Quantum Regularized Least Squares

Shantanav Chakraborty[1,2], Aditya Morolia[1,3], and Anurudh Peduri[4,1,2]

[1]Center for Quantum Science and Technology, IIIT Hyderabad, Telangana 500032, India
[2]Center for Security, Theory and Algorithmic Research, IIIT Hyderabad, Telangana 500032, India
[3]Center for Computational Natural Sciences and Bioinformatics, IIIT Hyderabad, Telangana 500032, India
[4]Faculty of Computer Science, Ruhr University Bochum, 44801 Bochum, Germany

Linear regression is a widely used technique to fit linear models and finds widespread applications across different areas such as machine learning and statistics. In most real-world scenarios, however, linear regression problems are often *ill-posed* or the underlying model suffers from *overfitting*, leading to erroneous or trivial solutions. This is often dealt with by adding extra constraints, known as regularization. In this paper, we use the frameworks of block-encoding and quantum singular value transformation (QSVT) to design the first quantum algorithms for quantum least squares with general ℓ_2-regularization. These include regularized versions of quantum ordinary least squares, quantum weighted least squares, and quantum generalized least squares. Our quantum algorithms substantially improve upon prior results on *quantum ridge regression* (polynomial improvement in the condition number and an exponential improvement in accuracy), which is a particular case of our result.

To this end, we assume approximate block-encodings of the underlying matrices as input and use robust QSVT algorithms for various linear algebra operations. In particular, we develop a variable-time quantum algorithm for matrix inversion using QSVT, where we use quantum singular value discrimination as a subroutine instead of gapped phase estimation. This ensures that substantially fewer ancilla qubits are required for this procedure than prior results. Owing to the generality of the block-encoding framework, our algorithms are applicable to a variety of input models and can also be seen as improved and generalized versions of prior results on standard (non-regularized) quantum least squares algorithms.

Shantanav Chakraborty: shchakra@iiit.ac.in
Aditya Morolia: aditya.morolia@research.iiit.ac.in
Anurudh Peduri: anurudh.peduri@rub.de

Contents

1 Introduction

The problem of fitting a theoretical model to a large set of experimental data appears across various fields ranging from the natural sciences to machine learning and statistics [Mur12]. Linear regression is one of the most widely used procedures for achieving this. By assuming that, for the underlying model, there exists a linear relationship between a dependent variable and one or more explanatory variables, linear regression constructs the best linear fit to the series of data points. Usually, it does so while minimizing the sum of squared errors - known as the least squares method.

In other words, suppose that we are given N data points $\{(a_i, b_i)\}_{i=1}^N$ where $\forall i : a_i \in \mathbb{R}^d, \forall i : b_i \in \mathbb{R}$. The assumption is that each b_i is linearly dependent on a_i up to some random noise of mean 0. Suppose A is the data matrix of dimension $N \times d$, such that its i^{th} row is the vector a_i and $b \in \mathbb{R}^N$ such that $b = (b_1, \cdots, b_N)^T$. Then the procedure, known as ordinary least squares, obtains a vector $x \in \mathbb{R}^d$ that minimizes the objective function $||Ax - b||_2^2$. This problem has a closed-form solution given by $x = (A^T A)^{-1} A^T b = A^+ b$, where A^+ denotes the Moore-Penrose inverse of the matrix A. Thus computationally, finding the best fit by linear regression reduces to finding the pseudoinverse of a matrix that represents the data, a task that is expensive for classical machines for large data sets.

In practice, however, least squares regression runs into problems such as *overfitting*. For instance, the solution might fit most data points, even those corresponding to random noise. Furthermore, the linear regression problem may also be *ill-posed*, for instance, when the number of variables exceeds the number of data points rendering it impossible to fit the data. These issues come up frequently with linear regression models and result in erroneous or trivial solutions. Furthermore, another frequent occurrence is that the data matrix A has linearly dependent columns. In this scenario, the matrix $A^T A$ is not full rank and therefore is not invertible.

Regularization is a widely used technique to remedy these problems, not just for linear regression but for inverse problems, in general [EHN96]. In the context of linear regression, broadly, this involves adding a penalty term to the objective function, which constrains the solution of the regression problem. For instance, in the case of ℓ_2-regularization, the objective is to obtain x that minimizes

$$\|Ax - b\|_2^2 + \lambda \|Lx\|_2^2 \tag{1}$$

where L is an appropriately chosen penalty matrix (or regularization matrix) of dimension $N \times d$ and $\lambda > 0$ is the regularization parameter, an appropriately chosen constant. This regularization technique is known as *general ℓ_2-regularization* or *Tikhonov regularization* in the literature [Hem75, HH93, Bis95, GHO99, vW15]. It is a generalization of *ridge regression* which corresponds to the case when L is the identity matrix [HK00, Mar70, Vin78]. The closed-form solution of the general ℓ_2-regularized ordinary least squares problem is given by

$$x = \left(A^T A + \lambda L^T L\right)^{-1} A^T b. \tag{2}$$

A straightforward observation is that even when $A^T A$ is singular, a judicious choice of the penalty matrix L can ensure that the effective condition number (ratio of the maximum and the minimum non-zero singular values) of the overall matrix is finite and $A^T A + \lambda L^T L$ is invertible.

In this paper, we develop quantum algorithms for linear regression with general ℓ_2-regularization. If the optimal solution is $x = (x_1, \cdots, x_d)^T$, then our quantum algorithm outputs a quantum state that is δ-close to $|x\rangle = \sum_{j=1}^d x_j |j\rangle / \|x\|$, assuming access to the matrices A, L, and the quantum state $|b\rangle$ via general quantum input models.

In several practical scenarios, depending on the underlying theoretical model, generalizations of the ordinary least squares (OLS) technique are more useful to fit the data. For instance, certain samples may be of more importance (and therefore have more weight) than the others, in which case weighted least squares (WLS) is preferred. Generalized least squares (GLS) is used when the underlying samples obtained are correlated. These techniques also suffer from the issues commonplace with OLS, warranting the need for regularization [vW15]. Consequently, we also design algorithms for quantum WLS with general ℓ_2-regularization and quantum GLS with general ℓ_2-regularization.

Organization of the paper: In the remainder of Section 1, we formally describe ℓ_2-regularized versions of OLS, WLS, and GLS (Section 1.1), discuss prior and related work (Section 1.2), and outline our contributions and results (Section 1.3). In Section 2, we briefly outline the framework of block-encoding and quantum input models that are particular instances of it (Section 2.2). We also briefly introduce quantum singular value transformation (QSVT) (Section 2.3) and variable time amplitude amplification (VTAA) (Section 2.4). Following this, in Section 3, we develop several algorithmic primitives involving arithmetic of block-encodings (Section 3.1), quantum singular value discrimination (Section 3.2) and quantum linear algebra using QSVT (Section 3.3). These are the technical building blocks for designing our quantum regularized regression algorithms. Using these algorithmic primitives, we design quantum algorithms for the quantum least squares with ℓ_2-regularization in Section 4. Finally, we conclude by discussing some possible future research directions in Section 5.

1.1 Linear regression with ℓ_2-regularization

Suppose we are given data points $\{(a_i, b_i)\}_{i=1}^N$, where $\forall i : a_i \in \mathbb{R}^d, \forall i : b_i \in \mathbb{R}$ such that $(a_i, b_i) \sim_{i.i.d} \mathcal{D}$, i.e. they are sampled i.i.d. from some unknown distribution $\mathcal{D}$, assumed to be linear. We want to find a vector $x \in \mathbb{R}^d$ such that the inner product $x^T a_j$ is a good predictor for the target b_j for some unknown a_j. This can be done by minimizing the total squared loss over the given data points,

$$\mathcal{L}_O := \sum_j (x^T a_j - b_j)^2, \tag{3}$$

leading to the ordinary least squares (OLS) optimization problem. The task then is to find $x \in \mathbb{R}^d$ that minimizes $\|Ax - b\|_2^2$, where A is the $N \times d$ data matrix such that the i^{th} row of A is a_i, and the i^{th} element of the vector b is b_i. Assuming that $A^T A$ is non-singular, the optimal x satisfies

$$x = (A^T A)^{-1} A^T b = A^+ b, \tag{4}$$

which corresponds to solving a linear system of equations.

Suppose that out of the samples present in the data, we have higher confidence in some of them than others. In such a scenario, the i^{th} observation can be assigned a weight $w_i \in \mathbb{R}$. This leads to a generalization of the OLS problem to *weighted least squares* (WLS). In order to obtain the best linear fit, the task is now to minimize the weighted version of the loss

$$\mathcal{L}_W := \sum_j w_j (x^T a_j - b_j)^2. \tag{5}$$

As before, assuming $A^T W A$ is non-singular, the above loss function has the following closed-form solution:

$$x = (A^T W A)^{-1} A^T W b, \tag{6}$$

where W is a diagonal matrix with w_i being the i^{th} diagonal element.

There can arise scenarios where there exists some correlation between any two samples. For *generalized least squares* (GLS), the presumed correlations between pairs of samples are given in a symmetric, non-singular covariance matrix Ω. This objective is to find the vector x that minimizes

$$\mathcal{L}_\Omega := \sum_{i,j} (\Omega^{-1})_{i,j} (x^T a_i - b_i)(x^T a_j - b_j). \tag{7}$$

Similarly, the closed-form solution for GLS is given by

$$x = (A^T \Omega^{-1} A)^{-1} A^T \Omega^{-1} b. \tag{8}$$

As mentioned previously, in several practical scenarios, the linear regression problem may be *ill-posed* or suffer from *overfitting*. Furthermore, the data may be such that some of the columns of the matrix A are linearly dependent. This shrinks the rank of A, and consequently of the matrix $A^T A$, rendering it singular and, therefore non-invertible. Recall that the closed-form solution of OLS exists only if $A^T A$ is non-singular, which is no longer the case. Such scenarios arise even for WLS and GLS problems [vW15].

In such cases, one resorts to *regularization* to deal with them. Let $\mathcal{L}$ be the loss function to be minimized for the underlying least squares problem (such as OLS, WLS, or GLS). Then general ℓ_2-regularization (*Tikhonov regularization*) involves an additional penalty term so that the objective now is to find the vector $x \in \mathbb{R}^d$ that minimizes

$$\mathcal{L} + \lambda \|Lx\|_2^2. \tag{9}$$

Here λ, known as the regularization parameter, is a positive constant that controls the size of the vector x, while L is known as the penalty matrix (or regularization matrix) that defines a (semi)norm on the solution through which the size is measured. The solution to the Tikhonov regularization problem also has a closed-form solution. For example, in the OLS problem, when $\mathcal{L} = \mathcal{L}_O$, we have that

$$x = (A^T A + \lambda L^T L)^{-1} A^T b. \tag{10}$$

It is worth noting that when $L = I$, the ℓ_2-regularized OLS problem is known as *ridge regression*. For the unregularized OLS problem, the singular values of A, σ_j are mapped to $1/\sigma_j$. The penalty term due to ℓ_2-regularization, results in a shrinkage of the singular values. This implies that even in the scenario where A has linearly dependent columns (some $\sigma_j = 0$) and $(A^T A)^{-1}$ does not exist, the inverse $(A^T A + \lambda L^T L)^{-1}$ is well defined for $\lambda > 0$ and any positive-definite L. Throughout this article, we refer to such an L (which is positive definite) as a *good regularizer*. The penalty matrix L allows for penalizing each regression parameter differently and leads to joint shrinkage among the elements of x. It also determines the rate and direction of shrinkage. In the special case of ridge regression, as $L = I$, the penalty shrinks each element of x equally along the unit vectors e_j. Also note that by definition, I is a *good regularizer*.

Closed-form expressions can also be obtained for the WLS and the GLS problem ($\mathcal{L} = \mathcal{L}_W, \mathcal{L}_\Omega$ respectively), and finding the optimal solution x reduces to solving a linear system. The quantum version of these algorithms output a quantum state that is ϵ-close $|x\rangle = \sum_j x_j |j\rangle / \|x\|$.

Throughout this work, while designing our quantum algorithms, we shall assume access (via a block-encoding) to the matrices A, W, Ω, and L and knowledge of the parameter λ. Classically, however, the regularization matrix L and the optimal parameter λ are obtained via several heuristic techniques [HH93, GHO99, vW15].

1.2 Prior work

Quantum algorithms for (unregularized) linear regression was first developed by Wiebe et al. [WBL12], wherein the authors made use of the HHL algorithm for solving a linear system of equations [HHL09]. Their algorithm assumes query access to a sparse matrix A (*sparse-access-model*) and to a procedure to prepare $|b\rangle = \sum_i b_i |i\rangle$. They first prepare a quantum state proportional to $A^T |b\rangle$, and then use the HHL algorithm to apply the operator $(A^T A)^{-1}$ to it. Overall the algorithm runs in a time scaling as κ_A^6 (the condition number of A) and inverse polynomial in the accuracy δ. Subsequent results have considered the problem of obtaining classical outputs for linear regression. For instance, in Ref. [Wan17], A^+ is directly applied to the quantum state $|b\rangle$, followed by amplitude estimation to obtain the entries of x. On the other hand, Ref. [SSP16] used the techniques of quantum principal component analysis in [LMR14] to predict a new data point for the regression problem. These algorithms also work in the *sparse access model* and run in a time that scales as poly $(\kappa, 1/\delta)$. Kerenidis and Prakash [KP20] provided a quantum algorithm for the WLS problem wherein they used a classical data structure to store the entries of A and W. Furthermore, they assumed QRAM access to this data structure [Pra14, KP17] that would allow the preparation of quantum states proportional to the entries of A and W efficiently. They showed that in this input model (*quantum data structure model*), an iterative quantum linear systems algorithm can prepare $|x\rangle$ in time $\widetilde{O}(\mu\kappa^3/\delta)$, where κ is the condition number of the matrix $A^T\sqrt{W}$ while $\mu = \left\|\sqrt{W}A\right\|_F$. Chakraborty et al. [CGJ19] applied the framework of block-encoding along with (controlled) Hamiltonian simulation of Low and Chuang [LC19] to design improved quantum algorithms for solving linear systems. Quantum algorithms developed in the block-encoding framework are applicable to a wide variety of input models, including the sparse access model and the quantum data structure model of [KP20]. They applied their quantum linear systems solver to develop quantum algorithms for quantum weighted least squares and generalized least squares. Their quantum algorithm for WLS has a complexity that is in $\widetilde{O}\left(\alpha\kappa\text{polylog}(Nd/\delta)\right)$, where $\alpha = s$, the sparsity of the matrix $A^T\sqrt{W}$ in the sparse access model while $\alpha = \left\|\sqrt{W}A\right\|_F$, for the quantum data structure input model. For GLS, their quantum algorithm outputs $|x\rangle$ in cost $\widetilde{O}\left(\kappa_A\kappa_\Omega(\alpha_A + \alpha_\Omega\kappa_\Omega)\text{polylog}(1/\delta)\right)$, where κ_A and κ_Ω are the condition numbers of A and Ω respectively while α_A and α_Ω are parameters that depend on how the matrices A and Ω are accessed in the underlying input model.

While quantum linear regression algorithms have been designed and subsequently improved over the years, quantum algorithms for regularized least squares have not been developed extensively. Yu et al. [YGW21] developed a quantum algorithm for *ridge regression* in the sparse access model using the LMR scheme [LMR14] for Hamiltonian simulation and quantum phase estimation, which they then used to determine the optimal value of the parameter λ. Their algorithm to output $|x\rangle$ has a cubic dependence on both κ and $1/\delta$. They use this as a subroutine to determine a good value of λ. A few other works [SX20, CYGL22] have considered the quantum ridge regression problem in the sparse access model, all of which can be implemented with poly$(\kappa, 1/\delta)$ cost.

Recently, Chen and de Wolf designed quantum algorithms for lasso (ℓ_1-regularization) and ridge regressions from the perspective of empirical loss minimization [CdW21]. For both lasso and ridge, their quantum algorithms output a classical vector $\widetilde{x}$ whose loss (mean squared error) is δ-close to the minimum achievable loss. In this context, they prove a quantum lower bound of $\Omega(d/\delta)$ for ridge regression which indicates that in their setting, the dependence on d cannot be improved on a quantum computer (the classical lower bound is also linear in d and there exists a matching upper bound). Note that $\widetilde{x}$ is not

necessarily close to the optimal solution x of the corresponding least squares problem, even though their respective loss values are. Moreover, their result (of outputting a classical vector $\widetilde{x}$) is incomparable to our objective of obtaining a quantum state encoding the optimal solution to the regularized regression problem.

Finally, Gilyén et al. obtained a "dequantized" classical algorithm for ridge regression assuming norm squared access to input data similar to the quantum data structure input model [GST22]. Furthermore, similar to the quantum setting where the output is the quantum state $|x\rangle = \sum_j x_j |j\rangle / \|x\|$ instead of x itself, their algorithm obtains samples from the distribution $x_j^2/\|x\|^2$. For the regularization parameter $\lambda = O\left(\|A\|\|A\|_F\right)$, the running time of their algorithm is in $\widetilde{O}\left(\kappa^{12} r_A^3/\delta^4\right)$, where r_A is the rank of A. Their result (and several prior results) does not have a polynomial dependence on the dimension of A and therefore rules out the possibility of generic exponential quantum speedup (except in δ) in the quantum data structure input model.

1.3 Our contributions

In this work, we design the first quantum algorithms for OLS, WLS, and GLS with general ℓ_2-regularization. We use the Quantum Singular Value Transformation (QSVT) framework introduced by Gilyén et al [GSLW19]. We assume that the relevant matrices are provided as input in the *block-encoding* model, in which access to an input matrix A is given by a unitary U_A whose top-left block is (close to) A/α. The parameter α takes specific values depending on the underlying input model. QSVT then allows us to implement nearly arbitrary polynomial transformations to a block of a unitary matrix using a series of parameterized, projector-controlled rotations (quantum signal processing [LC17b]).

More precisely, given approximate block-encodings of the data matrix A and the regularizing matrix L, and a unitary procedure to prepare the state $|b\rangle$, our quantum algorithms output a quantum state that is δ-close to $|x\rangle$, the quantum state proportional to the ℓ_2-regularized ordinary least squares (or weighted least squares or generalized least squares problem). We briefly summarize the query complexities of our results in Table 1.

For the OLS problem with general ℓ_2-regularization (Section 4.2, Theorem 32), we design a quantum algorithm which given an $(\alpha_A, a_A, \varepsilon_A)$-block-encoding of A (implemented in cost T_A), an $(\alpha_L, a_L, \varepsilon_L)$-block-encoding of L (implemented in cost T_L), a parameter $\lambda > 0$, and a procedure to prepare $|b\rangle$ (in cost T_b), outputs a quantum state which is δ-close to $|x\rangle$. The algorithm has a cost

$$\mathcal{O}\left(\kappa \log \kappa \left(\left(\frac{\alpha_A + \sqrt{\lambda}\alpha_L}{\|A\| + \sqrt{\lambda}\|L\|}\right) \log\left(\frac{\kappa}{\delta}\right)(T_A + T_L) + T_b\right)\right)$$

where κ can be thought of as a *modified condition number*, related to the effective condition numbers of A and L. When L is a good regularizer, this is given by the expression

$$\kappa = \kappa_L \left(1 + \frac{\|A\|}{\sqrt{\lambda}\|L\|}\right),$$

Notice that κ is independent of κ_A, the condition number of the data matrix A, which underscores the advantage of regularization. The parameters α_A and α_L take specific values depending on the underlying input model. For the sparse access input model, $\alpha_A = s_A$ and $\alpha_L = s_L$, the respective sparsities of the matrices A and L. On the other hand for the quantum data structure input model, $\alpha_A = \|A\|_F$ and $\alpha_L = \|L\|_F$. Consequently, the complexity of *Quantum Ridge Regression* can be obtained by substituting $L = I$ in the

above complexity as

$$\mathcal{O}\left(\log\kappa\left(\frac{\alpha_A}{\sqrt{\lambda}}\log\left(\frac{\kappa}{\delta}\right)T_A+\kappa T_b\right)\right)$$

where $\kappa = 1+\|A\|/\sqrt{\lambda}$, by noting that the block-encoding of I is trivial while the norm and condition number of the identity matrix is one. For this problem of *quantum ridge regression*, our quantum algorithms are substantially better than prior results [SX20, YGW21, CYGL22], exhibiting a polynomial improvement in κ and an exponential improvement in $1/\delta$.

For the ℓ_2-regularized GLS problem (Section 4, Theorem 42), we design a quantum algorithm that along with approximate block-encodings of A and L, takes as input an $(\alpha_\Omega, a_\Omega, \varepsilon_\Omega)$-lock-encoding of the matrix Ω (implementable at a cost of T_Ω) to output a state δ-close to $|x\rangle$ at a cost of

$$\mathcal{O}\left(\kappa\sqrt{\kappa_\Omega}\log\kappa\left(\left(\frac{\alpha_A}{\|A\|}T_A+\frac{\alpha_L}{\|L\|}T_L+\frac{\alpha_\Omega\kappa_\Omega}{\|\Omega\|}T_\Omega\right)\log^3\left(\frac{\kappa\kappa_\Omega\|A\|\|L\|}{\delta\|\Omega\|}\right)+T_b\right)\right)$$

In the above complexity, when L is a good regularizer, the modified condition number κ is defined as

$$\kappa=\kappa_L\left(1+\frac{\sqrt{\kappa_\Omega}\|A\|}{\sqrt{\lambda\|\Omega\|}\|L\|}\right)$$

The WLS problem is a particular case of GLS, wherein the matrix Ω is diagonal. However, we show that better complexities for the ℓ_2-regularized WLS problem can be obtained if we assume QRAM access to the diagonal entries of W (Section 4, Theorem 39 and Theorem 40).

Table 1 summarizes the complexities of our algorithms for quantum linear regression with general ℓ_2-regularization. For better exposition, here we assume that $\|A\|, \|L\|, \|\Omega\|$ and $\lambda = \Theta(1)$. For the general expression of the complexities, we refer the readers to Section 4.

Problem	**Unregularized**	**ℓ_2-Regularized**
Quantum OLS	$\widetilde{\mathcal{O}}\left(\alpha_A\kappa_A\log\left(1/\delta\right)\right)$	$\widetilde{\mathcal{O}}\left((\alpha_A+\alpha_L)\kappa_L\log\left(1/\delta\right)\right)$
Quantum GLS	$\widetilde{\mathcal{O}}\left(\left(\alpha_A+\alpha_\Omega\kappa_\Omega\right)\kappa_A\sqrt{\kappa_\Omega}\log^3\left(1/\delta\right)\right)$	$\widetilde{\mathcal{O}}\left(\left(\alpha_A+\alpha_L+\alpha_\Omega\kappa_\Omega\right)\kappa_L\sqrt{\kappa_\Omega}\log^3\left(1/\delta\right)\right)$

Table 1: Complexity of quantum linear regression algorithms with and without general ℓ_2-regularization. All of these algorithms require only $\Theta(\log\kappa)$ additional qubits.

In order to derive our results, we take advantage of the ability to efficiently perform arithmetic operations on block-encoded matrices, as outlined in Section 3. Along with this, we use QSVT to perform linear algebraic operations on block-encoded matrices. To this end, adapt the results in Refs. [GSLW19, MRTC21] to our setting. One of our contributions is that we work with robust versions of many of these algorithms. In prior works, QSVT is often applied to block-encoded matrices, assuming perfect block-encoding. For the quantum algorithms in this paper, we rigorously obtain the precision ε required to obtain a δ-approximation of the desired output state.

For instance, a key ingredient of our algorithm for regularized least squares is to make use of QSVT to obtain A^+, given an ε-approximate block-encoding of A. In order to obtain

a (near) optimal dependence on the condition number of A by applying variable-time amplitude amplification (VTAA) [Amb12], we recast the standard QSVT algorithm as a variable stopping-time quantum algorithm. Using QSVT instead of controlled Hamiltonian simulation ensures that the variable-time quantum procedure to prepare $A^+ |x\rangle$ has a slightly better running time (by a log factor) and considerably fewer additional qubits than Refs. [CKS17, CGJ19].

