separation oracle works as follows:

1. If $y \in [-1, 0]^n$, then return the statement that $y \in B(K_z, 0) = K_z$.

[14] Note that this is a slightly different version from the one used in Section 2.1.

2. If $y \notin [-1,1]^n$, then return a hyperplane that separates y from $[-1,1]^n$ (and hence from K_z).

3. Otherwise, let i be such that $y_i > 0$. Query z_i.

   (a) If $z_i = 1$ and i is the only index such that $y_i > 0$, then return that $y \in B(K_z, 0) = K_z$.

   (b) If $z_i = 1$ and there is a $j \neq i$ such that $y_j > 0$, return the separating hyperplane corresponding to $x_j \leq y_j$.

   (c) If $z_i = 0$, then return the separating hyperplane $x_i \leq y_i$.

We show that a validity query over K_z with the direction $c = \frac{1}{\sqrt{n}}(1, \ldots, 1) \in \mathbb{R}^n$, value $\gamma = \frac{1}{2\sqrt{n}}$ and error $\varepsilon = \frac{1}{5n}$ solves the search problem:

- If $|z| = 0$, then for all points $x \in K_0$ we have $\langle c, x \rangle \leq 0$. Thus, for all points $x \in B(K_0, \varepsilon)$ we have $\langle c, x \rangle \leq \varepsilon < \gamma - \varepsilon$. Hence the validity oracle will have to return that $\langle c, x \rangle \leq \gamma + \varepsilon$ holds for all $x \in B(K_0, -\varepsilon)$, since the other possible output is not true.

- If $|z| = 1$, then the point $z \in K_z$ satisfies $\langle z, c \rangle = \frac{1}{\sqrt{n}}$ and therefore $x = z - \varepsilon e \in B(K_z, -\varepsilon)$ satisfies $\langle c, x \rangle = \frac{1}{\sqrt{n}} - \sqrt{n}\varepsilon > \gamma + \varepsilon$. Hence the validity oracle will have to return that $\langle c, x \rangle \geq \gamma - \varepsilon$ holds for some $x \in B(K_z, \varepsilon)$, since the other possible output is not true.

□

5.3 Lower bound on number of SEP queries for OPT (without interior point)

We now lower bound the number of quantum queries to a separation oracle needed to solve the optimization problem, if our algorithm does not already know an interior point of K. In fact we prove a lower bound on finding a point close to K using separation queries, which implies the lower bound on the number of separation queries needed for optimization since OPT returns a point close to the set K.

We prove our lower bound by a reduction to the problem of learning z with *first-difference queries*. Here one needs to find an initially unknown n-bit binary string z via a guessing game. For a given guess $g \in \{0,1\}^n$ a query returns the first index in $[n]$ for which the binary strings z and g differ (or it returns $n+1$ if $z = g$). The goal is to recover z with as few guesses as possible. First we prove an $\Omega(n)$ quantum query lower bound for this problem.[15]

Theorem 26 (Quantum lower bound for learning z with first-difference queries)**.** *Let $z \in \{0,1\}^n$ be an unknown string accessible by an oracle acting as $O_z|g,b\rangle = |g, b \oplus f(g,z)\rangle$, where $f(g,z)$ is the first index for which z and g differ, more precisely $f(g,z) = \min\{i \in [n] : g_i \neq z_i\}$ if $g \neq z$ and $f(g,z) = n+1$ otherwise. Then every quantum algorithm that outputs z with high probability uses at least $\Omega(n)$ queries to O_z.*

[15]Note that this is a strengthening of the $\Omega(n)$ quantum query lower bound for binary search on a space of size 2^n by Ambainis [Amb99], since first-difference queries are at least as strong as the queries one makes in binary search.

Proof. We will use the general adversary bound [HLŠ07]. For this problem, we call $\Gamma \in \mathbb{R}^{2^n \times 2^n}$ an *adversary matrix* if it is a non-zero matrix with zero diagonal whose rows and columns are indexed by all $z \in \{0,1\}^n$. For $g \in \{0,1\}^n$ let us define $\Delta_g \in \{0,1\}^{2^n \times 2^n}$ such that the $[z, z']$ entry of Δ_g is 0 if and only if $f(g,z) = f(g,z')$. The general adversary bound tells us that for any adversary matrix Γ, the quantum query complexity of our problem is

$$\Omega\left(\frac{\|\Gamma\|}{\max_{g\in\{0,1\}^n}\|\Gamma \circ \Delta_g\|}\right), \tag{9}$$

where "$\circ$" denotes the Hadamard product and $\|\cdot\|$ the operator norm.

