Linear Regression by Quantum Amplitude Estimation and its Extension to Convex Optimization

Kazuya Kaneko,[1] Koichi Miyamoto,[2, 1, *] Naoyuki Takeda,[1] and Kazuyoshi Yoshino[1]

[1]*Mizuho-DL Financial Technology Co., Ltd.*
2-4-1 Kojimachi, Chiyoda-ku, Tokyo, 102-0083, Japan
[2]*Center for Quantum Information and Quantum Biology, Osaka University*
1-3 Machikaneyama, Toyonaka, Osaka, 560-8531, Japan

Linear regression is a basic and widely-used methodology in data analysis. It is known that some quantum algorithms efficiently perform least squares linear regression of an exponentially large data set. However, if we obtain values of the regression coefficients as classical data, the complexity of the existing quantum algorithms can be larger than the classical method. This is because it depends strongly on the tolerance error ϵ: the best one among the existing proposals is $O(\epsilon^{-2})$. In this paper, we propose a new quantum algorithm for linear regression, which has a complexity of $O(\epsilon^{-1})$ and keeps a logarithmic dependence on the number of data points N_D. In this method, we overcome bottleneck parts in the calculation, which take the form of the sum over data points and therefore have a complexity proportional to N_D, using quantum amplitude estimation, and other parts classically. Additionally, we generalize our method to some class of convex optimization problems.

This is a short version of [1]. All proofs are omitted in this document but presented in [1].

I. INTRODUCTION

Following the rapid advance of quantum computing technology, many quantum algorithms have been proposed and their applications to the wide range of practical problems have been studied in the recent researches. One prominent example is linear regression. Linear regression, which is based on the least squares method in many cases, is a basic and ubiquitous methodology for many fields in natural and social sciences. There are some quantum algorithms for linear regression [2–7] with complexity depending on the number of data points N_D as $O(\text{polylog}(N_D))$ [1]. This means the exponential speedup compared with the naive classical method explained in Sec. II B, whose complexity is proportional to N_D.

However, despite this, the existing quantum methods are not necessarily more beneficial than classical ones, when we want to obtain the values of the optimized regression coefficients *as classical data*, and N_D is *mid-sized*, say $O(10^3-10^5)$. Roughly speaking, in the existing methods such as the first proposed one [2], which is based on Harrow-Hassidim-Lloyd (HHL) algorithm [11] for solving systems of linear equations, the authors create quantum states in which the values of the coefficients are encoded and read out in classical form. Therefore, it is inevitable that the estimated coefficients are accompanied by errors and high-accuracy estimation leads to large complexity. As far as we know, the existing method with the best complexity with respect to the tolerance error ϵ is that in [4]: the complexity of estimating coefficients with additive error at most ϵ is

$$O\left(\frac{d^{5/2}\kappa^3}{\epsilon^2}\text{polylog}\left(\frac{d\kappa}{\epsilon}\right)\right), \tag{1}$$

where d is the number of the coefficients (or the explanatory variables) and κ is the condition number of the design matrix (defined in Sec. II B). On the other hand, a naive classical method, which is explained in Sec. II B, has complexity $O\left(d^2 N_D\right)$. Therefore, assuming the prefactors in these expressions of complexities are comparable, under an ordinary situation $\epsilon \sim 10^{-3}, d \sim 10, \kappa \sim 1$, the minimum N_D for which the quantum method is advantageous over classical one is $N_D \sim 10^6$. Although in some problems N_D is of such an order or much larger, cases where $N_D \sim 10^3$ or 10^4 are also ubiquitous, and in such cases the exiting quantum methods are inferior to classical ones.

The mid-sized regression problems are often time-consuming and desired to be sped up, although we might naively think that such problems can be solved by classical computers in short time. For example, if such regressions are repeated in the whole of the calculation flow, the total computational time can be long. One of such cases is *least square Monte Carlo (LSM)* [12]. LSM is a methodology used in pricing financial derivatives[2] with an early exercise option, that is, the contract term stating that either of two parties in the contract can terminate it before the final maturity[3]. In Monte Carlo pricing, we generate many (say, $O(10^4-10^5)$) paths of the random time evolution of underlying assets and estimate the price as the expected cashflow. In the case of early-exercisable products, we have to determine the optimal exercise time for each path. In LSM, we approximate the continuation value of the derivative at each exercise date by linear regression using certain functions of underlying asset prices as explanatory variables, whose number is typically $d \sim 10$. Since regression is done many times in pricing one contract and banks have numerous contracts, they have huge complexity in total and are meaningful targets of quantum speedup,

[*] koichi.miyamoto@qiqb.osaka-u.ac.jp

[1] There are also quantum-inspired classical methods [8–10].

[2] Financial derivatives, or simply derivatives, are financial contracts in which two parties exchanges cashflows whose amounts are determined by the price of some assets. As textbooks for derivatives and pricing of them, we refer to [13, 14].

[3] For the detail of LSM, see [12].

even though each of them is mid-sized.

Based on the above motivation, in this paper, we present a new quantum algorithm for linear regression, focusing on reducing the order of the inverse of ϵ in the expression of the complexity. In our method, unlike existing methods, we do not perform all calculation on a quantum computer. Instead, we use a quantum computer *only to perform a bottleneck part in the naive classical method.* That is, we perform some intermediate calculation, which is the sum over N_D terms and therefore has the complexity proportional to N_D naively, by quantum algorithm. Then, we classically solve the d-dimensional system of linear equations, whose coefficients and constant terms are outputs of the preceding quantum computation, and obtain the regression coefficients. A classical computer can perform this step in negligible time for $d \sim 10$, without inducing any error. Also note that we naturally obtain the regression coefficients as classical data.

In order to speed up the bottleneck sums, we use *quantum amplitude estimation (QAE)* [15–21], which is also used to speed up Monte Carlo integration [16, 22]. Then, as discussed in detail in Sec. III, we can perform the sums with complexity

$$O\left(\max\left\{\frac{d^{3/2}\kappa^4}{\epsilon}, d\kappa^2\right\} \times d^2 \log(d)\right),$$

and the total complexity of the whole of our method is almost equal to this. Compared with the naive classical method, whose complexity is $O(d^2 N_D)$, our method accomplishes speedup with respect to N_D. Besides, compared with the existing quantum methods, our method improves the dependency of the complexity on ϵ from ϵ^{-2} in (1) to ϵ^{-1}. Unfortunately, in out method, the dependencies on d and κ are worse than those of (1) and the naive classical method. Therefore, our method is more suitable to the situation such that

$$d, \kappa \ll \frac{1}{\epsilon} \ll N_D, \tag{2}$$

which includes typical mid-sized problems such as LSM.

