Quantum Algorithms and Lower Bounds for Linear Regression with Norm Constraints

Yanlin Chen
QuSoft and CWI, Amsterdam, The Netherlands

Ronald de Wolf
QuSoft and CWI, Amsterdam, The Netherlands
University of Amsterdam, The Netherlands

Abstract

Lasso and Ridge are important minimization problems in machine learning and statistics. They are versions of linear regression with squared loss where the vector $\theta \in \mathbb{R}^d$ of coefficients is constrained in either ℓ_1-norm (for Lasso) or in ℓ_2-norm (for Ridge). We study the complexity of quantum algorithms for finding ε-minimizers for these minimization problems. We show that for Lasso we can get a quadratic quantum speedup in terms of d by speeding up the cost-per-iteration of the Frank-Wolfe algorithm, while for Ridge the best quantum algorithms are linear in d, as are the best classical algorithms. As a byproduct of our quantum lower bound for Lasso, we also prove the first classical lower bound for Lasso that is tight up to polylog-factors.

2012 ACM Subject Classification Mathematics of computing → Mathematical optimization; Theory of computation → Quantum computation theory

Keywords and phrases Quantum algorithms, Regularized linear regression, Lasso, Ridge, Lower bounds

Digital Object Identifier 10.4230/LIPIcs.ICALP.2023.38

Category Track A: Algorithms, Complexity and Games

Related Version *Full Version*: https://arxiv.org/abs/2110.13086

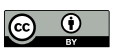
Funding *Ronald de Wolf*: Partially supported by the Dutch Research Council (NWO) through Gravitation-grant Quantum Software Consortium, 024.003.037, and through QuantERA ERA-NET Cofund project QuantAlgo 680-91-034.

Acknowledgements We thank Yi-Shan Wu and Christian Majenz for useful discussions, and Armando Bellante for pointing us to [11].

1 Introduction

1.1 Linear regression with norm constraints

One of the simplest, most useful and best-studied problems in machine learning and statistics is *linear regression*. We are given N data points $\{(x_i, y_i)\}_{i=0}^{N-1}$ where $x \in \mathbb{R}^d$ and $y \in \mathbb{R}$, and want to fit a line through these points that has small error. In other words, we want to find a vector $\theta \in \mathbb{R}^d$ of coefficients such that the inner product $\langle \theta, x \rangle = \sum_{j=1}^{d} \theta_j x_j$ is a good predictor for the y-variable. There are different ways to quantify the error ("loss") of such a θ-vector, the most common being the squared error $(\langle \theta, x \rangle - y)^2$, averaged over the N data points (or over an underlying distribution $\mathcal{D}$ that generated the data). If we let X be the $N \times d$ matrix whose N rows are the x-vectors of the data, then we want to find a $\theta \in \mathbb{R}^d$ that minimizes $\|X\theta - y\|_2^2$. This minimization problem has a well-known closed-form solution: $\theta = (X^T X)^+ X^T y$, where the superscript "+" indicates the Moore-Penrose pseudoinverse.



In practice, unconstrained least-squares regression sometimes has problems with overfitting and often yields solutions θ where all entries are non-zero, even when only a few of the d coordinates in the x-vector really matter and one would really hope for a sparse vector θ [42, see Chapters 2 and 13]. This may be improved by "regularizing" θ via additional constraints. The most common constraints are to require that the ℓ_1-norm or ℓ_2-norm of θ is at most some bound B.[1] Linear regression with an ℓ_1-constraint is called *Lasso* (due to Tibshirani [43]), while with an ℓ_2-constraint it is called *Ridge* (due to Hoerl and Kennard [29]).

Both Lasso and Ridge are widely used for robust regression and sparse estimation in ML problems and elsewhere [44, 15]. Consequently, there has been great interest in finding the fastest-possible algorithms for them. For reasons of efficiency, algorithms typically aim at finding not the exactly optimal solution but an ε-minimizer, i.e., a vector θ whose loss is only an additive ε worse than the minimal-achievable loss. The best known results on the time complexity of classical algorithms for Lasso are an upper bound of $\tilde{\mathcal{O}}(d/\varepsilon^2)$ [28] and a lower bound of $\Omega(d/\varepsilon)$ [16] (which we actually improve to a tight lower bound in this paper, see below); for Ridge the best bound is $\tilde{\Theta}(d/\varepsilon^2)$ [28], which is tight up to logarithmic factors.[2]

1.2 Our results

We focus on the *quantum* complexity of Lasso and Ridge, investigating to what extent quantum algorithms can solve these problems faster. Table 1 summarizes the results. The upper bounds are on time complexity (total number of elementary operations and queries to entries of the input vectors) while the lower bounds are on query complexity (which itself lower bounds time complexity).

Table 1 Classical and quantum upper and lower bounds for Lasso and Ridge.

	Upper bound	**Lower bound**
Lasso	Classical [28]: $\tilde{\mathcal{O}}(d/\varepsilon^2)$ Quantum [*this work*]: $\tilde{\mathcal{O}}(\sqrt{d}/\varepsilon^2)$	Classical [*this work*]: $\tilde{\Omega}(d/\varepsilon^2)$ Quantum [*this work*]: $\Omega(\sqrt{d}/\varepsilon^{1.5})$
Ridge	Classical [28]: $\tilde{\mathcal{O}}(d/\varepsilon^2)$	Classical [28]: $\Omega(d/\varepsilon^2)$ Quantum [*this work*]: $\Omega(d/\varepsilon)$

1.2.1 Lasso

We design a quantum algorithm that finds an ε-minimizer for Lasso in time $\tilde{\mathcal{O}}(\sqrt{d}/\varepsilon^2)$. This gives a quadratic quantum speedup over the best-possible classical algorithm in terms of d, while the ε-dependence remains the same as in the best known classical algorithm.

[1] For ease of presentation we will set $B = 1$. However, one can also set B differently or even do a binary search over its values, finding a good θ for each of those values and selecting the best one at the end. Instead of putting a hard upper bound B on the norm, one may also include it as a penalty term in the objective function itself, by just minimizing the function $\|X\theta - y\|_2^2 + \lambda \|\theta\|$, where λ is a Lagrange multiplier and the norm of θ could be ℓ_1 or ℓ_2 (and could also be squared). This amounts to basically the same thing as our setup.

[2] For such bounds involving additive error ε to be meaningful, one has to put certain normalization assumptions on X and y, which are given in the body of the paper. The $\tilde{\mathcal{O}}$ and $\tilde{\Theta}$-notation hides polylogarithmic factors. It is known that $N = \mathcal{O}((\log d)/\varepsilon^2)$ data points suffice for finding an ε-minimizer, which explains the absence of N as a separate variable in these bounds.

Our quantum algorithm is based on the Frank-Wolfe algorithm, a well-known iterative convex optimization method [22]. Frank-Wolfe, when applied to a Lasso instance, starts at the all-zero vector θ and updates this in $\mathcal{O}(1/\varepsilon)$ iterations to find an ε-minimizer. Each iteration looks at the gradient of the loss function at the current point θ and selects the best among $2d$ directions for changing θ (each of the d coordinates can change positively or negatively, whence $2d$ directions). The new θ will be a convex combination of the previous θ and this optimal direction of change. Note that Frank-Wolfe automatically generates *sparse* solutions: only one coordinate of θ can change from zero to nonzero in one iteration, so the number of nonzero entries in the final θ is at most the number of iterations, which is $\mathcal{O}(1/\varepsilon)$.

Our quantum version of Frank-Wolfe does not reduce the number of iterations, which remains $\mathcal{O}(1/\varepsilon)$, but it does reduce the cost per iteration. In each iteration it selects the best among the $2d$ possible directions for changing θ by using a version of quantum minimum-finding on top of a quantum approximation algorithm for entries of the gradient (which in turn uses amplitude estimation). Both this minimum-finding and our approximation of entries of the gradient will result in approximation errors throughout. Fortunately Frank-Wolfe is a very robust method which still converges if we carefully ensure those quantum-induced approximation errors are sufficiently small.

Our quantum algorithm assumes coherent quantum query access to the entries of the data points (x_i, y_i), as well as a relatively small QRAM (quantum-readable classical-writable classical memory). We use a variant of a QRAM data structure developed by Prakash and Kerenidis [37, 33], to store the nonzero entries of our current solution θ in such a way that we can (1) quickly generate θ as a quantum state, and (2) quickly incorporate the change of θ incurred by a Frank-Wolfe iteration.[3] Because our θ is $\mathcal{O}(1/\varepsilon)$-sparse throughout the algorithm, we only need $\tilde{\mathcal{O}}(1/\varepsilon)$ bits of QRAM.

We also prove a lower bound of $\Omega(\sqrt{d}/\varepsilon^{1.5})$ quantum queries for Lasso, showing that the d-dependence of our quantum algorithm is essentially optimal, while our ε-dependence might still be slightly improvable. Our lower bound strategy "hides" a subset of the columns of the data matrix X by letting those columns have slightly more $+1$s than -1, and observes that an approximate minimizer for Lasso allows us to recover this hidden set. We then use the composition property of the adversary lower bound [12] together with a worst-case to average-case reduction to obtain a quantum query lower bound for this hidden-set-finding problem, and hence for Lasso.

Somewhat surprisingly, no tight *classical* lower bound was known for Lasso prior to this work. To the best of our knowledge, the previous-best classical lower bound was $\Omega(d/\varepsilon)$, due to Cesa-Bianchi, Shalev-Shwartz, and Shamir [16]. As a byproduct of our quantum lower bound, we use the same set-hiding approach to prove for the first time the optimal (up to logarithmic factors) lower bound of $\tilde{\Omega}(d/\varepsilon^2)$ queries for classical algorithms for Lasso.

1.2.2 Ridge

What about Ridge? Because ℓ_2 is a more natural norm for quantum states than ℓ_1, one might hope that Ridge is more amenable to quantum speedup than Lasso. Unfortunately this turns out to be wrong: we prove a quantum lower bound of $\Omega(d/\varepsilon)$ queries for Ridge, using a similar strategy as for Lasso. This shows that the classical linear dependence of the runtime on d cannot be improved on a quantum computer. Whether the ε-dependence can be improved remains an open question.

[3] Each iteration will actually change all nonzero entries of θ because the new θ is a convex combination of the old θ and a vector with one nonzero entry. Our data structure keeps track of a global scalar, which saves us the cost of separately adjusting all nonzero entries of θ in the data structure in each iteration.

1.3 Related work

As already cited in Table 1, Hazan and Koren [28] obtained an optimal classical algorithm for Ridge, and the best known classical algorithm for Lasso. Cesa-Bianchi, Shalev-Shwartz, and Shamir [16] provided a non-optimal classical lower bound for Lasso, and their idea inspired us to hide a subset among the column of the data matrix and to use a Lasso solver to find that subset (our lower bound also benefited from the way composition of the adversary bound was used in [8]).

Du, Hsieh, Liu, You, and Tao [20] also showed a quantum upper bound for Lasso based on quantizing parts of Frank-Wolfe, though their running time $\tilde{\mathcal{O}}(N^{3/2}\sqrt{d})$ is substantially worse than ours. The main goal of their paper was to establish differential privacy, not so much to obtain the best-possible quantum speedup for Lasso. They also claim an $\Omega(\sqrt{d})$ lower bound for quantum algorithms for Lasso [20, Corollary 1], without explicit dependence on ε, but we do not fully understand their proof, which goes via a claimed equivalence with quantum SVMs. Bellante and Zanero [11] recently and independently used similar techniques as we use here for our Lasso upper bound (KP-trees and amplitude estimation) to give a polynomial quantum speedup for the classical matching-pursuit algorithm, which is a heuristic algorithm for the NP-hard problem of linear regression with a sparsity constraint, i.e., with an ℓ_0-regularizer.