Furthermore, for the variable time matrix inversion algorithm, a crucial requirement is the application of the inversion procedure to the portion of the input state that is spanned by singular values larger than a certain threshold. In order to achieve this, prior results have made use of *Gapped Phase Estimation* (GPE), which is a simple variant of the standard phase estimation procedure that decides whether the eigenvalue of a Hermitian operator is above or below a certain threshold [Amb12, CKS17, CGJ19]. However, GPE can only be applied to a Hermitian matrix and requires additional registers that store the estimates of the phases, which are never used for variable-time amplitude amplification. In this work, instead of GPE, we develop a robust version of *quantum singular value discrimination* (QSVD) using QSVT, which can be directly applied to non-Hermitian matrices. This algorithm decides whether some singular value of a matrix is above or below a certain threshold without storing estimates of the singular values. This leads to a *space-efficient* variable time quantum algorithm for matrix inversion by further reducing the number of additional qubits required by a factor of $O(\log^2(\kappa/\delta))$ as compared to prior results [CKS17, CGJ19]. Consequently, this also implies that in our framework, quantum algorithms for (unregularized) least squares (which are special cases of our result) have better complexities than those of Ref. [CGJ19].

2 Preliminaries

This section lays down the notation, and introduces the quantum singular value transformation (QSVT) and block-encoding frameworks, which are used to design the algorithm for quantum regression.

2.1 Notation

For a matrix $A \in \mathbb{R}^{N\times d}$, $A_{i,\cdot}$ denotes the i^{th} row of A, and $\|A_{i,\cdot}\|$ denotes the vector norm of $A^T_{i,\cdot}$. s^A_r and s^A_c denote the row and column sparsity of the matrix, which is the maximum number of non-zero entries in any row and any column, respectively.

Singular Value Decomposition. The decomposition $A = W\Sigma V^\dagger$, where W and V are unitary and Σ is a diagonal matrix, represents the *singular value decomposition* (SVD) of A. All matrices can be decomposed in this form. The diagonal entries of Σ, usually denoted by $\sigma(A) = \{\sigma_j\}$, is the multiset of all *singular values* of A, which are real and non-negative. $\sigma_{\max}$ and $\sigma_{\min}$ denote the maximum and minimum singular values of A. $r(A) = \mathrm{rank}(A)$ is the number of non-zero singular values of A. The columns of W, V (denoted by $\{|w_j\rangle\}$ and $\{|v_j\rangle\}$) are the left and right *singular vectors* of A. Thus $A = \sum_j^r \sigma_j |w_j\rangle \langle v_j|$. The singular vectors of A can be computed as the positive square roots of the eigenvalues of $A^\dagger A$ (which is positive semi-definite and therefore has non-negative real eigenvalues.)

Effective Condition Number. κ_A denotes (an upper bound on) the effective condition number of A, defined as the ratio of the maximum and minimum non-zero singular values of A. Let $\sigma_{\max}(A)$ be the largest singular value of A, and $\sigma_{\min}(A)$ be the smallest singular

value of A. Additionally, let $\widetilde{\sigma}_{\min}(A)$ be the smallest non-zero singular value of A. Then

$$\kappa_A \geq \frac{\sigma_{\max}(A)}{\widetilde{\sigma}_{\min}(A)} = \sqrt{\frac{\lambda_{\max}(A^\dagger A)}{\widetilde{\lambda}_{\min}(A^\dagger A)}}$$

If A is full-rank, then $\widetilde{\sigma}_{\min}(A) = \sigma_{\min}(A)$, and κ_A becomes the condition number of the matrix. In this text, unless stated otherwise, we always refer to κ_A as (an upper bound on) effective condition number of a matrix, and not the true condition number.

Norm. Unless otherwise specified, $\|A\|$ denotes the spectral norm of A, while $\|A\|_F$ denotes the Frobenius norm of A, defined as

$$\|A\| := \max_{x \neq 0} \frac{\|Ax\|}{\|x\|} = \sigma_{\max}(A)$$
$$\|A\|_F := \sqrt{\sum_{j=1}^{r} \sigma_j^2}$$

Unless otherwise specified, when A is assumed to be normalized, it is with respect to the spectral norm.

Soft-O Complexity. Finally, we use $f = \widetilde{\mathcal{O}}(g)$ to denote $f = \mathcal{O}(g \cdot \text{polylog}(g))$.

Controlled Unitaries. If U is a s-qubit unitary, then $C\text{-}U$ is a $(s+1)$-qubit unitary defined by

$$C\text{-}U = |0\rangle\langle 0| \otimes I_s + |1\rangle\langle 1| \otimes U$$

Throughout this text whenever we state that the time taken to implement a unitary U_A is T_A and the cost of an algorithm is $\mathcal{O}(nT_A)$, we imply that the algorithm makes n uses of the unitary U_A. Thus, if the circuit depth of U_A is T_A, the circuit depth of our algorithm is $\mathcal{O}(nT_A)$.

2.2 Quantum Input Models

The complexities of quantum algorithms often depend on how the input data is accessed. For instance, in quantum algorithms for linear algebra (involving matrix operations), it is often assumed that there exists a black-box that returns the positions of the non-zero entries of the underlying matrix when queried. The algorithmic running time is expressed in terms of the number of queries made to this black-box. Such an input model, known as the *Sparse Access Model*, helps design efficient quantum algorithms whenever the underlying matrices are sparse. Various other input models exist, and quantum algorithms are typically designed and optimized for specific input models.

Kerenidis and Prakash [KP17, Section 5.1] introduced a different input model, known as the *quantum data structure model*, which is more conducive for designing quantum machine learning algorithms. In this model, the input data (e.g: entries of matrices) arrive online and are stored in a classical data structure (often referred to as the KP-tree in the literature), which can be queried in superposition by using a QRAM. This facilitates efficiently preparing quantum states corresponding to the rows of the underlying matrix, that can then be used for performing several matrix operations. Subsequently, several quantum-inspired classical algorithms have also been developed following the breakthrough result of Tang [Tan19]. Such classical algorithms have the same underlying assumptions as the

quantum algorithms designed in the data structure input model and are only polynomially slower provided the underlying matrix is low rank.

In this work, we will consider the framework of *block-encoding*, wherein it is assumed that the input matrix A (up to some sub-normalization) is stored in the left block of some unitary. The advantage of the block-encoding framework, which was introduced in a series of works [LC19, Definition 1], [CGJ19, Section 1], [GSLW19, Section 1.3], is that it can be applied to a wide variety of input models. For instance, it can be shown that both the sparse access input model as well as the quantum data structure input model are specific instances of block-encoded matrices [CGJ19, Sections 2.2 and 2.4], [GSLW19, Section 5.2]. Here we formally define the framework of block-encoding and also express the sparse access model as well as the quantum data structure model as block-encodings. We refer the reader to [CGJ19, GSLW19] for proofs.

Definition 1 (Block Encoding, restated from [GSLW19], Definition 24)**.** *Suppose that A is an s-qubit operator, $\alpha, \varepsilon \in \mathbb{R}^+$ and $a \in \mathbb{N}$, then we say that the $(s+a)$-qubit unitary U_A is an (α, a, ε)-block-encoding of A, if*

$$\left\| A - \alpha(\langle 0|^{\otimes a} \otimes I) U_A (|0\rangle^{\otimes a} \otimes I) \right\| \leq \varepsilon. \tag{11}$$

Let $|\psi\rangle$ be an s-qubit quantum state. Then applying U_A to $|\psi\rangle |0\rangle^{\otimes a}$ outputs a quantum state that is $\frac{\varepsilon}{\alpha}$-close to

$$\frac{A}{\alpha} |\psi\rangle |0\rangle^{\otimes a} + |\Phi^{\perp}\rangle ,$$

where $\left(I_s \otimes |0\rangle\langle 0|^{\otimes a}\right) |\Phi^{\perp}\rangle = 0$. Equivalently, suppose $\tilde{A} := \alpha \left(\langle 0|^{\otimes a} \otimes I_s\right) U_A \left(|0\rangle^{\otimes a} \otimes I_s\right)$ denotes the actual matrix that is block-encoded into U_A, then $\left\| A - \tilde{A} \right\| \leq \varepsilon$.

In the subsequent sections, we provide an outline of the quantum data structure model and the sparse access model which are particular instances of the block encoding framework.

2.2.1 Quantum Data Structure Input Model

Kerenidis and Prakash introduced a quantum accessible classical data structure which has proven to be quite useful for designing several quantum algorithms for linear algebra [KP17]. The classical data structure stores entries of matrices or vectors and can be queried in superposition using a QRAM (quantum random access memory). We directly state the following theorem from therein.

Theorem 2 (Implementing quantum operators using an efficient data structure, [Pra14, KP17])**.** *Let $A \in \mathbb{R}^{N \times d}$, and w be the number of non-zero entries of A. Then there exists a data structure of size $\mathcal{O}\left(w \log^2 (dN)\right)$ that given the matrix elements (i, j, a_{ij}), stores them at a cost of $\mathcal{O}(\log (dN))$ operations per element. Once all the non-zero entries of A have been stored in the data structure, there exist quantum algorithms that are ε-approximations to the following maps:*

$$U : |i\rangle |0\rangle \mapsto \frac{1}{\|A_{i,\cdot}\|} \sum_{j=1}^{d} a_{i,j} |i, j\rangle = |\psi_i\rangle ,$$

$$V : |0\rangle |j\rangle \mapsto \frac{1}{\|A\|_F} \sum_{i=1}^{N} \|A_i, .\| \, |i, j\rangle = |\phi_j\rangle$$

where $\|A_{i,\cdot}\|$ *is the norm of the* i^{th} *row of* A *and the second register of* $|\psi_i\rangle$ *is the quantum state corresponding to the* i^{th} *row of* A*. These operations can be applied at a cost of* $\mathcal{O}(\text{polylog}(Nd/\varepsilon))$*.*

It was identified in Ref. [CGJ19] that if a matrix A is stored in this quantum accessible data structure, there exists an efficiently implementable block-encoding of A. We restate their result here.

Lemma 3 (Implementing block encodings from quantum data structures, [CGJ19], Theorem 4)**.** *Let the entries of the matrix* $A \in \mathbb{R}^{N\times d}$ *be stored in a quantum accessible data structure, then there exist unitaries* U_R, U_L *that can be implemented at a cost of* $\mathcal{O}(\text{polylog}(dN/\varepsilon))$ *such that* $U_R^\dagger U_L$ *is a* $(\|A\|_F, \lceil \log(d+N)\rceil, \varepsilon)$*-block-encoding of* A*.*

Proof. The unitaries U_R and U_L can be implemented via U and V in the previous lemma. Let $U_R = U$ and $U_L = V.\texttt{SWAP}$. Then for $s = \lceil \log(d+N)\rceil$ we have

$$U_R : \ |i\rangle\,|0^s\rangle \to |\psi_i\rangle\,,$$

and

$$U_L : \ |j\rangle\,|0^s\rangle \to |\phi_j\rangle\,,$$

So we have that the top left block of $U_R^\dagger U_L$ is

$$\sum_{i=1}^{N}\sum_{j=1}^{d} \langle\psi_i|\phi_j\rangle\,|i,0\rangle\,\langle j,0|$$

Now

$$\begin{aligned}\langle\psi_i|\phi_j\rangle &= \sum_{k=1}^{d}\sum_{\ell=1}^{N} \frac{a_{ik}}{\|A_{i,\cdot}\|}\cdot\frac{\|A_\ell\|}{\|A\|_F}\underbrace{\langle i,k|l,j\rangle}_{:=\delta_{i,l}.\delta_{k,j}} \\ &= \frac{a_{ij}}{\|A\|_F}.\end{aligned}$$

Moreover since only ε-approximations of U and V can be implemented we have that $U_R^\dagger U_L$ is a $(\|A\|_F, \lceil\log(n+d)\rceil, \varepsilon)$ block encoding of A implementable with the same cost as U and V. □

In Ref. [KP20] argued that in certain scenarios, storing the entries of $A^{(p)}, (A^{1-p})^\dagger$ might be useful as compared to storing A, for some $p \in [0,1]$. In such cases, the quantum data structure is a $(\mu_p, \lceil\log(N+d)\rceil, \varepsilon)$ block encoding of A, where $\mu_p(A) = \sqrt{s_{2p}(A).s_{2(1-p)}(A^T)}$ such that $s_p(A) := \max_j \|A_{j,\cdot}\|_q^q$. Throughout the work, whenever our results are expressed in the quantum data structure input model, we shall state our complexity in terms of μ_A. When the entries of A are directly stored in the data structure, $\mu_A = \|A\|_F$. Although, we will not state it explicitly each time, our results also hold when fractional powers of A are stored in the database and simply substituting $\mu_A = \mu_p(A)$, yields the required complexity.

2.2.2 Sparse Access Input Model

The sparse access input model considers that the input matrix $A \in \mathbb{R}^{N\times d}$ has row sparsity s_r and column sparsity s_c. Furthermore, it assumes that the entries of A can be queried via an oracle as

$$O_A : |i\rangle |j\rangle |0\rangle^{\otimes b} \mapsto |i\rangle |j\rangle |a_{ij}\rangle \quad \forall i \in [N], j \in [d],$$

and the indices of the non-zero elements of each row and column can be queried via the following oracles:

$$O_r : |i\rangle |j\rangle \mapsto |i\rangle |r_{ij}\rangle \quad \forall i \in [N], k \in [s_r],$$
$$O_c : |i\rangle |j\rangle \mapsto |c_{ij}\rangle |j\rangle \quad \forall i \in [d], k \in [s_c]$$

where r_{ij} is the j^{th} non-zero entry of the i^{th} row of A and c_{ij} is the i^{th} non-zero entry of the j^{th} column of A. Gilyén et al. [GSLW19] showed that a block encoding of a sparse A can be efficiently prepared by using these three oracles. We restate their lemma below.

Lemma 4 (Constructing a block-encoding from sparse-access to matrices, [GSLW18], Lemma 48)**.** *Let $A \in \mathbb{R}^{N \times d}$ be an s_r, s_c row, column sparse matrix given as a sparse access input. Then for all $\varepsilon \in (0,1)$, we can implement a $(\sqrt{s_c s_r}, \text{polylog}(Nd/\varepsilon), \varepsilon)$-block-encoding of A with $\mathcal{O}(1)$ queries to O_r, O_c, O_A and $\text{polylog}(Nd/\varepsilon)$ elementary quantum gates.*

Throughout the paper, we shall assume input matrices are accessible via approximate block-encodings. This also allows us to write down the complexities of our quantum algorithms in this general framework. Additionally, we state the complexities in both the sparse access input model as well as the quantum accessible data structure input model as particular cases.

2.3 Quantum Singular Value Transformation

In a seminal work, Gilyén et al. presented a framework to apply an arbitrary polynomial function to the singular values of a matrix, known as Quantum Singular Value Transformation (QSVT) [GSLW19]. QSVT is quite general: many quantum algorithms can be recast to this framework, and for several problems, better quantum algorithms can be obtained [GSLW19, MRTC21]. In particular, QSVT has been extremely useful in obtaining optimal quantum algorithms for linear algebra. For instance, using QSVT, given the block-encoding of a matrix A, one could obtain A^{-c} with $c \in [0, \infty)$ with optimal complexity and by using fewer additional qubits than prior art. This section briefly describes this framework, which is a generalization of Quantum Signal Processing (QSP) [LC19, Section 2], [LC17b, Theorem 2], [LYC16]. The reader may refer to [MRTC21] for a more pedagogical overview of these techniques.

Let us begin by discussing the framework of Quantum Signal Processing. QSP is a quantum algorithm to apply a d-degree bounded polynomial transformation with parity $d \mod 2$ to an arbitrary quantum subsystem, using a quantum circuit U_Φ consisting of only controlled single qubit rotations. This is achieved by interleaving a *signal rotation operator* W (which is an x-rotation by some fixed angle θ) and a *signal processing operator* S_ϕ (which is a z-rotation by a variable angle $\phi \in [0, 2\pi]$). In this formulation, the signal rotation operator is defined as

$$W(x) := \begin{pmatrix} x & i\sqrt{1-x^2} \\ i\sqrt{1-x^2} & x \end{pmatrix}, \tag{12}$$

which is an x-rotation by angle $\theta = -2\arccos(x)$, and the signal processing operator is defined as

$$S_\phi := e^{i\phi Z}, \tag{13}$$

which is a z-rotation by an angle -2ϕ. Interestingly, sandwiching them together for some $\Phi := (\phi_0, \phi_1, \dots \phi_d) \in \mathbb{R}^{d+1}$, as shown in Equation 14, gives us a matrix whose elements are polynomial transformations of x,

$$U_\Phi := e^{i\phi_0 Z} \prod_{j=1}^{j=d} \left(W(x) e^{i\phi_j Z} \right) \tag{14}$$

$$= \begin{pmatrix} P(x) & iQ(x)\sqrt{1-x^2} \\ iQ^*(x)\sqrt{1-x^2} & P^*(x) \end{pmatrix}, \tag{15}$$

such that

1. $\deg P \leq d$; $\deg Q \leq d-1$,
2. $P(x)$ has a parity $d \mod 2$,
3. $|P(x)|^2 + (1-x^2)|Q(x)|^2 = 1 \quad \forall x \in [-1, 1]$.

Following the application of the quantum circuit U_Φ for an appropriate Φ, one can project into the top left block of U_Φ to recover the polynomial $\langle 0| U_\Phi |0\rangle = P(x)$. Projecting to other basis allows the ability to perform more interesting polynomial transformations, which can be linear combinations of $P(x), Q(x)$, and their complex conjugates. For example, projecting to $\{|+\rangle, |-\rangle\}$ basis gives us

$$\langle +| U_\Phi |+\rangle = \Re(P(x)) + i\Re(Q(x))\sqrt{1-x^2}. \tag{16}$$

Quantum Signal Processing can be formally stated as follows.

Theorem 5 (Quantum Signal Processing, Corollary 8 from [GSLW19])**.** *Let $P \in \mathbb{C}[x]$ be a polynomial of degree $d \geq 2$, such that*

- *P has parity-$(d \mod 2)$,*
- *$\forall x \in [-1, 1] : |P(x)| \leq 1$,*
- *$\forall x \in (-\infty, -1] \cup [1, \infty) : |P(x)| \geq 1$,*
- *if d is even, then $\forall x \in \mathbb{R} : P(ix)P^*(ix) \geq 1$.*

Then there exists a $\Phi \in \mathbb{R}^d$ such that

$$\prod_{j=1}^{d} \left(e^{i\phi_j \sigma_z} W(x) \right) = \begin{pmatrix} P(x) & \cdot \\ \cdot & \cdot \end{pmatrix}. \tag{17}$$

Thus, QSP allows us to implement any polynomial $P(x)$ that satisfies the aforementioned requirements. Throughout this article, we refer to any such polynomial $P(x)$ as a *QSP polynomial*. Quantum Singular Value Transformation is a natural generalization of this procedure. It allows us to apply a QSP polynomial transformation to each singular value of an arbitrary block of a unitary matrix. In addition to this generalization, QSVT relies on the observation that several functions can be well-approximated by QSP polynomials. Thus, through QSVT one can transform each singular value of a block-encoded matrix by any function that can be approximated by a QSP polynomial. Since several linear algebra problems boil down to applying specific transformations to the singular values of a matrix, QSVT is particularly useful for developing fast algorithms for quantum linear algebra. Next, we introduce QSVT formally via the following theorem.

Theorem 6 (Quantum Singular Value Transformation [GSLW18], Section 3.2)**.** *Suppose $A \in \mathbb{R}^{N\times d}$ is a matrix with singular value decomposition $A = \sum_{j=1}^{d_{\min}} \sigma_j |v_j\rangle \langle w_j|$, where $d_{\min} = \min\{N, d\}$ and $|v_j\rangle$ ($|w_j\rangle$) is the left (right) singular vector with singular value σ_j. Furthermore, let U_A be a unitary such that $A = \widetilde{\Pi} U_A \Pi$, where Π and $\widetilde{\Pi}$ are orthogonal projectors. Then, for any QSP polynomial $P(x)$ of degree n, there exists a vector $\Phi = (\phi_1, \phi_2, \cdots \phi_n) \in \mathbb{R}^n$ and a unitary*

$$U_\Phi = \begin{cases} e^{i\phi_1(2\widetilde{\Pi}-I)} U_A \left[\prod_{k=1}^{(n-1)/2} e^{i\phi_{2k}(2\widetilde{\Pi}-I)} U_A^\dagger e^{i\phi_{2k+1}(2\widetilde{\Pi}-I)} U_A\right], & n \text{ is odd} \\ \left[\prod_{k=1}^{n/2} e^{i\phi_{2k-1}(2\widetilde{\Pi}-I)} U_A^\dagger e^{i\phi_{2k}(2\widetilde{\Pi}-I)} U_A\right], & n \text{ is even,} \end{cases} \tag{18}$$

such that

$$P^{SV}(A) = \begin{cases} \widetilde{\Pi} U_\Phi \Pi, & n \text{ is odd} \\ \Pi U_\Phi \Pi, & n \text{ is even,} \end{cases} \tag{19}$$

where $P^{SV}(A)$ is the polynomial transformation of the matrix A defined as

$$P^{SV}(A) := \begin{cases} \sum_j P(\sigma_j) |v_j\rangle \langle w_j|, & P \text{ is odd} \\ \sum_j P(\sigma_j) |w_j\rangle \langle w_j|, & P \text{ is even.} \end{cases} \tag{20}$$

Theorem 6 tells us that for any QSP polynomial P of degree n, we can implement $P^{SV}(A)$ using one ancilla qubit, $\Theta(n)$ applications of U_A, $U_A^\dagger$ and controlled reflections $I - 2\Pi$ and $I - 2\widetilde{\Pi}$. Furthermore, if in some well-defined interval, some function $f(x)$ is well approximated by an n-degree QSP polynomial $P(x)$, then Theorem 6 also allows us to implement a transformation that approximates $f(A)$, where

$$f(A) := \begin{cases} \sum_j f(\sigma_j) |v_j\rangle \langle w_j|, & P \text{ is odd} \\ \sum_j f(\sigma_j) |w_j\rangle \langle w_j|, & P \text{ is even.} \end{cases} \tag{21}$$

The following theorem from Ref. [GSLW18] deals with the robustness of the QSVT procedure, i.e. how errors propagate in QSVT. In particular, for two matrices A and $\tilde{A}$, it shows how close their polynomial transformations ($P^{SV}(A)$ and $P^{SV}(\tilde{A})$, respectively) are, as a function of the distance between A and $\tilde{A}$.

Lemma 7 (Robustness of Quantum Singular Value Transformation, [GSLW18], Lemma 23)**.** *Let $P \in \mathbb{C}[x]$ be a QSP polynomial of degree n. Let $A, \tilde{A} \in \mathbb{C}^{N\times d}$ be matrices of spectral norm at most 1, such that*

$$\left\|A - \tilde{A}\right\| + \left\|\frac{A + \tilde{A}}{2}\right\|^2 \leq 1.$$

Then,

$$\left\|P^{SV}(A) - P^{SV}(\tilde{A})\right\| \leq n \sqrt{\frac{2}{1 - \left\|\frac{A+\tilde{A}}{2}\right\|^2}} \left\|A - \tilde{A}\right\|.$$

We will apply this theorem to develop a robust version of QSVT. More precisely, in order to implement QSVT, we require access to a unitary U_A, which is a block-encoding of some matrix A. This block-encoding, in most practical scenarios, is not perfect: we only have access to a ε-approximate block-encoding of A. If we want an δ-accurate implementation of $P^{SV}(A)$, how precise should the block-encoding of A be? Such a robustness analysis has been absent from prior work involving QSVT and will allow us to develop robust versions of a number of quantum algorithms in subsequent sections. The following theorem determines the precision ε required in the block-encoding of A in terms of n, the degree of the QSP polynomial that we wish to implement and δ, the accuracy of $P^{SV}(A)$.