In addition to linear regression, we generalize our method to some class of convex optimization. As we see in IV, linear regression can be considered as an optimization problem of the sum of quadratic functions solved by Newton's method. Inspired by this, we consider a convex optimization problem where an objective function is written as sums of many terms similarly to linear regression, and present a quantum algorithm of Newton's method, in which calculation of the gradient and the Hessian is sped up by QAE. We also estimate complexity to obtain a solution with desired error level under some mathematical assumptions which are usually made in the convergence analysis of Newton's method.

The remaining parts of this paper are organized as follows. Sec. II is a preliminary section. In Sec. II A, we present some notations for the later discussion. In Sec. II B, we briefly review linear regression and the classical method for it. In Sec. III, we explain our method for linear regression in detail. In Sec. IV, we discuss the extension of our method to some convex optimization problems. Sec. V summarizes this paper.

II. PRELIMINARY

A. Notations

For a vector $\vec{v} \in \mathbb{R}^n$, $\left\|\vec{v}\right\|$ and $\left\|\vec{v}\right\|_\infty$ denotes the Euclidean norm and the max norm, respectively. For a matrix $A \in \mathbb{R}^{m\times n}$, $||A||$ denotes the spectral norm, which is equal to the maximum singular value of A. For a full-rank matrix $A \in \mathbb{R}^{m\times n}$, $\kappa(A)$ denotes its condition number.

B. Linear regression

We here define the problem of linear regression. Assume that we have N_D data points $\mathcal{D} := \{(\vec{x}_i, y_i)\}_{i=1,..,N_D}$, each of which consists of a vector of d explanatory variables[4] $\vec{x}_i = (x_i^{(1)}, ..., x_i^{(d)})^T \in \mathbb{R}^d$ and an objective variable $y_i \in \mathbb{R}$. Linear regression attempts to fit the linear combination of the explanatory variables to the objective variable, that is, finding $\vec{a}$ such that

$$\vec{y} \simeq X\vec{a}. \tag{3}$$

Here, $a_1, ..., a_d \in \mathbb{R}$ are model parameters called *regression coefficients* and $\vec{a} := (a_1, ..., a_d)^T \in \mathbb{R}^d$. $\vec{y} := (y_1, ..., y_{N_D})^T$ and

$$X := \begin{pmatrix} x_1^{(1)} & \cdots & x_1^{(d)} \\ \vdots & \ddots & \vdots \\ x_{N_D}^{(1)} & \cdots & x_{N_D}^{(d)} \end{pmatrix} \tag{4}$$

is called the *design matrix*. In this paper, as in many cases, we determine $\vec{a}$ by the *least squares method*, that is,

$$\vec{a} = \underset{\vec{a}'}{\mathrm{argmin}} \left\|\vec{y} - X\vec{a}'\right\|^2. \tag{5}$$

As is well known (see [23] for example), the solution of (5) is given by

$$\vec{a} = \left(X^T X\right)^{-1} X^T \vec{y}, \tag{6}$$

where we assume that X is full-rank, as stated again in Sec. III B as Assumption 2.

Here, let us introduce some symbols. We define $d\times d$ matrix W,

$$W := \frac{1}{N_D} X^T X, \tag{7}$$

whose (i, j)-element is

$$w_{ij} = \frac{1}{N_D} \sum_{k=1}^{N_D} x_k^{(i)} x_k^{(j)}. \tag{8}$$

[4] If we want to consider the intercept, we include a dummy variable (say $x_i^{(1)}$) in each explanatory variable vector and set it to 1: $x_1^{(1)} = ... = x_{N_D}^{(1)} = 1$.

W is invertible since we assumed that X is full-rank. We also define the d-dimensional vector $\vec{z}$,

$$\vec{z} := \frac{1}{N_D} X^T \vec{y}, \tag{9}$$

whose i-th element is

$$z_i = \frac{1}{N_D} \sum_{k=1}^{N_D} x_k^{(i)} y_k. \tag{10}$$

In (7) and (9), the prefactor $1/N_D$ is just for the later convenience. Using these, (6) becomes

$$\vec{a} = W^{-1} \vec{z}. \tag{11}$$

We here call the following classical computation *the naive classical method*: calculate w_{ij}'s and z_i's by simply repeating multiplications and additions N_D times, literally following the definitions (8) and (10), and then solve the d-dimensional system of linear equations (11) to find $\vec{a}$. Let us discuss the complexity of this method. Since we are considering the situation $d \ll N_D$, we focus only on the major contributions with respect to N_D. Calculating one w_{ij} or z_i takes complexity of $O(N_D)$. Since the numbers of w_{ij}'s and z_i's are $O(d^2)$ and $O(d)$, respectively, and their total is $O(d^2)$, the total complexity of calculating all of w_{ij}'s and z_i's is $O(d^2 N_D)$. Then, solving (11) takes the negligible complexity, since it does not depend on N_D. For example, even if we use the elementary method such as row reduction, the complexity is $O(d^3)$. In summary, the complexity of the naive classical method is $O(d^2 N_D)$, which dominantly comes from calculation of w_{ij}'s and z_i's.

III. LINEAR REGRESSION BY QUANTUM AMPLITUDE ESTIMATION

A. Quantum Amplitude Estimation

In this section, we present our method for linear regression. Before this, let us review the outline of QAE briefly.