Another quantum approach for solving (unregularized) least-squares linear regression is based on the linear-systems algorithm of Harrow, Hassidim, and Lloyd [27]. In this type of approach, the quantum algorithm very efficiently generates a solution vector θ *as a quantum state* $\frac{1}{\|\theta\|_2}\sum_i \theta_i |i\rangle$ (which is incomparable to our goal of returning θ as a classical vector). Chakraborty, Gilyén, and Jeffery [18] used the framework of block-encodings to achieve this. Subsequently Gilyén, Lloyd, and Tang [25] obtained a "dequantized" classical algorithm for (unregularized) least-squares linear regression assuming length square sampling access to the input data, which again is incomparable to our setup. The quantum algorithm was very recently improved with an ℓ_2-regularizer by Chakraborty, Morolia, and Peduri [19], thought still producing the final output as a quantum state rather than as a classical solution.

Norm-constrained linear regression is a special case of convex optimization. Quantum algorithms for various convex optimization problems have received much attention recently. For example, there has been a sequence of quantum algorithms for solving linear and semidefinite programs starting with Brandão and Svore [14, 5, 13, 6, 3]. There have also been some polynomial speedups for matrix scaling [7, 26] and for boosting in machine learning [9, 30], as well as some general speedups for converting membership oracles for a convex feasible set to separation oracles and optimization oracles [17, 4, 2]. On the other hand Garg, Kothari, Netrapalli, and Sherif [24] showed that the number of iterations for first-order algorithms for minimizing non-smooth convex functions cannot be significantly improved on a quantum computer; recently they generalized this result to higher-order algorithms [23]. Finally, there has also been work on quantum speedups for *non-convex* problems, for instance on escaping from saddle points [45].

2 Preliminaries

Throughout the paper, d will always be the dimension of the ambient space $\mathbb{R}^d$, and log without a base will be the binary logarithm. It will be convenient for us to index entries of vectors starting from 0, so the entries x_i of a d-dimensional vector x are indexed by $i \in \{0,\dots,d-1\} = \mathbb{Z}_d$. $\mathcal{U}_N = \mathcal{U}\{0,\dots,N-1\}$ is the discrete uniform distribution over integers $0,1,2,\dots,N-1$.

2.1 Computational model and quantum algorithms

Our computational model is a classical computer (a classical random-access machine) that can invoke a quantum computer as a subroutine. The input is stored in quantum-readable read-only memory (a QROM), whose bits can be queried. The classical computer can also write bits to a quantum-readable classical-writable classical memory (a QRAM). The classical computer can send a description of a quantum circuit to the quantum computer; the quantum computer runs the circuit (which may include queries to the input bits stored in QROM and to the bits stored by the computer itself in the QRAM), measures the full final state in the computational basis, and returns the measurement outcome to the classical computer. In this model, an algorithm has time complexity T if it uses at most T elementary classical operations and quantum gates, quantum queries to the input bits stored in QROM, and quantum queries to the QRAM. The query complexity of an algorithm only measures the number of queries to the input stored in QROM. We call a (quantum) algorithm *bounded-error* if (for every possible input) it returns a correct output with probability at least $9/10$.

We will represent real numbers in computer memory using a number of bits of precision that is polylogarithmic in d, N, and $1/\varepsilon$ (i.e., $\tilde{\mathcal{O}}(1)$ bits). This ensures all numbers are represented throughout our algorithms with negligible approximation error and we will ignore those errors later on for ease of presentation.

The following is a modified version of quantum minimum-finding, which in its basic form is due to Høyer and Dürr [21]. Our proof of the more general version below is given in our full version on arXiv, and is based on a result from [5]. We also use some other Grover-based quantum algorithms as subroutines, described in our full version.

▶ **Theorem 1** (min-finding with an approximate unitary). *Let $\delta_1, \delta_2, \varepsilon \in (0,1)$, $v_0, \ldots, v_{d-1} \in \mathbb{R}$. Suppose we have a unitary $\tilde{A}$ that maps $|j\rangle|0\rangle \to |j\rangle|\Lambda_j\rangle$ such that for every $j \in \mathbb{Z}_d$, after measuring the state $|\Lambda_j\rangle$, with probability $\geq 1-\delta_2$ the first register λ of the measurement outcome satisfies $|\lambda - v_j| \leq \varepsilon$. There exists a quantum algorithm that finds an index j such that $v_j \leq \min_{k\in\mathbb{Z}_d} v_k + 2\varepsilon$ with probability $\geq 1-\delta_1-1000\log(1/\delta_1)\cdot\sqrt{2d\delta_2}$, using $1000\sqrt{d}\cdot\log(1/\delta_1)$ applications of $\tilde{A}$ and $\tilde{A}^\dagger$, and $\tilde{\mathcal{O}}(\sqrt{d})$ elementary gates. In particular, if $\delta_2 \leq \delta_1^2/(2000000d\log(1/\delta_1))$, that finds such a j with probability $\geq 1-2\delta_1$.*

2.2 Expected and empirical loss

Let sample set $S = \{(x_i, y_i)\}_{i=0}^{N-1}$ be a set of i.i.d. samples from $\mathbb{R}^d \times \mathbb{R}$, drawn according to an unknown distribution $\mathcal{D}$. A *hypothesis* is a function $h : \mathbb{R}^d \to \mathbb{R}$, and $\mathcal{H}$ denotes a set of hypotheses. To measure the performance of the prediction, we use a convex loss function $\ell : \mathbb{R}^2 \to \mathbb{R}$. The *expected loss* of h with respect to $\mathcal{D}$ is denoted by $L_{\mathcal{D}}(h) = \mathbb{E}_{(x,y)\sim\mathcal{D}}[\ell(h(x), y)]$, and the *empirical loss* of h with respect to S is denoted by $L_S(h) = \frac{1}{N}\sum_{i\in\mathbb{Z}_N} \ell(h(x_i), y_i)$.

▶ **Definition 2.** *Let $\varepsilon > 0$. An $h \in \mathcal{H}$ is an ε-minimizer over $\mathcal{H}$ w.r.t. $\mathcal{D}$ if*

$$L_{\mathcal{D}}(h) - \min_{h'\in\mathcal{H}} L_{\mathcal{D}}(h') \leq \varepsilon.$$

▶ **Definition 3.** *Let $\varepsilon > 0$. An $h \in \mathcal{H}$ is an ε-minimizer over $\mathcal{H}$ w.r.t. sample set S if*

$$L_S(h) - \min_{h'\in\mathcal{H}} L_S(h') \leq \varepsilon.$$

2.3 Linear regression problems and their classical and quantum setup

In linear regression problems, the hypothesis class is the set of linear functions on $\mathbb{R}^d$. The goal is to find a vector θ for which the corresponding hypothesis $\langle \theta, x \rangle$ provides a good prediction of the target y. One of the most natural choices for regression problems is the squared loss

$$\ell(\hat{y}, y) = (\hat{y} - y)^2.$$

We can instantiate the expected and empirical losses as a function of θ using squared loss:

$$L_{\mathcal{D}}(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{D}}[\ell(\langle \theta, x \rangle, y)] = \mathbb{E}_{(x,y)\sim\mathcal{D}}[(\langle \theta, x \rangle - y)^2],$$

$$L_S(\theta) = \frac{1}{N} \sum_{i \in \mathbb{Z}_N} \ell(\langle \theta, x \rangle, y_i) = \frac{1}{N} \sum_{i \in \mathbb{Z}_N} (\langle \theta, x \rangle - y_i)^2.$$

We also write the empirical loss as $L_S(\theta) = \frac{1}{N}\|X\theta - y\|_2^2$, where matrix entry X_{ij} is the jth entry of the vector x_i, and y is the N-dimensional vector with entries y_i. As we will see below, if the instances in the sample set are chosen i.i.d. according to $\mathcal{D}$, and N is sufficiently large, then $L_S(\theta)$ and $L_{\mathcal{D}}(\theta)$ are typically close by the law of large numbers.

In the quantum case, we assume the sample set S is stored in a QROM, which we can access by means of queries to the oracles $O_X : |i\rangle |j\rangle |0\rangle \to |i\rangle |j\rangle |X_{ij}\rangle$ and $O_y : |i\rangle |0\rangle \to |i\rangle |y_i\rangle$.

2.3.1 Lasso

The *least absolute shrinkage and selection operator*, or *Lasso*, is a special case of linear regression with a norm constraint on the vector θ: it restricts solutions to the unit ℓ_1-ball, which we denote by B_1^d. For the purpose of normalization, we require that every sample (x, y) satisfies $\|x\|_\infty \leq 1$ and $|y| \leq 1$.[4] The goal is to find a $\theta \in B_1^d$ that (approximately) minimizes the expected loss. Since the expected loss is not directly accessible, we instead find an approximate minimizer of the empirical loss. Mohri, Rostamizadeh, and Talwalkar [34] showed that with high probability, an approximate minimizer for *empirical* loss is also a good approximate minimizer for *expected* loss.

▶ **Theorem 4** ([34], Theorem 11.16). *Let $\mathcal{D}$ be an unknown distribution over $[-1,1]^d \times [-1,1]$ and $S = \{(x_i, y_i)\}_{i=0}^{N-1}$ be a sample set containing N i.i.d. samples from $\mathcal{D}$. Then, for each $\delta > 0$, with probability $\geq 1 - \delta$ over the choice of S, the following holds for all $\theta \in B_1^d$:*

$$L_{\mathcal{D}}(\theta) - L_S(\theta) \leq 4\sqrt{\frac{2\log(2d)}{N}} + 4\sqrt{\frac{\log(1/\delta)}{2N}}.$$

This theorem implies that if $N = c\log(d/\delta)/\varepsilon^2$ for sufficiently large constant c, then finding (with error probability $\leq \delta$) an ε-minimizer for the *empirical* loss L_S, implies finding (with error probability $\leq 2\delta$ taken both over the randomness of the algorithm and the choice of the sample S) a 2ε-minimizer for the *expected* loss $L_{\mathcal{D}}$.

[4] Note that if $\theta \in B_1^d$ and $\|x\|_\infty \leq 1$, then $|\langle \theta, x \rangle| \leq 1$ by Hölder's inequality.

2.3.2 Ridge

Another special case of linear regression with a norm constraint is *Ridge*, which restricts solutions to the unit ℓ_2-ball B_2^d. For the purpose of normalization, we now require that every sample (x, y) satisfies $\|x\|_2 \leq 1$ and $|y| \leq 1$. Similarly to the Lasso case, Mohri, Rostamizadeh, and Talwalkar [34] showed that with high probability, an approximate minimizer for the empirical loss is also a good approximate minimizer for the expected loss.

▶ **Theorem 5** ([34], Theorem 11.11). *Let $\mathcal{D}$ be an unknown distribution over $B_2^d \times [-1, 1]$ and $S = \{(x_i, y_i)\}_{i=0}^{N-1}$ be a sample set containing N i.i.d. samples from $\mathcal{D}$. Then, for each $\delta > 0$, with probability $\geq 1 - \delta$ over the choice of S, the following holds for all $\theta \in B_2^d$:*

$$L_{\mathcal{D}}(\theta) - L_S(\theta) \leq 8\sqrt{\frac{1}{N}} + 4\sqrt{\frac{\log(1/\delta)}{2N}}.$$

2.4 The KP-tree data structure and efficient state preparation

Kerenidis and Prakash [37, 33] gave a quantum-accessible classical data structure to store a vector θ with support t (i.e., t nonzero entries) to enable efficient preparation of the state

$$|\theta\rangle = \sum_{j \in \mathbb{Z}_d} \sqrt{\frac{|\theta_j|}{\|\theta\|_1}} |j\rangle |sign(\theta_j)\rangle .$$

We modify their data structure such that for arbitrary $a, b \in \mathbb{R}$ and $j \in \mathbb{Z}_d$, we can efficiently update a data structure for the vector θ to a data structure for the vector $a\theta + be_j$, without having to individually update all nonzero entries of the vector. We only give the definition here; for more details and analysis, see our full version on arXiv.

