Results in Control and Optimization 14 (2024) 100380

Contents lists available at ScienceDirect

Results in Control and Optimization

journal homepage: www.elsevier.com/locate/rico

QUBO formulations for a system of linear equations

Kyungtaek Jun

Quantum research center, QTomo, Busan, Republic of Korea

A R T I C L E I N F O

Keywords:
QUBO
Quantum linear system
Quantum optimization
Quantum annealing

A B S T R A C T

With the advent of quantum computers, many quantum computing algorithms are being developed. Solving linear systems is one of the most fundamental problems in modern science and engineering. The Harrow Hassidim-Lloyd algorithm, a monumental quantum algorithm for solving linear systems on gate model quantum computers, was invented and several advanced variations have been developed. The algorithm was difficult to apply to general linear equations because it required various conditions for the matrix. In this paper, we introduce a new algorithm that can be applied to all linear systems. For a given general square matrix $A \in \mathbb{R}^{n\times n}$ and a vector $\overrightarrow{b} \in \mathbb{R}^n$, we will find quadratic unconstrained binary optimization (QUBO) models for a vector $\overrightarrow{x} \in R^n$ that satisfies $A\overrightarrow{x} = \overrightarrow{b}$. To formulate QUBO models for solving linear systems, we made use of a linear least-square problem with binary representation of the solution. We validated those QUBO models on the d-Wave system and discussed the results. For a simple system, we provide a Python code to calculate the matrix characterizing the relationship between the variables, and to print the test code that can be used directly in the d-Wave system.

Introduction

Quantum computing has opened up a new paradigm for approaching computing problems, providing unparalleled speeds for specific problems which classical (i.e., traditional) computing cannot compete with. A specific subset of quantum computing is quantum annealing, which is aimed at optimization problems. The goal of quantum annealing processors is to find the global energy minimum, obtaining the solution to the optimization problem [1]. One popular model of optimization problems on which quantum annealers are based on is the Ising model [2].

A monumental quantum algorithm for solving linear systems on gate model quantum computers was invented in 2008, and a variety of advanced variations were developed since then [3]. At that time, that algorithm and its variants had many limitations, such as the quantum state of the solution, the need for quantum random access memory (RAM), and the strong requirements of matrix conditions [4]. In particular, the quantum state of the solution limits the use of the algorithm to submodules of the entire system, or reduces the quantum speed improvement in real world implementations as it requires iterative computations to read the quantum state of each base. Since 2019, methods of using linear least squares were proposed to solve linear problems [5–7]. The researchers suggested a quadratic unconstrained binary optimization (QUBO) model of linear equations for m by n linear systems. When solving a linear problem using the proposed QUBO model, the number of floating-point operations per second (flops) for several classic methods such as a product of an orthogonal matrix and an upper triangular matrix (QR) decomposition and singular value decomposition (SVD) [8] were compared. There has been remarkable research [9–15] since this method thanks to the development and advancement of

E-mail address: ktfriends@gmail.com.

https://doi.org/10.1016/j.rico.2024.100380
Received 24 July 2023; Received in revised form 13 December 2023; Accepted 17 January 2024
Available online 5 February 2024

d-Wave quantum annealers.

Harrow, Hassidim, and Lloyd (HHL) introduced an algorithm that can compute sparse and well-conditioned linear systems in $O(\text{polylog}(n))$ using quantum states [3]. Since then, algorithms that improve computation time, preconditioning, and polylogarithmic over the HHL algorithm have been introduced [16–18]. Quantum machine learning algorithms have been developed based on quantum linear system algorithms. While they potentially have a few caveats [19–22], quantum linear system has inspired several works in the emerging research area of quantum machine learning [23–32]. In this paper, we present a quantum optimization model that computes linear systems in a new way. This method's strength lies in the fact that it can be applied to calculate a linear system under very lenient or relaxed conditions. We propose two representative QUBO models for $n \times n$ linear systems. Additionally, we show the overall CPU calculations when performing the new QUBO models. Classical algorithms such as QR decomposition, lower–upper (LU) decomposition, and SVD require $O(n^3)$ flops, but the number of flops required to compute the QUBO model is $O(n^2)$. Another advantage of the new algorithm is that parallel computation is possible when calculating the QUBO model. The new QUBO model can efficiently reduce the number of flops up to $O(n)$, and up to $O(\log_2 n)$. These results will be of great use with significant benefits over other models when solving linear equations for large n. We also provide a Python code to create QUBO models that can be used in d-Wave quantum annealers. The number of qubits that can be used in quantum annealers is approximately 5760 in d-Wave Advantage, so the size of the linear problem that can be solved using our QUBO model is not large. However, with the continual development of quantum annealers, the method we propose will have advantages in the future. It is also expected that other issues could be approached in a similar way to ours.

Method

Background

The Ising model is a mathematical model for ferromagnetism in statistical mechanics. The energy Hamiltonian (the cost function) is formulated as follows:

$$H(\vec{\sigma}) = -\sum_{i=1}^{N} h_i\sigma_i - \sum_{i<j}^{N} J_{i,j}\sigma_i\sigma_j \tag{1}$$

where $\vec{\sigma} = (\sigma_1, \cdots, \sigma_N)^T$ and $\sigma_i \in \{+1, -1\}$.