QAE is the algorithm to estimate a probability amplitude of a marked state in a superposition. Consider the system consisting of some qubits. We set the system to the initial state where all qubits are set to $|0\rangle$ and write such a state as $|0\rangle$ for simplicity. Then, we assume that there exists an unitary transformation A on the system such that

$$A|0\rangle = a|\Psi\rangle + \sqrt{1-a^2}|\Psi_\perp\rangle, \tag{12}$$

where $|\Psi\rangle$ is the 'marked state', $|\Psi_\perp\rangle$ is a state orthogonal to $|\Psi\rangle$ and $0 < a < 1$. Typically, $|\Psi\rangle$ and $|\Psi_\perp\rangle$ are the states where a specific qubit takes $|1\rangle$ and $|0\rangle$ respectively. In addition to A, we use the following unitary operators S_0 and S_Ψ, which are defined as

$$S_0|\phi\rangle = \begin{cases} -|0\rangle & ;\text{if } |\phi\rangle = |0\rangle \\ |\phi\rangle & ;\text{if } |\phi\rangle \text{ is orthogonal to } |0\rangle \end{cases}, \tag{13}$$

$$S_\Psi|\phi\rangle = \begin{cases} -|\Psi\rangle & ;\text{if } |\phi\rangle = |\Psi\rangle \\ |\phi\rangle & ;\text{if } |\phi\rangle \text{ is orthogonal to } |\Psi\rangle \end{cases}. \tag{14}$$

S_0 can be constructed using a multi-controlled Toffoli gate, and S_Ψ is simply a controlled-Z gate if $|\Psi\rangle$ is defined as the state where a specific qubit is $|1\rangle$. Then, defining

$$Q := -AS_0A^{-1}S_\Psi, \tag{15}$$

we can construct a quantum algorithm (see [15] for details) which makes

$$O\left(\frac{1}{\epsilon}\right) \tag{16}$$

uses of Q (therefore, $O(1/\epsilon)$ uses of A) and outputs an estimate of a with an ϵ-additive error.

We here make a comment on success probability. In the algorithm of QAE [15], the success probability, that is, the probability that the algorithm outputs the estimation with the desired additive error is not 1 but lower-bounded by $8/\pi^2$. However, we can enhance the success probability to an arbitrary level $1-\gamma$, where $\gamma \in (0,1)$, by repeating QAE $O\left(\log(\gamma^{-1})\right)$ times. That is, taking the median of the results in the $O\left(\log(\gamma^{-1})\right)$ runs of QAE, we can obtain the estimation with the additive error ϵ with probability $1-\gamma$ [22, 24]. Considering this point, we can write the number of calls to A in repeating QAE with an ϵ-additive error and a success probability larger than $1-\gamma$ as

$$O\left(\frac{\log(\gamma^{-1})}{\epsilon}\right). \tag{17}$$

If we set $1-\gamma$ to some fixed value, say 99%, (17) is reduced to (16).

B. Assumptions

Next, we present some assumptions which are necessary for the method. The first one is as follows.

Assumption 1. *The following oracles P_x and P_y are available:*

$$P_x : |i\rangle|k\rangle|0\rangle \mapsto |i\rangle|k\rangle|x_k^{(i)}\rangle, \tag{18}$$

$$P_y : |k\rangle|0\rangle \mapsto |k\rangle|y_k\rangle, \tag{19}$$

for any $i \in \{1,...,d\}$ and $k \in \{1,...,N_D\}$.

Here and hereafter, for a number x, the ket $|x\rangle$ corresponds to a computational basis state on a quantum register where the bit string represents the binary representation of x. (18) and (19) mean that P_x and P_y output the element of X and $\vec{y}$, respectively, for the specified index. Previous papers [2–7] also assume such oracles. We can construct P_x and P_y if quantum random access memories (QRAMs) [25] are available.

The second assumption is just a reproduction.

Assumption 2. *X defined as (4) is full-rank.*

Because of this, $\kappa(X)$, the condition number of X, can be defined, and W defined as (8) is invertible. Hereafter, we simply write $\kappa(X)$ as κ.

The third assumption is as follows.

Assumption 3.

$$0 \leq x_k^{(i)} \leq 1, 0 \leq y_k \leq 1 \tag{20}$$

for any $i \in \{1, ..., d\}$ and $k \in \{1, ..., N_D\}$.

That is, we assume that the explanatory variables and the objective variable are bounded by 0 and 1. Besides, we make the fourth assumption as follows.

Assumption 4. *There is a positive number c (say, $\frac{1}{2}$), which is independent of N_D, ϵ, κ and d, such that*

$$\forall i \in \{1, ..., d\}, w_{ii} = \frac{1}{N_D} \sum_{k=1}^{N_D} \left(x_k^{(i)}\right)^2 > c. \tag{21}$$

Since $w_{ii} \leq 1$ is immediately derived from Assumption 3, Assumption 4 means that $c < w_{ii} \leq 1$.

Although Assumption 3 and 4 are too strong seemingly, they should be assumed for successful regression, regardless of whether in the classical or quantum way, for the following reasons. First, we note that Assumption 3 is satisfied if we know the bounds for $x_k^{(i)}$ and y_k in advance and can rescale them. That is, for $i \in \{1, ..., d\}$, although in general $x_k^{(i)}$'s are not in $[0, 1]$, we can redefine

$$\tilde{x}_k^{(i)} := \frac{x_k^{(i)} - L_i}{U_i - L_i} \tag{22}$$

as $x_k^{(i)}$ if we know L_i, U_i such that

$$\forall k \in \{1, ..., N_D\}, L_i \leq x_k^{(i)} \leq U_i, \tag{23}$$

Then, redefined $x_k^{(i)}$'s are in $[0, 1]$. Similarly, redefining

$$\tilde{y}_k := \frac{y_k - L_y}{U_y - L_y} \tag{24}$$

as y_k with L_y, U_y such that

$$\forall k \in \{1, ..., N_D\}, L_y \leq y_k \leq U_y \tag{25}$$

leads to $0 \leq y_k \leq 1$. Assumption 3 is then satisfied. Besides, if we know the bounds which are not too far from the typical scale of the original $x_k^{(i)}$'s, that is, if we can take L_i, U_i such that $|L_i| \sim |U_i| \sim |x_k^{(i)}|$ for most k's, Assumption 4 is satisfied. In summary, Assumption 3 and 4 are naturally satisfied if we know the typical scales of $x_k^{(i)}$'s and y_k's. Practically, we *must* know the typical scales, since we have to address the problem of *outliers*. Data sets often contain points whose explanatory and/or objective variables are much larger than those of others, because of various reasons (for example, misrecord). Such points are called outliers. It is widely known that outliers lead to inaccurate regression and so we have to address them. Typically, we omit them from data points used for regression or replace the values of the explanatory and/or objective variables out of some range with the upper or lower bound of the range. For such a preprocess, we have to know the typical scales of the variables. In fact, the previous paper [4] makes assumptions similar to Assumption 3 and 4. Mentioning the necessity of preprocessing outliers, it assumes that the design matrix and the objective variable vector do not contain the extraordinarily large elements.