Theorem 8 (Robust QSVT). *Let $P \in \mathbb{C}[x]$ be a QSP polynomial of degree $n \geq 2$. Let $\delta \in [0,1]$ be the precision parameter. Let U be an (α, a, ε)-block-encoding of matrix $A \in \mathbb{C}^{N\times d}$ satisfying $\|A\| \leq \alpha/2$, implemented in cost T for some $\varepsilon \leq \alpha\delta/2n$. Then we can construct a $(1, a+1, \delta)$-block-encoding of $P(A/\alpha)$ in cost $\mathcal{O}(nT)$.*

Proof. Let $\tilde{A}$ be the encoded block of U, then $\left\|A - \tilde{A}\right\| \leq \varepsilon$. Applying QSVT on U with the polynomial P, we get a block-encoding for $P(\tilde{A}/\alpha)$, with $\mathcal{O}(n)$ uses of $U, U^\dagger$, and as many multiply-controlled NOT gates. Observe that $\left\|\frac{A}{\alpha} - \frac{\tilde{A}}{\alpha}\right\| \leq \frac{\varepsilon}{\alpha} \leq \frac{\delta}{2n} \leq \frac{1}{4}$, and,

$$\left\|\frac{\frac{A}{\alpha} + \frac{\tilde{A}}{\alpha}}{2}\right\|^2 = \left\|\frac{A}{\alpha} + \frac{\tilde{A} - A}{2\alpha}\right\|^2 \leq \left(\frac{\|A\|}{\alpha} + \frac{\left\|\tilde{A} - A\right\|}{2\alpha}\right)^2 \leq \left(\frac{1}{2} + \frac{1}{8}\right)^2 \leq \frac{1}{2}$$

Therefore the error in the final block-encoding is given by invoking Lemma 7 with matrices $A/\alpha, \tilde{A}/\alpha$:

$$\left\|P\left(\frac{A}{\alpha}\right) - P\left(\frac{\tilde{A}}{\alpha}\right)\right\| \leq n\sqrt{\frac{2}{1-\frac{1}{2}}}\,\frac{\varepsilon}{\alpha} = \frac{2n\varepsilon}{\alpha} \leq \delta.$$

□

In Section 3, we will make use of Theorem 8, to develop robust quantum algorithms for singular value discrimination, variable-time matrix inversion, positive and negative powers of matrices. Subsequently, in Sec. 4, we shall combine algorithmic primitives to design robust quantum regularized least squares algorithms.

2.4 Variable Time Amplitude Amplification

Ambainis [Amb12] defined the notion of a *variable-stopping-time quantum algorithm* and formulated the technique of *Variable Time Amplitude Amplification* (VTAA), a tool that can be used to amplify the success probability of a variable-stopping-time quantum algorithm to a constant by taking advantage of the fact that computation on some parts of an algorithm can complete earlier than on other parts. The key idea here is to look at a quantum algorithm $\mathcal{A}$ acting on a state $|\psi\rangle$ as a combination of m quantum sub-algorithms $\mathcal{A} = \mathcal{A}_m \cdot \mathcal{A}_{m-1} \cdot \ldots \mathcal{A}_1$, each acting on $|\psi\rangle$ conditioned on some ancilla flag being set. Formally, a variable stopping time algorithm is defined as follows

Definition 9 (Variable-stopping-time Algorithm, [Amb12]). *A quantum algorithm $\mathcal{A}$ acting on $\mathcal{H}$ that can be written as m quantum sub-algorithms, $\mathcal{A} = \mathcal{A}_m \cdot \mathcal{A}_{m-1} \cdot \ldots \mathcal{A}_1$ is called a variable stopping time algorithm if $\mathcal{H} = \mathcal{H}_C \otimes \mathcal{H}_\mathcal{A}$, where $\mathcal{H}_C = \otimes_{i=1}^m \mathcal{H}_{C_i}$ with $\mathcal{H}_{C_i} = \mathrm{span}(|0\rangle, |1\rangle)$, and each unitary $\mathcal{A}_j$ acts on $\mathcal{H}_{C_j} \otimes \mathcal{H}_\mathcal{A}$ controlled on the first $j-1$ qubits $|0\rangle^{\otimes j-1} \in \otimes_{i=1}^{j-1}\mathcal{H}_{C_i}$ being in the all zero state.*

Here $\mathcal{H}_{C_i}$ is a single qubit clock register. In VTAA, $\mathcal{H}_\mathcal{A}$ has a flag space consisting of a single qubit to indicate success, $\mathcal{H}_\mathcal{A} = \mathcal{H}_F \otimes \mathcal{H}_W$. Here $\mathcal{H}_F = \mathrm{Span}(|g\rangle, |b\rangle)$ flags the good and bad parts of the run. Furthermore, for $1 \leq i \leq m$, define the stopping times t_i such that $t_1 < t_2 < \cdots t_m = T_{\max}$, such that the algorithm $\mathcal{A}_j\mathcal{A}_{j-1}\cdots\mathcal{A}_1$ having (gate/query) complexity t_i halts with probability

$$p_j = \left\|\Pi_{C_j}\mathcal{A}_j\mathcal{A}_{j-1}\cdots\mathcal{A}_1|0\rangle_\mathcal{H}\right\|^2,$$

where $|0\rangle_{\mathcal{H}} \in \mathcal{H}$ is the all zero quantum state and Π_{C_j} is the projector onto $|1\rangle$ in $\mathcal{H}_{C_j}$. From this one can define the average stopping time of the algorithm $\mathcal{A}$ defined as

$$\|T\|_2 = \sqrt{\sum_{j=1}^{m} p_j t_j^2}.$$

For a variable stopping time algorithm if the average stopping time $\|T\|_2$ is less than the maximum stopping time $T_{\max}$, VTAA can amplify the success probability (p_{succ}) much faster than standard amplitude amplification. In this framework, the success probability of $\mathcal{A}$ is given by

$$p_{\text{succ}} = \|\Pi_F \mathcal{A}_m \mathcal{A}_{m-1} \cdots \mathcal{A}_1 |0\rangle_{\mathcal{H}}\|^2$$

While standard amplitude amplification requires time scaling as $\mathcal{O}(T_{\max}/\sqrt{p_{\text{succ}}})$, the complexity of VTAA is more involved. Following [CGJ19], we define the complexity of VTAA as follows.

Lemma 10 (Efficient variable time amplitude amplification [CGJ18], Theorem 23)**.** *Let U be a state preparation unitary such that $U|0\rangle^{\otimes k} = \sqrt{p_{\text{prep}}}|0\rangle|\psi_0\rangle + \sqrt{1-p_{\text{prep}}}|1\rangle|\psi_1\rangle$ that has a query complexity T_U. And let $\mathcal{A} = \mathcal{A}_m\mathcal{A}_{m-1}\cdots\mathcal{A}_1$ be a variable stopping time quantum algorithm that we want to apply to the state $|\psi_0\rangle$, with the following known bounds: $p_{\text{prep}} \geq p'_{\text{prep}}$ and $p_{\text{succ}} \geq p'_{\text{succ}}$. Define $T'_{\max} := 2T_{\max}/t_1$ and*

$$Q := \left(T_{\max} + \frac{T_U + k}{\sqrt{p_{\text{prep}}}}\right)\sqrt{\log\left(T'_{\max}\right)} + \frac{\left(\|T\|_2 + \frac{T_U+k}{\sqrt{p_{\text{prep}}}}\right)\log\left(T'_{\max}\right)}{\sqrt{p_{\text{succ}}}}.$$

Then with success probability $\geq 1-\delta$, we can create a variable-stopping time algorithm $\mathcal{A}'$ that prepares the state $a|0\rangle\mathcal{A}'|\psi_0\rangle + \sqrt{1-a^2}|1\rangle|\psi_{garbage}\rangle$, such that $a = \Theta(1)$ is a constant and $\mathcal{A}'$ has the complexity $\mathcal{O}(Q)$.

One cannot simply replace standard amplitude amplification with VTAA to boost the success probability of a quantum algorithm. A crucial task would be to recast the underlying algorithm in the VTAA framework. We will be applying VTAA to the quantum algorithm for matrix inversion by QSVT. So, first of all, in order to apply VTAA to the algorithm must be first recast into a variable-time stopping algorithm so that VTAA can be applied.

Originally, Ambainis [Amb12] used VTAA to improve the running time of the HHL algorithm from $\mathcal{O}(\kappa^2 \log N)$ to $\mathcal{O}\left(\kappa \log^3 \kappa \log N\right)$. Childs et al. [CKS17] designed a quantum linear systems algorithm with a polylogarithmic dependence on the accuracy. Additionally, they recast their algorithm into a framework where VTAA could be applied to obtain a linear dependence on κ. Later Chakraborty et al. [CGJ19] modified Ambainis' VTAA algorithm to perform variable time amplitude estimation.

In this work, to design quantum algorithms for ℓ_2-regularized linear regression, we use a quantum algorithm for matrix inversion by QSVT. We recast this algorithm in the framework of VTAA to achieve nearly linear dependence in κ (the effective condition number of the matrix to be inverted). Using QSVT instead of controlled Hamiltonian simulation improves the complexity of the overall matrix inversion algorithm (QSVT and VTAA) by a log factor. It also reduces the number of additional qubits substantially. Furthermore, we replace a gapped quantum phase estimation procedure with a more efficient quantum singular value discrimination algorithm using QSVT. This further reduces the number of additional qubits by $O(\log^2(\kappa/\delta))$ than in Refs. [CKS17, CGJ19], where κ is the condition number of the underlying matrix and δ is the desired accuracy. The details of the variable stopping time quantum algorithm for matrix inversion by QSVT are laid out in Section 3.3.

3 Algorithmic Primitives

This section introduces the building blocks of our quantum algorithms for quantum linear regression with general ℓ_2-regularization. As mentioned previously, we work in the block-encoding framework. We develop robust quantum algorithms for arithmetic operations, inversion, and positive and negative powers of matrices using quantum singular value transformation, assuming we have access to approximate block-encodings of these matrices. While some of these results were previously derived assuming perfect block-encodings [GSLW19, CGJ19], we calculate the precision required in the input block-encodings to output a block-encoding or quantum state arbitrarily close to the target.

Given a (α, a, ε)-block-encoding of a matrix A, we can efficiently amplify the subnormalization factor from α to a constant and obtain an amplified block-encoding of A. For our quantum algorithms in Sec. 4, we show working with pre-amplified block-encodings often yields better complexities. We state the following lemma which was proven in Ref. [LC17a]:

Lemma 11 (Uniform Block Amplification of Contractions, [LC17a])**.** *Let $A \in \mathbb{R}^{N\times d}$ such that $\|A\| \leq 1$ If $\alpha \geq 1$ and U is a (α, a, ε)-block-encoding of A that can be implemented at a cost of T_U, then there is a $(\sqrt{2}, a+1, \varepsilon+\gamma)$-block-encoding of A that can be implemented at a cost of $\mathcal{O}(\alpha T_U \log(1/\gamma))$.*

Corollary 12 (Uniform Block Amplification)**.** *Let $A \in \mathbb{R}^{N\times d}$ and $\delta \in (0, 1]$. Suppose U is a (α, a, ε)-block-encoding of A, such that $\varepsilon \leq \frac{\delta}{2}$, that can be implemented at a cost of T_U. Then a $(\sqrt{2}\|A\|, a+1, \delta)$-block-encoding of A can be implemented at a cost of $\mathcal{O}\Big(\frac{\alpha T_U}{\|A\|}\log(\|A\|/\delta)\Big)$.*

We now obtain the complexity of applying a block-encoded matrix to a quantum state, which is a generalization of a lemma proven in Ref. [CGJ19].

Lemma 13 (Applying a Block-encoded Matrix on a Quantum State)**.** *Let A be an s-qubit operator such that its non-zero singular values lie in $[\|A\|/\kappa, \|A\|]$. Also let $\delta \in (0, 1)$, and U_A be an (α, a, ε)-block-encoding of A, implementable in time T_A, such that*

$$\varepsilon \leq \frac{\delta\|A\|}{2\kappa}.$$

Furthermore, suppose $|b\rangle$ be an s-qubit quantum state, prepared in time T_b. Then we can prepare a state that is δ-close to $\frac{A|b\rangle}{\|A|b\rangle\|}$ with success probability $\Omega(1)$ at a cost of

$$\mathcal{O}\Big(\frac{\alpha\kappa}{\|A\|}(T_A + T_b)\Big)$$

Corollary 14 (Applying a pre-amplified Block-encoded Matrix on a Quantum State)**.** *Let A be an s-qubit operator such that its non-zero singular values lie in $[\|A\|/\kappa, \|A\|]$. Also let $\delta \in (0, 1)$, and U_A be an (α, a, ε)-block-encoding of A, implementable in time T_A, such that*

$$\varepsilon \leq \frac{\delta\|A\|}{4\kappa}.$$

Furthermore, suppose $|b\rangle$ be an s-qubit quantum state that can be prepared in time T_b. Then we can prepare a state that is δ-close to $\frac{A|b\rangle}{\|A|b\rangle\|}$ with success probability $\Omega(1)$ at a cost of

$$\mathcal{O}\Big(\frac{\alpha\kappa}{\|A\|}\log\Big(\frac{\kappa}{\delta}\Big)T_A + \kappa T_b\Big)$$

Now, it may happen that U_b prepares a quantum state that is only ε-close to the desired state $|b\rangle$. In such cases, we have the following lemma

Lemma 15 (Robustness of state preparation)**.** *Let A be an s-qubit operator such that its non-zero singular values lie in $[\|A\|/\kappa, \|A\|]$. Suppose $|b'\rangle$ is a quantum state that is $\varepsilon/2\kappa$-close to $|b\rangle$ and $|\psi\rangle$ is a quantum state that is $\varepsilon/2$-close to $A|b'\rangle/\|A|b'\rangle\|$. Then we have that $|\psi\rangle$ is ε-close to $A|b\rangle/\|A|b\rangle\|$.*

The proofs for Corollary 12, Lemma 13, Corollary 14, and Lemma 15 can be found in Appendix A.

3.1 Arithmetic with Block-Encoded Matrices

The block-encoding framework embeds a matrix on the top left block of a larger unitary U. It has been demonstrated that this framework allows us to obtain sums, products, linear combinations of block-encoded matrices. This is particularly useful for solving linear algebra problems in general. Here, we state some of the arithmetic operations on block-encoded matrices that we shall be using in order to design the quantum algorithms of Section 4 and tailor existing results to our requirements.

First we prove a slightly more general form of linear combination of unitaries in the block-encoding framework, presented in [GSLW19]. To do this we assume that we are given optimal state preparation pairs, defined as follows.

Definition 16 (Optimal State Preparation Unitary)**.** *Let $m \in \mathbb{Z}^+$, and $s = \lceil \log m \rceil$. Let $\eta \in \mathbb{R}_+^m$. Then we call a s-qubit unitary P a η state-preparation unitary if*

$$P|0\rangle = \frac{1}{\sum_j \eta_j} \sum_j \sqrt{\eta_j}\,|j\rangle$$

Lemma 17 (Linear Combination of Block Encoded Matrices)**.** *For each $j \in \{0, \ldots, m-1\}$, let A_j be an s-qubit operator, and $y_j \in \mathbb{R}^+$. Let U_j be a $(\alpha_j, a_j, \varepsilon_j)$-block-encoding of A_j, implemented in time T_j. Define the matrix $A = \sum_j y_j A_j$, and the vector $\eta \in \mathbb{R}^m$ s.t. $\eta_j = y_j \alpha_j$. Let U_η be a η state-preparation unitary, implemented in time T_η. Then we can implement a*

$$\left(\sum_j y_j \alpha_j, \max_j(a_j) + s, \sum_j y_j \varepsilon_j \right)$$

block-encoding of A at a cost of $\mathcal{O}\left(\sum_j T_j + T_\eta\right)$.

The proof is similar to the one in Ref. [GSLW19], with some improvements to the bounds. The detailed proof can be found in Appendix A. We now specialize the above lemma for the case where we need a linear combination of just two unitaries. This is the case used in this work, and we obtain a better error scaling for this by giving an explicit state preparation unitary.

Corollary 18 (Linear Combination of Two Block Encoded Matrices)**.** *For $j \in \{0, 1\}$, let A_j be an s-qubit operator and $y_j \in \mathbb{R}^+$. Let U_j be a $(\alpha_j, a_j, \varepsilon_j)$-block-encoding of A_j, implemented in time T_j. Then we can implement a $(y_0\alpha_0 + y_1\alpha_1, 1 + \max(a_0, a_1), y_0\varepsilon_0 + y_1\varepsilon_1)$ encoding of $y_0 A_0 + y_1 A_1$ in time $\mathcal{O}(T_0 + T_1)$.*

Proof. Let $\alpha = y_0\alpha_0 + y_1\alpha_1$ and $P = \frac{1}{\sqrt{\alpha}}\begin{pmatrix} \sqrt{y_0\alpha_0} & -\sqrt{y_1\alpha_1} \\ \sqrt{y_1\alpha_1} & \sqrt{y_0\alpha_0} \end{pmatrix}$. By Definition 16, P is a $\{y_0\alpha_0, y_1\alpha_1\}$ state-preparation-unitary. Invoking Lemma 17 with P, we get the required unitary.

□

Given block-encodings of two matrices A and B, it is easy to obtain a block-encoding of AB.

Lemma 19 (Product of Block Encodings, [GSLW18], Lemma 53)**.** *If U_A is an (α, a, δ)-block-encoding of an s-qubit operator A implemented in time T_A, and U_B is a (β, b, ε)-block-encoding of an s-qubit operator B implemented in time T_B, then $(I^{\otimes b} \otimes U_A)(I^{\otimes a} \otimes U_B)$ is an $(\alpha\beta, a+b, \alpha\varepsilon + \beta\delta)$-block-encoding of AB implemented at a cost of $\mathcal{O}(T_A + T_B)$.*

Directly applying Lemma 19 results in a block-encoding of $\frac{AB}{\alpha\beta}$. If α and β are large, then the sub-normalization factor $\alpha\beta$ might incur an undesirable overhead to the cost of the algorithm that uses it. In many cases, the complexity of obtaining products of block-encodings can be improved if we first amplify the block-encodings (using Lemma 12) and then apply Lemma 19. We prove the following lemma:

Lemma 20 (Product of Amplified Block-Encodings)**.** *Let $\delta \in (0, 1]$. If U_A is an $(\alpha_A, a_A, \varepsilon_A)$-block-encoding of an s-qubit operator A implemented in time T_A, and U_B is a $(\alpha_B, a_B, \varepsilon_B)$-block-encoding of an s-qubit operator B implemented in time T_B, such that $\varepsilon_A \leq \frac{\delta}{4\sqrt{2}\|B\|}$ and $\varepsilon_B \leq \frac{\delta}{4\sqrt{2}\|A\|}$. Then we can implement a $(2\|A\|\|B\|, a_A + a_B + 2, \delta)$-block-encoding of AB implemented at a cost of*

$$\mathcal{O}\left(\left(\frac{\alpha_A}{\|A\|}T_A + \frac{\alpha_B}{\|B\|}T_B\right)\log\left(\frac{\|A\|\|B\|}{\delta}\right)\right).$$

Proof. Using Corollary 12 for some $\delta_A \geq 2\varepsilon_A$ we get a $(\sqrt{2}\|A\|, a_A + 1, \delta_A)$-block-encoding of A at a cost of

$$\mathcal{O}\left(\frac{\alpha_A T_A}{\|A\|}\log\left(\|A\|/\delta_A\right)\right).$$

Similarly for some $\delta_B \geq 2\varepsilon_B$ we get a $(\sqrt{2}\|B\|, a_B + 1, \delta_B)$-block-encoding of B at a cost of

$$\mathcal{O}\left(\frac{\alpha_B T_B}{\|B\|}\log\left(\|B\|/\delta_B\right)\right).$$

Now using Lemma 19 we get a $(2, a_A + a_B + 2, \sqrt{2}\left(\|A\|\delta_B + \|B\|\delta_A\right))$-block-encoding of AB. We can choose $\delta_A = \frac{\delta}{2\sqrt{2}\|B\|}$ and $\delta_B = \frac{\delta}{2\sqrt{2}\|A\|}$ which bounds the final block-encoding error by δ. □

Observe that we have assumed that A and B are s-qubit operators. For any two matrices of dimension $N \times d$ and $d \times K$, such that $N, d, K \leq 2^s$, we can always pad them with rows and columns of zero entries and convert them to s-qubit operators. Thus, in the scenario where A and B are not s-qubit operators, one can consider block encodings of padded versions of these matrices. Note that this does not affect the operations on the sub-matrix blocks encoding A and B. Thus, the above results can be used to perform block-encoded matrix products for arbitrary (compatible) matrices.

Next we show how to find the block encoding of tensor product of matrices from their block encodings. This procedure will be useful in creating the dilated matrices required for regularization. The proof can be found in Appendix A.

Lemma 21 (Tensor Product of Block Encoded Matrices)**.** *Let U_1 and U_2 be $(\alpha, a, \varepsilon_1)$ and $(\beta, b, \varepsilon_2)$-block-encodings of A_1 and A_2, s and t-qubit operators, implemented in time T_1 and T_2 respectively. Define $S := \Pi_{i=1}^{s} \texttt{SWAP}_{a+b+i}^{a+i}$. Then, $S(U_1 \otimes U_2)S^\dagger$ is an $(\alpha\beta, a+b, \alpha\varepsilon_2 + \beta\varepsilon_1 + \varepsilon_1\varepsilon_2)$ block-encoding of $A_1 \otimes A_2$, implemented at a cost of $\mathcal{O}(T_1 + T_2)$.*

We will now use Lemma 21 to augment one matrix into another, given their approximate block-encodings.

Lemma 22 (Block-encoding of augmented matrix)**.** *If U_A is an $(\alpha_A, a_A, \varepsilon_A)$-block encoding of an s-qubit operator A that can be implemented in time T_A and U_B is an $(\alpha_B, a_B, \varepsilon_B)$-block encoding of an s-qubit operator B that can be implemented in time T_B, then we an implement an $(\alpha_A + \alpha_B, \max(a_A, a_B) + 2, \varepsilon_A + \varepsilon_B)$-block-encoding of*

$$A_B = \begin{pmatrix} A & 0 \\ B & 0 \end{pmatrix}$$

at a cost of $\mathcal{O}(T_A + T_B)$.

Proof. Let $M_A = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$. Then the SWAP gate is a $(1, 1, 0)$ block encoding of M_A. By Lemma 21, we can implement U'_A, an $(\alpha_A, a_A + 1, \varepsilon_A)$-block-encoding of their tensor product $M_A \otimes A = \begin{pmatrix} A & 0 \\ 0 & 0 \end{pmatrix}$ at a cost of $\mathcal{O}(T_A)$. Similarly, Let $M_B = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}$. Then $(I \otimes X) \cdot \texttt{SWAP}$ is a $(1, 1, 0)$-block-encoding of M_B. Similarly Lemma 21, we can implement U'_B, an $(\alpha_B, a_B + 1, \varepsilon_B)$-block-encoding of $M_B \otimes B = \begin{pmatrix} 0 & 0 \\ B & 0 \end{pmatrix}$ at a cost of $\mathcal{O}(T_B)$. We add them by using Corollary 18 on U'_A and U'_B, to implement U_{A_B}, an $(\alpha_A + \alpha_B, 2 + \max(a_A, a_B), \varepsilon_A + \varepsilon_B)$-block-encoding of $A_B = \begin{pmatrix} A & 0 \\ B & 0 \end{pmatrix}$. This can be implemented at a cost of $\mathcal{O}(T_A + T_B)$. □

3.2 Robust Quantum Singular Value Discrimination

The problem of deciding whether the eigenvalues of a Hamiltonian lie above or below a certain threshold, known as *eigenvalue discrimination*, finds widespread applications. For instance, the problem of determining whether the ground energy of a generic local Hamiltonian is $< \lambda_a$ or $> \lambda_b$ is known to be QMA-Complete [KKR06]. Nevertheless, quantum eigenvalue discrimination has been useful in preparing ground states of Hamiltonians. Generally, a variant of quantum phase estimation, which effectively performs a projection onto the eigenbasis of the underlying Hamiltonian, is used to perform eigenvalue discrimination [GTC19]. Recently, it has been shown that QSVT can be used to approximate a projection onto the eigenspace of an operator by implementing a polynomial approximation of the *sign function* [LT20a]. This was then used to design improved quantum algorithms for ground state preparation.