▶ **Definition 6** (KP-tree). *Let $\theta \in \mathbb{R}^d$ have support t. Define a KP-tree KP_θ of θ as:*

- *KP_θ is a rooted binary tree with depth $\lceil \log d \rceil$ and with $\mathcal{O}(t \log d)$ vertices.*
- *The root stores a scalar $A \in \mathbb{R} \setminus \{0\}$ and the support t of θ.*
- *Each edge of the tree is labelled by a bit.*
- *For each $j \in supp(\theta)$, there is one corresponding leaf storing $\frac{\theta_j}{A}$. The number of leaves is t.*
- *The bits on the edges of the path from the root to the leaf corresponding to the j^{th} entry of θ, form the binary description of j.*
- *Each intermediate node stores the sum of its children's absolute values.*

For $\ell \in \mathbb{Z}_{\lceil \log d \rceil}$ and $j \in \mathbb{Z}_{2^\ell}$, we define $KP_\theta(\ell, j)$ as the value of the j^{th} node in the ℓ^{th} layer, i.e., the value stored in the node that we can reach by the path according to the binary representation of j from the root. Also, we let $KP_\theta(0, 0)$ be the sum of all absolute values stored in the leaves. If there is no corresponding j^{th} node in the ℓ^{th} layer (that is, we cannot reach a node by the path according to the binary representation of j from the root), then $KP_\theta(\ell, j)$ is defined as 0. Note that both the numbering of the layer and the numbering of nodes start from 0. In the special case where θ is the all-0 vector, the corresponding tree will just have a root node with $t = 0$.

3 Quantum Algorithm for Lasso

3.1 The classical Frank-Wolfe algorithm

Below is a description of the Frank-Wolfe algorithm with approximate linear solvers. For now this is for an arbitrary convex objective function L and arbitrary compact convex domain $\mathcal{X}$ of feasible solutions; for Lasso we will later instantiate these to the quadratic loss function and

ℓ_1-ball, respectively. Frank-Wolfe finds an ε-approximate solution to a convex optimization problem, using $O(1/\varepsilon)$ iterations. It is a first-order method: each iteration assumes access to the gradient of the objective function at the current point. The algorithm considers the linearization of the objective function, and moves towards a minimizer of this linear function without ever leaving the domain $\mathcal{X}$ (in contrast to for instance projected gradient descent).

Algorithm 1 The Frank-Wolfe algorithm with approximate linear subproblems.

input : number of iterations $T > 0$; convex differentiable function L; compact convex domain $\mathcal{X}$;
Let C_L be the curvature constant of L;
Let θ^0 be an arbitrary point in $\mathcal{X}$;
for $t \leftarrow 0$ **to** T **do**
  $\tau_t = \frac{2}{t+2}$;
  find $s \in \mathcal{X}$ such that $\langle s, \nabla L(\theta^t)\rangle \leq \min_{s' \in \mathcal{X}} \langle s', \nabla L(\theta^t)\rangle + \frac{\tau_t C_L}{4}$;
  $\theta^{t+1} = (1-\tau_t)\theta^t + \tau_t s$;
end
output : θ^T;

The convergence rate of the Frank-Wolfe algorithm is affected by the "non-linearity" of the objective function L, as measured by the curvature constant C_L:

▶ **Definition 7.** *The curvature constant C_L of a convex and differentiable function $L : \mathbb{R}^d \to \mathbb{R}$ with respect to a convex domain $\mathcal{X}$ is defined as*

$$C_L \equiv \sup_{\substack{x,s \in \mathcal{X}, \gamma \in [0,1], \\ y = x + \gamma(s-x)}} \frac{2}{\gamma^2}(L(y) - L(x) - \langle \nabla L(x), (y-x)\rangle).$$

Next we give an upper bound for the curvature constant of the empirical loss function for Lasso.

▶ **Theorem 8.** *Let $S = \{(x_i, y_i)\}_{i=0}^{N-1}$ with all entries of x_i and y_i in $[-1, 1]$. Then the curvature constant C_{L_S} of L_S w.r.t. B_1^d is ≤ 8.*

Proof. We know

$$L_S(\theta) = \frac{1}{N}\|X\theta - y\|_2^2 = \frac{(X\theta - y)^T(X\theta - y)}{N} = \frac{\theta^T X^T X\theta - y^T X\theta - \theta^T X^T y + y^T y}{N},$$

which implies the Hessian of L_S is $\nabla^2 L_S(z) = \frac{2X^T X}{N}$, independent of z. By replacing sup by max because the domain is compact, we have

$$\begin{aligned} C_{L_S} &= \max_{\substack{x,s \in \mathcal{X}, \gamma \in [0,1], \\ y = x + \gamma(s-x)}} \frac{2}{\gamma^2}(L_S(y) - L_S(x) - \langle \nabla L_S(x), (y-x)\rangle) \\ &= \max_{x,s \in \mathcal{X}, \gamma \in [0,1]} \langle (s-x), \nabla^2 L_S \cdot (s-x)\rangle = \max_{x,s \in \mathcal{X}} \frac{2}{N}\|X(s-x)\|_2^2. \end{aligned}$$

Each coefficient of X is at most 1 in absolute value, and $s - x \in 2B_1^d$, hence each entry of the vector $X(s-x)$ has magnitude at most 2. Therefore $\max_{x,y \in B_1^d} \frac{2}{N}\|X(s-x)\|_2^2$ is at most 8. ◀

The original Frank-Wolfe algorithm [22] assumed that the minimization to determine the direction-of-change s was done exactly, without the additive error term $\tau_t C_{L_S}/4$ that we wrote in Algorithm 1. However, the following theorem, due to Jaggi [31], shows that solving approximate linear subproblems is sufficient for the Frank-Wolfe algorithm to converge at an $O(C_{L_S}/T)$ rate, which means one can find an ε-approximate solution with $T = O(C_{L_S}/\varepsilon)$ iterations.

▶ **Theorem 9** ([31], Theorem 1). *For each iteration $t \geq 1$, the corresponding θ^t of Algorithm 1 satisfies*

$$L_S(\theta^t) - \min_{\theta' \in B_1^d} L_S(\theta') \leq \frac{3C_{L_S}}{t+2}.$$

3.2 Approximating the quadratic loss function and entries of its gradient

In this subsection, we give a quantum algorithm to estimate the quadratic loss function $L_S(\theta)$ and entries of its gradient, given query access to entries of the vectors in $S = \{(x_i, y_i)\}_{i=0}^{N-1}$ and given a KP-tree for $\theta \in B_1^d$. One can estimate these numbers with additive error β in time roughly $1/\beta$.

We start with estimating entries of the gradient of the loss function at a given θ:

▶ **Theorem 10.** *Let $\theta \in B_1^d$, and $\beta, \delta > 0$. Suppose we have a KP-tree KP_θ of vector θ and can make quantum queries to $O_{KP_\theta} : |\ell, k\rangle |0\rangle \to |\ell, k\rangle |KP_\theta(\ell, k)\rangle$. One can implement $\tilde{U}_{\nabla L_S} : |j\rangle |0\rangle \to |j\rangle |\Lambda\rangle$ such that for all $j \in \mathbb{Z}_d$, after measuring the state $|\Lambda\rangle$, with probability $\geq 1-\delta$ the first register λ of the outcome will satisfy $|\lambda - \nabla_j L_S(\theta)| \leq \beta$, by using $\tilde{\mathcal{O}}(\frac{\log(1/\delta)}{\beta})$ applications of O_X, $O_X^\dagger$, O_y, $O_y^\dagger$, O_{KP_θ}, $O_{KP_\theta}^\dagger$, and elementary gates.*

Next we show how to estimate the value of the loss function itself at a given θ:

▶ **Theorem 11.** *Let $\theta \in B_1^d$, and $\beta, \delta > 0$. Suppose we have a KP-tree KP_θ of vector θ and can make quantum queries to $O_{KP_\theta} : |\ell, k\rangle |0\rangle \to |\ell, k\rangle |KP_\theta(\ell, k)\rangle$. Then we can implement $\tilde{U}_{L_S} : |0\rangle \to |\Lambda\rangle$ such that after measuring the state $|\Lambda\rangle$, with probability $\geq 1-\delta$ the first register λ of the outcome will satisfy $|\lambda - L_S(\theta)| \leq \beta$, by using $\tilde{\mathcal{O}}(\frac{\log(1/\delta)}{\beta})$ applications of O_X, $O_X^\dagger$, O_y, $O_y^\dagger$, O_{KP_θ}, $O_{KP_\theta}^\dagger$, and elementary gates.*

If we have multiple vectors $\theta^0, \ldots, \theta^{m-1}$, then we can apply the previous theorem conditioned on the index of the vector we care about:

▶ **Corollary 12.** *Let $\theta^0, \theta^1, \ldots, \theta^{m-1} \in B_1^d$, and $\beta, \delta > 0$. Suppose for all $h \in \mathbb{Z}_m$, we have a KP-tree KP_{θ^h} of vector θ^h and can make quantum queries to $O_{KP_\theta} : |h, \ell, k\rangle |0\rangle \to |h, \ell, k\rangle |KP_{\theta^h}(\ell, k)\rangle$. Then we can implement $\tilde{U}_{L_S} : |h\rangle |0\rangle \to |h\rangle |\Lambda\rangle$ such that for all $h \in \mathbb{Z}_m$, after measuring the state $|\Lambda\rangle$, with probability $\geq 1-\delta$ the first register λ of the outcome will satisfy $|\lambda - L_S(\theta^h)| \leq \beta$, by using $\tilde{\mathcal{O}}(\frac{\log(1/\delta)}{\beta})$ applications of O_X, $O_X^\dagger$, O_y, $O_y^\dagger$, O_{KP_θ}, $O_{KP_\theta}^\dagger$, and elementary gates.*

3.3 Quantum algorithms for Lasso with respect to S

In this subsection, we will show how to find an approximate minimizer for Lasso with respect to a given sample set S. The following algorithm simply applies the Frank-Wolfe algorithm to find an ε-minimizer for Lasso with respect to the sample set S given C, a guess for the curvature constant C_{L_S} (which our algorithm does not know in advance). Note that to find an $s \in B_1^d$ such that $\langle s, \nabla L_S(\theta^t)\rangle \leq \min_{s' \in \mathcal{X}} \langle s', \nabla L_S(\theta^t)\rangle + \tau_t C_{L_S}/4$, it suffices to only check

$s \in \{\pm e_0, \ldots, \pm e_{d-1}\}$ because the domain is B_1^d and ∇L_S is a linear function in θ. Also, by Theorem 8, the curvature constant C_{L_S} of loss function L_S is at most 8 because (x_i, y_i) is in $[-1,1]^d \times [-1,1]$ for all $i \in \mathbb{Z}_N$.