C. Details of our method

We now explain our method in detail. First, we present a lemma on the error in the solution of a system of linear equations where coefficients and constant terms contain errors.

Lemma 1. *Let Assumptions 2 to 4 be satisfied. For given symmetric $\hat{W} \in \mathbb{R}^{d \times d}$ and $\hat{\vec{z}} \in \mathbb{R}^d$, consider a system of linear equations*

$$\hat{W}\hat{\vec{a}} = \hat{\vec{z}}, \tag{26}$$

where $\hat{\vec{a}} \in \mathbb{R}^d$. For a given $\epsilon > 0$, if each element of $\delta W := \hat{W} - W$ and $\delta\vec{z} := \hat{\vec{z}} - \vec{z}$ has an absolute value smaller than ϵ' such that

$$\epsilon' < \min\left\{\frac{c}{d\kappa^2}, \frac{c^2\epsilon}{2d^{3/2}\kappa^4}\right\} \tag{27}$$

then $\hat{\vec{a}}$ is uniquely determined by solving Eq. (26), i.e., $\hat{\vec{a}} = \hat{W}^{-1}\hat{\vec{z}}$, and becomes an $O(\epsilon)$-additive approximation of $\vec{a}$, which means that

$$\|\hat{\vec{a}} - \vec{a}\|_\infty = O(\epsilon). \tag{28}$$

This lemma means that, if we want a solution of a system of linear equations with an $O(\epsilon)$-additive error, it is sufficient to calculate coefficients and constant terms with an additive error ϵ' satisfying (27).

We therefore propose the following method for linear regression: we first estimate W in (8) and $\vec{z}$ in (10) by a quantum method with ϵ'-additive error and then calculate (11) by some classical method. Since classical methods basically introduces no additional error, we can obtain a solution with $O(\epsilon)$-additive error.

Then, we state a theorem on the complexity of our method.

Theorem 1. *Given $\epsilon > 0$, accesses to oracles P_x and P_y which satisfy Assumptions 1, and $\{x_k^{(i)}\}_{\substack{i=1,...,d \\ k=1,...,N_D}}$, $\{y_k\}_{k=1,...,N_D}$ which satisfy Assumption 2 to 4, there is a quantum algorithm that makes*

$$O\left(\max\left\{\frac{d^{3/2}\kappa^4}{\epsilon}, d\kappa^2\right\} \times d^2 \log(d)\right) \tag{29}$$

uses of P_x and

$$O\left(\max\left\{\frac{d^{3/2}\kappa^4}{\epsilon}, d\kappa^2\right\} \times d\log(d)\right) \tag{30}$$

uses of P_y and, with a probability larger than 99%, outputs an $O(\epsilon)$-additive approximation of $\vec{a}$, which is defined as (11).

We present only the concrete procedure of our algorithm here, giving the proof on complexity in [1]. The algorithm consists of the following steps.

1. Estimate the elements of W and $\vec{z}$ in (11) using a quantum algorithm based on QAE. Let the matrix and the vector consisting of the estimation results be $\hat{W}$ and $\vec{\hat{z}}$.
2. Solve $\hat{W}\vec{a} = \vec{\hat{z}}$ by some classical solver of systems of linear equations (for example, row reduction). Let the solution be $\vec{\hat{a}}$. This is an output of our algorithm.

Since the second step is a simple classical calculation, we focus on the step 1. As we explained in Sec. III A, we can obtain an estimation $\hat{w}_{ij}$ of w_{ij} by QAE if we construct the following operators A_{ij}. A_{ij} transforms $|0\rangle$, a state in which all qubits are 0, to a state in the form of

$$\sqrt{w_{ij}}\,|\psi\rangle + \sqrt{1-w_{ij}}\,|\psi_\perp\rangle, \tag{31}$$

where $|\psi\rangle$ and $|\psi_\perp\rangle$ are some orthogonal states. Such an operator is constructed as follows.

(i) Prepare quantum registers $R_1, ..., R_5$, which have enough qubits, and a single qubit register R_6. Set R_1, R_2 and the others to $|i\rangle, |j\rangle$ and $|0\rangle$, respectively.

(ii) Create $\frac{1}{\sqrt{N_D}}\sum_{k=1}^{N_D}|k\rangle$, that is, an equiprobable superposition of $|1\rangle, ..., |N_D\rangle$ on R_3.

(iii) Apply P_x to a block of R_1, R_3 and R_4, which outputs $x_k^{(i)}$ on R_4. Similarly, apply P_x to a block of R_2, R_3 and R_5, which outputs $x_k^{(j)}$ on R_5.

(iv) Using $x_k^{(i)}$ on R_4 and $x_k^{(j)}$ on R_5, transform R_6 from $|0\rangle$ to $\left(\sqrt{1-x_k^{(i)}x_k^{(j)}}\,|0\rangle + \sqrt{x_k^{(i)}x_k^{(j)}}\,|1\rangle\right)$ by some arithmetic circuits and controlled rotations. Then, the resultant state is in the form of (31).

Through the steps (i) to (iv), the state is transformed as follows.

$$\begin{aligned}
&|i\rangle|j\rangle|0\rangle|0\rangle|0\rangle|0\rangle \\
\xrightarrow{\text{(ii)}}\ &|i\rangle|j\rangle\left(\frac{1}{\sqrt{N_D}}\sum_{k=0}^{N_D}|k\rangle\right)|0\rangle|0\rangle|0\rangle \\
\xrightarrow{\text{(iii)}}\ &|i\rangle|j\rangle\left(\frac{1}{\sqrt{N_D}}\sum_{k=0}^{N_D}|k\rangle|x_k^{(i)}\rangle|x_k^{(j)}\rangle\right)|0\rangle \\
\xrightarrow{\text{(iv)}}\ &|i\rangle|j\rangle\left[\frac{1}{\sqrt{N_D}}\sum_{k=0}^{N_D}|k\rangle|x_k^{(i)}\rangle|x_k^{(j)}\rangle\right. \\
&\qquad\left.\otimes\left(\sqrt{1-x_k^{(i)}x_k^{(j)}}\,|0\rangle + \sqrt{x_k^{(i)}x_k^{(j)}}\,|1\rangle\right)\right]
\end{aligned} \tag{32}$$

We can obtain an estimation $\hat{z}_i$ of z_i in the similar way by replacing one of two P_x's with P_y and $x_k^{(j)}$ with y_k.