In our work, we design a more general primitive, known as *Quantum Singular Value Discrimination* (QSVD). Instead of eigenvalues, the algorithm distinguishes whether a singular value σ is $\leq \sigma_a$ or $\geq \sigma_b$. This is particularly useful when the block-encoded matrix is not necessarily Hermitian and hence, may not have well-defined eigenvalues. We use this procedure to develop a more space-efficient variable stopping time matrix inversion algorithm in Section 3.3. Owing to the widespread use of singular values in a plethora of fields, we believe that our QSVD procedure is of independent interest.

Let us define the *sign function* $\text{sign} : \mathbb{R} \to \mathbb{R}$ as follows:

$$\text{sign}(x) = \begin{cases} -1 & x < 0 \\ 0 & x = 0 \\ 1 & x > 0. \end{cases} \tag{22}$$

Given a threshold singular value c, Low and Chuang [LC17a] showed that there exists a polynomial approximation to $\text{sign}(c - x)$ (based on its approximation of the *erf function*). We use the result of Ref. [MRTC21], where such a polynomial of even parity was considered. This is crucial, as for even polynomials, QSVT maps right (left) singular vectors to right (left) singular vectors, which enables us to use the polynomial in [MRTC21] for singular value discrimination.

Lemma 23 (Polynomial approximation to the sign function [LC17a, Low17, MRTC21])**.** *For any $\varepsilon, \Delta, c \in (0,1)$, there exists an efficiently computable even polynomial $P_{\varepsilon,\Delta,c}(x)$ of degree $l = \mathcal{O}\left(\frac{1}{\Delta}\log(1/\varepsilon)\right)$ such that*

1. $\forall x \in [0,1] : |P_{\varepsilon,\Delta,c}(x)| \leq 1$

2. $\forall x \in [0,1] \setminus \left(c - \frac{\Delta}{2}, c + \frac{\Delta}{2}\right) : |P_{\varepsilon,\Delta,c}(x) - \text{sign}(c - x)| \leq \varepsilon$

Therefore, given a matrix A with singular values between $[0,1]$, we can use QSVT to implement $P_{\varepsilon,\Delta,c}(A)$ which correctly distinguishes between singular values of A whose value is less than $c - \Delta/2$ and those whose value is greater than $c + \Delta/2$. For our purposes, we shall consider that we are given U_A, which is an (α, a, ε) block-encoding of a matrix A. Our goal would be to distinguish whether a certain singular value σ satisfies $0 \leq \sigma \leq \varphi$ or $2\varphi \leq \sigma \leq 1$. Since U_A (approximately) implements A/α, the task can be rephrased as distinguishing whether a singular value of A/α is in $[0, \varphi/\alpha]$ or in $[2\varphi/\alpha, 1]$. For this, we develop a robust version of quantum singular value discrimination ($QSVD(\phi, \delta)$), which indicates the precision ε required to commit an error that is at most δ.

Theorem 24 (Quantum Singular Value Discrimination using QSVT)**.** *Suppose $A \in \mathbb{C}^{N \times N}$ is an s-qubit operator (where $N = 2^s$) with singular value decomposition $A = \sum_{j \in [N]} \sigma_j |u_j\rangle\langle v_j|$ such that all σ_j lie in the range $[0,1]$. Let $\varphi \in \left(0, \frac{1}{2}\right)$ and $\delta \in (0,1]$ be some parameters. Suppose that for some $\alpha \geq 2$ and ε satisfying*

$$\varepsilon = o\left(\frac{\delta\varphi}{\log(1/\delta)}\right)$$

we have access to U_A, an (α, a, ε)-block-encoding of A implemented in cost T_A. Then there exists a quantum algorithm $QSVD(\varphi, \delta)$ which implements a $(1, a+1, \delta)$-block-encoding of some $(s+1)$-qubit operator $D \in \mathbb{C}^{2N \times 2N}$ satisfying the following constraints for all $j \in [N]$:

- $\sigma_j \leq \varphi \implies D|0\rangle|v_j\rangle = |0\rangle|v_j\rangle$
- $\sigma_j \geq 2\varphi \implies D|0\rangle|v_j\rangle = |1\rangle|v_j\rangle$

This algorithm has a cost of

$$\mathcal{O}\left(\frac{\alpha}{\varphi}\log\left(\frac{1}{\delta}\right)T_A\right).$$

Proof. We invoke Lemma 23 with parameters $\varepsilon' := \frac{\delta}{2}$, $c := \frac{3\varphi}{2\alpha}$ and $\Delta := \frac{\varphi}{2\alpha}$, to construct an even polynomial $P := P_{\varepsilon',\Delta,c}$ of degree $n := \mathcal{O}\left(\frac{\alpha}{\varphi}\log\left(\frac{1}{\varepsilon'}\right)\right)$, which is an ε'-approximation of $f(x) := \operatorname{sign}\left(\frac{3\varphi}{2\alpha} - x\right)$ for $x \in \left[0, \frac{\varphi}{\alpha}\right] \cup \left[\frac{2\varphi}{\alpha}, 1\right]$. Invoking Theorem 8 with P and U_A, we get U_B – a $(1, a+1, \gamma)$-block-encoding of $B := P(A/\alpha)$, implemented in cost $\mathcal{O}(nT_A)$, where ε must satisfy $\varepsilon \leq \alpha\gamma/2n$.

Now consider the following unitary W that acts on $s + a + 2$ qubits:

$$W := \mathtt{SWAP}^{\dagger}_{[s,s+a+1]}(H \otimes I_{s+a+1})\,(C\text{-}U_B)\,(H \otimes I_{s+a+1})\mathtt{SWAP}_{[s,s+a+1]}$$

W is the required block-encoding of D, and $\mathtt{SWAP}_{[l,r]}$ sequentially swaps adjacent qubits with indices in range $[l, r]$ effectively moving qubit indexed l to the right of qubit r. (where qubits are zero-indexed, with higher indices for ancillas). Let $\tilde{B}$ be the top-left block of U_B (therefore $\left\|B - \tilde{B}\right\| \leq \gamma$). Then we can extract $\tilde{D}$, the top-left block of W as follows:

$$\begin{aligned}\tilde{D} &= \left(\langle 0|^{\otimes a+1} \otimes I_{s+1}\right)\mathtt{SWAP}^{\dagger}_{[s,s+a+1]}\left(|+\rangle\langle+| \otimes I_{s+a+1} + |-\rangle\langle-| \otimes U_B\right)\mathtt{SWAP}_{[s,s+a+1]}\left(|0\rangle^{\otimes a+1} \otimes I_{s+1}\right)\\ &= |+\rangle\langle+| \otimes I_s + |-\rangle\langle-| \otimes \tilde{B}\end{aligned}$$

Let us define index sets $L, R \subseteq [N]$ where $L := \{j \in [N] \mid \sigma_j \leq \varphi\}$ and $R := \{j \in [N] \mid \sigma_j \geq 2\varphi\}$; and the corresponding subspace projections $\Pi_L := \sum_{j\in L} |v_j\rangle\langle v_j|$, $\Pi_R := \sum_{j\in R} |v_j\rangle\langle v_j|$, and $\Pi_\perp := I_s - \Pi_L - \Pi_R$. Using these we pick our required operator D as follows:

$$D := I \otimes \Pi_L + X \otimes \Pi_R + \tilde{D}(I \otimes \Pi_\perp)$$

That is, D behaves as expected on the required subspace, and acts identical to $\tilde{D}$ on the remaining space. The error in the block-encoding can be computed as

$$\begin{aligned}\left\|D - \tilde{D}\right\| &= \left\|I \otimes \Pi_L + X \otimes \Pi_R + \tilde{D}(I \otimes \Pi_\perp) - \tilde{D}\right\|\\ &= \left\|I \otimes \Pi_L + X \otimes \Pi_R - \tilde{D}(I \otimes (\Pi_L + \Pi_R))\right\|\\ &= \left\|(I \otimes I_s - \tilde{D})(I \otimes \Pi_L) + (X \otimes I_s - \tilde{D})(I \otimes \Pi_R)\right\|\\ &= \left\|\left(|-\rangle\langle-| \otimes (I_s - \tilde{B})\right)(I \otimes \Pi_L) - \left(|-\rangle\langle-| \otimes (I_s + \tilde{B})\right)(I \otimes \Pi_R)\right\|\\ &= \left\|(I_s - \tilde{B})\Pi_L - (I_s + \tilde{B})\Pi_R\right\|\\ &= \left\|(I_s - B)\Pi_L - (I_s + B)\Pi_R + (B - \tilde{B})(\Pi_L - \Pi_R)\right\|\\ &\leq \|(I_s - P(A/\alpha))\Pi_L - (I_s + P(A/\alpha))\Pi_R\| + \left\|B - \tilde{B}\right\|\|\Pi_L - \Pi_R\|\\ &\leq \varepsilon' + \gamma\end{aligned}$$

We can choose $\gamma = \delta/2$, therefore

$$\varepsilon \leq \frac{\alpha\delta}{4n} = o\left(\frac{\delta\varphi}{\log\left(\frac{1}{\delta}\right)}\right)$$

□

In Section 3.3, we develop a variable stopping time quantum algorithm for matrix inversion using QSVT. In order to recast the usual matrix inversion to the VTAA framework, we

need to be able to apply this algorithm to specific ranges of the singular values of the matrix. This is achieved by applying a controlled QSVD algorithm, to determine whether the input singular vector corresponds to an singular value less than (or greater than) a certain threshold. Based on the outcome of controlled QSVD, the standard inversion algorithm is applied. These two steps correspond to sub-algorithms $\mathcal{A}_j$ of the VTAA framework.

In prior works such as Refs. [Amb12, CKS17, CGJ19], gapped phase estimation (GPE) was used to implement this. GPE requires an additional register of $\mathcal{O}(\log(\kappa)\log(1/\delta))$ qubits to store the estimated phases. For the whole VTAA procedure, $\log\kappa$ such registers are needed. As a result, substituting GPE with QSVD, we save $\mathcal{O}\Big(\log^2(\kappa)\log(1/\delta)\Big)$ qubits.

3.3 Variable-Time Quantum Algorithm for Matrix Inversion using QSVT

Matrix inversion by QSVT applies a polynomial approximation of $f(x) = 1/x$, satisfying the constraints laid out in Section 2.3. Here, we make use of the result of [MRTC21] to implement A^+. We adapt their result to the scenario where we have an approximate block-encoding of A as input. Finally, we convert this to a variable stopping time quantum algorithm and apply VTAA to obtain a linear dependence on the condition number of A.

Lemma 25 (Matrix Inversion polynomial (Appendix C of [MRTC21]))**.** *Given $\kappa \geq 1, \varepsilon \in \mathbb{R}^+$, there exists an odd QSP polynomial $P^{MI}_{\varepsilon,\kappa}$ of degree $\mathcal{O}(\kappa\log(\kappa/\varepsilon))$, which is an $\frac{\varepsilon}{2\kappa}$ approximation of the function $f(x) = \frac{1}{2\kappa x}$ in the range $\mathcal{D} := [-1, -\frac{1}{\kappa}] \cup [\frac{1}{\kappa}, 1]$. Also in this range $P^{MI}_{\varepsilon,\kappa}$ is bounded from above by 1, i.e. $\forall x \in \mathcal{D} : \left|P^{MI}_{\varepsilon,\kappa}(x)\right| \leq 1$.*

Theorem 26 (Inverting Normalized Matrices using QSVT)**.** *Let A be a normalized matrix with non-zero singular values in the range $[1/\kappa_A, 1]$ for some $\kappa_A \geq 1$. Let $\delta \in (0, 1]$. For some $\varepsilon = o\left(\frac{\delta}{\kappa_A^2\log(\kappa_A/\delta)}\right)$ and $\alpha \geq 2$, let U_A be an (α, a, ε)-block-encoding of A, implemented in time T_A. Then we can implement a $(2\kappa_A, a+1, \delta)$-block-encoding of A^+ at a cost of*

$$\mathcal{O}\left(\kappa_A\alpha\log\left(\frac{\kappa_A}{\delta}\right)T_A\right).$$

Proof. We use the matrix inversion polynomial defined in Lemma 25, $P := P^{MI}_{\phi,\kappa}$ for this task, with $\kappa = \kappa_A\alpha$ and an appropriate ϕ. This has a degree of $n := \mathcal{O}(\kappa_A\alpha\log(\kappa_A\alpha/\phi))$. We invoke Theorem 8 to apply QSVT using the polynomial P above, block-encoding U_A, and an appropriate error parameter γ such that $\varepsilon \leq \alpha\gamma/2n$, to get the unitary U, a $(1, a+1, \gamma)$-block-encoding of $P(A/\alpha)$. As P is a $(\phi/2\kappa)$-approximation of $f(x) := 1/2\kappa x$, we have

$$\|f(A/\alpha) - P(A/\alpha)\| \leq \frac{\phi}{2\kappa},$$

which implies U is a $(1, a+1, \gamma+\phi/2\kappa)$-block-encoding of $f(A/\alpha)$. And because $f(A/\alpha) = \frac{\alpha A^+}{2\kappa} = A^+/2\kappa_A$, we can re-interpret U as a $(2\kappa_A, a+1, 2\kappa_A\gamma + \phi/\alpha)$-block-encoding of A^+. Choosing $2\kappa_A\gamma = \phi/\alpha = \delta/2$, the final block-encoding has an error of δ. This gives us $\phi = \alpha\delta/2$ and $\gamma = \delta/4\kappa_A$, and

$$\varepsilon \leq \frac{\alpha\gamma}{2n} = \frac{\alpha\delta}{8\kappa_A n} = \mathcal{O}\left(\frac{\delta}{\kappa_A^2\log(\kappa_A/\delta)}\right)$$

□

Next, we design a map $W(\gamma, \delta)$ that uses QSVT to invert the singular values of a matrix if they belong to a particular domain. This helps us recast the usual matrix inversion algorithm as a variable-stopping-time algorithm and will be a key subroutine for boosting the success probability of this algorithm using VTAA. This procedure was also used in Refs. [CKS17, CGJ19] for the quantum linear systems algorithms.

Theorem 27 (Efficient inversion of block-encoded matrix)**.** *Let A be a normalized matrix with non-zero singular values in the range $[1/\kappa, 1]$, for some $\kappa \geq 1$. Let $\delta \in (0, 1]$; $0 < \gamma \leq 1$. Let U_A be an (α, a, ε)-block-encoding of A implemented in time T_A, such that $\alpha \geq 2$ and $\varepsilon = o\left(\frac{\delta\gamma^2}{\log\left(\frac{1}{\delta\gamma}\right)}\right)$. Then for any quantum state $|b\rangle$ that is spanned by the left singular vectors of A corresponding to the singular values in the range $[\gamma, 1]$, there exists a unitary $W(\gamma, \delta)$ that implements*

$$W(\gamma, \delta) : |0\rangle_F |0\rangle_Q |b\rangle_I \mapsto \frac{1}{a_{\max}} |1\rangle_F |0\rangle_Q f(A) |b\rangle_I + |0\rangle_F |\perp\rangle_{QI} \tag{23}$$

where $a_{\max} = \mathcal{O}(\kappa_A)$ is a constant independent of γ, $|\perp\rangle_{QI}$ is an unnormalized quantum state orthogonal to $|0\rangle_Q$ and $\|f(A)|b\rangle - A^+|b\rangle\| \leq \delta$. Here F is a 1-qubit flag register, Q is an α-qubit ancilla register, and I is the $\log N$-qubit input register. This unitary has a cost

$$\mathcal{O}\left(\frac{\alpha}{\gamma} \log\left(\frac{1}{\gamma\delta}\right) T_A\right) \tag{24}$$

Proof. Since we only need to invert the singular values in a particular range, we can use the procedure in Theorem 26 with κ_A modified to the restricted range. That gives us the description of a quantum circuit $\widetilde{W}(\gamma, \delta)$ that can implement the following map

$$\widetilde{W}(\gamma, \delta) : |b\rangle_I |0\rangle_Q \mapsto \frac{\gamma}{2} f(A) |b\rangle_I |0\rangle_Q + |\perp\rangle_{QI},$$

where $|\perp\rangle$ is an unnomalized state with no component along $|0\rangle_Q$. This has the same cost as Equation 24. Here $\|f(A)|\psi\rangle - A^+|\psi\rangle\| \leq \delta$ whenever $|\psi\rangle$ is a unit vector in the span of the singular vectors of A corresponding to the singular values in $[\gamma, 1]$. This follows from the sub-multiplicativity property of the matrix-vector product.

Next, we must transform the amplitude of the good part of the state to $\Theta(\kappa)$, independent of γ. To achieve this, we will have to flag it with an ancillary qubit to use a controlled rotation to modify the amplitude. Thus we add a single qubit $|0\rangle_F$ register and flip this register controlled on register Q being in the state $|0\rangle$ (the good part). This gives us the transformation

$$\widetilde{W}'(\gamma, \delta) : |0\rangle_F |b\rangle_I |0\rangle_Q \mapsto \frac{\gamma}{2} |1\rangle_F f(A) |b\rangle_I |0\rangle_Q + |0\rangle_F |\perp\rangle_{QI}$$

Then we use a controlled rotation to replace the amplitude $\gamma/2$ with some constant $a_{\max}^{-1}$ which is independent of γ, which is achieved by introducing the relevant phase to the flag space

$$|1\rangle_F \mapsto \frac{2}{\gamma a_{\max}} |1\rangle_F + \sqrt{1 - \frac{4}{\gamma^2 a_{\max}^2}} |0\rangle_F.$$

This gives us the desired $W(\gamma, \delta)$ as in Equation 23. □

Given such a unitary $W(\gamma, \delta)$, Ref. [CGJ19] laid out a procedure for a variable time quantum algorithm $\mathcal{A}$ that takes as input the block encoding of an $N \times d$ matrix A, and a state preparation procedure $U_b : |0\rangle^{\otimes n} \mapsto |b\rangle$, and outputs a quantum state that is a

bounded distance away from $A^+ |b\rangle / \|A^+ |b\rangle\|$. In order to determine the branches of the algorithm on which to apply VTAA at a particular iteration, [CKS17, CGJ19, Amb12] use the technique of gapped phase estimation, which given a unitary U, a threshold ϕ and one of its eigenstate $|\lambda\rangle$, decides if the corresponding eigenvalue is a bounded distance below the threshold, or a bounded distance above it. In this work, we replace gapped phase estimation with the QSVD algorithm (Theorem 24) which can be applied directly to any block-encoded (not necessarily Hermitian) matrix A, and allows for saving on $\mathcal{O}\Big(\log^2(\kappa/\delta)\Big)$ qubits.

The Variable time Algorithm: This algorithm will be a sequence of m sub-algorithms $\mathcal{A} = \mathcal{A}_m \cdot \mathcal{A}_{m-1} \cdot \ldots \mathcal{A}_1$, where $m = \lceil \log \kappa \rceil + 1$. The overall algorithm acts on the following registers:

- m single qubit clock registers $C_i : i \in [m]$.
- An input register I, initialized to $|0\rangle^{\otimes s}$.
- Ancillary register space Q for the block encoding of A, initialized to $|0\rangle^{\otimes a}$.
- A single qubit flag register $|0\rangle_F$ used to flag success of the algorithm.

Once we have prepared the above state space, we use the state preparation procedure to prepare the state $|b\rangle$. Now we can define how each $\mathcal{A}_j$ acts on the state space. Let $\varepsilon' = \frac{\delta}{a_{\max} m}$. The action of $\mathcal{A}_j$ can be broken down into two parts:

1. If $C_{j-1} \ldots C_1$ is in state $|0\rangle^{\otimes(j-1)}$, apply QSVD$(2^{-j}, \varepsilon')$, (Theorem 24) to the state $|b\rangle$. The output is to be written to the clock register C_j.
2. If the state of C_j is now $|1\rangle$, apply $W(2^{-j}, \varepsilon')$ to $I \otimes F \otimes Q$.

Additionally, we would need algorithms $\mathcal{A}' = \mathcal{A}'_m \cdots \mathcal{A}'_1$ which are similar to $\mathcal{A}$, except that in Step 2, it implements W' which sets the flag register to 1. That is,

$$W' |b\rangle_I |0\rangle_F |0\rangle_Q = |b\rangle_I |1\rangle_F |0\rangle_Q .$$

Now we are in a position to define the variable time quantum linear systems algorithm using QSVT.

Theorem 28 (Variable Time Quantum Linear Systems Algorithm Using QSVT)**.** *Let $\varepsilon, \delta > 0$. Let A is a normalized $N \times d$ matrix such that its non-zero singular values lie in $[1/\kappa, 1]$. Suppose that for*

$$\varepsilon = o\left(\frac{\delta}{\kappa^3 \log^2\left(\frac{\kappa}{\delta}\right)}\right),$$

we have access to U_A which is an (α, a, ε)-block-encoding of A, implemented with cost T_A. Let $|b\rangle$ be a state vector which is spanned by the left singular vectors of A. Suppose there exists a procedure to prepare the state $|b\rangle$ in cost T_b. Then there exists a variable time quantum algorithm that outputs a state that is δ-close $\frac{A^+|b\rangle}{\|A^+|b\rangle\|}$ at a cost of

$$\mathcal{O}\left(\kappa \log \kappa \left(\alpha T_A \log\left(\frac{\kappa}{\delta}\right) + T_b\right)\right) \tag{25}$$

using $\mathcal{O}(\log(\kappa))$ additional qubits.