Algorithm 2 The algorithm for Lasso with a guess C for the value of the curvature constant.

input : a positive value C; additive error ε;
Let θ^0 be the d-dimensional all-zero vector;
Let $T = 6 \cdot \lceil \frac{C}{\varepsilon} \rceil$;
for $t \leftarrow 0$ **to** T **do**
  $\tau_t = \frac{2}{t+2}$;
  Let $s \in \{\pm e_0, \ldots, \pm e_{d-1}\}$ be such that $\langle \nabla L_S(\theta^t), s \rangle \leq \min_{j' \in \mathbb{Z}_d} -|\nabla_{j'} L_S(\theta^t)| + \frac{C}{8t+16}$;
  $\theta^{t+1} = (1-\tau_t)\theta^t + \tau_t s$;
end
output : θ^T;

It is worth mentioning that Algorithm 2 also outputs an ε-minimizer if its input C equals the curvature constant C_{L_S} approximately instead of exactly. For example, suppose we only know that the curvature constant C_{L_S} is between C and $2C$, where C is the input in Algorithm 2. Then the output of Algorithm 2 is still an ε-minimizer. We can see this by first observing that the error we are allowed to make for the linear subproblem in iteration t is $\frac{C_{L_S}}{4t+8} \geq \frac{C}{8t+16}$, and hence by Theorem 9, after $T = 6 \cdot \lceil \frac{C}{\varepsilon} \rceil$ iterations, the output θ^T is a $\frac{3C}{(T+2)} = \frac{3C}{6 \cdot \lceil C/\varepsilon \rceil + 2}$-minimizer for L_S. Because $\frac{3C}{6 \cdot \lceil C/\varepsilon \rceil + 2} \leq \varepsilon$, the output θ^T is therefore an ε-minimizer.

In the Lasso case, we do not know how to find a positive number C such that $C_{L_S} \in [C, 2C]$, but we know $C_{L_S} \leq 8$ by Theorem 8. Hence we can try different intervals of possible values for C_{L_S}: we apply Algorithm 2 with different input $C = 8, 4, 2, 1, 1/2, \ldots, 2^{-\lceil \log(1/\varepsilon) \rceil}$, and then we collect all outputs of Algorithm 2 with those different inputs, as candidates. After that, we compute the objective values of all those candidates, and output the one with minimum objective value. If $C_{L_S} \in (\varepsilon, 8]$, then at least one of the values we tried for C will be within a factor of 2 of the actual curvature constant C_{L_S}. Hence one of our candidates is an ε-minimizer.

However, we also need to deal with the case that $C_{L_S} \leq \varepsilon$. In this case, we consider the "one-step" version of the Frank-Wolfe algorithm, where the number of iterations is 1. But now we do not estimate $\langle \nabla L_S(\theta^t), s \rangle$ anymore (i.e., we do not solve linear subproblems anymore). We find that the only possible directions are the vertices of the ℓ_1-ball, and θ^0 is the all-zero vector, implying that θ^1, the output of one-step Frank-Wolfe, must be in $I = \{\pm e_0/3, \ldots, \pm e_{d-1}/3\}$ by the update rule of Frank-Wolfe. Besides, $C_{L_S} \leq \varepsilon$ implies that θ^1 is a $\frac{3C_{L_S}}{1+2} \leq \varepsilon$-minimizer for Lasso. Hence we simply output a $v = \arg\min_{v' \in I} L_S(v')$ if $C_{L_S} \leq \varepsilon$.

Combining the above arguments gives the following algorithm:

▶ **Theorem 13.** *Let $S = \{(x_i, y_i)\}_{i=0}^{N-1}$ be the given sample set stored in QROM. For each $\varepsilon \in (0, 0.5)$, there exists a bounded-error quantum algorithm that finds an ε-minimizer for Lasso w.r.t. sample set S using $\tilde{\mathcal{O}}(\frac{\sqrt{d}}{\varepsilon^2})$ time and $\tilde{\mathcal{O}}(\frac{1}{\varepsilon})$ QRAM and classical space.*

Proof. We will implement Algorithm 3 in $\tilde{\mathcal{O}}(\frac{\sqrt{d}}{\varepsilon^2})$ time and $\tilde{\mathcal{O}}(\frac{1}{\varepsilon})$ QRAM space. Below we analyze its different components.

Algorithm 3 The algorithm for Lasso.

input : ε;
Let $v \in \{\pm e_0/3, \ldots, \pm e_{d-1}/3\}$ be such that $L_S(v) - \min_{j\in\mathbb{Z}_d} L_S(\pm e_j/3) \le \varepsilon/10$;
Let candidate set $A = \{v\}$;
for $C \leftarrow 8, 4, 2, 1, \frac{1}{2}, \ldots, 2^{-\lceil\log(1/\varepsilon)\rceil-1}$ **do**
  RUN Algorithm 2 with inputs C and $\varepsilon/10$;
  ADD the output of Algorithm 2 to A;
end
output : $\arg\min_{w\in A} L_S(w)$;

3.3.1 Analysis of Algorithm 2

We first show that we can implement Algorithm 2 in $\tilde{\mathcal{O}}(\frac{\sqrt{d}}{\varepsilon^2})$ time. Because $C_{L_S} \le 8$ (Theorem 8), the number of iterations for Algorithm 2 with input $C = C_{L_S}$ is at most $6 \cdot \lceil\frac{8}{\varepsilon}\rceil$. However, as we mentioned above, we don't know how large C_{L_S} is exactly, so we try all possible inputs (of Algorithm 2) in Algorithm 3. Note that for every input $C \in \{8, 4, 2, 1, \frac{1}{2}, \ldots, 2^{-\lceil\log(1/\varepsilon)\rceil-1}\}$ and for every number of iterations $t \in \{1, \ldots, 6 \cdot \lceil\frac{C}{\varepsilon}\rceil\}$, $\frac{C}{4t+8}$ is at least $\frac{\varepsilon}{10}$, so it suffices to ensure that in each iteration in each of our runs of Algorithm 2, the additive error for the approximate linear subproblem is $\le \frac{\varepsilon}{10}$.

Suppose we have KP_{θ^t} for each iteration t of Algorithm 2, and suppose we can make queries to $O_{KP_{\theta^t}}$, then by Theorem 10, one can implement $\tilde{U}_{\nabla L_S} : |j\rangle|0\rangle \to |j\rangle|\Lambda\rangle$ such that for all $j \in \mathbb{Z}_d$, after measuring the state $|\Lambda\rangle$, with probability $\ge 1 - \frac{\varepsilon^2}{2d\cdot 10^{20}\cdot\log^6(1/\varepsilon)}$ the first register λ of the measurement outcome will satisfy $|\lambda - \nabla_j L_S(\theta)| \le \frac{\varepsilon}{20}$, by using $\tilde{\mathcal{O}}(\frac{\log(d/\varepsilon)}{\varepsilon})$ time and queries to $O_{KP_{\theta^t}}$, $O^\dagger_{KP_{\theta^t}}$. Then by Theorem 1, with failure probability at most $\frac{\varepsilon}{10000\log(1/\varepsilon)}$, one can find $s \in \{\pm e_0, \ldots, \pm e_{d-1}\}$ such that $\langle\nabla L_S(\theta^t), s\rangle \le \min_{j'\in\mathbb{Z}_d} -|\nabla_{j'} L_S(\theta^t)| + 2\cdot\frac{\varepsilon}{20}$, by using $\tilde{\mathcal{O}}(\sqrt{d}\cdot\log(1/\varepsilon))$ applications of $\tilde{U}_{\nabla L_S}$ and $\tilde{U}^\dagger_{\nabla L_S}$, and $\tilde{\mathcal{O}}(\sqrt{d})$ elementary gates.

For each iteration t in Algorithm 2, we also maintain KP_{θ^t} and hence we can make quantum queries to $O_{KP_{\theta^t}}$. The cost for constructing KP_{θ^0} and the cost for updating KP_{θ^t} to $KP_{\theta^{t+1}}$ is $\tilde{\mathcal{O}}(1)$ for both time and space by (shown in our full version). Moreover, the total number of iterations T is at most $6\cdot\lceil\frac{8}{\varepsilon}\rceil$ in Algorithm 2 because $C_{L_S} \le 8$, and hence the space cost for maintaining KP_{θ^t} and implementing $O_{KP_{\theta^t}}$ is $\tilde{\mathcal{O}}(\frac{1}{\varepsilon})$ bits. Hence we can implement Algorithm 2 with failure probability at most $\lceil\frac{8}{\varepsilon}\rceil\cdot\frac{6\varepsilon}{10000\log(1/\varepsilon)}$ using $\tilde{\mathcal{O}}(\frac{\sqrt{d}}{\varepsilon^2})$ time and $\tilde{\mathcal{O}}(\frac{1}{\varepsilon})$ bits of QRAM and classical space.

3.3.2 Analysis of Algorithm 3

Now we show how to implement Algorithm 3 with failure probability at most 1/10 using $\tilde{\mathcal{O}}(\frac{\sqrt{d}}{\varepsilon^2})$ time. By Corollary 12, one can implement $\tilde{U}_{L_S} : |j\rangle|0\rangle \to |j\rangle|\Lambda\rangle$ such that for all $j \in \mathbb{Z}_d$, after measuring the state $|\Lambda\rangle$, with failure probability at most $\frac{1}{2d\cdot 10^{16}}$ the first register λ of the outcome will satisfy $|\lambda - L_S(e_j/3)| \le \varepsilon/20$ using $\tilde{\mathcal{O}}(\frac{1}{\varepsilon})$ time. Then by Theorem 1, with failure probability at most $0.0001 + 1000\cdot\log(1000)\sqrt{\frac{2d}{2d\cdot 10^{16}}} \le \frac{2}{1000}$ we can find $v \in \{\pm e_0/3, \ldots, \pm e_{d-1}/3\}$ such that $L_S(v) - \min_{j\in\mathbb{Z}_d} L_S(\pm e_j/3) \le 2\cdot\varepsilon/20 = \varepsilon/10$ by using $\tilde{\mathcal{O}}(\sqrt{d})$ applications of $\tilde{U}_{L_S}$ and $\tilde{U}^\dagger_{L_S}$ and $\tilde{\mathcal{O}}(\sqrt{d})$ elementary gates, hence $\tilde{\mathcal{O}}(\frac{\sqrt{d}}{\varepsilon})$ time.