We claim that Equation (9) gives a lower bound of $\Omega(n)$ for the adversary matrix Γ defined as

$$\Gamma[z,z'] = \begin{cases} 2^{f(z,z')} & \text{if } z \neq z' \\ 0 & \text{if } z = z' \end{cases}$$

It is easy to see that Γ is indeed an adversary matrix since it is zero on the diagonal and non-zero everywhere else. Furthermore, the all-one vector e is an eigenvector of Γ with eigenvalue $n2^n$:

$$(\Gamma e)_z = \sum_{z'\in\{0,1\}^n} \Gamma[z,z'] = \sum_{d=1}^{n} 2^d \cdot |\{z' \in \{0,1\}^n \,:\, f(z,z') = d\}| = \sum_{d=1}^{n} 2^d 2^{n-d} = n2^n.$$

So $\Gamma e = n2^n e$ and hence $\|\Gamma\| \geq n2^n$.

From the definition of Δ_g it follows that

$$(\Gamma \circ \Delta_g)[z,z'] = 2^{f(z,z')}\chi_{[f(g,z)\neq f(g,z')]},$$

where $\chi_{[f(g,z)\neq f(g,z')]}$ stands for the indicator function of the condition $f(g,z) \neq f(g,z')$. Let $\Gamma_g := \Gamma \circ \Delta_g$. We will show an upper bound on $\|\Gamma_g\|$. We decompose Γ_g in an "upper-triangular" and a "lower-triangular" part:

$$\begin{aligned} \Gamma_g^U[z,z'] &:= 2^{f(z,z')}\chi_{[f(g,z)<f(g,z')]} = 2^{f(g,z)}\chi_{[f(g,z)<f(g,z')]}, \\ \Gamma_g^L[z,z'] &:= 2^{f(z,z')}\chi_{[f(g,z')<f(g,z)]} = 2^{f(g,z')}\chi_{[f(g,z')<f(g,z)]}. \end{aligned} \tag{10}$$

So $\Gamma_g = \Gamma_g^U + \Gamma_g^L$ and $\Gamma_g^U = (\Gamma_g^L)^T$. Hence by the triangle inequality we have

$$\|\Gamma_g\| \leq \left\|\Gamma_g^U\right\| + \left\|\Gamma_g^L\right\| = 2\left\|\Gamma_g^U\right\|. \tag{11}$$

It thus suffices to upper bound $\left\|\Gamma_g^U\right\|$. Notice that as (10) shows, $\Gamma_g^U[z,z']$ only depends on the values $f(g,z)$, $f(g,z')$. Since the range of $f(g,\cdot)$ is $[n+1]$, we can think of Γ_g^U as an $(n+1)\times(n+1)$ block-matrix, where the blocks are determined by the values of $f(g,z)$ and $f(g,z')$, and within a block all matrix elements are the same. Also observe that for all $k \in [n]$ there are 2^{n-k} bitstrings $y \in \{0,1\}^n$ such that $f(g,y) = k$, which tells us the sizes of the blocks are $2^{n-k} \times 2^{n-k}$. Motivated by these observations we define an orthonormal set of vectors in $\mathbb{R}^{2^n}$ by $v_{n+1} := e_g$, and for all $k \in [n]$

$$v_k := \sum_{y:f(g,y)=k} \frac{e_y}{\sqrt{2^{n-k}}}.$$

Since the row and column spaces of Γ_g^U are spanned by $\{v_k : k \in [n+1]\}$, we can reduce Γ_g^U to an $(n+1) \times (n+1)$-dimensional matrix G:

$$\Gamma_g^U = \left(\sum_{k=1}^{n+1} v_k v_k^T\right)\Gamma_g^U\left(\sum_{\ell=1}^{n+1} v_\ell v_\ell^T\right) = \left(\sum_{k=1}^{n+1} v_k e_k^T\right)\underbrace{\left(\sum_{k=1}^{n+1} e_k v_k^T\right)\Gamma_g^U\left(\sum_{\ell=1}^{n+1} v_\ell e_\ell^T\right)}_{G:=}\left(\sum_{\ell=1}^{n+1} e_\ell v_\ell^T\right).$$