IV. EXTENTION TO A CLASS OF CONVEX OPTIMIZATION

A. Linear regression as optimization by Newton's method

In this section, we extend our method for linear regression to more general optimization problems. Before this, we firstly present another interpretation of (6), the formula for the solution of linear regression. We regard linear regression as an optimization problem of

$$F(\vec{a}) = \frac{1}{2N_D}\|\vec{y} - X\vec{a}\|^2 = \frac{1}{2N_D}\sum_{k=1}^{N_D}\left(\sum_{i=1}^{d} a_i x_k^{(i)} - y_k\right)^2. \tag{33}$$

This can be rewritten as

$$F(\vec{a}) = \frac{1}{N_D}\sum_{k=1}^{N_D} f(\vec{a}; \vec{x}_k, y_k), \tag{34}$$

where

$$f(\vec{a}; \vec{x}_k, y_k) = \frac{1}{2}\left(\sum_{i=1}^{d} a_i x_k^{(i)} - y_k\right)^2. \tag{35}$$

The first and second derivatives of F are

$$\frac{\partial}{\partial a_i}F(\vec{a}) = \frac{1}{N_D}\sum_{k=1}^{N_D}\frac{\partial}{\partial a_i}f(\vec{a}; \vec{x}_k, y_k) = \frac{1}{N_D}\sum_{k=1}^{N_D} x_k^{(i)}\left(\sum_{j=1}^{d} a_j x_k^{(j)} - y_k\right), \tag{36}$$

$$\frac{\partial^2}{\partial a_i \partial a_j}F(\vec{a}) = \frac{1}{N_D}\sum_{k=1}^{N_D}\frac{\partial^2}{\partial a_i \partial a_j}f(\vec{a}; \vec{x}_k, y_k) = \frac{1}{N_D}\sum_{k=1}^{N_D} x_k^{(i)} x_k^{(j)}, \tag{37}$$

respectively. This means that, $W = \frac{1}{N_D}X^T X$ and $-\vec{z} = -\frac{1}{N_D}X^T\vec{y}$ are the Hessian matrix and the gradient vector of F at $\vec{a} = \vec{0}$, respectively. Besides, the updating formula in Newton's method is

$$\vec{a}_{n+1} = \vec{a}_n - H_F^{-1}(\vec{a_n})\vec{g}_F(\vec{a}_n), \tag{38}$$

where H_F and $\vec{g}_F$ are the Hessian and the gradient of the function F, respectively, and $\vec{a}_n$ is the optimization variable after the n-th update. Then, we can interpret (6) as an one-time update in Newton's method from the initial point $\vec{a}_0 = \vec{0}$. Note that Newton's method gives the exact solution by only one update from any initial point, if the objective function is quadratic.

In summary, we can consider our method as optimization of the objective function (33) by Newton's method, where calculation of the gradient and the Hessian, which is time-consuming in the classical method, are done by the QAE-based method.

B. Extension of our method : the QAE-based Newton's method

On the basis of the above discussion, it is now straightforward to extend our method to a more general class of optimization problems. That is, keeping the updating formula (38), we can perform Newton's method based on the gradient and the Hessian estimated by QAE.

Concretely, we consider an optimization problem in which the objective function can be written as a sum of the values of some function with different inputs:

$$F(\vec{a}) = \frac{1}{N_D}\sum_{k=1}^{N_D} f(\vec{a}, \vec{c}_k). \qquad (39)$$

Here, f is the real-valued twice-differentiable function, which are shared by all the terms. Its inputs are the optimization variables $\vec{a} \in \mathbb{R}^d$ and some parameters $\vec{c}_k$, which are different in each term. Some conditions on F necessary for convergence analysis are given in Sec. IV C. It is obvious that the objective functions (33) fall into (39). For the objective function like (39), the gradient $\vec{g}_F(\vec{a}) = (g_{F,1}(\vec{a}), ..., g_{F,d}(\vec{a}))^T$ and the Hessian $H_F(\vec{a}) = (h_{F,ij}(\vec{a}))_{\substack{1\le i\le d\\ 1\le j\le d}}$ are given as

$$g_{F,i}(\vec{a}) = \frac{\partial}{\partial a_i}F(\vec{a}) = \frac{1}{N_D}\sum_{k=1}^{N_D} f_i(\vec{a}, \vec{c}_k), \qquad (40)$$

$$h_{F,ij}(\vec{a}) = \frac{\partial^2}{\partial a_i \partial a_i}F(\vec{a}) = \frac{1}{N_D}\sum_{k=1}^{N_D} f_{ij}(\vec{a}, \vec{c}_k), \qquad (41)$$

where we simply write $\frac{\partial}{\partial a_i}f$ and $\frac{\partial^2}{\partial a_i \partial a_j}f$ as f_i and f_{ij}, respectively.

Then, we estimate the gradient and the Hessian as follows. We assume the availability of the followings:

- P_c, which outputs $\vec{c}_k$ for given k:

$$P_c : |k\rangle|0\rangle \mapsto |k\rangle|\vec{c}_k\rangle. \qquad (42)$$

  This can be constructed by QRAM.

- P_i for $i = 1, .., d$, which outputs $f_i(\vec{a}, \vec{c}_k)$ for given $\vec{a}$ and $\vec{c}_k$:

$$P_i : |\vec{a}\rangle|\vec{c}_k\rangle|0\rangle \mapsto |\vec{a}\rangle|\vec{c}_k\rangle|f_i(\vec{a}, \vec{c}_k)\rangle. \qquad (43)$$

- P_{ij} for $i, j = 1, ..., d$, which outputs $f_{ij}(\vec{a}, \vec{c}_k)$ for given $\vec{a}$ and $\vec{c}_k$:

$$P_{ij} : |\vec{a}\rangle|\vec{c}_k\rangle|0\rangle \mapsto |\vec{a}\rangle|\vec{c}_k\rangle|f_{ij}(\vec{a}, \vec{c}_k)\rangle. \qquad (44)$$