Because Algorithm 3 runs Algorithm 2 $\lceil\log(1/\varepsilon)\rceil$ times and each run fails with probability at most $\lceil\frac{8}{\varepsilon}\rceil\cdot\frac{6\varepsilon}{10000\log(1/\varepsilon)}$, the candidate set A, with failure probability $\lceil\frac{8}{\varepsilon}\rceil\cdot\frac{6\varepsilon}{10000\log(1/\varepsilon)}\cdot\lceil\log(1/\varepsilon)\rceil + \frac{2}{1000} \le \frac{1}{20}$, contains an $\frac{\varepsilon}{10}$-minimizer. To output $\arg\min_{w\in A} L_S(w)$, we use

Theorem 11 to evaluate $L_S(w)$ for all $w \in A$ with additive error $\frac{\varepsilon}{10}$ with failure probability at most $\frac{1}{40\log(1/\varepsilon)}$, and hence we find an $\varepsilon/10$-minimizer among A with probability at least $1 - 1/20 - \lceil \log(1/\varepsilon) \rceil \cdot \frac{1}{40\log(1/\varepsilon)} \geq 0.9$. Because the candidate set A contains an $\frac{\varepsilon}{10}$-minimizer for Lasso, the $\frac{\varepsilon}{10}$-minimizer among A is therefore an ε-minimizer for Lasso. The QRAM and classical space cost for each run is at most $\tilde{\mathcal{O}}(\frac{1}{\varepsilon})$ because the space cost for Algorithm 2 is $\tilde{\mathcal{O}}(\frac{1}{\varepsilon})$. Hence the total cost for implementing Algorithm 3 is $\tilde{\mathcal{O}}(\frac{\sqrt{d}}{\varepsilon^2})$ time and $\tilde{\mathcal{O}}(\frac{1}{\varepsilon})$ bits of QRAM and classical space. ◀

3.4 Quantum algorithms for Lasso with respect to $\mathcal{D}$

In the previous subsection, we showed that we can find an ε-minimizer for Lasso with respect to sample set S. Here we show how we can find an ε-minimizer for Lasso with respect to distribution $\mathcal{D}$. First sample a set S of $N = \tilde{\mathcal{O}}((\log d)/\varepsilon^2)$ i.i.d. samples from $\mathcal{D}$, which is the input that will be stored in QROM, and then find an $\varepsilon/2$-minimizer for Lasso with respect to S by Theorem 13. By Theorem 4, with high probability, an $\varepsilon/2$-minimizer for Lasso with respect to S will be an ε-minimizer for Lasso with respect to distribution $\mathcal{D}$. Hence we obtain the following corollary:

▶ **Corollary 14.** *Let $S = \{(x_i, y_i)\}_{i=0}^{N-1}$ be the given sample set, sampled i.i.d. from $\mathcal{D}$. For arbitrary $\varepsilon > 0$, if $N = \tilde{\mathcal{O}}(\frac{\log d}{\varepsilon^2})$, then there exists a bounded-error quantum algorithm that finds an ε-minimizer for Lasso w.r.t. distribution $\mathcal{D}$ using $\tilde{\mathcal{O}}(\frac{\sqrt{d}}{\varepsilon^2})$ queries to O_X, O_y and elementary gates, and using $\tilde{\mathcal{O}}(\frac{1}{\varepsilon})$ space (QRAM and classical bits).*

In our full version on arXiv we show that we can also avoid the usage of QRAM in the above corollary with $\tilde{\mathcal{O}}(1/\varepsilon)$ extra overhead.

▶ **Corollary 15.** *Let $S = \{(x_i, y_i)\}_{i=0}^{N-1}$ be the given sample set, sampled i.i.d. from $\mathcal{D}$. For arbitrary $\varepsilon > 0$, if $N = \tilde{\mathcal{O}}(\frac{\log d}{\varepsilon^2})$, then there exists a bounded-error quantum algorithm that finds an ε-minimizer for Lasso w.r.t. $\mathcal{D}$ using $\tilde{\mathcal{O}}(\frac{\sqrt{d}}{\varepsilon^3})$ queries to O_X, O_y and elementary gates, and using $\tilde{\mathcal{O}}(\frac{1}{\varepsilon})$ classical bits.*

4 Quantum query lower bounds for Lasso

In this section we prove a quantum lower bound of $\Omega(\sqrt{d}/\varepsilon^{1.5})$ queries for Lasso. To show such a lower bound, we define a certain set-finding problem, and show how it can be solved by an algorithm for Lasso. After that, we show that the worst-case set-finding problem can be seen as the composition of two problems, which have query complexities $\Omega(\sqrt{d/\varepsilon})$ and $\Omega(1/\varepsilon)$, respectively. Then the composition property of the quantum adversary bound implies a $\Omega(\sqrt{d/\varepsilon} \cdot 1/\varepsilon) = \Omega(\sqrt{d}/\varepsilon^{1.5})$ query lower bound for Lasso.

4.1 Finding a hidden set W using a Lasso solver

Let $p \in (0, 1/2)$, $W \subset \mathbb{Z}_d$, and $\overline{W} = \mathbb{Z}_d \setminus W$. Define the distribution $\mathcal{D}_{p,W}$ over $(x, y) \in \{-1, 1\}^d \times \{-1, 1\}$ as follows. For each $j' \in \overline{W}$, $x_{j'}$ is generated according to $\Pr[x_{j'} = 1] = \Pr[x_{j'} = -1] = 1/2$, and for each $j \in W$, x_j is generated according to $\Pr[x_j = 1] = 1/2 + p$. And y is generated according to $\Pr[y = 1] = 1$. The goal of the distributional set-finding problem $\mathrm{DSF}_{\mathcal{D}_{p,W}}$ with respect to $\mathcal{D}_{p,W}$ is to output a set $\tilde{W}$ such that $|\tilde{W} \Delta W| \leq w/200$, given M samples from $\mathcal{D}_{p,W}$. One can think of the $M \times d$ matrix of samples as "hiding" the set W: the columns corresponding to $j \in W$ are likely to have more 1s than -1s, while the columns corresponding to $j \in \overline{W}$ have roughly as many 1s as -1s. A Lasso-solver can help us to find the hidden set W approximately. Precisely, algorithms that find an $\varepsilon/8000$-minimizer for Lasso with respect to $\mathcal{D}_{p,W}$ can also find a set $\tilde{W} \subset \mathbb{Z}_d$ such that $|W \Delta \tilde{W}| \leq w/200$.

▶ **Theorem 16.** *Let $\varepsilon \in (2/d, 1/100)$, w be eitehr $\lfloor 1/\varepsilon \rfloor$ or $\lfloor 1/\varepsilon \rfloor - 1$, $p = 1/(2\lfloor 1/\varepsilon \rfloor)$, and $W \subset \mathbb{Z}_d$ be a set of size w. Let θ be an $\varepsilon/8000$-minimizer for Lasso w.r.t. $\mathcal{D}_{p,W}$. Then the set $\tilde{W}$ that contains the indices of the entries of θ whose absolute value is $\geq \varepsilon/3$ satisfies $|W \Delta \tilde{W}| \leq w/200$.*

4.2 Worst-case quantum query lower bound for the set-finding problem

Here we will define the worst-case set-finding problem and then provide a quantum query lower bound for it. Before we step into the query lower bound for the worst-case set-finding problem, we have to introduce the lower bounds for the following problems first.

consider the *exact set-finding problem*: given input $x = x_0 \dots x_{d-1} \in \{0,1\}^d$ with at most w 1s, find the set W of all indices j with $x_j = 1$ (equivalently, learn x). To see the query lower bound for this problem, we consider the identity function where both domain and codomain are $\mathcal{Z} = \{z \in \{0,1\}^d : |z| = w\}$, and give a lower bound for computing this. If we can compute the identity function, then we can simply check the output string $x_0, x_1, \dots, x_{d-1}$ and collect all indices j with $x_j = 1$.

▶ **Theorem 17.** *Let w be an integer satisfying $0 < w \leq d/2$, $W \subset \mathbb{Z}_d$ with size w, and $x \in \{0,1\}^d$ such that $x_j = 1$ if $j \in W$ and $x_{j'} = 0$ if $j' \in \overline{W}$. Suppose we have query access to x. Then every quantum bounded-error algorithm to find W makes at least $\frac{1}{8}\sqrt{dw}$ queries.*

Using the same method, we give a lower bound for the *approximate* set-finding problem $\mathrm{ASF}_{d,w}$, which is to find a set $\tilde{W} \subset \mathbb{Z}_d$ such that $|W \Delta \tilde{W}| \leq w/200$. The intuition is that if we could find such a $\tilde{W}$ then we can "correct" it to W itself using a small number of Grover searches, so finding a good approximation $\tilde{W}$ is not much easier than finding W itself.

▶ **Theorem 18.** *Let w be an integer satisfying $0 < w \leq d/2$, $W \subset \mathbb{Z}_d$ with size w, and $x \in \{0,1\}^d$ such that $x_j = 1$ if $j \in W$ and $x_{j'} = 0$ if $j' \in \overline{W}$. Suppose we have query access to x. Then every bounded-error quantum algorithm that outputs $\tilde{W} \subset \mathbb{Z}_d$ satisfying $|W \Delta \tilde{W}| \leq w/200$ makes $\Omega(\sqrt{dw})$ queries.*

Next we consider the *Hamming-weight distinguisher problem* $\mathrm{HD}_{\ell,\ell'}$: given a $z \in \{0,1\}^N$ of Hamming weight ℓ or ℓ', distinguish these two cases. The adversary bound gives the following bound (a special case of Nayak and Wu [35] based on the polynomial method [10]).

▶ **Theorem 19.** *Let $N \in 2\mathbb{Z}_+$, $z \in \{0,1\}^N$, and $p \in (0, 0.5)$ be multiple of $1/N$. Suppose we have query access to z. Then every bounded-error quantum algorithm that computes $HD_{\frac{N}{2},N(\frac{1}{2}+p)}$ makes $\Omega(1/p)$ queries.*

The above theorem implies a lower bound of $\Omega(1/p)$ queries for $\mathrm{HD}_{\frac{N}{2},N(\frac{1}{2}+p)}$. One can also think of the input bits as ± 1 and in this case, the goal is to distinguish whether the entries add up to 0 or to $2pN$. For convenience, we abuse the notation $\mathrm{HD}_{\frac{N}{2},N(\frac{1}{2}+p)}$ also for the problem with ± 1 inputs. Now we are ready to prove a lower bound for the *worst-case set-finding problem* $\mathrm{WSF}_{d,w,p,N}$: given a matrix $X \in \{-1,1\}^{N \times d}$ where each column-sum is either $2pN$ or 0, the goal is to find a set $\tilde{W} \subset \mathbb{Z}_d$ such that $|\tilde{W} \Delta W| \leq w/200$, where W is the set of indices for those columns whose entries add up to $2pN$ and $w = |W|$. One can see that this problem is actually a composition of the approximate set-finding problem and the Hamming-weight distinguisher problem. Composing the relational problem $\mathrm{ASF}_{d,w}$ with d valid inputs of $\mathrm{HD}_{\frac{N}{2},N(\frac{1}{2}+p)}$, exactly w of which evaluate to 1, we can see that the d-bit string given by the values of $\mathrm{HD}_{\frac{N}{2},N(\frac{1}{2}+p)}$ on these d inputs, is a valid input for $\mathrm{ASF}_{d,w}$. In other words, the set of valid inputs for $\mathrm{WSF}_{d,w,p,N}$, or equivalently, the set of valid inputs for the composed problem $\mathrm{ASF}_{d,w} \circ (\mathrm{HD}_{\frac{N}{2},N(\frac{1}{2}+p)})^d$ is

$$\{(x^{(1)},\dots,x^{(d)}) \in \mathcal{P}^d : |\mathrm{HD}_{\frac{N}{2},N(\frac{1}{2}+p)}(x^{(1)}) \dots \mathrm{HD}_{\frac{N}{2},N(\frac{1}{2}+p)}(x^{(d)})| = w\},$$

where $\mathcal{P} = \{x \in \{0,1\}^N : |x| \in \{N/2, N/2 + pN\}\}$. The next theorem by Belovs and Lee shows that the quantum query complexity of the composed problem $\mathrm{ASF}_{d,w} \circ (\mathrm{HD}_{\frac{N}{2},\frac{N+2pN}{2}})^d$ is at least the product of the complexities of the two composing problems:

▶ **Theorem 20** ([12], Corollary 27). *Let $f \subseteq S \times T$, with $S \subseteq \{0,1\}^d$, be a relational problem with bounded-error quantum query complexity L. Assume that f is efficiently verifiable, that is given some $t \in T$ and oracle access to $x \in S$, there exists a bounded-error quantum algorithm that verifies whether $(x,t) \in f$ using $o(L)$ queries to x. Let $D \subseteq \{0,1\}^N$ and $g : D \to \{0,1\}$ be a Boolean function whose bounded-error quantum query complexity is Q. Then the bounded-error quantum query complexity of the relational problem $f \circ g^d$, restricted to inputs $x \in \{0,1\}^{dN}$ such that $g^d(x) \in S$, is $\Omega(LQ)$.*

Applying Theorem 20 with the lower bounds of Theorem 19 and Theorem 18, we obtain:

▶ **Corollary 21.** *Let $N \in 2\mathbb{Z}_+$ and $p \in (0, 0.5)$ be an integer multiple of $1/N$. Given a matrix $X \in \{-1,+1\}^{N\times d}$ such that there exists a set $W \subseteq \mathbb{Z}_d$ with size w and*

- *For every $j \in W$, $\sum_{i\in\mathbb{Z}_N} X_{ij} = 2pN$.*
- *For every $j' \in \overline{W}$, $\sum_{i\in\mathbb{Z}_N} X_{ij'} = 0$.*

Suppose we have query access to X. Then every bounded-error quantum algorithm that computes $\tilde{W}$ such that $|W \Delta \tilde{W}| \le w/200$, uses $\Omega(\sqrt{dw}/p)$ queries to O_X.