It follows from the above identity, together with the orthonormality of $\{v_1, \ldots, v_n, v_{n+1}\}$, that

$$\left\|\Gamma_g^U\right\| = \left\|\left(\sum_{k=1}^{n+1} e_k v_k^T\right)\Gamma_g^U\left(\sum_{\ell=1}^{n+1} v_\ell e_\ell^T\right)\right\| = \|G\|. \tag{12}$$

$G \in \mathbb{R}^{(n+1)\times(n+1)}$ is a strictly upper-triangular matrix, with the following entries for $k, \ell \in [n]$:

$$\begin{aligned} G[k,\ell] &= v_k^T \Gamma_g^U v_\ell \\ &= \left(\sum_{z:f(g,z)=k} \frac{e_z^T}{\sqrt{2^{n-k}}}\right)\Gamma_g^U\left(\sum_{z':f(g,z')=\ell} \frac{e_{z'}}{\sqrt{2^{n-\ell}}}\right) \\ &= \frac{2^{\frac{k+\ell}{2}}}{2^n}\left(\sum_{z:f(g,z)=k} e_z^T\right)\Gamma_g^U\left(\sum_{z':f(g,z')=\ell} e_{z'}\right) \\ &= \frac{2^{\frac{k+\ell}{2}}}{2^n}\sum_{z:f(g,z)=k}\sum_{z':f(g,z')=\ell} \Gamma_g^U[z,z'] \end{aligned}$$

By Equation (10) this is further equal to

$$\begin{aligned} G[k,\ell] &= \frac{2^{\frac{k+\ell}{2}}}{2^n}\sum_{z:f(g,z)=k}\sum_{z':f(g,z')=\ell} 2^k \chi_{[k<\ell]} \\ &= \frac{2^{\frac{k+\ell}{2}}}{2^n} 2^{n-k} 2^{n-\ell} 2^k \chi_{[k<\ell]} \\ &= 2^{n-\frac{\ell-k}{2}} \chi_{[k<\ell]}. \end{aligned}$$

Similarly for $\ell = n+1$ we get that $G[k,\ell] = \sqrt{2}\, 2^{n-\frac{\ell-k}{2}} \chi_{[k<\ell]}$ for all $k \in [n+1]$. For each $d \in [n]$ define $G_d \in \mathbb{R}^{(n+1)\times(n+1)}$ such that $G_d[k,\ell] = G[k,\ell]\chi_{[d=\ell-k]}$. This G_d is only non-zero on one non-main diagonal (namely the (k,ℓ)-entries where $d = \ell - k$), and its non-zero entries are all upper bounded by $\sqrt{2}\, 2^n 2^{-\frac{d}{2}}$. We have $G = \sum_{d=1}^n G_d$ and therefore

$$\|G\| \leq \sum_{d=1}^n \|G_d\| \leq \sum_{d=1}^n \sqrt{2}\, 2^n 2^{-\frac{d}{2}} = 2^n \sum_{d=0}^{n-1} (\sqrt{2})^{-d} \leq \frac{2^n}{1 - 1/\sqrt{2}} \leq 2^{n+2}. \tag{13}$$

Inequalities (11)-(13) give that $\|\Gamma_g\| \leq 2^{n+3}$ and hence (9) yields a lower bound of $\Omega\left(\frac{n2^n}{2^{n+3}}\right) = \Omega(n)$ on the number of quantum queries to O_z needed to learn z. □

Theorem 27. *Finding a point in $B_\infty(K, 1/7)$ for an unknown convex set K such that $K \subseteq B_\infty(0,2) \subseteq \mathbb{R}^n$ requires $\Omega(n)$ quantum queries to a separation oracle $\mathrm{SEP}_{0,0}(K)$, even if we are promised there exists some unknown $x \in \mathbb{R}^n$ such that $B_\infty(x, 1/3) \subseteq K$.*

Proof. We will prove an $\Omega(n)$ quantum query lower bound for this problem by a reduction from learning with first-difference queries. Let $z \in \{0,1\}^n$ be an unknown binary string, and let us define $K_z := B_\infty(z, 1/3) \subset \mathbb{R}^n$ as a small box around the corner of the hypercube corresponding to z. Then clearly $K_z \subset B_\infty(0,2)$, and finding a point close enough to K_z is enough to recover z.