Proof. The correctness of the algorithm is similar to that of Refs. [CKS17, CGJ19], except here, we use QSVD instead of gapped phase estimation. According to Lemma 10, we need $T_{\max}$ (the maximum time any of the sub-algorithms $\mathcal{A}_j$ take), $\|T\|_2^2$ (the ℓ_2-averaged stopping time of the sub-algorithms), and $\sqrt{p_{\text{succ}}}$ (the square root of the success probability.) Now each sub-algorithm consists of two steps, implementing QEVD with precision 2^{-j} and error ε', followed by $W(2^{-j}, \varepsilon')$. From Theorem 24, the first step costs

$$\mathcal{O}\left(\alpha T_A 2^j \log\left(\frac{1}{\varepsilon'}\right)\right),$$

and the cost of implementing $W(2^{-j}, \varepsilon')$ is as described in Equation 24. Thus the overall cost of $\mathcal{A}_j$, which is the sum of these two costs, turns out to be

$$\mathcal{O}\left(\alpha T_A 2^j \log\left(\frac{2^j}{\varepsilon'}\right)\right) \tag{26}$$

Note that the time t_j required to implement $\mathcal{A}_j \dots \mathcal{A}_1$ is also the same as Equation 26. Also,

$$\begin{aligned}
T_{\max} &= \max_j \; t_j \\
&= \max_j \; \mathcal{O}\left(\alpha T_A 2^j \log\left(\frac{2^j}{\varepsilon'}\right)\right) \\
&= \mathcal{O}\left(\alpha T_A \kappa \log\left(\frac{\kappa}{\varepsilon'}\right)\right) \\
&= \mathcal{O}\left(\alpha T_A \kappa \log\left(\frac{\kappa \log(\kappa)}{\delta}\right)\right).
\end{aligned}$$

The $\|T\|_2^2$ is dependent on the probability that $\mathcal{A}$ stops at the j^{th} step. This is given by $p_j = \left\|\Pi_{C_j} \mathcal{A}_j \dots \mathcal{A}_1 |\psi\rangle_I |0\rangle_{CFPQ}\right\|^2$, where Π_{C_j} is the projector on $|1\rangle_{C_j}$, the j^{th} clock register. From this, $\|T\|_2^2$ can be calculated as

$$\begin{aligned}
\|T\|_2^2 &= \sum_j p_j t_j^2 \\
&= \sum_j \left\|\Pi_{C_j} \mathcal{A}_j \dots \mathcal{A}_1 |\psi\rangle_I |0\rangle_{CFPQ}\right\|^2 t_j^2 \\
&= \sum_k |c_k|^2 \sum_j \left(\left\|\Pi_{C_j} \mathcal{A}_j \dots \mathcal{A}_1 |v_k\rangle_I |0\rangle_{CFPQ}\right\|^2 t_j^2\right) \\
&= \mathcal{O}\left(\alpha^2 T_A^2 \sum_k \log^2\left(\frac{1}{\sigma_k \varepsilon'}\right) \frac{|c_k|^2}{\sigma_k^2}\right)
\end{aligned}$$

Therefore

$$\|T\|_2 = \mathcal{O}\left(\alpha T_A \log\left(\frac{\kappa \log \kappa}{\delta}\right) \sqrt{\sum_k \frac{|c_k|^2}{\sigma_k^2}}\right). \tag{27}$$

Next we calculate the success probability.

$$\sqrt{p_{\text{succ}}} = \left\|\Pi_F \frac{A^{-1}}{\alpha_{\max}} |b\rangle_I |\phi\rangle_{CFPQ}\right\| + \mathcal{O}(m\varepsilon')$$

$$= \frac{1}{\alpha_{\max}} \sqrt{\sum_j \frac{|c_j|^2}{\sigma_j^2}} + \mathcal{O}\left(\frac{\delta}{\alpha_{\max}}\right)$$
$$= \Omega\left(\frac{1}{\kappa} \sqrt{\sum_j \frac{|c_j|^2}{\sigma_j^2}}\right)$$

Given these, we can use Lemma 10 to write the final complexity of matrix inversion with VTAA:

$$T_{\max} + T_b + \frac{(\|T\|_2 + T_b) \log(T'_{\max})}{\sqrt{p_{\text{succ}}}} = \mathcal{O}\left(\kappa \log \kappa \left(\alpha T_A \log\left(\frac{\kappa}{\delta}\right) + T_b\right)\right)$$

The upper bound on the precision required for the input block-encoding, ε, can be calculated from the bounds on the precisions for $W(\kappa, \varepsilon')$ (Theorem 27) and $\text{QSVD}(\kappa, \varepsilon')$ (Theorem 24) as follows:

$$\varepsilon = o\left(\min\left(\frac{\varepsilon'}{\kappa^2 \log\left(\frac{\kappa}{\varepsilon'}\right)}, \frac{\varepsilon'}{\kappa \log\left(\frac{1}{\varepsilon'}\right)}\right)\right) = o\left(\frac{\varepsilon'}{\kappa^2 \log\left(\frac{\kappa}{\varepsilon'}\right)}\right) = o\left(\frac{\delta}{\kappa^3 \log^2\left(\frac{\kappa}{\delta}\right)}\right)$$

□

The overall complexity is better by a log factor and requires $\mathcal{O}\left(\log^2(\kappa/\delta)\right)$ fewer additional qubits as compared to the variable time algorithms in Refs. [CKS17, CGJ19].

3.4 Negative Powers of Matrices using QSVT

We consider the problem: given an approximate block-encoding of a matrix A, we need to prepare a block-encoding of A^{-c}, where $c \in (0, 1)$. This procedure will be used to develop algorithms for ℓ_2-regularized versions of GLS. We will directly use the results of [GSLW19].

Lemma 29 (Polynomial approximations of negative power functions, [GSLW18], Corollary 67)**.** *Let $\varepsilon, \delta \in (0, \frac{1}{2}], c > 0$ and let $f(x) := \frac{\delta^c}{2} x^{-c}$, then there exist even/odd polynomials $P_{c,\varepsilon,\delta}, P'_{c,\varepsilon,\delta} \in \mathbb{R}[x]$ such that $\|P_{c,\varepsilon,\delta} - f\|_{[\delta,1]} \leq \varepsilon$, $\|P_{c,\varepsilon,\delta}\|_{[-1,1]} \leq 1$ and $\left\|P'_{c,\varepsilon,\delta} - f\right\|_{[\delta,1]} \leq \varepsilon$, $\left\|P'_{c,\varepsilon,\delta}\right\|_{[-1,1]} \leq 1$. Moreover the degree of the polynomials are $\mathcal{O}\left(\frac{\max(1,c)}{\delta} \log\left(\frac{1}{\varepsilon}\right)\right)$.*

Theorem 30 (Negative fractional powers of a normalized matrix using QSVT)**.** *Let $c \in (0, 1)$ be some constant and $\delta \in (0, 1]$ Let A be a normalized matrix with non-zero singular values in the range $[1/\kappa, 1]$. Let U_A be a (α, a, ε)-block-encoding of a matrix A, implemented in time T_A such that $\alpha \geq 2$ and*

$$\varepsilon = o\left(\frac{\delta}{\kappa^{c+1} \log(\kappa/\delta)}\right)$$

Then we can construct a $(2\kappa^c, a + 1, \delta)$-block-encoding of A^{-c} at a cost of

$$\mathcal{O}\left(\alpha\kappa \log\left(\frac{\kappa}{\delta}\right) \; T_A\right).$$

Proof. From Lemma 29, using $\Delta := \frac{1}{\kappa\alpha}$ and an appropriate $\varphi \in (0, \frac{1}{2}]$, we get an even QSP polynomial $P := P_{c,\varphi,\Delta}$ which is φ-close to $f(x) := \frac{1}{2\kappa^c \alpha^c x^c}$, and has degree n such that $n = \mathcal{O}\left(\alpha\kappa \log\left(\frac{1}{\varphi}\right)\right)$. Therefore

$$\|f(A/\alpha) - P(A/\alpha)\| \leq \varphi.$$

Using Theorem 8 we can construct U_P, a $(1, a+1, \gamma)$-block-encoding of $P(A/\alpha)$, given that $\varepsilon \leq \frac{\alpha\gamma}{2n}$. Then from triangle inequality it follows that it is a $(1, a+1,\ \varphi+\gamma)$-block-encoding of $f(A/\alpha)$. And because $f(A/\alpha) = \frac{A^{-c}}{2\kappa^c}$, U_P can be re-interpreted as a $(2\kappa^c, a+1, 2\kappa^c(\varphi+\gamma))$-block-encoding of A^{-c}. We therefore choose $\varphi = \gamma = \frac{\delta}{4\kappa^c}$, and choose ε as

$$\varepsilon = o\left(\alpha\frac{\delta}{4\kappa^c}\frac{1}{\alpha\kappa\log(4\kappa^c/\delta)}\right) = o\left(\frac{\delta}{\kappa^{c+1}\log(\kappa/\delta)}\right)$$

□

Having discussed the necessary algorithmic primitives, we are now in a position to design quantum algorithms for linear regression with general ℓ_2-regularization. We will first deal with ordinary least squares followed by weighted and generalized least squares.

4 Quantum Least Squares with General ℓ_2-Regularization

In this section, we derive the main results of our paper, namely quantum algorithms for quantum ordinary least squares (OLS), quantum weighted least squares (WLS) and quantum generalized least squares (GLS) with ℓ_2-regularization.

4.1 Quantum Ordinary Least Squares

Given N data points $\{a_i, b_i\}_{i=1}^N$ such that $a_i \in \mathbb{R}^d$ and $b_i \in \mathbb{R}$, the objective of linear regression is to find $x \in \mathbb{R}^d$ that minimizes the loss function

$$\mathcal{L}_O = \sum_{j=1}^{N}(x^T a_i - b_i)^2. \tag{28}$$

Consider the $N \times d$ matrix A (known as the data matrix) such that the i^{th} row of A is the vector a_i transposed and the column vector $b = (b_1 \cdots b_N)^T$. Then, the solution to the OLS problem is given by $x = (A^T A)^{-1} A^T b = A^+ b$.

For the ℓ_2-regularized version of the OLS problem, a penalty term is added to its objective function. This has the effect of shrinking the singular values of A which helps overcome problems such as rank deficiency and *overfitting* for the OLS problem. The loss function to be minimized is of the form

$$\|Ax - b\|_2^2 + \|Lx\|_2^2, \tag{29}$$

where L is the $N \times d$ penalty matrix and $\lambda > 0$ is the optimal regularizing parameter. The solution $x \in \mathbb{R}^d$ satisfies

$$x = (A^T A + \lambda L^T L)^{-1} A^T b. \tag{30}$$

Therefore, for quantum ordinary least squares with general ℓ_2-regularization, we assume that we have access to approximate block-encodings of the data matrix A, L and a procedure to prepare the quantum state $|b\rangle = \sum_{j=1}^N b_j |j\rangle / \|b\|$. Our algorithm outputs a quantum state that is close to

$$|x\rangle = \frac{(A^T A + \lambda L^T L)^{-1} A^T |b\rangle}{\|(A^T A + \lambda L^T L)^{-1} A^T |b\rangle\|}. \tag{31}$$

In order to implement a quantum algorithm that implements this, a straightforward approach would be the following: We first construct block-encodings of $A^T A$ and $L^T L$,

given block encodings of A and L, respectively (Using Lemma 19). We could then implement a block-encoding of $A^TA + \lambda L^TL$ using these block encodings (By Lemma 17). On the other hand, we could also prepare a quantum state proportional to $A^T|b\rangle$ by using the block-encoding for A and the unitary preparing $|b\rangle$. Finally, using the block encoding of $A^TA+\lambda L^TL$, we could implement a block-encoding of $(A^TA+\lambda L^TL)^{-1}$ (using Theorem 26) and apply it to the state $A^T|b\rangle$. Although this procedure would output a quantum state close to $|x\rangle$, it is not efficient. It is easy to see that the inverse of $A^TA + \lambda L^TL$, would be implemented with a complexity that has a quadratic dependence on the condition numbers of A and L. This would be undesirable as it would perform worse than the unregularized quantum least squares algorithm, where one is able to implement A^+ directly. However, it is possible to design a quantum algorithm that performs significantly better than this.

The first observation is that it is possible to recast this problem as finding the pseudoinverse of some augmented matrix. Given the data matrix $A \in \mathbb{R}^{N\times d}$, the regularizing matrix $L \in \mathbb{R}^{N\times d}$, let us define the following augmented matrix

$$A_L := \begin{pmatrix} A & 0 \\ \sqrt{\lambda}L & 0 \end{pmatrix}. \tag{32}$$

It is easy to see that the top left block of $A_L^+ = (A^TA + \lambda L^TL)^{-1}A^T$, which is the required linear transformation to be applied to b. Consequently, our strategy would be to implement a block-encoding of A_L, given block-encodings of A and L. Following this, we use matrix inversion by QSVT to implement $A_L^+|b\rangle|0\rangle$. The first register is left in the quantum state given in Equation 31.

From this, it is clear that the complexity of our quantum algorithm would depend on the effective condition number of the augmented matrix A_L. In this regard, we shall assume that the penalty matrix L is a *good regularizer*. That is, L is chosen such that it does not have zero singular values (positive definite). This is a fair assumption as if L has only non-zero singular values, the minimum singular value of A_L is guaranteed to be lower bounded by the minimum singular value of L. This ensures that the effective condition number of A_L depends on κ_L, even when the data matrix A has zero singular values and A^TA is not invertible. Consequently, this also guarantees that regularized least squares provide an advantage over their unregularized counterparts.

Next, we obtain bounds on the effective condition number of the augmented matrix A_L for a good regularizer L via the following lemma:

Lemma 31 (Condition number and Spectral Norm of A_L)**.** *Let the data matrix A and the positive definite penalty matrix L have spectral norms $\|A\|$ and $\|L\|$, respectively. Furthermore, suppose their effective condition numbers be upper bounded by κ_A and κ_L. Then the ratio between the maximum and minimum (non-zero) singular value of A_L is upper bounded by*

$$\kappa = \kappa_L\left(1 + \frac{\|A\|}{\sqrt{\lambda}\|L\|}\right)$$

We can also bound the spectral norm as

$$\|A_L\| = \Theta\left(\|A\| + \sqrt{\lambda}\|L\|\right)$$

Proof. To bound the spectral norm and condition number of A_L, consider the eigenvalues of the following matrix:

$$A_L^TA_L = \begin{pmatrix} A^TA + \lambda L^TL & 0 \\ 0 & 0 \end{pmatrix}$$

This implies that the non-zero eigenvalues of $A_L^T A_L$ are the same as those of $A^T A + \lambda L^T L$. Therefore, using triangle inequality, the spectral norm of A_L can be upper-bounded as follows:

$$\|A_L\| = \sqrt{\|A_L^T A_L\|} = \sqrt{\|A^T A + \lambda L^T L\|} \leq \sqrt{\|A^T A\| + \lambda \|L^T L\|} = \sqrt{\|A\|^2 + \lambda \|L\|^2} \leq \|A\| + \sqrt{\lambda}\|L\|$$

Similarly $\|A_L\| \geq \|A\|$ and $\|A_L\| \geq \sqrt{\lambda}\|L\|$, which effectively gives the tight bound for $\|A_L\|$.

As $L^T L$ is positive definite, we have that its minimum singular value is $\sigma_{\min(L)} = \|L\|/\kappa_L$. And we also know that $A^T A$ is positive semidefinite, so by Weyl's inequality, the minimum singular value of A_L is lower bounded by

$$\sigma_{\min}(A_L) \geq \sqrt{\sigma_{\min}(A)^2 + \lambda \sigma_{\min}(L)^2} \geq \sqrt{\lambda \frac{\|L\|^2}{\kappa_L^2}} = \sqrt{\lambda}\frac{\|L\|}{\kappa_L}$$

Thus,

$$\frac{\sigma_{\max}(A_L)}{\sigma_{\min}(A_L)} \leq \kappa = \kappa_L \left(1 + \frac{\|A\|}{\sqrt{\lambda}\|L\|}\right)$$

□

In the theorems and lemmas for regularized quantum linear regression and its variants that we develop in this section, we consider that L is a *good regularizer* in order to provide a simple expression for κ. However, this is without loss of generality. When L is not a good regularizer, the expressions for the respective complexities will remain unaltered, except that κ would now correspond to the condition number of the augmented matrix.

Now it might be possible that $|b\rangle$ does not belong to the row space of $(A^T A + \lambda L^T L)^{-1} A^T$ which is equivalent to saying $|b\rangle|0\rangle$ may not lie in $\text{row}(A_L^+)$. However, it is reasonable to expect that the initial hypothesis of the underlying model being close to linear is correct. That is, we expect $|b\rangle$ to have a good overlap with $\text{row}\left(A_L^+\right) = \text{col}(A_L)$. The quantity that quantifies how far the model is from being linear is the so called *normalized residual sum of squares*. For ℓ_2-regularized ordinary least squares, this is given by

$$\mathcal{S}_O = \frac{\left\|\left(I - \Pi_{\text{col}(A_L)}\right)|b\rangle|0\rangle\right\|^2}{\||b\rangle\|^2} = 1 - \left\|\Pi_{\text{col}(A_L)}|b\rangle|0\rangle\right\|^2. \tag{33}$$

If the underlying data can indeed be fit by a linear function, $\mathcal{S}_O$ will be low. Subsequently, we assume that $\mathcal{S}_O = 1 - \left\|\Pi_{\text{col}(A_L)}|b\rangle|0\rangle\right\|^2 \leq \gamma < 1/2$. This in turn implies that $\left\|\Pi_{\text{col}(A_L)}|b\rangle|0\rangle\right\|^2 = \Omega(1)$, implying that the data can be reasonably fit by a linear model.[i]

Now we are in a position to present our quantum algorithm for the quantum least squares problem with general ℓ_2-regularization. We also present an improved quantum algorithm for the closely related quantum ridge regression, which is a special case of the former.

Theorem 32 (Quantum Ordinary Least Squares with General ℓ_2-Regularization)**.** *Let $A, L \in \mathbb{R}^{N\times d}$ be the data and penalty matrices with effective condition numbers κ_A and*

[i]Our results also hold if we assume that $\mathcal{S}_O \leq \gamma$ for some $\gamma \in (0,1)$. That is, $\left\|\Pi_{\text{col}(A_L)}\right\| \geq 1 - \gamma$. In such a scenario our complexity to prepare $A_L^+|b,0\rangle / \left\|A_L^+|b,0\rangle\right\|$ is rescaled by $1/\sqrt{1-\gamma}$.

κ_L *respectively, and* $\lambda \in \mathbb{R}^+$ *be the regression parameter. Let* U_A *be a* $(\alpha_A, a_A, \varepsilon_A)$*-block-encoding of* A *implemented in time* T_A *and* U_L *be a* $(\alpha_L, a_L, \varepsilon_L)$*-block-encoding of* L *implemented in time* T_L*. Furthermore, suppose* U_b *be a unitary that prepares* $|b\rangle$ *in time* T_b *and define*

$$\kappa = \kappa_L \left(1 + \frac{\|A\|}{\sqrt{\lambda}\|L\|}\right)$$

Then for any $\delta \in (0,1)$ *such that*

$$\varepsilon_A, \sqrt{\lambda}\varepsilon_L = o\left(\frac{\delta}{\kappa^3 \log^2\left(\frac{\kappa}{\delta}\right)}\right) \tag{34}$$

we can prepare a state that is δ*-close to*

$$\frac{\left(A^T A + \lambda L^T L\right)^{-1} A^T |b\rangle}{\left\|\left(A^T A + \lambda L^T L\right)^{-1} A^T |b\rangle\right\|}$$

with probability $\Theta(1)$*, at a cost of*

$$\mathcal{O}\left(\kappa \log \kappa \left(\left(\frac{\alpha_A + \sqrt{\lambda}\alpha_L}{\|A\| + \sqrt{\lambda}\|L\|}\right) \log\left(\frac{\kappa}{\delta}\right)(T_A + T_L) + T_b\right)\right) \tag{35}$$

using only $\mathcal{O}(\log \kappa)$ *additional qubits.*

Proof. We invoke Lemma 22, to obtain a unitary U, which is a $(\alpha_A + \sqrt{\lambda}\alpha_L, \max(a_A, a_L) + 2, \varepsilon_A + \sqrt{\lambda}\varepsilon_L)$-block-encoding of the matrix A_L, implemented at a cost of $\mathcal{O}(T_A + T_L)$. Note that in Lemma 22, A and L are considered to be s-qubit operators. For $N \times d$ matrices, such that $N, d \leq 2^s$, we can pad them with zero entries. Padding A and L with zeros may result in the augmented matrix A_L having some zero rows between A and L. However, this is also not an issue as we are only interested in the top left block of A_L^+ which remains unaffected.

Note that U can be reinterpreted as a $\left(\frac{\alpha_A + \sqrt{\lambda}\alpha_L}{\|A_L\|}, \max(a_A, a_L) + 2, \frac{\varepsilon_A + \sqrt{\lambda}\varepsilon_L}{\|A_L\|}\right)$-block-encoding of the normalized matrix $A_L/\|A_L\|$. Furthermore, we can prepare the quantum state $|b\rangle|0\rangle$ in time T_b. Now by using Theorem 28 with U and an appropriately chosen δ specified above, we obtain a quantum state that is δ-close to

$$\frac{(A^T A + \lambda L^T L)^{-1} A^T |b\rangle}{\|(A^T A + \lambda L^T L)^{-1} A^T |b\rangle\|}$$

in the first register. □

In the above complexity, when L is a good regularizer, κ is independent of κ_A. κ can be made arbitrarily smaller than κ_A by an appropriate choice of L. Thus the regularized version has significantly better time complexity than the unregularized case. One such example of a good regularizer is in case of *Quantum Ridge Regression*, where we use the identity matrix to regularize. The corollary below elucidates this.

Corollary 33 (Quantum Ridge Regression)**.** *Let* A *be a matrix of dimension* $N \times d$ *with effective condition number* κ_A *and* $\lambda \in \mathbb{R}^+$ *be the regression parameter. Let* U_A *be a* (α, a, ε)*-block-encoding of* A *implemented in time* T_A*. Let* U_b *be a unitary that prepares* $|b\rangle$ *in time* T_b*. If* $\kappa = 1 + \|A\|/\sqrt{\lambda}$ *then for any* δ *such that*

$$\varepsilon = o\left(\frac{\delta}{\kappa^3 \log^2\left(\frac{\kappa}{\delta}\right)}\right)$$

we can prepare a state δ-close to

$$\frac{\left(A^T A + \lambda I\right)^{-1} A^T |b\rangle}{\left\|\left(A^T A + \lambda I\right)^{-1} A^T |b\rangle\right\|}$$

at a cost of

$$\mathcal{O}\left(\log\kappa\left(\frac{\alpha_A}{\sqrt{\lambda}}\log\left(\frac{\kappa}{\delta}\right)T_A + \kappa T_b\right)\right) \tag{36}$$

with probability $\Theta(1)$ using only $\mathcal{O}(\log\kappa)$ additional qubits.

Proof. The identity matrix I is a trivial $(1,0,0)$-block-encoding of itself, and $\kappa_I = 1$. We invoke Theorem 32 with $L = I$ to obtain the solution. □

Being in the block-encoding framework allows us to express the complexity of our quantum algorithm in specific input models such as the *quantum data structure input model* and the *sparse access model*. We express these complexities via the following corollaries.

Corollary 34 (Quantum Ordinary Least Squares with ℓ_2-Regularization in the Quantum Data Structure Input Model)**.** *Let $A, L \in \mathbb{R}^{N\times d}$ with effective condition numbers κ_A, κ_L respectively. Let $\lambda \in \mathbb{R}^+$ and $b \in \mathbb{R}^N$. Let κ be the effective condition number of the augmented matrix A_L. Suppose that A, L and b are stored in a quantum accessible data structure. Then for any $\delta > 0$ there exists a quantum algorithm to prepare a quantum state δ-close to*

$$\frac{\left(A^T A + \lambda L^T L\right)^{-1} A^T |b\rangle}{\left\|\left(A^T A + \lambda L^T L\right)^{-1} A^T |b\rangle\right\|}$$

with probability $\Theta(1)$, at a cost of

$$\mathcal{O}\left(\kappa\left(\frac{\mu_A + \sqrt{\lambda}\mu_L}{\|A\| + \sqrt{\lambda}\|L\|}\right)\text{polylog}\left(Nd, \kappa, \frac{1}{\delta}, \lambda\right)\right). \tag{37}$$

Proof. Since b is stored in the data structure, for some $\varepsilon_b > 0$, we can prepare the state $|b'\rangle$ that is ε_b-close to $|b\rangle = \sum_i b_i |i\rangle / \|b\|$ using $T_b = \mathcal{O}(\text{polylog}(N/\varepsilon_b))$ queries to the data structure (see Section 2.2.1.) Similarly, for some parameters $\varepsilon_A, \varepsilon_L > 0$, we can construct a $(\mu_A, \lceil\log(d+N)\rceil, \varepsilon_A)$-block-encoding of A using $T_A = \mathcal{O}(\text{polylog}(Nd/\varepsilon_A))$ queries to the data structure and a $(\mu_L, \lceil\log(d+N)\rceil, \varepsilon_B)$-block-encoding of L using $T_L = \mathcal{O}(\text{polylog}(Nd/\varepsilon_B))$ queries.