4.3 Worst-case to average-case reduction for the set-finding problem

Our goal is to prove a lower bound for Lasso algorithms that have high success probability w.r.t. the distribution $\mathcal{D}_{p,W}$, yet the lower bound of the previous subsection is for *worst-case* instances. In this subsection, we will connect these by providing a worst-case to average-case reduction for the set-finding problem. After that, by simply combining with the query lower bound for the worst-case set-finding problem and the reduction from the distributional set-finding problem to Lasso, we obtain an $\Omega(\sqrt{d}/\varepsilon^{1.5})$ query lower bound for Lasso.

▶ **Theorem 22.** *Let $N \in 2\mathbb{Z}_+$, $p \in (0, 0.5)$ be an integer multiple of $1/N$, w be a natural number between 2 to $d/2$, and M be a natural number. Suppose $X \in \{-1,+1\}^{N\times d}$ is a valid input for $\mathrm{WSF}_{d,w,p,N}$, and let $W \subset \mathbb{Z}_d$ be the set of the w indices of the columns of X whose entries add up to $2pN$. Let $R \in \mathbb{Z}_N^{M\times d}$ be a matrix whose entries are i.i.d. samples from $\mathcal{U}_N$, and define $X' \in \{-1,1\}^{M\times d}$ as $X'_{ij} = X_{R_{ij}j}$. Then the M vectors $(X'_i, 1)$, where X'_i is the ith row of X' and $i \in \mathbb{Z}_M$, are i.i.d. samples from $\mathcal{D}_{p,W}$.*

Proof. Every entry of R is a sample from $\mathcal{U}_N$, so $X_{R_{ij}j}$ is uniformly chosen from the entries of the jth column of X. Moreover, because every valid input W for $\mathrm{WSF}_{d,w,p,N}$ satisfies that for every $j \in W$, $\Pr_{i\sim\mathcal{U}_N}[X_{ij} = 1] = 1/2 + p$ and for every $j' \in \overline{W}$, $\Pr_{i\sim\mathcal{U}_N}[X_{ij'} = 1] = 1/2$, we know $(X'_i, 1)$ is distributed as $D_{p,W}$. ◀

The above theorem tells us that we can convert an instance of $\mathrm{WSF}_{d,w,p,N}$ to an instance of $\mathrm{DSF}_{\mathcal{D}_{p,W}}$. Note that we can produce matrix R offline and therefore we can construct the oracle $O_{X'} : |i\rangle|j\rangle|0\rangle \to |i\rangle|j\rangle|X_{R_{ij}j}\rangle$ using 1 query to $O_X : |i\rangle|j\rangle|0\rangle \to |i\rangle|j\rangle|X_{ij}\rangle$ (and some other elementary gates, which is irrelevant to the number of queries). Also observe that if $M = 10^{12} \cdot \lceil \log d \rceil \cdot \lfloor 1/\varepsilon \rfloor^2 = \mathcal{O}((\log d)/\varepsilon^2)$ and hence $S' = \{(X'_i, 1)\}_{i=0}^{M-1}$ is a sample

set with M i.i.d. samples from $\mathcal{D}_{p,W}$, then by Theorem 4, with probability $\geq 9/10$, an $\varepsilon/16000$-minimizer for Lasso with respect to S' is also an $\varepsilon/8000$-minimizer for Lasso with respect to distribution $\mathcal{D}_{p,W}$. By Theorem 16, an $\varepsilon/8000$-minimizer for Lasso with respect to distribution $\mathcal{D}_{p,W}$ can be used to output a set $\tilde{W} \subset \mathbb{Z}_d$ such that $|\tilde{W} \Delta W| \leq w/200$, where W is the set of indices for those columns of X whose entries add up to $2pN$. Hence we have a reduction from the worst-case set-finding problem to Lasso. By the reduction above and by plugging $w = \lfloor 1/\varepsilon \rfloor$ and $p = 1/(2\lfloor 1/\varepsilon \rfloor)$ in Corollary 21 (and N an arbitrary natural number such that $pN \in \mathbb{N}$), we obtain a lower bound of $\Omega(\sqrt{d}/\varepsilon^{1.5})$ queries for $\mathrm{WSF}_{d,w,p,N}$, and hence the main result of this section: a lower bound of $\Omega(\sqrt{d}/\varepsilon^{1.5})$ for Lasso.

▶ **Corollary 23.** *Let $\varepsilon \in (2/d, 1/100)$, $w = \lfloor 1/\varepsilon \rfloor$, $p = 1/(2\lfloor 1/\varepsilon \rfloor)$, and $W \subset \mathbb{Z}_d$ with size w. Every bounded-error quantum algorithm that computes an ε-minimizer for Lasso w.r.t. $\mathcal{D}_{p,W}$ uses $\Omega(\sqrt{d}/\varepsilon^{1.5})$ queries.*

4.4 Classical lower bound for Lasso

In the full version of this paper on arXiv we show how this quantum lower bound approach can be modified to prove, for the first time, a lower bound of $\tilde{\Omega}(d/\varepsilon^2)$ on the classical query complexity of Lasso. This lower bound is optimal up to logarithmic factors.

5 Quantum query lower bound for Ridge

Recall that Ridge's setup assumes the vectors in the sample set are normalized in ℓ_2 rather than ℓ_∞ as in Lasso. We modify the distribution to $\mathcal{D}'_{p,W}$ over $(x, y) \in \{-1/\sqrt{d}, 1/\sqrt{d}\}^d \times \{-1, 1\}$ as follows. Let $p \in (0, 1/4)$, $W \subset \mathbb{Z}_d$, and $\overline{W} = \mathbb{Z}_d \setminus W$. For each $j' \in \overline{W}$, $x_{j'}$ is generated according to $\Pr[x_{j'} = -1/\sqrt{d}] = 1/2 + p$; for each $j \in W$, x_j is generated according to $\Pr[x_j = 1/\sqrt{d}] = 1/2 + p$; y is generated according to $\Pr[y = 1] = 1$. Now again we want to solve a distributional set-finding problem with respect to $\mathcal{D}'_{p,W}$, given M samples from $\mathcal{D}'_{p,W}$. Similar to the Lasso case, one can think of the $M \times d$ matrix of samples as "hiding" the set W: the columns corresponding to $j \in W$ are likely to have more $1/\sqrt{d}$'s than $-1/\sqrt{d}$'s, while the columns corresponding to $j \in \overline{W}$ are likely to have more $-1/\sqrt{d}$'s than $1/\sqrt{d}$'s.

In this section let $\theta^* = \sum_{j \in \mathbb{Z}_d} \frac{e_j}{\sqrt{d}} (-1)^{[j \in \overline{W}]}$ and note that for every $\theta \in \mathbb{R}^d$,

$$\begin{aligned}
L_{\mathcal{D}'_{p,W}}(\theta) =& \mathbb{E}_{(x,y)\sim\mathcal{D}'_{p,W}}[\langle \theta, x\rangle^2] - 2\mathbb{E}_{(x,y)\sim\mathcal{D}'_{p,W}}[\langle \theta, x\rangle] + 1 \\
=& (\mathbb{E}_{(x,y)\sim\mathcal{D}'_{p,W}}[\langle \theta, x\rangle^2] - \mathbb{E}_{(x,y)\sim\mathcal{D}'_{p,W}}[\langle \theta, x\rangle]^2) \\
&+ \mathbb{E}_{(x,y)\sim\mathcal{D}'_{p,W}}[\langle \theta, x\rangle]^2 - 2\mathbb{E}_{(x,y)\sim\mathcal{D}'_{p,W}}[\langle \theta, x\rangle] + 1 \\
=& \|\theta\|_2^2 \cdot (1 - 4p^2)/d + (\mathbb{E}_{(x,y)\sim\mathcal{D}'_{p,W}}[\langle \theta, x\rangle] - 1)^2 \\
=& \|\theta\|_2^2 \cdot (1 - 4p^2)/d + (2p\langle \theta, \theta^*\rangle - 1)^2,
\end{aligned}$$

where the third equality holds because $\langle \theta, x\rangle$ is a sum of independent random variables and hence its variance is the sum of the variances of the terms $\theta_i x_i$ (which are $\theta_i^2(1 - 4p^2)/d$).

Next we show that θ^* is the minimizer for Ridge with respect to $\mathcal{D}'_{p,W}$.

▶ **Theorem 24.** *Let $w = \lfloor d/2 \rfloor$ and $W \subset \mathbb{Z}_d$ be a set of size w, and let $\varepsilon \in (1000/d, 1/10000)$ and $p = 1/\lfloor 1/\varepsilon \rfloor$. Then $\theta^* = \sum_{j \in \mathbb{Z}_d} \frac{e_j}{\sqrt{d}} (-1)^{[j \in \overline{W}]}$ is the minimizer for Ridge w.r.t. $\mathcal{D}'_{p,W}$.*

Proof. Let $\theta = \sum_{j\in\mathbb{Z}_d} \theta_j e_j \in B_2^d$ be a minimizer. We want to show $\theta_j = \theta_j^*$ for every $j \in \mathbb{Z}_d$. Note that if $\theta_j \cdot (-1)^{[j\in\overline{W}]} < 0$, then we can flip the sign of θ_j to get a smaller objective value, that is,

$$\begin{aligned} L_{\mathcal{D}'_{p,W}}(\theta') - L_{\mathcal{D}'_{p,W}}(\theta) &= (\|\theta'\|_2^2 - \|\theta\|_2^2)\cdot(1-4p^2)/d + (2p\langle\theta',\theta^*\rangle - 1)^2 - (2p\langle\theta,\theta^*\rangle - 1)^2 \\ &= (2p\langle\theta'-\theta,\theta^*\rangle)(2p\langle\theta'+\theta,\theta^*\rangle - 2) \\ &= (-4p\theta_j\cdot(-1)^{[j\in\overline{W}]})(2p\langle\theta'+\theta,\theta^*\rangle - 2) < 0, \end{aligned}$$

where $\theta' = \sum_{k\in\mathbb{Z}_d\setminus\{j\}} \theta_k e_k - \theta_j e_j$, and the last inequality is because $-4p\theta_j\cdot(-1)^{[j\in\overline{W}]} > 0$ and $2p\langle\theta'+\theta,\theta^*\rangle \le 2p\|\theta'+\theta\|_2\cdot\|\theta^*\|_2 \le 4p \le 1$. Since θ was assumed a minimizer, for all $j\in\mathbb{Z}_d$ the sign of θ_j must be $(-1)^{[j\in\overline{W}]}$.