We can easily reduce a separation oracle query to a first-difference query to z, as follows. Suppose y is the vector for which we need to answer a SEP query:

1. If y is outside $[-1/3, 4/3]^n$, then output a hyperplane separating y from $[-1/3, 4/3]^n$.
2. If y is in $[-1/3, 4/3]^n$, then let g be the nearest corner of the hypercube.
3. Let i be the result of a first-difference query to z with g.
   (a) If $i = n+1$, indicating that $z = g$, then we know K_z exactly, so we can find a separating hyperplane or conclude that $y \in K_z$.
   (b) If $z \neq g$, then return e_i if $g_i = 1$, and $-e_i$ if $g_i = 0$.

Hence our $\Omega(n)$ quantum lower bound on learning z with first-difference queries implies an $\Omega(n)$ lower bound on the number of quantum queries to a separation oracle needed for finding a point close to a convex set. □

Since optimization over a set K gives a point close to the set K, this also implies a lower bound on the number of separation queries needed for optimization. This theorem is tight up to logarithmic factors, since it is known that $\widetilde{\mathcal{O}}(n)$ classical separation queries suffice for optimization, even without knowing a point in the convex set [LSW15]. Finally we remark that, due to our improved algorithm for optimization using validity queries (by combining Section 6 with Theorem 23), this also gives an $\widetilde{\Omega}(n)$ lower bound on the number of separation queries needed to implement validity.[16]

6 Consequences of convex polarity

Here we justify the central symmetry of Figure 1 using the results of Grötschel, Lovász, and Schrijver [GLS88, Section 4.4]. We first need to recall the definition and some basic properties of the polar K^* of a set $K \subseteq \mathbb{R}^n$. This is the closed convex set defined as follows:

$$K^* = \{y \in \mathbb{R}^n : \langle y, x\rangle \leq 1 \text{ for all } x \in K\}.$$

It is straightforward to verify that if $B(0,r) \subseteq K \subseteq B(0,R)$, then $B(0,1/R) \subseteq K^* \subseteq B(0,1/r)$, moreover $(K^*)^* = K$ for closed convex sets.[17] For the remainder of this section we assume that K is a closed convex set such that $B(0,r) \subseteq K \subseteq B(0,R)$.

We will observe that for the polar K^* of a set K the following holds:

$$\mathrm{MEM}(K^*) \leftrightarrow \mathrm{VAL}(K), \qquad \mathrm{SEP}(K^*) \leftrightarrow \mathrm{VIOL}(K), \tag{14}$$

where $\mathrm{MEM}(K^*) \leftrightarrow \mathrm{VAL}(K)$ means we can implement a weak validity oracle for K using a single query to a weak membership oracle for K^*, and vice versa. Since $\mathrm{VIOL}(K)$

[16] It is easy to modify Theorem 26 to prove a lower bound on computing the majority function applied to z, which would imply an $\Omega(n)$ lower bound on the number of separation queries needed to implement a validity oracle, without the log factors.

[17] Note that K^* is a dual representation of the convex set K. Each point in K^* corresponds to a (normalized) valid inequality for K. This duality is not to be confused with Lagrangian duality.

and $\mathrm{OPT}(K)$ are equivalent up to reductions that use $\widetilde{\Theta}(1)$ queries (via binary search), this justifies the central symmetry of Figure 1, because it shows that algorithms that implement $\mathrm{VIOL}(K)$ given $\mathrm{VAL}(K)$ are equivalent to algorithms that implement $\mathrm{SEP}(K^*)$ given $\mathrm{MEM}(K^*)$, and similarly algorithms that implement $\mathrm{SEP}(K)$ given $\mathrm{VIOL}(K)$ are equivalent to algorithms that implement $\mathrm{VIOL}(K^*)$ given $\mathrm{SEP}(K^*)$.

Grötschel, Lovász, and Schrijver [GLS88, Section 4.4] showed that the weak membership problem for K^* can be solved using a single query to a weak validity oracle for K, and that the weak separation problem for K^* can be solved using a single query to a weak violation oracle for K. Using similar arguments one can show the reverse directions as well, which justifies (14). Here we only motivate the equivalences between the above-mentioned weak oracles by showing the equivalence of the strong oracles (i.e., where ρ and ε are 0).