Then, preparing appropriate registers, we can perform the following computation:

$$\begin{aligned}
&|\vec{a}\rangle|0\rangle|0\rangle|0\rangle|0\rangle \\
\to &\left(\frac{1}{\sqrt{N_D}}\sum_{k=0}^{N_D}|\vec{a}\rangle|k\rangle\right)|0\rangle|0\rangle|0\rangle \\
\to &\left(\frac{1}{\sqrt{N_D}}\sum_{k=0}^{N_D}|\vec{a}\rangle|k\rangle|\vec{c}_k\rangle\right)|0\rangle|0\rangle \\
\to &\left(\frac{1}{\sqrt{N_D}}\sum_{k=0}^{N_D}|\vec{a}\rangle|k\rangle|\vec{c}_k\rangle\left|f_i(\vec{a};\vec{c}_k)\right\rangle\right)|0\rangle \\
\to &\frac{1}{\sqrt{N_D}}\sum_{k=0}^{N_D}|\vec{a}\rangle|k\rangle|\vec{c}_k\rangle\left|f_i(\vec{a};\vec{c}_k)\right\rangle \\
&\otimes\left(\sqrt{1-f_i(\vec{a};\vec{c}_k)}|0\rangle + \sqrt{f_i(\vec{a};\vec{c}_k)}|1\rangle\right), \qquad (45)
\end{aligned}$$

where P_c and P_i are used at the second and third arrows, respectively. We then obtain an estimation of $g_i(\vec{a})$ by estimating the probability that the last qubit takes 1 by QAE. We can also estimate $h_{ij}(\vec{a})$ similarly, replacing P_i with P_{ij} and $f_i(\vec{a}, \vec{c}_k)$ with $f_{ij}(\vec{a}, \vec{c}_k)$. Again, in order to estimate one $g_i(\vec{a})$ or $h_{ij}(\vec{a})$ with ϵ-additive error, the number of calling P_i and P_{ij} is at most $O(1/\epsilon)$, which means the exponential speedup with respect to N_D compared with classical iterative calculation.

Using the gradient and the Hessian estimated as above, we update $\vec{a}_n$ to $\vec{a}_{n+1}$ similarly to (38), that is,

$$\vec{a}_{n+1} = \vec{a}_n - \hat{H}_F^{-1}(\vec{a_n})\vec{\hat{g}}_F(\vec{a}_n), \qquad (46)$$

where $\vec{\hat{g}}_F(\vec{a_n})$ and $\hat{H}_F(\vec{a_n})$ are the estimated gradient and Hessian, respectively. After sufficiently many iterations, we obtain the approximated solution of the optimization. Hereafter, we call this method the *QAE-based Newton's method*.

Note that $\hat{H}_F(\vec{a_n})$ must be invertible so that the update (46) can be defined. Hereafter, we consider the situation where the original Hessian $H_F(\vec{a_n})$ is positive-definite and therefore invertible. In order to keep such a property, we have to obtain $\hat{H}_F(\vec{a_n})$ accurately enough.

C. Convergence analysis of the QAE-based Newton's method

Then, let us estimate the complexity of the QAE-based Newton's method to obtain an approximated solution with ϵ-additive error. For a mathematically rigorous discussion, we first make some assumptions on the objective function $F : \mathbb{R}^d \to \mathbb{R}$.

Assumption 5. *F is twice-differentiable.*

This is reproduced since we assumed that f in (39) is twice-differentiable.

Assumption 6. *F is μ-strongly convex, that is, there exists a positive number μ such that*

$$\forall\vec{a}, \vec{b} \in \mathbb{R}^d, F(\vec{a}) \geq F(\vec{b}) + \vec{g}_F(\vec{b})\cdot(\vec{a}-\vec{b}) + \frac{\mu}{2}||\vec{a}-\vec{b}||^2. \quad (47)$$

This assumption means that the eigenvalues of $H_F(\vec{a})$ are greater than or equal to μ for any $\vec{a} \in \mathbb{R}^d$, that is,

$$\forall \vec{a} \in \mathbb{R}^d, H_F(\vec{a}) \geq \mu I_d, \tag{48}$$

where I_d is the $d \times d$ identity matrix and for $A, B \in \mathbb{R}^d$, $A \geq B$ means that $A - B$ is positive-semidefinite. This immediately leads to

$$\forall a \in \mathbb{R}^d, \|(H_F(\vec{a}))^{-1}\| \leq \frac{1}{\mu}. \tag{49}$$

Assumption 7. *F has an M-Lipschitz Hessian, that is,*

$$\forall \vec{a}, \vec{b} \in \mathbb{R}^d, \|H_F(\vec{a}) - H_F(\vec{b})\| \leq M\|\vec{a} - \vec{b}\|. \tag{50}$$

This assumption leads to the following inequality:

$$\forall \vec{a}, \vec{b} \in \mathbb{R}^d, \|\vec{g}_F(\vec{a}) - \vec{g}_F(\vec{b}) - H_F(\vec{b})(\vec{a} - \vec{b})\| \leq \frac{M}{2}\|\vec{a} - \vec{b}\|^2. \tag{51}$$

Assumptions 5 to 7 are usually made in the discussion on convergence properties of the *ordinary* Newton's method, that is, cases where the gradient and the Hessian can be exactly computed in the classical way (see, for example, [26]). We do not make any additional assumptions for the QAE-based Newton's method.

Next, let us define some quantities. Since QAE introduces errors in the estimated Hessian and gradient, we have to consider Newton's method in which the update difference contains some error. For $\vec{a} \in \mathbb{R}^d$, we define $\vec{\Delta}_F(\vec{a})$ as

$$\vec{\Delta}_F(\vec{a}) := (\hat{H}_F(\vec{a}))^{-1}\vec{\hat{g}}_F(\vec{a}) - (H_F(\vec{a}))^{-1}\vec{g}_F(\vec{a}). \tag{52}$$

Besides, we write the minimum[5] that we search as $\vec{a}^\star := \operatorname*{argmin}_{\vec{a} \in \mathbb{R}^d} F(\vec{a})$, the optimization variable after the n-th update as $\vec{a}_n$ and the difference between $\vec{a}^\star$ and $\vec{a}_n$ as $\delta_n := \|\vec{a}_n - \vec{a}^\star\|$.

Then, we can show the following lemma, which is repeatedly used.