We invoke Theorem 32 with a precision $\delta/2$ by choosing ε_A and ε_L such that equation Equation 34 is satisfied. This gives us a state that is $\delta/2$-close to

$$\frac{\left(A^T A + \lambda L^T L\right)^{-1} A^T |b'\rangle}{\left\|\left(A^T A + \lambda L^T L\right)^{-1} A^T |b'\rangle\right\|}$$

To compute the final precision as δ, we use Lemma 15 by choosing $\varepsilon_b = \frac{\delta}{2\kappa}$. The complexity can be calculated by plugging in the relevant values in Equation 35 □

In the previous corollary $\mu_A = \|A\|_F$ and $\mu_L = \|L\|_F$ when the matrix A and L are stored in the data structure. Similarly, $\mu_A = \mu_p(A)$ and $\mu_L = \mu_p(L)$ when the matrices $A^{(p)}, A^{(1-p)}$ and $L^{(p)}, L^{(1-p)}$ are stored in the data structure.

Now we discuss the complexity of quantum ordinary least squares with ℓ_2-regularization in the *sparse access input model*. We call a matrix M as (s_r, s_c) row-column sparse if it has a row sparsity s_r and column sparsity s_c.

Corollary 35 (Quantum Ordinary least squares with ℓ_2-regularization in the sparse access model)**.** *Let $A \in \mathbb{R}^{N\times d}$ be (s_r^A, s_c^A) row-column sparse, and similarly, let $L \in \mathbb{R}^{N\times d}$ be (s_r^L, s_c^L) row-column sparse, with effective condition numbers κ_A and κ_L respectively. Let $\lambda \in \mathbb{R}^+$ and $\delta > 0$. Suppose there exists a unitary that prepares $|b\rangle$ at a cost, T_b. Then there is a quantum algorithm to prepare a quantum state that is δ-close to*

$$\frac{(A^TA + \lambda L^TL)^{-1}A^T|b\rangle}{\|(A^TA + \lambda L^TL)^{-1}A^T|b\rangle\|}$$

with probability $\Theta(1)$, at a cost of

$$\mathcal{O}\left(\kappa\left(\frac{\sqrt{s_r^A s_c^A} + \sqrt{\lambda s_r^L s_c^L}}{\|A\| + \sqrt{\lambda}\|L\|}\right)\text{polylog}\left(Nd, \kappa, \frac{1}{\delta}, \lambda\right) + \kappa \log \kappa T_b\right). \tag{38}$$

Proof. The proof is similar to Corollary 34 but with $\alpha_A = \sqrt{s_r^A s_c^A}$ and $\alpha_L = \sqrt{s_r^L s_c^L}$. □

4.2 Quantum Weighted And Generalized Least Squares

This technique of working with a augmented matrix will also hold for the other variants of ordinary least squares. In this section, we begin by briefly describing these variants before moving on to designing quantum algorithms for the corresponding problems.

Weighted Least Squares: For the WLS problem, each observation $\{a_i, b_i\}$ is assigned some weight $w_i \in \mathbb{R}^+$ and the objective function to be minimized is of the form

$$\mathcal{L}_W := \sum_j w_j(x^Ta_j - b_j)^2. \tag{39}$$

If $W \in \mathbb{R}^{N\times N}$ is the diagonal matrix with w_i being the i^{th} diagonal entry, then the optimal x satisfies

$$x = (A^TWA)^{-1}A^TWb. \tag{40}$$

The ℓ_2-regularized version of WLS satisfies

$$x = (A^TWA + \lambda L^TL)^{-1}A^TWb \tag{41}$$

Our quantum algorithm outputs a state that is close to

$$|x\rangle = \frac{(A^TWA + \lambda L^TL)^{-1}A^TW|b\rangle}{\|(A^TWA + \lambda L^TL)^{-1}A^TW|b\rangle\|} \tag{42}$$

given approximate block-encodings of A, W and L. Much like Equation 32, finding the optimal solution reduces to finding the pseudo inverse of an augmented matrix A_L given by

$$A_L := \begin{pmatrix} \sqrt{W}A & 0 \\ \sqrt{\lambda}L & 0 \end{pmatrix}.$$

The top left block of $A_L^+ = (A^TWA + \lambda L^TL)^{-1}A^T\sqrt{W}$, which is the required linear transformation to be applied to the vector $y = \sqrt{W}b$. The ratio between the minimum and maximum singular values of A_L, κ, can be obtained analogously to Lemma 31. For the ℓ_2-regularized WLS problem, *normalized residual sum of squares* is given by

$$\mathcal{S}_W = \frac{\left\|(I - \Pi_{\text{col}(A_L)})|y\rangle|0\rangle\right\|^2}{\||y\rangle\|^2} = 1 - \left\|\Pi_{\text{col}(A_L)}|y\rangle|0\rangle\right\|^2. \tag{43}$$

Subsequently, we assume that $\mathcal{S}_W = 1 - \left\|\Pi_{\mathrm{col}(A_L)} |y\rangle |0\rangle\right\|^2 \leq \gamma < 1/2$. This in turn implies that $\left\|\Pi_{\mathrm{col}(A_L)} |y\rangle |0\rangle\right\|^2 = \Omega(1)$, implying that the data can be reasonably fit by a linear model.

Generalized Least Squares. Similarly, we can extend this to GLS problem, where there the input data may be correlated. These correlations are given by the non-singular covariance matrix $\Omega \in \mathbb{R}^{N\times N}$. The WLS problem is a special case of the GLS problem, corresponding to when Ω is a diagonal matrix. The objective function to be minimized is

$$\mathcal{L}_\Omega := \sum_{i,j} (\Omega^{-1})_{ij} (x^T a_i - b_i)(x^T a_j - b_j). \tag{44}$$

The optimal $x \in \mathbb{R}^d$ satisfies

$$x = (A^T \Omega^{-1} A)^{-1} A^T \Omega^{-1} b \tag{45}$$

Similarly, the ℓ_2-regularized GLS solver outputs x such that

$$x = (A^T \Omega^{-1} A + \lambda L^T L)^{-1} A^T \Omega^{-1} b. \tag{46}$$

So, given approximate block-encodings of A, Ω and L a quantum GLS solver outputs a quantum state close to

$$|x\rangle = \frac{(A^T \Omega^{-1} A + \lambda L^T L)^{-1} A^T \Omega^{-1} |b\rangle}{\|(A^T \Omega^{-1} A + \lambda L^T L)^{-1} A^T \Omega^{-1} |b\rangle\|} \tag{47}$$

The augmented matrix A_L is defined as

$$A_L := \begin{pmatrix} \Omega^{-1/2} A & 0 \\ \sqrt{\lambda} L & 0 \end{pmatrix}.$$

Then top left block of A_L^+ to the vector $y = \Omega^{-1/2} b$ yields the optimal x. Thus the quantum GLS problem with ℓ_2-regularization first prepares $\Omega^{-1/2} |b\rangle |0\rangle$ and then uses the matrix inversion algorithm by QSVT to implement $A_L^+ \Omega^{-1/2} |b\rangle |0\rangle$. Analogous to OLS and WLS, we assume that the normalized residual sum of squares $\mathcal{S}_\Omega \leq \gamma < 1/2$.

4.2.1 Quantum Weighted Least Squares

In this section, we derive the complexity of the ℓ_2-regularized WLS problem. We assume that we have a diagonal weight matrix $W \in \mathbb{R}^{N\times N}$ such that its smallest and largest diagonal entries are $w_{\min}$ and $w_{\max}$, respectively. This implies that $\|W\| = w_{\max}$ and $\kappa_W = w_{\max}/w_{\min}$. We take advantage of the fact that the matrix W is diagonal and then apply controlled rotations to directly implement a block encoding of $\sqrt{W}A$. Additionally, given a state preparation procedure for $|b\rangle$, we can easily prepare a state proportional to $\sqrt{W} |b\rangle$. We then use Theorem 32 to solve QWLS.

We first formalize this idea in Theorem 36, assuming direct access to (i) a block encoding of $B = \sqrt{W}A$, and (ii) a procedure for preparing the state $|b_w\rangle = \frac{\sqrt{W}|b\rangle}{\|\sqrt{W}|b\rangle\|}$. Subsequently, for the specific input models, we show that we can indeed efficiently obtain a block-encoding of B and prepare the state $|b_w\rangle$.

Theorem 36 (Quantum Weighted Least Squares with General ℓ_2-Regularization)**.** *Let $A, L \in \mathbb{R}^{N\times d}$, be the data and penalty matrix, with effective condition numbers κ_A and κ_L, respectively. Let $\lambda \in \mathbb{R}^+$ be the regularizing parameter. Let $W \in \mathbb{R}^{N\times N}$ be a diagonal weight matrix with the largest and smallest diagonal entries being w_{max}, w_{min}, respectively. Let U_B be a $(\alpha_B, a_B, \varepsilon_B)$ block encoding of $B := \sqrt{W}A$ implemented in time T_B and let U_L be a $(\alpha_L, a_L, \varepsilon_L)$ block encoding of L implemented in time T_L, such that $\varepsilon_B = o\left(\frac{\delta}{\kappa^3 \log^2\left(\frac{\kappa}{\delta}\right)}\right)$ and $\varepsilon_L = o\left(\frac{\delta}{\sqrt{\lambda}\kappa^3 \log^2\left(\frac{\kappa}{\delta}\right)}\right)$. Let U_{b_w} be a unitary that prepares $\frac{\sqrt{W}|b\rangle}{\|\sqrt{W}|b\rangle\|}$ in time T_{b_w}. Define*

$$\kappa := \kappa_L \left(1 + \frac{\sqrt{w_{max}}\|A\|}{\sqrt{\lambda}\|L\|}\right)$$

Then for any $\delta > 0$ we can prepare a quantum state that is δ-close to

$$\frac{(A^T W A + \lambda L^T L)^{-1} A^T W |b\rangle}{\|(A^T W A + \lambda L^T L)^{-1} A^T W |b\rangle\|}$$

with probability $\Theta(1)$, at a cost of

$$\mathcal{O}\left(\kappa \log \kappa \left(\frac{\alpha_B + \sqrt{\lambda}\alpha_L}{\sqrt{w_{max}}\|A\| + \sqrt{\lambda}\|L\|} \log\left(\frac{\kappa}{\delta}\right)(T_B + T_L) + T_{b_w}\right)\right), \tag{48}$$

using only $\mathcal{O}(\log \kappa)$ additional qubits.

Proof. We then invoke Theorem 32 with B and L as the data and regularization matrices, respectively. This requires that $\varepsilon_B, \varepsilon_L$ such that

$$\varepsilon_B + \sqrt{\lambda}\varepsilon_L = o\left(\frac{\delta}{\kappa^3 \log^2\left(\frac{\kappa}{\delta}\right)}\right).$$

Thus, we get the upper bounds on the precision ε_B, ε_L required. This gives us a quantum state δ-close to

$$\frac{(A^T W A + \lambda L^T L)^{-1} A^T W |b\rangle}{\|(A^T W A + \lambda L^T L)^{-1} A^T W |b\rangle\|}.$$

□

Next, we construct the block encodings for $\sqrt{W}A$ and the state $\frac{\sqrt{W}|b\rangle}{\|\sqrt{W}|b\rangle\|}$ efficiently in the quantum data structure input model. This construction would also apply to the sparse access input model with slight modifications.

Lemma 37 (Efficiently preparing $\sqrt{W}A$ in the Quantum Data Structure Model)**.** *Let $W \in \mathbb{R}^{N\times N}$ such that $W = \mathrm{diag}(w_1, w_2 \ldots w_N)$ and $w_{\max} := \max_i w_i$, and $A \in \mathbb{R}^{N\times d}$ be stored in a quantum-accessible data structure. Then for any $\delta > 0$ there exists a*

$$(\sqrt{w_{\max}}\|A\|_F, \lceil \log(N + d) \rceil, \delta)$$

block-encoding of $\sqrt{W}A$ that can be implemented at the cost $\mathcal{O}(\mathrm{polylog}(Nd/\delta))$.

Proof. $\forall j \in [N]$, define

$$|\psi_j\rangle := \sqrt{\frac{w_j}{w_{\max}}} |j\rangle \frac{1}{\|A_{j,\cdot}\|} \sum_{k\in[d]} A_{j,k} |k\rangle.$$

Similarly, $\forall k \in [d]$, define

$$|\phi_k\rangle := \frac{1}{\|A\|_F} \left(\sum_{j \in [N]} \|A_{j,\cdot}\| \, |j\rangle \right) |k\rangle \, .$$

Observe that $\forall j \in [N], k \in [d]$,

$$\langle \psi_j | \phi_k \rangle = \sqrt{\frac{w_j}{w_{\max}}} \frac{A_{j,k}}{\|A\|_F} = \frac{\langle j | \sqrt{W} A \, |k\rangle}{\sqrt{w_{\max}} \|A\|_F}.$$

Given quantum data structure accesses to W and A, one can construct quantum circuits W_R and W_L similar to U_L and U_R from Lemma 3 that prepare $|\phi_k\rangle$ and $|\psi_j\rangle$ above. $|\phi_k\rangle$ can be prepared just as in Lemma 3, while $|\psi_j\rangle$ can be prepared using controlled rotations on the state $|\frac{w_j}{w_{\max}}\rangle$ (which can be constructed from the QRAM access to W) after adding an ancilla qubit and the QRAM access to A. Thus, $W_R^\dagger W_L$ is the required block encoding, which according to Theorem 2 can be implemented using $\text{polylog}(Nd/\delta)$ queries. □

Lemma 38 (Efficiently preparing $\sqrt{W}\,|b\rangle$ in the Quantum Data Structure Model)**.** *Let $b \in \mathbb{R}^N$ and $W \in \mathbb{R}^{N \times N}$. Suppose that b and W are stored in a quantum-accessible data structure such that we have a state preparation procedure that acts as*

$$U_W : |j\rangle |0\rangle \mapsto |j\rangle |w_j\rangle \, ,$$
$$U_b : |0\rangle \mapsto \sum_j \frac{b_j}{\|b\|} |j\rangle \, .$$

Then for any $\delta > 0$ we can prepare the quantum state that is δ-close to $\frac{\sqrt{W}|b\rangle}{\|\sqrt{W}|b\rangle\|}$ with constant success probability and at a cost of $\mathcal{O}\left(\sqrt{\frac{w_{\max}}{w_{\min}}} \text{polylog}\left(\frac{N}{\delta}\right)\right)$.

Proof. Use U_b to prepare the state

$$|b\rangle = \frac{1}{\|b\|} \sum_j b_j |j\rangle$$

in time $\text{polylog}(N)$. Then, apply the following transformation

$$\begin{aligned} |j\rangle |0\rangle |0\rangle &\mapsto |j\rangle |w_j\rangle |0\rangle \\ &\mapsto |j\rangle |w_j\rangle \left(\sqrt{\frac{w_j}{w_{\max}}} |0\rangle + \sqrt{1 - \frac{w_j}{w_{\max}}} |1\rangle \right) \\ &\mapsto |j\rangle |0\rangle \left(\sqrt{\frac{w_j}{w_{\max}}} |0\rangle + \sqrt{1 - \frac{w_j}{w_{\max}}} |1\rangle \right) \end{aligned}$$

which can again be applied using some controlled rotations, a square root circuit and U_W. This gives us the state (ignoring some blank registers)

$$\sum_j \left(\sqrt{\frac{w_j}{w_{\max}}} |0\rangle + \sqrt{1 - \frac{w_j}{w_{\max}}} |1\rangle \right) \frac{b_j}{\|b\|} |j\rangle \, . \tag{49}$$

The probability for the ancilla to be in $|0\rangle$ state is

$$\Omega \left(\frac{w_{\min}}{w_{\max}} \right).$$

Thus performing $\mathcal{O}\left(\sqrt{\frac{w_{\max}}{w_{\min}}}\right)$ rounds of amplitude amplification on $|0\rangle$ gives us a constant probability of observing $|0\rangle$, and therefore obtaining the desired state $\frac{\sqrt{W}|b\rangle}{\|\sqrt{W}|b\rangle\|}$. □

Using the above two theorems, and the quantum OLS solver (Theorem 32), we can construct an algorithm for regularized quantum WLS.

Theorem 39 (Quantum Weighted Least Squares with General ℓ_2-Regularization in the Quantum Data Structure Model)**.** *Let $A, L \in \mathbb{R}^{N\times d}$ with effective condition numbers κ_A, κ_L respectively be stored in an efficient quantum accessible data structure. Let $W \in \mathbb{R}^{N\times N}$ be a diagonal matrix with largest and smallest singular values w_{max}, w_{min} respectively, which is also stored in an efficient quantum accessible data structure. Furthermore, suppose the entries of the vector $b \in \mathbb{R}^N$ are also stored in a quantum-accessible data structure and define,*

$$\kappa := \kappa_L \left(1 + \frac{\sqrt{w_{max}}\|A\|}{\sqrt{\lambda}\|L\|}\right)$$

Then for any $\delta > 0$ we can prepare a quantum state that is δ-close to

$$\frac{(A^TWA + \lambda L^TL)^{-1}A^TW\,|b\rangle}{\|(A^TWA + \lambda L^TL)^{-1}A^TW\,|b\rangle\|}$$

with probability $\Theta(1)$, at a cost of

$$\mathcal{O}\left(\kappa\left(\frac{\sqrt{w_{max}}\|A\|_F + \sqrt{\lambda}\|L\|_F}{\sqrt{w_{max}}\|A\| + \sqrt{\lambda}\|L\|} + \sqrt{\frac{w_{max}}{w_{min}}}\right)\text{polylog}\left(Nd, \kappa, \frac{1}{\delta}\right)\right) \tag{50}$$

Proof. Choose some precision parameter $\varepsilon > 0$ for accessing the data structure. Given access to W and A, we can use Lemma 37 to prepare a $(\sqrt{w_{\max}}\|A\|_F, \lceil\log(N+d)\rceil, \varepsilon)$-block-encoding of $\sqrt{W}A$, using $T_A := \mathcal{O}(\text{polylog}(Nd/\varepsilon))$ queries to the data structure. Similarly, Lemma 3 allows us to build a $(\|L\|_F, \lceil\log(N+d)\rceil, \varepsilon)$-block-encoding of L using $T_L := \mathcal{O}(\text{polylog}(Nd/\varepsilon))$ queries to the data structure.

Next, using Lemma 38, for any $\varepsilon_b > 0$, we can prepare a state ε_b-close to $|b'\rangle := \frac{\sqrt{W}|b\rangle}{\|\sqrt{W}|b\rangle\|}$. This procedure requires $T_b := \mathcal{O}\left(\sqrt{\frac{w_{\max}}{w_{\min}}}\text{polylog}\left(N/\varepsilon_b\right)\right)$ queries to the data structure. Now we can invoke the OLS solver in Theorem 32 with a precision of δ_b, by considering $\sqrt{W}A$ as the data matrix and $\frac{\sqrt{W}|b\rangle}{\|\sqrt{W}|b\rangle\|}$ as the input state. In order for the input block-encoding precision to satisfy the bound in Equation 34, we choose ε such that

$$\varepsilon = o\left(\frac{\delta_b}{\kappa^3\log^2\left(\frac{\kappa}{\delta_b}\right)}\right).$$

Finally, for the output state to be δ-close to the required state, we choose $\delta_b = \delta/2$ and $\varepsilon_b = \delta/2\kappa$ to use the robustness result from Lemma 15. This gives us

$$\begin{aligned}\log\left(\frac{1}{\varepsilon}\right) &= \mathcal{O}\left(\log\left(\frac{\kappa^3\log^2\left(\frac{\kappa}{\delta_b}\right)}{\delta_b}\right)\right)\\ &= \mathcal{O}\left(\log\left(\frac{\kappa}{\delta}\right)\right)\end{aligned}$$

Now we can substitute the cost of the individual components in Equation 35 to obtain the final cost as

$$\mathcal{O}\left(\kappa\log\kappa\left(\frac{\sqrt{w_{\max}}\|A\|_F + \sqrt{\lambda}\|L\|_F}{\sqrt{w_{\max}}\|A\| + \sqrt{\lambda}\|L\|}\log\left(\frac{\kappa}{\delta}\right)\text{polylog}\left(\frac{Nd}{\varepsilon}\right) + \sqrt{\frac{w_{\max}}{w_{\min}}}\text{polylog}\left(\frac{N\kappa}{\delta}\right)\right)\right)$$

$$= \mathcal{O}\left(\kappa\left(\frac{\sqrt{w_{\max}}\|A\|_F + \sqrt{\lambda}\|L\|_F}{\sqrt{w_{\max}}\|A\| + \sqrt{\lambda}\|L\|} + \sqrt{\frac{w_{\max}}{w_{\min}}}\right)\operatorname{polylog}\left(\frac{Nd\kappa}{\delta}\right)\right)$$

□

Now, for the sparse access model, we can obtain a block encoding similar to Lemma 37 and a quantum state similar to Lemma 38, with the same query complexities. Thus we have an algorithm similar to Theorem 39 in the sparse access model as well. We directly state the complexity of this algorithm.

Theorem 40 (Quantum Weighted Least Squares with General ℓ_2-Regularization in the Sparse Access Model)**.** *Let $A \in \mathbb{R}^{N\times d}$ be (s_r^A, s_c^A) row-column sparse, and similarly, let $L \in \mathbb{R}^{N\times d}$ be (s_r^L, s_c^L) row-column sparse, with effective condition numbers κ_A and κ_L respectively. Let $\lambda \in \mathbb{R}^+$. Let $W \in \mathbb{R}^{N\times N}$ be a diagonal matrix with the largest and the smallest diagonal entries being w_{max}, w_{min}, respectively. Suppose that the diagonal entries of W are stored in a QROM such that, for any $\delta > 0$, we can compute $|j\rangle\, 0 \mapsto |j\rangle\, |w_j\rangle$ in cost $O\left(\operatorname{polylog}\left(Nd/\delta\right)\right)$ as well as w_{max}. Furthermore, suppose there exists a unitary that prepares $|b\rangle$ at a cost T_b and define,*

$$\kappa := \kappa_L\left(1 + \frac{\sqrt{w_{max}}\|A\|}{\sqrt{\lambda}\|L\|}\right)$$

Then for any $\delta > 0$ we can prepare a quantum state that is δ-close to

$$\frac{(A^TWA + \lambda L^TL)^{-1}A^TW\,|b\rangle}{\|(A^TWA + \lambda L^TL)^{-1}A^TW\,|b\rangle\|}$$

with probability $\Theta(1)$, at a cost of

$$\mathcal{O}\left(\kappa\left(\frac{\sqrt{w_{max}}\sqrt{s_r^A s_c^A} + \sqrt{\lambda}\sqrt{s_r^L s_c^L}}{\sqrt{w_{max}}\|A\| + \sqrt{\lambda}\|L\|} + \sqrt{\frac{w_{max}}{w_{min}}}\,T_b\right)\operatorname{polylog}\left(Nd, \kappa, \frac{1}{\delta}\right)\right) \tag{51}$$

4.2.2 Quantum Generalized Least Squares

In this section, we assume that we have block-encoded access to the correlation matrix $\Omega \in \mathbb{R}^{N\times N}$, with condition number κ_Ω. We begin by preparing a block encoding of $\Omega^{-1/2}$, given an approximate block-encoding of Ω.

Lemma 41 (Preparing $\Omega^{-1/2}$)**.** *Let $\Omega \in \mathbb{R}^{N\times N}$ be a matrix with condition number κ_Ω Let U_Ω be an $(\alpha_\Omega, a_\Omega, \varepsilon_\Omega)$-block-encoding of Ω, implemented in time T_Ω. For any δ such that*

$$\varepsilon_\Omega = o\left(\frac{\sqrt{\|\Omega\|}\delta}{\kappa^{1.5}\log\left(\frac{\kappa}{\sqrt{\|\Omega\|}\delta}\right)}\right),$$

we can prepare a $(2\sqrt{\kappa_\Omega/\|\Omega\|}, a_\Omega + 1, \delta)$-block-encoding of $\Omega^{-1/2}$ at a cost of

$$\mathcal{O}\left(\frac{\alpha_\Omega\kappa_\Omega}{\|\Omega\|}\log\left(\frac{\kappa_\Omega}{\delta\sqrt{\|\Omega\|}}\right)T_\Omega\right)$$

Moreover, the condition number of $\Omega^{-1/2}$ is bounded by $\sqrt{\kappa_\Omega}$.