Strong membership on K^* is equivalent to strong validity on K. First, for a given vector $c \in \mathbb{R}^n$ and a $\gamma > 0$ observe the following:

$$\frac{c}{\gamma} \notin \mathrm{int}(K^*) \quad \Longleftrightarrow \quad \exists y \in K \text{ s.t. } \langle c/\gamma, y\rangle \geq 1 \quad \Longleftrightarrow \quad \exists y \in K \text{ s.t. } \langle c, y\rangle \geq \gamma.$$

Hence, a strong membership query to K^* with a point c can be implemented by querying a strong validity oracle for K with the vector c and the value 1. Likewise, a strong validity query to K with a point c and value[18] $\gamma > 0$ can be implemented using a strong membership query to K^* with c/γ.

Strong separation on K^* is equivalent to strong violation on K. To implement a strong separation query on K^* for a vector $y \in \mathbb{R}^n$ we do the following. Query the strong violation oracle for K with y and the value 1. If the answer is that $\langle y, x\rangle \leq 1$ for all $x \in K$, then $y \in K^*$. If instead we are given a vector $x \in K$ with $\langle y, x\rangle \geq 1$, then x separates y from K^* (indeed, for all $z \in K^*$, we have $\langle z, x\rangle \leq 1 \leq \langle y, x\rangle$).

For the reverse direction, to implement a strong violation oracle for K on the vector c and value[18] $\gamma > 0$ we do the following. Query the strong separation oracle for K^* with the point c/γ. If the answer is that $c/\gamma \in K^*$ then $\langle c, x\rangle \leq \gamma$ for all $x \in K$. If instead we are given a non-zero vector $y \in \mathbb{R}^n$ that satisfies $\langle c/\gamma, y\rangle \geq \langle z, y\rangle$ for all $z \in K^*$, then $\tilde{y} = y/\langle c/\gamma, y\rangle$ will be a valid answer for the strong violation oracle for K. Indeed, we have $\tilde{y} \in K$ because $\langle z, \tilde{y}\rangle \leq 1$ for all $z \in K^*$ and $K = (K^*)^*$, and by construction $\langle c, \tilde{y}\rangle = \gamma$.

7 Discussion and future work

We mention several open problems for future work:

- Our current implementation of an optimization query using $\widetilde{\mathcal{O}}(n)$ quantum membership queries is quadratically better than the best known classical randomized algorithm, which uses roughly n^2 membership queries. However, to the best of our knowledge it is open whether this quadratic classical bound is optimal (a quadratic classical lower bound is known for *deterministic* algorithms [Yao75]).
- Can we improve our $\Omega(\sqrt{n})$ lower bound on the number of separation (or membership) queries needed to implement an optimization oracle when our algorithm knows a point in K? We conjecture that the correct bound is $\widetilde{\Theta}(n)$, in which case knowing a point in K does not confer much benefit for query complexity.

[18] Observe that validity queries with value $\gamma \leq 0$ can be answered trivially, since $0 \in K$.

- Are there interesting convex optimization problems where separation is much harder than membership for classical computers?[19] Such problems would be good candidates for quantum speed-up in optimization in the real, non-oracle setting of time complexity. It is known that given a deterministic algorithm for function evaluation, an algorithm with roughly the same complexity can be constructed to compute the gradient of that function [GW08]. Hence for strong, deterministic oracles, separation is not much harder than membership queries. This, however, still leaves weak / randomized / quantum membership oracles to be considered.

- The algorithms that give an $\widetilde{\mathcal{O}}(n)$ upper bound on the number of separation queries for optimization (for example [LSW15, Theorem 42]) give the best theoretical results for many convex optimization problems. However, due to the large constants in these algorithms they are rarely used in a practical setting. A natural question is whether the algorithms used in practice lend themselves to quantum speed-ups. Very recent work by Kerenidis and Prakash [KP18] on quantum interior point methods is a first step in this direction.

Acknowledgments. We thank Shouvanik Chakrabarti, Andrew Childs, Tongyang Li, and Xiaodi Wu for sending us a preliminary version of their paper [CCLW18], and for useful comments and coordination between our two papers. AG thanks Márió Szegedy for insightful discussions. Many thanks to the anonymous referees of QIP'19 and Quantum for their constructive comments.