Second, we show that we must have $|\theta_0| = |\theta_1| = \cdots = |\theta_{d-1}|$. Suppose, towards a contradiction, that this is not the case. Consider $\theta' = \sum_{j\in\mathbb{Z}_d} ue_j\cdot(-1)^{[j\in\overline{W}]}$, where $u = \sqrt{\sum_{j\in\mathbb{Z}_d}|\theta_j|^2/d}$. We have

$$\begin{aligned} L_{\mathcal{D}'_{p,W}}(\theta') - L_{\mathcal{D}'_{p,W}}(\theta) &= (2p\langle\theta'-\theta,\theta^*\rangle)(2p\langle\theta'+\theta,\theta^*\rangle - 2) \\ &= (2p/\sqrt{d})\cdot(du - \sum_{j\in\mathbb{Z}_d}|\theta_j|)\cdot(2p\langle\theta'+\theta,\theta^*\rangle - 2) < 0. \end{aligned}$$

The last inequality holds because again $2p\langle\theta'+\theta,\theta^*\rangle \le 4p \le 1$ and in addition,

$$d\cdot\sum_{j\in\mathbb{Z}_d}|\theta_j|^2 > (\sum_{j\in\mathbb{Z}_d}|\theta_j|)^2$$

by the Cauchy–Schwarz inequality (which is strict if the $|\theta_j|$ are not all equal). Hence if θ is indeed a minimizer, then its entries must all have the same magnitude.

Now we know a minimizer θ must be in the same direction as θ^*, we just don't know yet that the magnitudes of its entries are $1/\sqrt{d}$. Suppose $\|\theta\|_2 = u \le 1$ and $\theta = u\cdot\theta^*$, then

$$L_{\mathcal{D}'_{p,W}}(\theta) = \|\theta\|_2^2\cdot(1-4p^2)/d + (2p\langle\theta,\theta^*\rangle - 1)^2 = (u^2(1-4p^2)/d + (2pu-1)^2).$$

The discriminant of $f(u) = u^2(1-4p^2)/d + (2pu-1)^2$ is less than 0, and $u = \frac{2p}{4p^2+(1-4p^2)/d}$ is the global minimizer of $f(u)$. Note that $u = \frac{2p}{4p^2+(1-4p^2)/d} > 1$, and hence $f(1) \le f(u)$ for every $u \le 1$. Therefore we know θ^* is the minimizer for Ridge with respect to $\mathcal{D}'_{p,W}$. ◀

Next we show that the inner product between the minimizer and an approximate minimizer for Ridge will be close to 1.

▶ **Theorem 25.** *Let $w = \lfloor d/2\rfloor$, $W \subset \mathbb{Z}_d$ be a set of size w, $\varepsilon \in (1000/d, 1/10000)$, and $p = 1/\lfloor 1/\varepsilon\rfloor$. Suppose $\theta \in B_2^d$ is an $\varepsilon/1000$-minimizer for Ridge w.r.t. $\mathcal{D}'_{p,W}$. Then $\langle\theta,\theta^*\rangle \ge 0.999$.*

Proof. Because θ is an $\varepsilon/1000$-minimizer, we have

$$\begin{aligned} &0.001\varepsilon \ge L_{\mathcal{D}'_{p,W}}(\theta) - L_{\mathcal{D}'_{p,W}}(\theta^*) = (1-4p^2)\cdot(\|\theta\|_2^2 - 1)/d + (2p\langle\theta,\theta^*\rangle - 1)^2 - (2p-1)^2 \\ &\implies 2p\langle\theta,\theta^*\rangle \ge 1 - \sqrt{1-4p+4p^2+0.001\varepsilon - (1-4p^2)\cdot(\|\theta\|_2^2-1)/d}. \end{aligned}$$

Letting $z = 4p - 4p^2 - 0.001\varepsilon + (1-4p^2)\cdot(\|\theta\|_2^2 - 1)/d$, we have

$$\begin{aligned} 2p\langle\theta,\theta^*\rangle \geq& 1-\sqrt{1-z} \geq 1-(1-z/2) = z/2 \\ =& 2p - 2p^2 + (1-4p^2)\cdot(\|\theta\|_2^2-1)/d - 0.001\varepsilon, \end{aligned}$$

where the second inequality holds because $z \in (0,1)$. Dividing both sides by $2p$, we have

$$\langle\theta,\theta^*\rangle \geq 1 - p + (1-4p^2)\cdot(\|\theta\|_2^2-1)/(2pd) - 0.0005\varepsilon/p.$$

Because $\theta \in B_2^d$, $p = 1/\lfloor 1/\varepsilon \rfloor$, and $\varepsilon \in (1000/d, 1/10000)$, we get $\langle\theta,\theta^*\rangle \geq 0.999$. ◀

Combining the above theorem with the following theorem, we can see how to relate the entries of an approximate minimizer for Ridge with respect to $\mathcal{D}'_{p,W}$ to the elements of the hidden set W.

▶ **Theorem 26.** *Suppose $\theta \in B_2^d$ satisfies $\langle\theta,\theta^*\rangle \geq 1 - 0.001$. Then $\#\{j \in \mathbb{Z}_d \mid \theta_j \cdot \theta^*_j \leq 0\} \leq d/500$.*

Proof. If $\theta_j \cdot \theta^*_j \leq 0$ then $|\theta_j - \theta^*_j| \geq |\theta^*_j| = \frac{1}{\sqrt{d}}$, hence using Theorem 25 we have

$$\begin{aligned} \frac{1}{d}\#\{j \in \mathbb{Z}_d \mid \theta_j\cdot\theta^*_j \leq 0\} \leq \|\theta-\theta^*\|_2^2 &= \|\theta\|_2^2 + \|\theta^*\|_2^2 - 2\langle\theta,\theta^*\rangle \\ &\leq 2 - 2(1-0.001) = 1/500. \end{aligned}$$

◀

We know $\theta^* = \sum_{j\in\mathbb{Z}_d} \frac{e_j}{\sqrt{d}}(-1)^{[j\in\overline{W}]}$, so by looking at the signs of entries of θ, we can find an index set $\tilde{W} = \{j \in \mathbb{Z}_d : \theta_j > 0\}$ satisfying that $|W\Delta\tilde{W}| \leq d/500 \leq w/200$ because $w = \lfloor d/2 \rfloor$. Therefore, once we have an $\varepsilon/1000$-minimizer for Ridge with respect to $\mathcal{D}'_{p,W}$, we can solve $\mathrm{DSF}_{\mathcal{D}'_{p,W}}$.

With the reduction from $\mathrm{DSF}_{\mathcal{D}'_{p,W}}$ to Ridge, we here show (similar to Lasso) a lower bound for the *worst-case symmetric set-finding problem* $\mathrm{WSSF}_{d,w,p,N}$: given a matrix $X \in \{-1/\sqrt{d}, 1/\sqrt{d}\}^{N\times d}$ where each column-sum is either $2pN/\sqrt{d}$ or $-2pN/\sqrt{d}$, the goal is to find a set $\tilde{W} \subset \mathbb{Z}_d$ such that $|\tilde{W}\Delta W| \leq w/200$, where W is the set of indices for those columns whose entries add up to $2pN/\sqrt{d}$ and $w = |W|$. This problem is again a composition of the approximate set finding problem in Section 4.2 and the Hamming-weight distinguisher problem $\mathrm{HD}_{\ell,\ell'}$ with $\ell = \frac{N}{2} - pN$ and $\ell' = \frac{N}{2} + pN$ up to a scalar $1/\sqrt{d}$. Following the proof of Theorem 19, we prove a lower bound of $\Omega(1/p)$ queries for this problem.

▶ **Theorem 27.** *Let $N \in 2\mathbb{Z}_+$, $z \in \{0,1\}^N$, and $p \in (0, 0.5)$ be an integer multiple of $1/N$. Suppose we have query access to z. Then every bounded-error quantum algorithm that computes $HD_{\frac{N}{2}-pN,\frac{N}{2}+pN}$ makes $\Omega(1/p)$ queries.*

Again we think of the input bits as ± 1 and abuse the notation $\mathrm{HD}_{\frac{N}{2}-pN,\frac{N}{2}+pN}$ for the problem with ± 1 input. Also, by the composition property of the adversary bound from Belovs and Lee [12] (Theorem 20), we have a lower bound of $\Omega(\sqrt{dw}/p)$ for $\mathrm{WSSF}_{d,w,p,N}$ from the $\Omega(\sqrt{dw})$ lower bound for $\mathrm{ASF}_{d,w}$ and the $\Omega(1/p)$ lower bound for $\mathrm{HD}_{\frac{N}{2}-pN,\frac{N}{2}+pN}$.

▶ **Corollary 28.** *Let $N \in 2\mathbb{Z}_+$ and $p \in (0, 0.5)$ be an integer multiple of $1/N$. Given a matrix $X \in \{-1/\sqrt{d}, +1/\sqrt{d}\}^{N\times d}$ such that there exists a set $W \subseteq \mathbb{Z}_d$ with size w and*

- *For every $j \in W$, $\sum_{i\in\mathbb{Z}_N} X_{ij} = 2pN/\sqrt{d}$.*
- *For every $j' \in \overline{W}$, $\sum_{i\in\mathbb{Z}_N} X_{ij'} = -2pN/\sqrt{d}$.*

Then every bounded-error quantum algorithm that computes $\tilde{W}$ such that $|W\Delta\tilde{W}| \leq w/200$, takes $\Omega(\sqrt{dw}/p)$ queries.

The final step for proving a lower bound for Ridge, using the same arguments as in Section 4.3, is to provide a worst-case to average-case reduction for the symmetric set-finding problem. We follow the same proof in Theorem 22 and immediately get the following theorem:

▶ **Theorem 29.** *Let $N \in 2\mathbb{Z}_+$, $p \in (0, 0.5)$ be an integer multiple of $1/N$, w be a natural number between 2 to $d/2$, and M be a natural number. Suppose $X \in \{-1/\sqrt{d}, +1/\sqrt{d}\}^{N\times d}$ is a valid input for $WSSF_{d,w,p,N}$, and let $W \subset \mathbb{Z}_d$ be the set of the w indices of the columns of X whose entries add up to $2pN/\sqrt{d}$. Let $R \in \mathbb{Z}_N^{M\times d}$ be a matrix whose entries are i.i.d. samples from $\mathcal{U}_N$, and define $X' \in \{-1/\sqrt{d}, 1/\sqrt{d}\}^{M\times d}$ as $X'_{ij} = X_{R_{ij}j}$. Then the vectors $(X'_i, 1)$, where X'_i is the ith row of X' and $i \in \mathbb{Z}_M$, are i.i.d. samples from $\mathcal{D}'_{p,W}$.*

By setting $M = 10^{10} \cdot \lceil \log d \rceil \cdot \lfloor 1/\varepsilon \rfloor^2 = \mathcal{O}((\log d)/\varepsilon^2)$ and letting $S' = \{(X'_i, 1)\}_{i=0}^{M-1}$ be a sample set with M i.i.d. samples from $\mathcal{D}'_{p,W}$, with probability $\geq 9/10$, an $\varepsilon/2000$-minimizer for Ridge with respect to S' is also an $\varepsilon/1000$-minimizer for Ridge with respect to distribution $\mathcal{D}'_{p,W}$ from Theorem 5. By Theorem 26 and Theorem 25, an $\varepsilon/1000$-minimizer for Ridge with respect to distribution $\mathcal{D}'_{p,W}$ gives us a set $\tilde{W} \subset \mathbb{Z}_d$ such that $|\tilde{W} \Delta W| \leq w/200$, where W is the set of indices for those columns of X whose entries add up to $2pN/\sqrt{d}$. Hence we have a reduction from the worst-case symmetric set-finding problem to Ridge. By this reduction and by plugging $w = \lfloor d/2 \rfloor$ and $p = 1/\lfloor 1/\varepsilon \rfloor$ in Corollary 28 (and N an arbitrary natural number such that $pN \in \mathbb{N}$), we obtain a lower bound of $\Omega(d/\varepsilon)$ queries for $\mathrm{WSSF}_{d,w,p,N}$, and hence for Ridge as well, which is the main result of this section.