Proof. U_Ω can be re-interpreted as a $(\frac{\alpha}{\|\Omega\|},\ a,\ \frac{\varepsilon}{\|\Omega\|})$-block-encoding of $\frac{\Omega}{\|\Omega\|}$. We can then prepare the required unitary by invoking Theorem 30 on U_Ω with $c = 1/2$ and some γ such that we get a $(2\sqrt{\kappa_\Omega},\ a+1,\ \gamma)$ block encoding of $\sqrt{\|\Omega\|}\Omega^{-1/2}$, which is a $(2\sqrt{\frac{\kappa}{\|\Omega\|}},\ a+1,\ \frac{\gamma}{\sqrt{\|\Omega\|}})$ block encoding of $\Omega^{-1/2}$. Fixing $\gamma = \sqrt{\|\Omega\|}\delta$ gives us the required result. □

We will now use this lemma in conjunction with Theorem 32 to develop quantum algorithms for GLS with general ℓ_2-regularization.

Theorem 42 (Quantum Generalized Least Squares with General ℓ_2-regularization)**.** *Let $A, L \in \mathbb{R}^{N\times d}$ be the data and penalty matrices with effective condition numbers κ_A, κ_L respectively. Let $\Omega \in \mathbb{R}^{N\times N}$ be the covariance matrix with condition number κ_Ω. Let $\delta > 0$ be the precision parameter. Define κ as*

$$\kappa := \kappa_L\left(1 + \frac{\sqrt{\kappa_\Omega}\|A\|}{\sqrt{\lambda\|\Omega\|}\|L\|}\right).$$

For some ε_A such that

$$\varepsilon_A = o\left(\frac{\delta\sqrt{\|\Omega\|}}{\kappa^3\sqrt{\kappa_\Omega}\log^2\frac{\kappa}{\delta}}\right)$$

we have access to U_A, an $(\alpha_A, a_A, \varepsilon_A)$-block-encoding of A implemented in time T_A. For some ε_L such that

$$\varepsilon_L = o\left(\frac{\delta}{\sqrt{\lambda}\kappa^3\log^2\frac{\kappa}{\delta}}\right)$$

we have access to U_L, an $(\alpha_L, a_L, \varepsilon_L)$-block-encoding of L implemented in time T_L. For some ε_Ω such that

$$\varepsilon_\Omega = o\left(\frac{\delta}{\|A\|\kappa^3{\kappa_\Omega}^{1.5}\log^3\frac{\kappa}{\delta}\log\left(\frac{\kappa_\Omega}{\|A\|\|\Omega\|}\right)}\right)$$

we have access to U_Ω, an $(\alpha_\Omega, a_\Omega, \varepsilon_\Omega)$-block-encoding of Ω implemented in time T_Ω. Let U_b be a unitary that prepares the state $|b\rangle$ in time T_b.

Then we can prepare the quantum state that is δ-close to

$$\frac{\left(A^T\Omega^{-1}A + \lambda L^TL\right)^{-1}A^T\Omega^{-1}|b\rangle}{\left\|\left(A^T\Omega^{-1}A + \lambda L^TL\right)^{-1}A^T\Omega^{-1}|b\rangle\right\|}$$

with probability $\Theta(1)$, at a cost of

$$\mathcal{O}\left(\kappa\sqrt{\kappa_\Omega}\log\kappa\left(\left(\frac{\alpha_A}{\|A\|}T_A + \frac{\alpha_L}{\|L\|}T_L + \frac{\alpha_\Omega\kappa_\Omega}{\|\Omega\|}T_\Omega\right)\log^3\left(\frac{\kappa\kappa_\Omega\|A\|\|L\|}{\delta\|\Omega\|}\right) + T_b\right)\right) \tag{52}$$

using only $\mathcal{O}(\log\kappa)$ additional qubits.

Proof. Observe that by choosing $A' := \Omega^{-1/2}A, L' := L, |b'\rangle := \Omega^{-1/2}|b\rangle$ (upto normalization) in the quantum ordinary least squares, we get a state proportional to $(A'^TA' + \lambda L'^TL')^{-1}A'^T|b'\rangle = (A^T\Omega^{-1}A + \lambda L^TL)A^T\Omega^{-1}|b\rangle$, which is the desired state.

For convenience, let us define the matrix $B := \Omega^{-1/2}$ (and therefore $\kappa_B = \sqrt{\kappa_\Omega}$ and $\|B\| = \sqrt{\kappa_\Omega/\|\Omega\|}$). We now need to prepare a block-encoding of BA and the quantum state $\frac{B|b\rangle}{\|B|b\rangle\|}$, which we then use to invoke Theorem 32.

We begin by using Lemma 41 with some precision ε_B to construct a $(\alpha_B, a_B, \varepsilon_B)$-block-encoding of $B = \Omega^{-1/2}$, where $\alpha_B = 2\sqrt{\frac{\kappa_\Omega}{\|\Omega\|}} = 2\|B\|$, and $a_B = a_\Omega + 1$. This bounds ε_Ω as

$$\varepsilon_\Omega = o\left(\frac{\sqrt{\|\Omega\|}\varepsilon_B}{{\kappa_\Omega}^{1.5}\log\left(\frac{\kappa_\Omega}{\sqrt{\|\Omega\|\varepsilon_B}}\right)}\right),$$

and has a cost of

$$T_B := \mathcal{O}\left(\frac{\alpha_\Omega\kappa_\Omega}{\|\Omega\|}\log\left(\frac{\kappa_\Omega}{\varepsilon_B\sqrt{\|\Omega\|}}\right)T_\Omega\right)$$

Then using Lemma 20 with precision γ satisfying $\gamma \geq 4\sqrt{2}\max\left(\|B\|\varepsilon_A, \|A\|\varepsilon_B\right)$, we get a
$(2\|A\|\|B\|, a_A + a_B + 3, \gamma)$-block-encoding of $A' := BA = \Omega^{-1/2}A$ at a cost

$$T_{A'} := \mathcal{O}\left(\left(\frac{\alpha_A}{\|A\|}T_A + \frac{\alpha_B}{\|B\|}T_B\right)\log\left(\frac{\|A\|\|B\|}{\gamma}\right)\right).$$

To prepare $\frac{B|b\rangle}{\|B|b\rangle\|}$, we use Lemma 13 with precision $\varepsilon_b \geq 2\varepsilon_B\kappa_B/\|B\|$. This prepares a state that is ε_b-close to $|b'\rangle := \frac{B|b\rangle}{\|B|b\rangle\|}$ with constant success probability at a cost of

$$T_{b'} := \mathcal{O}\left(\frac{\alpha_B\kappa_B}{\|B\|}(T_B + T_b)\right) = \mathcal{O}(\kappa_B(T_B + T_b))$$

We could invoke OLS directly using the above two, but that ends up with a product of sub-normalization factors (α terms) in the complexity. We want to avoid this, because in most common cases α-s for block-encodings are quite large. So we also pre-amplify U_L using Corollary 12: for any $\delta_L \geq 2\varepsilon_L$ we get a $(\sqrt{2}\|L\|, a_L + 1, \delta_L)$-encoding of L at a cost of

$$T_{L'} := \mathcal{O}\left(\frac{\alpha_L}{\|L\|}T_L\log\left(\frac{\|L\|}{\delta_L}\right)\right).$$

Now that we have these, we can use Theorem 32 to get a quantum state δ'-close to $|\psi\rangle := \frac{A_L^+|b'\rangle}{\|A_L^+|b'\rangle\|}$, where $A_L^+ = (A^T\Omega^{-1}A + \lambda L^T L)^{-1}A^T\Omega^{-1/2}$. This would require that $\gamma, \sqrt{\lambda}\delta_L \in o\left(\frac{\delta'}{\kappa^3\log^2\left(\frac{\kappa}{\delta'}\right)}\right)$ and would cost

$$\mathcal{O}\left(\kappa\log\kappa\left(\left(\frac{2\|A\|\|B\| + \sqrt{2\lambda}\|L\|}{\|BA\| + \sqrt{\lambda}\|L\|}\right)\log\left(\frac{\kappa}{\delta'}\right)(T_{A'} + T_{L'}) + T_{b'}\right)\right).$$

To simplify the ratio of norms term, we can first lower-bound $\|BA\| \geq \|A\|/\|B^{-1}\| = \|A\|/\sqrt{\|\Omega\|}$. And as $\|B\| = \sqrt{\kappa_\Omega/\|\Omega\|}$, the whole term can be simplified to $\mathcal{O}(\sqrt{\kappa_\Omega})$. This simplifies the cost expression to $\mathcal{O}(\kappa\log\kappa\left(\sqrt{\kappa_\Omega}\log\left(\kappa/\delta'\right)(T_{A'} + T_{L'}) + T_{b'}\right))$.

We can compute the error between $|\psi\rangle$ and the expected state by using Lemma 15. For the final error to be δ, we have to choose $\varepsilon_b = \delta/2\kappa$ and $\delta' = \delta/2$. Therefore

$$\varepsilon_B \leq \frac{\varepsilon_b\|B\|}{4\kappa_B} = \Theta\left(\frac{\delta}{\kappa\sqrt{\|\Omega\|}}\right)$$

$$\gamma, \sqrt{\lambda}\delta_L \in o\left(\frac{\delta}{\kappa^3\log^2(\kappa/\delta)}\right)$$

$$\implies \log\left(\frac{1}{\gamma}\right) = o\left(\log\left(\frac{\kappa}{\delta}\right)\right), \;\; \log\left(\frac{1}{\delta_L}\right) = o\left(\log\left(\frac{\sqrt{\lambda}\kappa}{\delta}\right)\right)$$

$$\varepsilon_A = o\left(\frac{\gamma}{\|B\|}\right), \;\; \varepsilon_B = o\left(\frac{\gamma}{\|A\|}\right)$$

Combining both bounds of ε_B by using sums or products, we can effectively bound

$$\varepsilon_\Omega = o\left(\frac{\delta}{\|A\|\kappa^3{\kappa_\Omega}^{1.5}\log^3\frac{\kappa}{\delta}\log\left(\frac{\kappa_\Omega}{\|A\|\|\Omega\|}\right)}\right)$$

Finally for the final costs, we calculate the respective coefficients of terms T_A, T_Ω, T_L and T_b, (excluding the common factor of $\kappa\sqrt{\kappa_\Omega}\log\kappa$ for brevity). Let us label these "coefficient extraction" functions as $\mathcal{C}$ with matching subscripts, and the total cost as T.

$$\begin{aligned}
\mathcal{C}_A(T) &= \mathcal{O}\left(\log\left(\frac{\kappa}{\delta}\right)\mathcal{C}_A(T_{A'})\right) \\
&= \mathcal{O}\left(\log\left(\frac{\kappa}{\delta}\right)\frac{\alpha_A}{\|A\|}\log\left(\frac{\|A\|\|B\|}{\gamma}\right)\right) \\
&= \mathcal{O}\left(\frac{\alpha_A}{\|A\|}\log^2\left(\frac{\kappa\kappa_\Omega\|A\|}{\delta\|\Omega\|}\right)\right) \\
\mathcal{C}_L(T) &= \mathcal{O}\left(\log\left(\frac{\kappa}{\delta}\right)\mathcal{C}_L(T_{L'})\right) \\
&= \mathcal{O}\left(\log\left(\frac{\kappa}{\delta}\right)\frac{\alpha_L}{\|L\|}\log\left(\frac{\|L\|}{\delta_L}\right)\right) \\
&= \mathcal{O}\left(\frac{\alpha_L}{\|L\|}\log^2\left(\frac{\kappa\|L\|}{\delta}\right)\right) \\
\mathcal{C}_\Omega(T) &= \mathcal{O}\left(\log\left(\frac{\kappa}{\delta}\right)\mathcal{C}_\Omega(T_{A'}) + \frac{\mathcal{C}_\Omega(T_{b'})}{\sqrt{\kappa_\Omega}}\right) \\
&= \mathcal{O}\left(\left(\log\left(\frac{\kappa}{\delta}\right)\log\left(\frac{\|A\|\|B\|}{\gamma}\right)+1\right)\mathcal{C}_\Omega(T_B)\right) \\
&= \mathcal{O}\left(\log^2\left(\frac{\kappa\kappa_\Omega\|A\|}{\delta\|\Omega\|}\right)\frac{\alpha_\Omega\kappa_\Omega}{\|\Omega\|}\log\left(\frac{\kappa_\Omega}{\varepsilon_B\sqrt{\|\Omega\|}}\right)\right) \\
&= \mathcal{O}\left(\frac{\alpha_\Omega\kappa_\Omega}{\|\Omega\|}\log^3\left(\frac{\kappa\kappa_\Omega\|A\|}{\delta\|\Omega\|}\right)\right) \\
\mathcal{C}_b(T) &= \mathcal{O}\left(\frac{\mathcal{C}_\Omega(T_{b'})}{\sqrt{\kappa_\Omega}}\right) = \mathcal{O}(1)
\end{aligned}$$

And hence the final complexity is given by the expression

$$\begin{aligned}
T &= \mathcal{O}(\kappa\sqrt{\kappa_\Omega}\log\kappa\left(\mathcal{C}_A(T)\cdot T_A + \mathcal{C}_L(T)\cdot T_L + \mathcal{C}_\Omega(T)\cdot T_\Omega + \mathcal{C}_b(T)\cdot T_b\right)) \\
&= \mathcal{O}\left(\kappa\sqrt{\kappa_\Omega}\log\kappa\left(\frac{\alpha_A}{\|A\|}\log^2\left(\frac{\kappa\kappa_\Omega\|A\|}{\delta\|\Omega\|}\right)T_A + \frac{\alpha_L}{\|L\|}\log^2\left(\frac{\kappa\|L\|}{\delta}\right)T_L + \frac{\alpha_\Omega\kappa_\Omega}{\|\Omega\|}\log^3\left(\frac{\kappa\kappa_\Omega\|A\|}{\delta\|\Omega\|}\right)T_\Omega + T_b\right)\right) \\
&= \mathcal{O}\left(\kappa\sqrt{\kappa_\Omega}\log\kappa\left(\left(\frac{\alpha_A}{\|A\|}T_A + \frac{\alpha_L}{\|L\|}T_L + \frac{\alpha_\Omega\kappa_\Omega}{\|\Omega\|}T_\Omega\right)\log^3\left(\frac{\kappa\kappa_\Omega\|A\|\|L\|}{\delta\|\Omega\|}\right) + T_b\right)\right)
\end{aligned}$$

□

One immediate observation is that for the special case of the (unregularized) quantum GLS problem (when $L = 0$ and $\lambda = 0$), our algorithm has a slightly better complexity than [CGJ19] and requires fewer additional qubits. Now, we will state the complexities of this algorithm in specific input models, namely the quantum data structure model and the sparse-access input model.

Corollary 43 (Quantum Generalized Least Squares with General ℓ_2-Regularization in the Quantum Data Structure Model)**.** *Let $A, L \in \mathbb{R}^{N\times d}$ be the data and penalty matrices with effective condition numbers κ_A, κ_L respectively. and $\Omega \in \mathbb{R}^{N\times N}$ be the covariance matrix with condition number κ_Ω. Let the matrices A, L, Ω and the vector b be stored in a quantum-accessible data structure. Define κ as*

$$\kappa := \kappa_L \left(1 + \frac{\sqrt{\kappa_\Omega}\|A\|}{\sqrt{\lambda\|\Omega\|}\|L\|}\right)$$

Then for any $\delta > 0$, we can prepare the quantum state that is δ-close to

$$\frac{\left(A^T\Omega^{-1}A + \lambda L^T L\right)^{-1} A^T\Omega^{-1}\,|b\rangle}{\left\|\left(A^T\Omega^{-1}A + \lambda L^T L\right)^{-1} A^T\Omega^{-1}\,|b\rangle\right\|}$$

with probability $\Theta(1)$, at a cost of

$$\mathcal{O}\left(\kappa\sqrt{\kappa_\Omega}\left(\frac{\mu_A}{\|A\|} + \frac{\mu_L}{\|L\|} + \frac{\kappa_\Omega\mu_\Omega}{\|\Omega\|}\right)\text{polylog}\left(Nd, \kappa, \frac{1}{\delta}, \frac{\kappa_\Omega}{\|\Omega\|}, \|A\|, \|L\|, \lambda\right)\right) \tag{53}$$

Proof. The proof is very similar to Corollary 34 with the extra input of Ω. We can use the data structure to prepare the block-encodings for A, L, Ω and the state $|b\rangle$, with precisions $\varepsilon_A, \varepsilon_L, \varepsilon_\Omega, \varepsilon_b$ respectively. We invoke Theorem 42 with a precision of δ_b, and choose the above ε terms to be equal to their corresponding upper-bounds. And finally we use Lemma 15 with $\varepsilon_b = \delta/2\kappa$ and $\delta_b = \delta/2$ to get the final error as δ. □

Now, $\mu_A = \|A\|_F$ (similarly for μ_L and μ_Ω). As $\|A\|_F \leq \sqrt{r(A)}\|A\|$, where $r(A)$ is the rank of A, we have that the complexity of Corollary 43 can be re-expressed as

$$\mathcal{O}\left(\kappa\sqrt{\kappa_\Omega}\left(\sqrt{r(A)} + \sqrt{r(L)} + \sqrt{r(\Omega)}\kappa_\Omega\right)\text{polylog}\left(\frac{Nd\kappa}{\delta}\right)\right). \tag{54}$$

Corollary 44 (Quantum Generalized Least Squares with General ℓ_2-Regularization in the Sparse Access Model)**.** *Let $A \in \mathbb{R}^{N\times d}$ be a (s_r^A, s_c^A) row-column sparse data matrix. Let $L \in \mathbb{R}^{N\times d}$ be a (s_r^L, s_c^L) row-column sparse penaly matrix. Let $\Omega \in \mathbb{R}^{N\times N}$ be a (s_r^Ω, s_c^Ω) row-column sparse covariance matrix. Suppose we have a procedure to prepare $|b\rangle$ in cost T_b. Define κ as*

$$\kappa := \kappa_L \left(1 + \frac{\sqrt{\kappa_\Omega}\|A\|}{\sqrt{\lambda\|\Omega\|}\|L\|}\right)$$

Then for any $\delta > 0$, we can prepare the quantum state that is δ-close to

$$\frac{\left(A^T\Omega^{-1}A + \lambda L^T L\right)^{-1} A^T\Omega^{-1}\,|b\rangle}{\left\|\left(A^T\Omega^{-1}A + \lambda L^T L\right)^{-1} A^T\Omega^{-1}\,|b\rangle\right\|}$$

with probability $\Theta(1)$, at a cost of

$$\mathcal{O}\left(\kappa\sqrt{\kappa_\Omega}\left(\frac{\sqrt{s_r^A s_c^A}}{\|A\|} + \frac{\sqrt{s_r^L s_c^L}}{\|L\|} + \frac{\kappa_\Omega\sqrt{s_r^\Omega s_c^\Omega}}{\|\Omega\|} + T_b\right)\text{polylog}\left(Nd, \kappa, \frac{1}{\delta}, \frac{\kappa_\Omega}{\|\Omega\|}, \|A\|, \|L\|, \lambda\right)\right) \tag{55}$$

Proof. The algorithm is similar to Corollary 43, but with $\alpha_A = \sqrt{s_r^A s_c^A}$, $\alpha_L = \sqrt{s_r^L s_c^L}$, $\alpha_\Omega = \sqrt{s_r^\Omega s_c^\Omega}$. □

5 Future Directions

Our algorithms for quantum linear regression with general ℓ_2-regularization made use of QSVT to implement various several matrix operations. However, it is possible to use QSVT directly to obtain the solution to *quantum ridge regression*. This requires computing a polynomial approximation for the transformation $\sigma \mapsto \sigma/(\sigma^2 + \lambda)$, to be applied on the singular values of A, which lie between $[1/\kappa_A, 1]$. However, it is unclear how to extend this while considering general ℓ_2-regularization. For instance, even when the data matrix and the penalty matrix share the same right singular vectors, this approach involves obtaining polynomial approximations to directly implement transformations of the form $\sigma \mapsto \sigma/(\sigma^2 + \lambda\widetilde{\sigma}^2)$, where $\widetilde{\sigma}$ is a singular value of the penalty matrix L. A monomial is no longer sufficient to approximate this quantum singular value transformation. It would be interesting to explore whether newly developed ideas of M-QSVT [RC22] can be used to implement such transformations directly with improved complexity.

While developing quantum machine learning algorithms, it is essential to point out the caveats, even at the risk of being repetitive [Aar15]. Our quantum algorithms output a quantum state $|x\rangle$ whose amplitudes encode the solution of the classical (regularized) linear regression problem. While given access to the data matrix and the penalty matrix, we achieve an exponential advantage over classical algorithms, this advantage is not generic. If similar assumptions (ℓ_2-sample and query access) are provided to a classical device, Gilyén et al. developed a quantum algorithm [GST22] for ridge regression (building upon [CGL+20]) which has a running time in $\mathcal{O}(\text{poly}(\kappa, \text{rank}(A), 1/\delta))$. This implies that any quantum algorithm for this problem can be at most polynomially faster in κ under these assumptions. One might posit that similar quantum-inspired classical algorithms for general ℓ_2-regression can also be developed. The exponential quantum speedup, however, is retained when the underlying matrices are sparse.

Another future direction of research would be to recast our algorithms in the framework of adiabatic quantum computing (AQC) following the works of [LT20b, AL22]. Quantum algorithms for linear systems in this framework have the advantage that a linear dependence on κ can be obtained without using complicated subroutines like variable-time amplitude amplification. The strategy is to implement these problems in the AQC model and then use time-dependent Hamiltonian simulation [LW18] to obtain their complexities in the circuit model. One caveat is that, so far, time-dependent Hamiltonian simulation algorithms have only been developed in the sparse-access model and therefore the advantage of the generality of the block-encoding framework is lost.

In the future, it would also be interesting to explore other quantum algorithms for machine learning such as principal component regression and linear support vector machines [RML14] using QSVT. Finally, following the results of [CdW21], it would be interesting to investigate techniques for quantum machine learning that do not require the quantum linear systems algorithm as a subroutine.

Acknowledgements

SC thanks András Gilyén, Stacey Jeffery and Jérémie Roland for useful discussions. SC acknowledges funding from the Science and Engineering Board, Department of Science

and Technology (SERB-DST), Government of India via grant number SRG/2022/000354. SC is also supported by IIIT Hyderabad via the Faculty Seed Grant. AP thanks Michael Walter for useful discussions. AP acknowledges support by the BMBF through project Quantum Methods and Benchmarks for Resource Allocation (QuBRA).

References

[Aar15] Scott Aaronson. Read the fine print. Nature Physics, 11(4):291–293, Apr 2015. doi:10.1038/nphys3272.

[AL22] Dong An and Lin Lin. Quantum linear system solver based on time-optimal adiabatic quantum computing and quantum approximate optimization algorithm. ACM Transactions on Quantum Computing, 3(2), mar 2022. doi:10.1145/3498331.

[Amb12] Andris Ambainis. Variable time amplitude amplification and quantum algorithms for linear algebra problems. In Christoph Dürr and Thomas Wilke, editors, 29th International Symposium on Theoretical Aspects of Computer Science (STACS 2012), volume 14 of Leibniz International Proceedings in Informatics (LIPIcs), pages 636–647, Dagstuhl, Germany, 2012. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.STACS.2012.636.

[Bis95] Chris M. Bishop. Training with noise is equivalent to tikhonov regularization. Neural Computation, 7(1):108–116, 1995. doi:10.1162/neco.1995.7.1.108.

[CdW21] Yanlin Chen and Ronald de Wolf. Quantum algorithms and lower bounds for linear regression with norm constraints. arXiv preprint, 2021. doi:10.48550/ARXIV.2110.13086.