JvA and SG are supported by the Netherlands Organization for Scientific Research (NWO), grant number 617.001.351. AG and RdW are supported by ERC Consolidator Grant 615307-QPROGRESS. RdW is also partially supported by NWO through Gravitation-grant Quantum Software Consortium - 024.003.037, and through QuantERA project QuantAlgo 680-91-034.

References

[Amb99] Andris Ambainis. A better lower bound for quantum algorithms searching an ordered list. In *Proceedings of the 40th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 352–357, 1999. arXiv: quant-ph/9902053

[vAG19] Joran van Apeldoorn and András Gilyén. Improvements in quantum SDP-solving with applications. In *Proceedings of the 46th International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 99:1–99:15, 2019. arXiv: 1804.05058

[vAGGdW17] 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

[AŠ06] Andris Ambainis and Robert Špalek. Quantum algorithms for matching and network flows. In *Proceedings of the 23rd Symposium on Theoreti-*

[19] Moment polytopes are promising candidates for such examples. Recently an efficient weak membership oracle was constructed by Bürgisser et al. [BFG+18] for a class of these polytopes. However, to the best of our knowledge it is unknown how to directly implement separation oracles for them, so one might get a quantum speed-up for implementing separation oracles using few queries to their membership oracle.

cal Aspects of Computer Science (STACS), pages 172–183, 2006. arXiv: quant-ph/0508205

[BBBV97] 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. arXiv: quant-ph/9701001

[BFG+18] Peter Bürgisser, Cole Franks, Ankit Garg, Rafael Oliveira, Michael Walter, and Avi Wigderson. Efficient algorithms for tensor scaling, quantum marginals, and moment polytopes. In *Proceedings of the 59th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 883–897, 2018. arXiv: 1804.04739

[BKL+19] 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. In *Proceedings of the 46th International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 27:1–27:14, 2019. arXiv: 1710.02581

[BS17] 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

[Bub15] Sébastien Bubeck. Convex optimization: Algorithms and complexity. *Foundations and Trends in Machine Learning*, 8(3–4):231–357, 2015. arXiv: 1405.4980

[CCLW18] Shouvanik Chakrabarti, Andrew M. Childs, Tongyang Li, and Xiaodi Wu. Quantum algorithms and lower bounds for convex optimization. arXiv: 1809.01731, 2018.

[DH96] Christoph Dürr and Peter Høyer. A quantum algorithm for finding the minimum. arXiv: quant-ph/9607014, 1996.

[DHHM06] Christoph Dürr, Mark Heiligman, Peter Høyer, and Mehdi Mhalla. Quantum query complexity of some graph problems. *SIAM Journal on Computing*, 35(6):1310–1328, 2006. arXiv: quant-ph/0401091

[GAW19] 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

[GLS88] Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer, 1988.

[Gro96] Lov K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the 28th ACM Symposium on the Theory of Computing (STOC)*, pages 212–219, 1996. arXiv: quant-ph/9605043

[GW08] Andreas Griewank and Andrea Walther. *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*. SIAM, second edition, 2008.

[HLŠ07] Peter Høyer, Troy Lee, and Robert Špalek. Negative weights make adversaries stronger. In *Proceedings of the 39th ACM Symposium on the Theory of Computing (STOC)*, pages 526–535, 2007. arXiv: quant-ph/0611054

[Jor05] Stephen P. Jordan. Fast quantum algorithm for numerical gradient estimation. *Physical Review Letters*, 95(5):050501, 2005. arXiv: quant-ph/0405146

[Jor08] Stephen P. Jordan. *Quantum Computation Beyond the Circuit Model*. PhD thesis, Massachusetts Institute of Technology, 2008. arXiv: 0809.2307

[KP18] Iordanis Kerenidis and Anupam Prakash. A quantum interior point method for LPs and SDPs. arXiv: 1808.09266, 2018.

[LSV18] Yin Tat Lee, Aaron Sidford, and Santosh S. Vempala. Efficient convex optimization with membership oracles. In *Proceedings of the 31st Conference On Learning Theory (COLT)*, pages 1292–1294, 2018. arXiv: 1706.07357

[LSW15] 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

[NC00] Michael A. Nielsen and Isaac L. Chuang. *Quantum computation and quantum information*. Cambridge University Press, 2000.

[Szeg04] Mário Szegedy. Quantum speed-up of Markov chain based algorithms. In *Proceedings of the 45th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 32–41, 2004. arXiv: quant-ph/0401053

[Yao75] Andrew Chi-Chih Yao. On computing the minima of quadratic forms (preliminary report). In *Proceedings of the 7th ACM Symposium on the Theory of Computing (STOC)*, pages 23–26, 1975.