Lemma 2. *Let Assumptions 5 to 7 be satisfied. Then, for any non-negative number ϵ and any $\vec{a} \in \mathbb{R}^d$ such that*

$$\|\vec{\Delta}_F(\vec{a})\| \leq \epsilon, \tag{53}$$

the following inequality holds:

$$\delta' \leq \frac{M}{2\mu}\delta^2 + \epsilon, \tag{54}$$

where $\delta := \|\vec{a} - \vec{a}^\star\|$, $\delta' := \|\vec{a}' - \vec{a}^\star\|$, and $\vec{a}' = \vec{a} - (\hat{H}_F(\vec{a}))^{-1}\vec{\hat{g}}_F(\vec{a})$. Furthermore, when

$$\frac{2M}{\mu}\epsilon < 1, \tag{55}$$

is satisfied, the following hold:

$$\begin{cases} \delta' < \delta; \text{ if } \delta_- < \delta < \delta_+ \\ \delta' \leq \delta_-; \text{ if } \delta \leq \delta_- \end{cases}, \tag{56}$$

where

$$\delta_\pm := \frac{\mu}{M}\left(1 \pm \sqrt{1 - \frac{2M}{\mu}\epsilon}\right). \tag{57}$$

Here, let us comment on what Lemma 2 implies. Equation (54) indicates that, even when the update differences in Newton's method contain errors at most ϵ, the difference δ between the optimization variables $\vec{a}$ and the optimal point $\vec{a}^\star$ quadratically converges like Newton's method with no error, as long as $\epsilon \lesssim \frac{M}{2\mu}\delta^2$. More strictly, while $\delta_- < \delta$, δ decreases at every update, as shown in (56). On the other hand, after δ reaches δ_-, ϵ is not negligible in (54), and therefore δ does not necessarily decreases. Nevertheless, δ does not exceeds δ_-, once it goes below. Since $\delta_- < 2\epsilon$, we can make $\vec{a}$ converge with desired accuracy if we can suppress ϵ.

Using Lemma 2, we obtain the following lemma, which shows how many updates are sufficient for δ to reach 2ϵ in Newton's method with erroneous update differences.

Lemma 3. *Let Assumptions 5 to 7 be satisfied. Suppose that we repeatedly updates $\vec{a} \in \mathbb{R}^d$ by (46) from some initial point $\vec{a}_0$, where we write the result of n-times updates as $\vec{a}_n$ and define $\delta_n := \|\vec{a}_n - \vec{a}^\star\|$ for $n = 0, 1, 2, \ldots$. Then, for any positive number ϵ satisfying (55) and any $\vec{a}_0$ such that*

$$\|\vec{a}_0 - \vec{a}^\star\| < \frac{\mu}{M}, \tag{58}$$

if (53) holds for $\vec{a} = \vec{a}_n, n = 0, 1, 2, \ldots$, there exists a non-negative integer $n_{\rm it}$ such that

$$\delta_n \leq 2\epsilon \tag{59}$$

for any $n \geq n_{\rm it}$, where

$$n_{\rm it} := \max\left\{\left\lceil \log_2\left(\frac{\log\left(\frac{2M\epsilon}{\mu}\right)}{2\log\left(\frac{M\delta_0}{\mu}\right)}\right)\right\rceil + 1, 1\right\}, \tag{60}$$

Next, we consider how accurate we should estimate the gradient and the Hessian in order to suppress the error in the update difference to the order of ϵ. We have the following Lemma 4.

Lemma 4. *Let Assumptions 5 to 7 be satisfied. Besides, suppose that we are given a positive integer n, a positive number ϵ satisfying (55), and $\vec{a} \in \mathbb{R}^d$ satisfying $\delta < \frac{2\mu}{M}$, where $\delta := \|\vec{a} - \vec{a}^\star\|$. Then, in order for $\Delta_F(\vec{a})$ defined as (52) to satisfy $\|\vec{\Delta}_F(\vec{a})\| \leq \epsilon$, it is sufficient that each component of the estimated gradient $\vec{\hat{g}}_F(\vec{a})$ and Hessian $\hat{H}_F(\vec{a})$ has an additive error ϵ'_g and ϵ'_H such that*

$$\epsilon'_g \leq \frac{\mu\epsilon}{2d^{1/2}}. \tag{61}$$

[5] Since we are considering the convex optimization as stated in Assumption 6, there is only one global minimum.

and

$$\epsilon'_H \le \frac{\mu\epsilon}{4\tilde{\delta}d}, \tag{62}$$

where $\tilde{\delta} := \max\{\delta, \delta_-\}$, *respectively.*

This leads to the following Lemma 5.

Lemma 5. *Let Assumptions 5 to 7 be satisfied. Besides, suppose that we are given a positive number ϵ satisfying (55) and an initial point $\vec{a}_0 \in \mathbb{R}^d$ satisfying $\delta_0 := \|\vec{a}_0 - \vec{a}^\star\| < \delta_+$. Furthermore, let $\vec{a}_n \in \mathbb{R}^d$ be a vector given by n-times updates by (46) from the initial point $\vec{a}_0$. Then, $\Delta_F(\vec{a}_n)$ defined as (52) satisfies*

$$\|\Delta_F(\vec{a}_n)\| \le \epsilon \tag{63}$$

for any positive integer n, if each component of $\hat{\vec{g}}_F(\vec{a})$ and $\hat{H}_F(\vec{a})$ has an additive error ϵ'_g and ϵ'_H satisfying (61) and

$$\epsilon'_H \le \frac{\mu\epsilon}{4\tilde{\delta}_0 d}, \tag{64}$$

where $\tilde{\delta}_0 := \max\{\delta_0, \delta_-\}$, *respectively.*

Combining Lemma 3 and Lemma 5, we immediately obtain the following theorem.

Theorem 2. *Let Assumptions 5 to 7 be satisfied. Besides, suppose that we are given a positive number ϵ satisfying (55) and $\vec{a}_0$ satisfying (58). Then, using the QAE-based Newton's method which is based on the updating formula (46), we obtain a 2ϵ-additive approximation of $\vec{a}^\star$ by $n_{\rm it}$-times updates, where $n_{\rm it}$ is given by (60), with a success probability higher than 99%. In the process, the total number of calls $P_i, i = 1, ..., d$ is*

$$N_{\rm 1stDer} = O\left(\frac{d^{3/2}}{\mu\epsilon} n_{\rm it} \log(n_{\rm it} d^2)\right), \tag{65}$$

that for $P_{ij}, i, j = 1, ..., d$ is

$$N_{\rm 2ndDer} = O\left(\frac{\tilde{\delta}_0 d^3}{\mu\epsilon} n_{\rm it} \log(n_{\rm it} d^2)\right), \tag{66}$$