[CGJ18] Shantanav Chakraborty, András Gilyén, and Stacey Jeffery. The power of block-encoded matrix powers: Improved regression techniques via faster hamiltonian simulation. arXiv preprint, 2018. doi:10.48550/arXiv.1804.01973.

[CGJ19] Shantanav Chakraborty, András Gilyén, and Stacey Jeffery. The Power of Block-Encoded Matrix Powers: Improved Regression Techniques via Faster Hamiltonian Simulation. In Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi, editors, 46th International Colloquium on Automata, Languages, and Programming (ICALP 2019), volume 132 of Leibniz International Proceedings in Informatics (LIPIcs), pages 33:1–33:14, Dagstuhl, Germany, 2019. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.ICALP.2019.33.

[CGL+20] Nai-Hui Chia, András Gilyén, Tongyang Li, Han-Hsuan Lin, Ewin Tang, and Chunhao Wang. Sampling-Based Sublinear Low-Rank Matrix Arithmetic Framework for Dequantizing Quantum Machine Learning, page 387–400. Association for Computing Machinery, New York, NY, USA, 2020. doi:10.1145/3357713.3384314.

[CKS17] Andrew M. Childs, Robin Kothari, and Rolando D. Somma. Quantum algorithm for systems of linear equations with exponentially improved dependence on precision. SIAM Journal on Computing, 46(6):1920–1950, Jan 2017. doi:10.1137/16M1087072.

[CYGL22] Menghan Chen, Chaohua Yu, Gongde Guo, and Song Lin. Faster quantum ridge regression algorithm for prediction. International Journal of Machine Learning and Cybernetics, Apr 2022. doi:10.1007/s13042-022-01526-6.

[EHN96] H.W. Engl, M. Hanke, and A. Neubauer. Regularization of Inverse Problems. Mathematics and Its Applications. Springer Netherlands, 1996. URL: https://link.springer.com/book/9780792341574.

[GHO99] Gene H. Golub, Per Christian Hansen, and Dianne P. O'Leary. Tikhonov regularization and total least squares. SIAM Journal on Matrix Analysis and Applications, 21(1):185–194, 1999. doi:10.1137/S0895479897326432.

[GSLW18] András Gilyén, Yuan Su, Guang Hao Low, and Nathan Wiebe. Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics. arXiv preprint, jun 2018. doi:10.48550/arXiv.1806.01838.

[GSLW19] András Gilyén, Yuan Su, Guang Hao Low, and Nathan Wiebe. Quantum singular value transformation and beyond: Exponential improvements for quantum matrix arithmetics. In Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, page 193–204, New York, NY, USA, 2019. Association for Computing Machinery. doi:10.1145/3313276.3316366.

[GST22] András Gilyén, Zhao Song, and Ewin Tang. An improved quantum-inspired algorithm for linear regression. Quantum, 6:754, June 2022. doi:10.22331/q-2022-06-30-754.

[GTC19] Yimin Ge, Jordi Tura, and J. Ignacio Cirac. Faster ground state preparation and high-precision ground energy estimation with fewer qubits. Journal of Mathematical Physics, 60(2):022202, 2019. doi:10.1063/1.5027484.

[Hem75] William J. Hemmerle. An explicit solution for generalized ridge regression. Technometrics, 17(3):309–314, 1975. URL: http://www.jstor.org/stable/1268066, doi:10.2307/1268066.

[HH93] Martin Hanke and Per Christian Hansen. Regularization methods for large-scale problems. Surv. Math. Ind, 3(4):253–315, 1993.

[HHL09] Aram W. Harrow, Avinatan Hassidim, and Seth Lloyd. Quantum algorithm for linear systems of equations. Physical Review Letters, 103(15), Oct 2009. doi:10.1103/physrevlett.103.150502.

[HK00] Arthur E. Hoerl and Robert W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. Technometrics, 42(1):80–86, 2000. URL: http://www.jstor.org/stable/1271436, doi:10.2307/1271436.

[KKR06] Julia Kempe, Alexei Kitaev, and Oded Regev. The complexity of the local hamiltonian problem. SIAM Journal on Computing, 35(5):1070–1097, 2006. doi:10.1137/S0097539704445226.

[KP17] Iordanis Kerenidis and Anupam Prakash. Quantum Recommendation Systems. In 8th Innovations in Theoretical Computer Science Conference (ITCS 2017), volume 67 of Leibniz International Proceedings in Informatics (LIPIcs), pages 49:1–49:21, Dagstuhl, Germany, 2017. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.ITCS.2017.49.

[KP20] Iordanis Kerenidis and Anupam Prakash. Quantum gradient descent for linear systems and least squares. Phys. Rev. A, 101:022316, Feb 2020. doi:10.1103/PhysRevA.101.022316.

[LC17a] Guang Hao Low and Isaac L Chuang. Hamiltonian simulation by uniform spectral amplification. arXiv:1707.05391, 2017. doi:10.48550/ARXIV.1707.05391.

[LC17b] Guang Hao Low and Isaac L. Chuang. Optimal hamiltonian simulation by

quantum signal processing. Phys. Rev. Lett., 118:010501, Jan 2017. doi:10.1103/PhysRevLett.118.010501.

[LC19] Guang Hao Low and Isaac L. Chuang. Hamiltonian simulation by qubitization. Quantum, 3:163, Jul 2019. doi:10.22331/q-2019-07-12-163.

[LMR14] Seth Lloyd, Masoud Mohseni, and Patrick Rebentrost. Quantum principal component analysis. Nature Physics, 10(9):631–633, Sep 2014. doi:10.1038/nphys3029.

[Low17] Guang Hao Low. Quantum signal processing by single-qubit dynamics. PhD thesis, Massachusetts Institute of Technology, 2017. URL: http://hdl.handle.net/1721.1/115025.

[LT20a] Lin Lin and Yu Tong. Near-optimal ground state preparation. Quantum, 4:372, December 2020. doi:10.22331/q-2020-12-14-372.

[LT20b] Lin Lin and Yu Tong. Optimal polynomial based quantum eigenstate filtering with application to solving quantum linear systems. Quantum, 4:361, November 2020. doi:10.22331/q-2020-11-11-361.

[LW18] Guang Hao Low and Nathan Wiebe. Hamiltonian simulation in the interaction picture. arXiv preprint, 2018. doi:10.48550/arXiv.1805.00675.

[LYC16] Guang Hao Low, Theodore J. Yoder, and Isaac L. Chuang. Methodology of resonant equiangular composite quantum gates. Phys. Rev. X, 6:041067, Dec 2016. doi:10.1103/PhysRevX.6.041067.

[Mar70] Donald W. Marquardt. Generalized inverses, ridge regression, biased linear estimation, and nonlinear estimation. Technometrics, 12(3):591–612, 1970. URL: http://www.jstor.org/stable/1267205, doi:10.2307/1267205.

[MRTC21] John M. Martyn, Zane M. Rossi, Andrew K. Tan, and Isaac L. Chuang. Grand unification of quantum algorithms. PRX Quantum, 2:040203, Dec 2021. doi:10.1103/PRXQuantum.2.040203.

[Mur12] Kevin P Murphy. Machine learning: a probabilistic perspective. MIT press, 2012. URL: https://mitpress.mit.edu/books/machine-learning-1.

[Pra14] Anupam Prakash. Quantum Algorithms for Linear Algebra and Machine Learning. PhD thesis, EECS Department, University of California, Berkeley, Dec 2014. URL: http://www2.eecs.berkeley.edu/Pubs/TechRpts/2014/EECS-2014-211.html.

[RC22] Zane M. Rossi and Isaac L. Chuang. Multivariable quantum signal processing (M-QSP): prophecies of the two-headed oracle. Quantum, 6:811, September 2022. doi:10.22331/q-2022-09-20-811.

[RML14] Patrick Rebentrost, Masoud Mohseni, and Seth Lloyd. Quantum support vector machine for big data classification. Phys. Rev. Lett., 113:130503, Sep 2014. doi:10.1103/PhysRevLett.113.130503.

[SSP16] Maria Schuld, Ilya Sinayskiy, and Francesco Petruccione. Prediction by linear regression on a quantum computer. Phys. Rev. A, 94:022342, Aug 2016. doi:10.1103/PhysRevA.94.022342.

[SX20] Changpeng Shao and Hua Xiang. Quantum regularized least squares solver with parameter estimate. Quantum Information Processing, 19(4):113, Feb 2020. doi:10.1007/s11128-020-2615-9.

[Tan19] Ewin Tang. A quantum-inspired classical algorithm for recommendation systems. In Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, page 217–228, New York, NY, USA, 2019. Association for Computing Machinery. doi:10.1145/3313276.3316310.

[Vin78] Hrishikesh D. 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. URL: http://www.jstor.org/stable/1924340, doi:10.2307/1924340.

[vW15] Wessel N. van Wieringen. Lecture notes on ridge regression, 2015. doi:10.48550/ARXIV.1509.09169.

[Wan17] Guoming Wang. Quantum algorithm for linear regression. Phys. Rev. A, 96:012335, Jul 2017. doi:10.1103/PhysRevA.96.012335.

[WBL12] Nathan Wiebe, Daniel Braun, and Seth Lloyd. Quantum algorithm for data fitting. Phys. Rev. Lett., 109:050505, Aug 2012. doi:10.1103/PhysRevLett.109.050505.

[YGW21] Chao-Hua Yu, Fei Gao, and Qiao-Yan Wen. An improved quantum algorithm for ridge regression. IEEE Transactions on Knowledge and Data Engineering, 33(3):858–866, 2021. doi:10.1109/TKDE.2019.2937491.

A Algorithmic Primitives

This appendix contains detailed proofs for certain lemmas and corollaries in Section 3, for completeness. These proofs are not necessary to understand the techniques and results of the paper, but may help the reader develop a better intuition for the methods used.

Corollary 12 (Uniform Block Amplification)**.** *Let $A \in \mathbb{R}^{N\times d}$ and $\delta \in (0,1]$. Suppose U is a (α, a, ε)-block-encoding of A, such that $\varepsilon \leq \frac{\delta}{2}$, that can be implemented at a cost of T_U. Then a $(\sqrt{2}\|A\|, a+1, \delta)$-block-encoding of A can be implemented at a cost of $\mathcal{O}\Big(\frac{\alpha T_U}{\|A\|}\log\left(\|A\|/\delta\right)\Big)$.*

Proof. We can re-interpret U as a $(\alpha/\|A\|, a, \varepsilon/\|A\|)$-block-encoding of $A/\|A\|$. Invoking Lemma 11 with $\gamma = \frac{\delta}{2\|A\|}$, we get U', a $(\sqrt{2}, a+1, \varepsilon/\|A\| + \frac{\delta}{2\|A\|})$-block-encoding of $A/\|A\|$, implemented at a cost of $\mathcal{O}\Big(\frac{\alpha}{\|A\|}T_U \log\left(\|A\|/\delta\right)\Big)$ which is a $(\sqrt{2}\|A\|, a+1, \delta)$-block-encoding of A. □

Lemma 13 (Applying a Block-encoded Matrix on a Quantum State)**.** *Let A be an s-qubit operator such that its non-zero singular values lie in $[\|A\|/\kappa, \|A\|]$. Also let $\delta \in (0,1)$, and U_A be an (α, a, ε)-block-encoding of A, implementable in time T_A, such that*

$$\varepsilon \leq \frac{\delta\|A\|}{2\kappa}.$$

Furthermore, suppose $|b\rangle$ be an s-qubit quantum state, prepared in time T_b. Then we can prepare a state that is δ-close to $\frac{A|b\rangle}{\|A|b\rangle\|}$ with success probability $\Omega(1)$ at a cost of

$$\mathcal{O}\Big(\frac{\alpha\kappa}{\|A\|}(T_A + T_b)\Big)$$

Proof. The proof is similar to Lemma 24 of [CGJ19]. We have $\|A|b\rangle\| \geq \frac{\|A\|}{\kappa}$. By applying U_A to $|0\rangle|b\rangle$ (implementable at a cost of $T_A + T_B$), followed by $\frac{\alpha\kappa}{\|A\|}$-rounds of amplitude amplification (conditioned on having $|0\rangle$ in the first register) , we obtain a quantum state that within δ of $|0\rangle \otimes \frac{A|b\rangle}{\|A|b\rangle\|}$. □

Corollary 14 (Applying a pre-amplified Block-encoded Matrix on a Quantum State)**.** *Let A be an s-qubit operator such that its non-zero singular values lie in $[\|A\|/\kappa, \|A\|]$. Also let $\delta \in (0,1)$, and U_A be an (α, a, ε)-block-encoding of A, implementable in time T_A, such that*

$$\varepsilon \leq \frac{\delta \|A\|}{4\kappa}.$$

Furthermore, suppose $|b\rangle$ be an s-qubit quantum state that can be prepared in time T_b. Then we can prepare a state that is δ-close to $\frac{A|b\rangle}{\|A|b\rangle\|}$ with success probability $\Omega(1)$ at a cost of

$$\mathcal{O}\left(\frac{\alpha\kappa}{\|A\|}\log\left(\frac{\kappa}{\delta}\right)T_A + \kappa T_b\right)$$

Proof. We first pre-amplify the unitary using Corollary 12 with some $\gamma \geq 2\varepsilon$. We get a $(\sqrt{2}\|A\|, a+1, \gamma)$-block-encoding of A implemented at a cost of

$$T_{A'} := \mathcal{O}\left(\frac{\alpha T_A}{\|A\|}\log\left(\frac{\|A\|}{\gamma}\right)\right)$$

Now we invoke Lemma 13 with $\delta = \frac{2\kappa\gamma}{\|A\|}$ and the above unitary to prepare the state, which has a time complexity of

$$\mathcal{O}(\kappa(T_{A'} + T_b)) = \mathcal{O}\left(\frac{\alpha\kappa}{\|A\|}\log\left(\frac{\kappa}{\delta}\right)T_A + \kappa T_b\right)$$

□

Lemma 15 (Robustness of state preparation)**.** *Let A be an s-qubit operator such that its non-zero singular values lie in $[\|A\|/\kappa, \|A\|]$. Suppose $|b'\rangle$ is a quantum state that is $\varepsilon/2\kappa$-close to $|b\rangle$ and $|\psi\rangle$ is a quantum state that is $\varepsilon/2$-close to $A|b'\rangle/\|A|b'\rangle\|$. Then we have that $|\psi\rangle$ is ε-close to $A|b\rangle/\|A|b\rangle\|$.*

Proof. We know that

$$\||b\rangle - |b'\rangle\| \leq \frac{\varepsilon}{2\kappa}$$

and

$$\left\||\psi\rangle - \frac{A|b'\rangle}{\|A|b'\rangle\|}\right\| \leq \frac{\varepsilon}{2}$$

For small enough $\varepsilon \ll \kappa$, we can assume that $\|A|b\rangle\| \approx \|A|b'\rangle\|$. We can derive the final error as

$$\begin{aligned}
\left\||\psi\rangle - \frac{A|b\rangle}{\|A|b\rangle\|}\right\| &= \left\||\psi\rangle - \frac{A|b\rangle - A|b'\rangle + A|b'\rangle}{\|A|b\rangle\|}\right\| \\
&= \left\||\psi\rangle - \frac{A|b'\rangle}{\|A|b\rangle\|} + \frac{A|b'\rangle - A|b'\rangle}{\|A|b\rangle\|}\right\| \\
&\leq \left\||\psi\rangle - \frac{A|b'\rangle}{\|A|b'\rangle\|}\right\| + \left\|\frac{A|b'\rangle - A|b'\rangle}{\|A|b\rangle\|}\right\| \\
&\leq \frac{\varepsilon}{2} + \frac{\|A\|\||b\rangle - |b'\rangle\|}{\|A|b\rangle\|} \\
&\leq \frac{\varepsilon}{2} + \frac{\varepsilon}{2} \\
&= \varepsilon
\end{aligned}$$

□

A.1 Arithmetic with Block-Encoded Matrices

Lemma 17 (Linear Combination of Block Encoded Matrices)**.** *For each $j \in \{0, \ldots, m-1\}$, let A_j be an s-qubit operator, and $y_j \in \mathbb{R}^+$. Let U_j be a $(\alpha_j, a_j, \varepsilon_j)$-block-encoding of A_j, implemented in time T_j. Define the matrix $A = \sum_j y_j A_j$, and the vector $\eta \in \mathbb{R}^m$ s.t. $\eta_j = y_j \alpha_j$. Let U_η be a η state-preparation unitary, implemented in time T_η. Then we can implement a*

$$\left(\sum_j y_j \alpha_j, \max_j(a_j) + s, \sum_j y_j \varepsilon_j \right)$$

block-encoding of A at a cost of $\mathcal{O}\left(\sum_j T_j + T_\eta\right)$.

Proof. Let $a = max_j(a_j) + s$ and $\alpha = \sum_j y_j \alpha_j$. For each $j \in \{0, \ldots, m-1\}$, construct the extended unitary U_j' by padding ancillas to U_j, i.e. $U_j' = I_{a-s-a_j} \otimes U_j$. Note that U_j' is a $(\alpha_j, a-s, \varepsilon_j)$-block-encoding of A_j. Let $B_j = (\langle 0|^{a_j} \otimes I_s) U_j (|0\rangle^{a_j} \otimes I_s)$ denote the top left block of U_j and U_j', and observe that $\|A_j - \alpha_j B_j\| \leq \varepsilon_j$. We also construct P — an η state-preparation unitary s.t. $P|0\rangle = \sum_j \sqrt{y_j \alpha_j} |j\rangle$ — by invoking Definition 16.

Consider the unitary $W = (P^\dagger \otimes I_{a-1} \otimes I_s)(\sum_j |j\rangle\langle j| \otimes U_j')(P \otimes I_{a-1} \otimes I_s)$. This is a (α, a, ε)-block-encoding of $A = \sum_j y_j A_j$, where ε is computed as:

$$\begin{aligned}
\|A - \alpha(\langle 0|^a \otimes I_s) W (|0\rangle^a \otimes I_s)\| &= \left\| \sum_{j=0}^{m-1} y_j A_j - \alpha(\langle 0|^a \otimes I_s) W (|0\rangle^a \otimes I_s) \right\| \\
&= \left\| \sum_j y_j A_j - \alpha(\langle 0|^a \otimes I_s)(\sum_j P^\dagger |j\rangle\langle j| P \otimes U_j')(|0\rangle^a \otimes I_s) \right\| \\
&= \left\| \sum_j y_j A_j - \alpha \sum_j \langle 0| P^\dagger |j\rangle\langle j| P |0\rangle \otimes B_j \right\| \\
&= \left\| \sum_j \left(y_j A_j - \alpha \langle 0| P^\dagger |j\rangle\langle j| P |0\rangle B_j \right) \right\| \\
&= \left\| \sum_j \left(y_j A_j - \alpha \left(\frac{y_j \alpha_j}{\alpha} \right) B_j \right) \right\| \\
&\leq \sum_j y_j \|A_j - \alpha_j B_j\| \\
&\leq \sum_j y_j \varepsilon_j = \varepsilon
\end{aligned}$$

□

Lemma 21 (Tensor Product of Block Encoded Matrices)**.** *Let U_1 and U_2 be $(\alpha, a, \varepsilon_1)$ and $(\beta, b, \varepsilon_2)$-block-encodings of A_1 and A_2, s and t-qubit operators, implemented in time T_1 and T_2 respectively. Define $S := \Pi_{i=1}^{s} \mathit{SWAP}_{a+b+i}^{a+i}$. Then, $S(U_1 \otimes U_2)S^\dagger$ is an $(\alpha\beta, a+b, \alpha\varepsilon_2 + \beta\varepsilon_1 + \varepsilon_1\varepsilon_2)$ block-encoding of $A_1 \otimes A_2$, implemented at a cost of $\mathcal{O}(T_1 + T_2)$.*

Proof. From the property of Kronecker products $(A \otimes B)(C \otimes D) = (AC) \otimes (BD)$. For $j \in \{1, 2\}$ let $\tilde{A}_j = \left(\langle 0|^{\otimes a_j} \otimes I_s\right) U_j \left(|0\rangle^{\otimes a_j} \otimes I_s\right)$. Then it follows that

$$\left(\langle 0|^{\otimes a} \otimes I_s \otimes \langle 0|^{\otimes b} \otimes I_t\right) (U_1 \otimes U_2) \left(|0\rangle^{\otimes a} \otimes I_s \otimes |0\rangle^{\otimes b} \otimes I_t\right) = \tilde{A}_1 \otimes \tilde{A}_2 \tag{56}$$

Therefore $\tilde{A}_1 \otimes \tilde{A}_2$ is block-encoded in $U_1 \otimes U_2$ as a non-principal block-encoding, and we can use SWAP gates to move it to the principal block as follows.

$$\begin{aligned} S\left(|0\rangle^{\otimes a} \otimes I_s |0\rangle^{\otimes b} \otimes I_t\right) &= \Pi_{i=1}^{s} \mathtt{SWAP}_{a+b+i}^{a+i} \left(|0\rangle^{\otimes a} \otimes I_s |0\rangle^{\otimes b} \otimes I_t\right) \\ &= \Pi_{i=1}^{s-1} \mathtt{SWAP}_{a+b+i}^{a+i} \mathtt{SWAP}_{a+b+s}^{a+s} \left(|0\rangle^{\otimes a} \otimes I_s |0\rangle^{\otimes b} \otimes I_t\right) \\ &= \Pi_{i=1}^{s-1} \mathtt{SWAP}_{a+b+i}^{a+i} \left(|0\rangle^{\otimes a} \otimes I_{s-1} |0\rangle^{\otimes b} \otimes I_{t+1}\right) \\ &= \dots \\ &= |0\rangle^{\otimes a+b} \otimes I_{s+t} \end{aligned}$$

Similarly,

$$\left(\langle 0|^{\otimes a} \otimes I_s \otimes \langle 0|^{\otimes b} \otimes I_t\right) S^\dagger = \langle 0|^{\otimes a+b} \otimes I_{s+t}.$$

From Equation 56 we have

$$\begin{aligned} \tilde{A}_1 \otimes \tilde{A}_2 &= \left(\langle 0|^{\otimes a} \otimes I_s \otimes \langle 0|^{\otimes b} \otimes I_t\right) S^\dagger S (U_1 \otimes U_2) S^\dagger S \left(|0\rangle^{\otimes a} \otimes I_s |0\rangle^{\otimes b} \otimes I_t\right) \\ &= \left(\langle 0|^{\otimes a+b} \otimes I_{s+t}\right) S (U_1 \otimes U_2) S^\dagger \left(|0\rangle^{\otimes a+b} \otimes I_{s+t}\right) \end{aligned}$$

Next, we look at the subnormalization and error terms.

$$\begin{aligned} \left\| A_1 \otimes A_2 - \alpha\beta \tilde{A}_1 \otimes \tilde{A}_2 \right\|_2 &\leq \left\| (\alpha \tilde{A}_1 + \varepsilon_1 I_s) \otimes (\beta \tilde{A}_2 + \varepsilon_2 I_t) - \alpha \tilde{A}_1 \otimes \beta \tilde{A}_2 \right\|_2 \\ &= \left\| \alpha \tilde{A}_1 \otimes \varepsilon_2 I_2 + \varepsilon_1 I_s \otimes \beta \tilde{A}_2 + \varepsilon_1 I_s \otimes \varepsilon_2 I_2 \right\|_2 \\ &\leq \alpha\varepsilon_2 \left\| \tilde{A}_1 \right\|_2 + \beta\varepsilon_2 \left\| \tilde{A}_2 \right\|_2 + \varepsilon_1\varepsilon_2 \\ &= \alpha\varepsilon_2 + \beta\varepsilon_1 + \varepsilon_1\varepsilon_2 \end{aligned}$$

where we have used $\|A_1\|_2 \leq \alpha \left\| \tilde{A}_1 \right\|_2 + \varepsilon_1$ and $\left\| \tilde{A}_1 \right\|_2 \leq 1$ and similarly for A_2.

□