A Quantum gradient computation using relational oracles

In this appendix we extend the result of Corollary 15 to functions given by a relational input oracle. As a direct consequence this shows that the algorithm from Theorem 23 also works when the input is given as a relational membership oracle instead of a standard oracle.

Definition 28 (Unitary δ-approximator)**.** *Let X be a finite set and let Y denote a set of fixed-point b-bit numbers. Let $f\colon X \to Y$ be a function. We say that a relational quantum oracle U on X is a* b-bit unitary δ-approximator of f *if the valid answers for each $x \in X$ differ at most δ from $f(x)$ (i.e., $\mathcal{F}(x) = \{y \in Y : |f(x) - y| \leq \delta\}$), and the success probability is at least $\frac{2}{3}$.*

Corollary 29 (Gradient computation using a unitary δ-approximator)**.** *Let $\delta, B, r, c \in \mathbb{R}$, $\rho \in (0, 1/3]$. Let $x_0, g \in \mathbb{R}^n$ with $\|g\|_\infty \leq \frac{B}{r}$. Let $m := \left\lceil \log_2\left(\frac{B}{28\pi\delta}\right)\right\rceil$ and suppose $f : (x_0 + rG_m^n) \to \mathbb{R}$ is such that*

$$|f(x_0 + rx) - \langle g, rx\rangle - c| \leq \delta$$

for 99.9% of the points $x \in G_m^n$, and we have access to U, an $\mathcal{O}\left(\log\left(\frac{B}{\delta}\right)\right)$-bit unitary δ-approximator of f over the domain $(x_0 + rG_m^n)$. Then we can compute a vector $\tilde{g} \in \mathbb{R}^n$ such that

$$\Pr\left[\, \|\tilde{g} - g\|_\infty > \frac{8 \cdot 42\pi\delta}{r} \right] \leq \rho,$$

with $\mathcal{O}\left(\log\left(\frac{n}{\rho}\right)\right)$ queries to U and $U^\dagger$ and gate complexity $\mathcal{O}\left(n \log\left(\frac{n}{\rho}\right)\log\left(\frac{B}{\delta}\right)\mathrm{loglog}\left(\frac{n}{\rho}\right)\mathrm{loglog}\left(\frac{B}{\delta}\right)\right)$.

Proof. The algorithm is the same as in the less general Corollary 15 presented in Section 3.2, we just need to analyze it a bit more carefully. The main idea is still to implement an approximate version of the phase oracle $\mathrm{O}: |x,0,0\rangle \mapsto e^{2\pi i \frac{M}{3B} f(x_0+rx)}|x,0,0\rangle$, and then use Jordan's gradient computation algorithm. We approximate O by first approximately computing f using U, then applying[20] a controlled phase operation cP acting as $\mathrm{cP}: |y\rangle \mapsto e^{2\pi i \frac{M}{3B} y}|y\rangle$ (where $M = \frac{3B}{84\pi\delta}$ as in the proof of Corollary 15), and finally applying $U^\dagger$ to approximately uncompute f.

We can assume without loss of generality that our unitary δ-approximator is such that the probability of $|f(x)-y| > \delta$ is at most $\frac{1}{1200}$. If this is not the case, we can improve the success probability by querying U a few times and taking the median of the results.

Let us define $\mathcal{F}(x) := \{y \in Y : |f(x)-y| \le \delta\}$ as in Definition 28. Observe that

$$\begin{aligned}\Big\|\mathrm{O}|x,0,0\rangle - U^\dagger(I\otimes \mathrm{cP}\otimes I)U|x,0,0\rangle\Big\|^2 &= \Big\|\Big(I\otimes (e^{2\pi i\frac{M}{3B}f(x_0+rx)}I - \mathrm{cP})\otimes I\Big)U|x,0,0\rangle\Big\|^2 \\ &= \left\|\sum_{y\in Y}\Big(e^{2\pi i\frac{M}{3B}f(x_0+rx)} - e^{2\pi i\frac{M}{3B}y}\Big)\alpha_{x,y}|x,y,\psi_{x,y}\rangle\right\|^2.\end{aligned}$$