and that for P_c is

$$N_c = N_{\rm 1stDer} + N_{\rm 2ndDer}. \tag{67}$$

V. SUMMARY

In this paper, we proposed a quantum algorithm for linear regression, or, more concretely, estimation of regression coefficients as classical data. Existing algorithms such as [2, 4] create the quantum state encoding coefficients in its amplitude by the HHL algorithm [11] or its modification and then read out the coefficients. On the other hand, in our method, we estimate the elements of $W = \frac{1}{N_D} X^T X$ and $\vec{z} = \frac{1}{N_D} X^T \vec{y}$, where $X = (x_k^{(i)})_{\substack{1 \le i \le d \\ 1 \le k \le N_D}}$ is the design matrix, $\vec{y} = (y_1, ..., y_k)^T$ is the objective variable vector and N_D is the number of the data points, and then find the coefficients by classical computation of $\vec{a} = W^{-1}\vec{z}$. Since, as shown in (8) and (10), the elements have the form of the sum of $x_k^{(i)} x_k^{(j)}$ or $x_k^{(i)} y_k$ over data points, we can estimate them by QAE [15–21], assuming availability of the oracles P_x and P_y which output $x_k^{(i)}$ and y_k, respectively, for specified i and k. The query complexity of our method is given as (29) and (30), which means exponential speedup with respect to N_D compared with the naive classical method, and improvement with respect to the tolerance error ϵ compared with the previous quantum methods such as [2, 4].

Finally, we extended our method to more general optimization problems, that is, convex optimization with an objective function consisting of many similar terms like (39). In light of interpretation of linear regression as Newton's method, we proposed the QAE-based Newton's method, in which the gradient and the Hessian are estimated by QAE. Introducing effects of estimation errors in the ordinary discussion on convergence of Newton's method, we derived the convergence property and the query complexity of the QAE-based Newton's method. Even if there are estimation errors, the method shows well-known quadratic convergence and reaches the solution in a small number of iterations.

Obtaining the improved dependence of complexity on ϵ, we expect that we can apply our method to mid-sized but many-times-repeated regression problems like LSM. Generally speaking, our method can be better than the previous quantum methods and the naive classical method when $d, \kappa \ll \frac{1}{\epsilon} \ll N_D$. On the other hand, since the complexity of our method depends on d and κ more strongly than the previous quantum methods, they will be better when $\frac{1}{\epsilon} \ll d$ or $\frac{1}{\epsilon} \ll \kappa$. Since the naive classical method does not induce any error as quantum ones, it will be better when $N_D \ll \frac{1}{\epsilon}$.

In future work, we will consider implementation of LSM on quantum computers using our method for linear regression, as a concrete and practical use case of quantum computing in financial industry.

ACKNOWLEDGMENT

This work was supported by MEXT Quantum Leap Flagship Program (MEXT Q-LEAP) Grant Number JPMXS0120319794.

[1] K. Kaneko et al., “Linear regression by quantum amplitude estimation and its extension to convex optimization”, Phys. Rev. A 104, 022430 (2021)
[2] N. Wiebe et al., “Quantum Data Fitting”, Phys. Rev. Lett. 109, 050505 (2012)
[3] M. Schuld et al., “Prediction by linear regression on a quantum computer”, Phys. Rev. A 94, 022342 (2016)
[4] G. Wang, “Quantum Algorithm for Linear Regression”, Phys. Rev. A 96, 012335 (2017)
[5] C.-H. Yu et al., “Quantum algorithms for ridge regression”, IEEE Transactions on Knowledge and Data Engineering 29, 37491 (2019)
[6] S. Chakraborty, “The power of block-encoded matrix powers: improved regression techniques via faster Hamiltonian simulation”, Proceedings of the 46th International Colloquium on Automata, Languages, and Programming (ICALP), pp. 33:1-33:14 (2019)
[7] I. Kerenidis and A. Prakash, “Quantum gradient descent for linear systems and least squares”, Phys. Rev. A 101, 022316 (2020)
[8] N.-H. Chia, et al., “Sampling-based sublinear low-rank matrix arithmetic framework for dequantizing quantum machine learning”, Proceedings of the 52nd ACM Symposium on the Theory of Computing (STOC), 387 (2020)
[9] A. Gilyen et al., “An improved quantum-inspired algorithm for linear regression”, arXiv:2009.07268
[10] C. Shao and A. Montanaro, “Faster quantum-inspired algorithms for solving linear systems”, arXiv:2103.10309
[11] A. W. Harrow et al., “Quantum algorithm for solving linear systems of equations”, Phys. Rev. Lett. 103, 150502 (2009)
[12] F. A. Longstaff and E. S. Schwartz, “Valuing American Options by Simulations: A Simple Least-Squares Approach”, Review of Financial Studies, 14, 113 (2001)
[13] J. C. Hull, “Options, Futures, and Other Derivatives”, Prentice Hall (1995)
[14] S. Shreve, “Stochastic Calculus for Finance I & II”, Springer (2004)
[15] G. Brassard et. al., “Quantum amplitude amplification and estimation”, Contemporary Mathematics, 305, 53 (2002)
[16] Y. Suzuki et. al., “Amplitude Estimation without Phase Estimation”, Quantum Information Processing, 19, 75 (2020)
[17] S. Aaronson and P. Rall, “Quantum approximate counting, simplified”, Symposium on Simplicity in Algorithms”, 24-32, SIAM (2020)
[18] D. Grinko et al., “Iterative quantum amplitude estimation”, arXiv:1912.05559
[19] K. Nakaji, “Faster Amplitude Estimation”, arXiv:2003.02417
[20] E. G. Brown et al., “Quantum Amplitude Estimation in the Presence of Noise”, arXiv:2006.14145
[21] T. Tanaka, et al., “Amplitude estimation via maximum likelihood on noisy quantum computer”, arXiv:2006.16223
[22] A. Montanaro, “Quantum speedup of Monte Carlo methods”, Proc. Roy. Soc. Ser. A, 471, 2181 (2015)
[23] G. H. Golub and C. F. Van Loan, “Matrix Computations”, Johns Hopkins University Press (1983)
[24] M. Jerrum et al., “Random generation of combinatorial structures from a uniform distribution”, Theoretical Computer Science, 43, 169 (1986)
[25] V. Giovannetti et al., “Architectures for a quantum random access memory”, Phys. Rev. A78, 052310 (2008)
[26] Y. Nesterov, “Introductory lectures on convex optimization: a basic course (Applied Optimization (87))”, Springer (2004)