▶ **Corollary 30.** *Let $\varepsilon \in (2/d, 1/1000)$, $w = \lfloor d/2 \rfloor$, $p = 1/\lfloor 1/\varepsilon \rfloor$, and $W \subset \mathbb{Z}_d$ with size w. Every bounded-error quantum algorithm that computes an ε-minimizer for Ridge w.r.t. $\mathcal{D}'_{p,W}$ uses $\Omega(d/\varepsilon)$ queries.*

6 Future work

We mention a few directions for future work:

- While the d-dependence of our quantum bounds for Lasso is essentially optimal, the ε-dependence is not: upper bound $\sqrt{d}/\varepsilon^2$ vs lower bound $\sqrt{d}/\varepsilon^{1.5}$. Can we shave off a $1/\sqrt{\varepsilon}$ factor from our upper bound, maybe using a version of accelerated gradient descent [36] with $O(1/\sqrt{\varepsilon})$ iterations instead of Frank-Wolfe's $O(1/\varepsilon)$ iterations? Or can we somehow improve our lower bound by embedding harder query problems into Lasso?
- Similar question for Ridge: the linear d-dependence of our quantum bounds is tight, but we should improve the ε-dependence of our upper and/or lower bounds. The most interesting outcome would be a quantum algorithm for Ridge with better ε-dependence than the optimal classical complexity of $\tilde{\Theta}(d/\varepsilon^2)$; currently we do not know of any quantum speedup for Ridge.
- Can we speed up some other methods for (smooth) convex optimization? In particular, can we find a classical iterative method where quantum algorithms can significantly reduce the number of iterations, rather than just the cost per iteration as we did here?
- There are many connections between Lasso and Support Vector Machines [32], and there are recent quantum algorithms for optimizing SVMs [38, 41, 39, 1, 40]. We would like to understand this connection better.

References

1 Jonathan Allcock and Chang-Yu Hsieh. A quantum extension of SVM-perf for training nonlinear SVMs in almost linear time. *Quantum*, 4:342, 2020. arXiv:2006.10299.

2 Joran van Apeldoorn. *A quantum view on convex optimization.* PhD thesis, Universiteit van Amsterdam, 2020.

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

4 Joran van Apeldoorn, András Gilyén, Sander Gribling, and Ronald de Wolf. Convex optimization using quantum oracles. *Quantum*, 4:220, 2020. arXiv:1809.00643.

5 Joran van Apeldoorn, András Gilyén, Sander Gribling, and Ronald de Wolf. Quantum SDP-solvers: better upper and lower bounds. *Quantum*, 4:230, 2020. Earlier version in FOCS'17. arXiv:1705.01843.

6 Joran van Apeldoorn and András Gilyén. Improvements in quantum SDP-solving with applications. In *Proceedings of 46th International Colloquium on Automata, Languages, and Programming*, volume 132 of *Leibniz International Proceedings in Informatics*, pages 99:1–99:15, 2019. arXiv:1804.05058.

7 Joran van Apeldoorn, Sander Gribling, Yinan Li, Harold Nieuwboer, Michael Walter, and Ronald de Wolf. Quantum algorithms for matrix scaling and matrix balancing. In *Proceedings of 48th International Colloquium on Automata, Languages, and Programming*, volume 198 of *Leibniz International Proceedings in Informatics*, pages 110:1–17, 2021. arXiv:2011.12823.

8 Simon Apers and Ronald de Wolf. Quantum speedup for graph sparsification, cut approximation and Laplacian solving. In *Proceedings of 61st IEEE Annual Symposium on Foundations of Computer Science*, pages 637–648, 2020. arXiv:1911.07306.

9 Srinivasan Arunachalam and Reevu Maity. Quantum boosting. In *Proceedings of 37th International Conference on Machine Learning (ICML'20)*, 2020. arXiv:2002.05056.

10 Robert Beals, Harry Buhrman, Richard Cleve, Michele Mosca, and Ronald de Wolf. Quantum lower bounds by polynomials. *Journal of the ACM*, 48(4):778–797, 2001. Earlier version in FOCS'98. quant-ph/9802049.

11 Armando Bellante and Stefano Zanero. Quantum matching pursuit: A quantum algorithm for sparse representations. *Physical Review A*, 105:022414, 2022.

12 Aleksandrs Belovs and Troy Lee. The quantum query complexity of composition with a relation, 2020. arXiv:2004.06439.

13 Fernando Brandão, Amir Kalev, Tongyang Li, Cedric Yen-Yu Lin, Krysta Svore, and Xiaodi Wu. Quantum SDP solvers: Large speed-ups, optimality, and applications to quantum learning. In *Proceedings of 46th International Colloquium on Automata, Languages, and Programming*, volume 132 of *Leibniz International Proceedings in Informatics*, pages 27:1–27:14, 2019. arXiv:1710.02581.

14 Fernando Brandão and Krysta Svore. Quantum speed-ups for solving semidefinite programs. In *Proceedings of 58th IEEE Annual Symposium on Foundations of Computer Science, FOCS*, pages 415–426, 2017. arXiv:1609.05537.

15 Peter Bühlmann and Sara van de Geer. *Statistics for High-Dimensional Data: Methods, Theory and Applications*. Springer, 2011.

16 Nicolò Cesa-Bianchi, Shai Shalev-Shwartz, and Ohad Shamir. Efficient learning with partially observed attributes. *Journal of Machine Learning Research*, 12:2857–2878, 2011. arXiv:1004.4421.

17 Shouvanik Chakrabarti, Andrew Childs, Tongyang Li, and Xiaodi Wu. Quantum algorithms and lower bounds for convex optimization. *Quantum*, 4:221, 2020. arXiv:1809.01731.

18 Shantanav Chakraborty, András Gilyén, and Stacey Jeffery. The power of block-encoded matrix powers: improved regression techniques via faster Hamiltonian simulation. In *Proceedings of 46th International Colloquium on Automata, Languages, and Programming*, volume 132 of *Leibniz International Proceedings in Informatics*, pages 33:1–33:14, 2019. arXiv:1804.01973.

19 Shantanav Chakraborty, Aditya Morolia, and Anurudh Peduri. Quantum regularized least squares, 2022. arXiv:2206.13143.

20 Yuxuan Du, Min-Hsiu Hsieh, Tongliang Liu, Shan You, and Dacheng Tao. Quantum differentially private sparse regression learning, 2020. arXiv:2007.11921.

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

22 Marguerite Frank and Philip Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3(1-2):95–110, 1956.

23 Ankit Garg, Robin Kothari, Praneeth Netrapalli, and Suhail Sherif. Near-optimal lower bounds for convex optimization for all orders of smoothness. In *Proceedings of 35th Conference on Neural Information Processing Systems*, 2021.

24 Ankit Garg, Robin Kothari, Praneeth Netrapalli, and Suhail Sherif. No quantum speedup over gradient descent for non-smooth convex optimization. In *Proceedings of 12th Innovations in Theoretical Computer Science Conference*, volume 185 of *Leibniz International Proceedings in Informatics*, pages 53:1–53:20, 2021. arXiv:2010.01801.

25 András Gilyén, Seth Lloyd, and Ewin Tang. Quantum-inspired low-rank stochastic regression with logarithmic dependence on the dimension, 2018. arXiv:1811.04909.

26 Sander Gribling and Harold Nieuwboer. Improved quantum lower and upper bounds for matrix scaling. In *Proceedings of 39th International Symposium on Theoretical Aspects of Computer Science (STACS 2022)*, volume 219 of *Leibniz International Proceedings in Informatics*, pages 35:1–35:23, 2022. arXiv:2109.15282.

27 Aram Harrow, Avinatan Hassidim, and Seth Lloyd. Quantum algorithm for solving linear systems of equations. *Physical Review Letters*, 103(15):150502, 2009. arXiv:0811.3171.

28 Elad Hazan and Tomer Koren. Linear regression with limited observation. In *Proceedings of the 29th International Conference on Machine Learning*, 2012. arXiv:1206.4678 . More extensive version at arXiv:1108.4559.

29 Arthur Hoerl and Robert Kennard. Ridge regression: biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

30 Adam Izdebski and Ronald de Wolf. Improved quantum boosting, 2020. arXiv:2009.08360.

31 Martin Jaggi. Revisiting Frank-Wolfe: Projection-free sparse convex optimization. In *Proceedings of the 30th International Conference on Machine Learning*, volume 28, pages 427–435, 2013.

32 Martin Jaggi. An equivalence between the Lasso and Support Vector Machines. In Johan Suykens, Marco Signoretto, and Andreas Argyriou, editors, *Regularization, Optimization, Kernels, and Support Vector Machines*. CRC Press, 2014. arXiv:1303.1152.

33 Iordanis Kerenidis and Anupam Prakash. Quantum recommendation systems. In *Proceedings of 8th Innovations in Theoretical Computer Science Conference*, volume 67 of *Leibniz International Proceedings in Informatics*, pages 49:1–49:21, 2017. arXiv:1603.08675.

34 Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of Machine Learning*. Adaptive Computation and Machine Learning series. MIT Press, second edition, 2018.

35 Ashwin Nayak and Felix Wu. The quantum query complexity of approximating the median and related statistics. In *Proceedings of the 31st Annual ACM Symposium on Theory of Computing*, pages 384–393. ACM, 1999. arXiv:quant-ph/9804066.

36 Yurii Nesterov. A method for solving the convex programming problem with convergence rate $\mathcal{O}(1/k^2)$. *Proceedings of the USSR Academy of Sciences*, 269:543–547, 1983.

37 Anupam Prakash. *Quantum Algorithms for Linear Algebra and Machine Learning*. PhD thesis, University of California, Berkeley, 2014.

38 Patrick Rebentrost, Masoud Mohseni, and Seth Lloyd. Quantum support vector machine for big data classification. *Physical Review Letters*, 113(13):130503, 2014. arXiv:1307.0471.

39 Seyran Saeedi and Tom Arodz. Quantum sparse support vector machines, 2019. arXiv:1902.01879.

40 Seyran Saeedi, Aliakbar Panahi, and Tom Arodz. Quantum semi-supervised kernel learning. *Quantum Machine Intelligence*, 3:24, 2021.

41 Maria Schuld and Nathan Killoran. Quantum machine learning in feature Hilbert spaces. *Physical Review Letters*, 122(13):040504, 2019. arXiv:1803.07128.

42 Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning - From Theory to Algorithms*. Cambridge University Press, 2014.

43 Robert Tibshirani. Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society*, 58:267–288, 1996.

44 Hrishikesh Vinod. A survey of Ridge regression and related techniques for improvements over ordinary least squares. *The Review of Economics and Statistics*, 60(1):121–131, 1978.

45 Chenyi Zhang, Jiaqi Leng, and Tongyang Li. Quantum algorithms for escaping from saddle points. *Quantum*, 5:229, 2021. `arXiv:2007.10253`.