We bound the above quantity in two parts using the triangle inequality as follows:

$$\left\|\sum_{y\in Y\setminus\mathcal{F}(x)}\Big(e^{2\pi i\frac{M}{3B}f(x_0+rx)} - e^{2\pi i\frac{M}{3B}y}\Big)\alpha_{x,y}|x,y,\psi_{x,y}\rangle\right\|^2 \le \sum_{y\in Y\setminus\mathcal{F}(x)}|2\alpha_{x,y}|^2 \le \frac{1}{300};$$

$$\begin{aligned}\left\|\sum_{y\in \mathcal{F}(x)}\Big(e^{2\pi i\frac{M}{3B}f(x_0+rx)} - e^{2\pi i\frac{M}{3B}y}\Big)\alpha_{x,y}|x,y,\psi_{x,y}\rangle\right\|^2 &\le \sum_{y\in\mathcal{F}(x)}\left|2\pi i\frac{M}{3B}(f(x_0+rx)-y)\alpha_{x,y}\right|^2 \\ &\le \sum_{y\in Y_x}\left|2\pi i\frac{M}{3B}\delta\right|^2|\alpha_{x,y}|^2 \\ &\le \left|2\pi i\frac{M}{3B}\delta\right|^2 = \frac{1}{42^2}.\end{aligned}$$

Thus for all $x \in G_m^n$ we have that

$$\Big\|\mathrm{O}|x,0,0\rangle - U^\dagger(I\otimes\mathrm{cP}\otimes I)U|x,0,0\rangle\Big\| \le \sqrt{\frac{1}{300}+\frac{1}{42^2}} < \frac{1}{16}. \tag{15}$$

We can assume without loss of generality that our approximate phase oracle does not change the value of the input register. Otherwise we can just copy $|x\rangle$ to another register, then apply our approximate phase oracle on the second copy, then (approximately) erase the second copy of $|x\rangle$ using mod 2 bitwise addition with the first copy. Under this assumption by (15) we get that

$$\Big\|\mathrm{O}|\psi\rangle - U^\dagger(I\otimes\mathrm{cP}\otimes I)U|\psi\rangle\Big\| < \frac{1}{16}, \text{ for any quantum state } |\psi\rangle = \sum_{x\in G_m^n}\alpha_x|x,0,0\rangle. \tag{16}$$

From now on the proof is the same as the proof of Corollary 15. In that proof we showed that if we use the phase oracle O in Jordan's gradient computation algorithm,

[20] If y is a b-bit fixed-point binary number, then this can be implemented using b single-qubit phase gates as follows: we can assume without loss of generality that $y = a_0 + a\cdot\sum_{j=1}^b y_j 2^j$ for some fixed $a_0, a \in \mathbb{R}$. Then $e^{2\pi i\frac{M}{3B}y} = e^{2\pi i\frac{M}{3B}a_0}\prod_{j=1}^b e^{2\pi i\frac{M}{3B}a y_j 2^j}$. The global phase is irrelevant, and the other phase factors can be implemented by using b single-qubit phase gates, each acting as $|y_j\rangle \mapsto e^{2\pi i\frac{M}{3B}a y_j 2^j}|y_j\rangle$.

then we would get a gradient estimate where each individual coordinate has the required approximation quality with probability at least $\frac{2}{3}$. Equation (16) implies that if instead we use our approximate implementation of the phase oracle, $U^\dagger(I \otimes cP \otimes I)U$, then the outcome probability distribution changes by at most $\frac{1}{16}$ in total variation distance. So one run of Jordan's algorithm using this approximate phase oracle still outputs a vector $v \in \mathbb{R}^n$ such that

$$\Pr\left[\left|g_i - \frac{3B}{r}v_i\right| > \frac{8 \cdot 42\pi\delta}{r}\right] \leq \frac{1}{3} + \frac{1}{16} < \frac{2}{5} \text{ for every } i \in [n].$$

As in the proof of Corollary 15, repeating the whole procedure $\mathcal{O}\left(\log(\frac{n}{\rho})\right)$ times, and taking the median of the resulting vectors coordinatewise, gives a gradient approximator $\tilde{g}$ of the desired quality. The gate complexity analysis follows from [GAW19, Theorem 21], noting that each controlled phase operation cP can be implemented using $\mathcal{O}\left(\log\left(\frac{B}{\delta}\right)\right)$ single-qubit phase gates. □