Quadratic unconstrained binary optimization (QUBO) is a combinatorial optimization problem in computer science. In this problem, a cost function f is defined on an $n-$ dimensional binary vector space $\mathbb{B}^n$ onto $\mathbb{R}$.

$$f(\vec{q}) = \vec{q}^T Q \vec{q} \tag{2}$$

where Q is an upper diagonal matrix, $\vec{q} = (q_1, \cdots, q_N)^T$, and q_i is a binary element of $\vec{q}$. In this paper, the matrix Q is referred to as the QUBO matrix. The problem is to find $\vec{q}^*$ which minimizes the cost function f among vectors $\vec{q}$. Since we have $q_i^2 = q_i$, the cost function is reformulated as follows:

$$f(\vec{x}) = \sum_{i=1}^{N} Q_{i,i}x_i + \sum_{i<j}^{N} Q_{i,j}x_ix_j \tag{3}$$

In Q, the diagonal terms $Q_{i,i}$ and the off-diagonal terms $Q_{i,j}$ represent the linear terms and the quadratic terms, respectively. The unknowns of the Ising model σ and the unknowns of the QUBO model q have the linear relation

$$\sigma \rightarrow 2q - 1 \text{ or } q \rightarrow \frac{1}{2}(\sigma + 1) \tag{4}$$

Therefore, we can choose the appropriate format depending on the binary unknown.

Given a matrix $A \in R^{n\times n}$, a column vector of variables $\vec{x} \in R^n$, and a column vector $\vec{b} \in R^n$, we find $\vec{x}$ that satisfy $A\vec{x} = \vec{b}$. The linear least squares problem is to find the $\vec{x}$ that minimizes $\| A\vec{x} - \vec{b} \|$. In other words, it can be described as follows:

$$\underset{\vec{x}}{\arg\min} \| A\vec{x} - \vec{b} \| \tag{5}$$

To solve Eq. (5) let us begin by writing out $\| A\vec{x} - \vec{b} \|$:

$$A\vec{x} - \vec{b} = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} - \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix} \tag{6}$$

Taking the squared 2-norm of the resultant vector of Eq. (6), we get the following:

$$\| A\vec{x} - \vec{b}\|_2^2 = \vec{x}^T A^T A\vec{x} - 2\vec{b}^T A\vec{x} + \vec{b}^T\vec{b} \tag{7}$$

$$= \sum_{k=1}^{n}\left\{\left(\sum_{i=1}^{n} a_{k,i}x_i\right)^2 - 2b_k\sum_{i=1}^{n} a_{k,i}x_i + b_k^2\right\} \tag{8}$$

$$= \sum_{k=1}^{n}\left\{\sum_{i=1}^{n}\left(a_{k,i}x_i\right)^2 + 2\sum_{i<j} a_{k,i}a_{k,j}x_ix_j - 2b_k\sum_{i=1}^{n} a_{k,i}x_i + b_k^2\right\} \tag{9}$$

QUBO model 1

If we were solving binary least squares then each x_i would be represented by the combination of qubits $q_{i,j} \in \{0,1\}$. In the work by O'Malley and Vesselinov [33], the radix 2 representation of positive real value x_i is given by

$$x_i \approx \sum_{l=-m}^{m} 2^l q_{i,l} \tag{10}$$

where positive integer l is the number of digits of x_i and negative l is the number of digits of fractional terms. Now, we can represent real value x_i is below

$$x_i \approx \sum_{l=-m}^{m} 2^l q_{i,l}^{+} - \sum_{l=-m}^{m} 2^l q_{i,l}^{-} \tag{11}$$

To represent both positive and negative numbers, $q_{i,l}^{+}$ and $q_{i,l}^{-}$ are involved. Note that this representation can have the same value with different binary combinations.

To drive the QUBO model, we insert Eq. (11) into Eq. (9). We obtain the two summation terms of the first term in Eq. (9) as below:

$$\sum_{k=1}^{n}\sum_{i=1}^{n}\left(a_{k,i}x_i\right)^2 \approx \sum_{k=1}^{n}\sum_{i=1}^{n} a_{k,i}^2\left(\sum_{l=-m}^{m} 2^l q_{i,l}^{+} - \sum_{l=-m}^{m} 2^l q_{i,l}^{-}\right)^2 \tag{12}$$

$$= \sum_{k=1}^{n}\sum_{i=1}^{n} a_{k,i}^2\left(\left(\sum_{l=-m}^{m} 2^l q_{i,l}^{+}\right)^2 + \left(\sum_{l=-m}^{m} 2^l q_{i,l}^{-}\right)^2\right) \tag{13}$$

$$= \sum_{k=1}^{n}\sum_{i=1}^{n} a_{k,i}^2\left(\sum_{l=-m}^{m} 2^{2l}\left(\left(q_{i,l}^{+}\right)^2 + \left(q_{i,l}^{-}\right)^2\right) + \sum_{l_1<l_2} 2^{l_1+l_2+1}\left(q_{i,l_1}^{+}q_{i,l_2}^{+} + q_{i,l_1}^{-}q_{i,l_2}^{-}\right)\right) \tag{14}$$

$$= \sum_{k=1}^{n}\sum_{i=1}^{n}\sum_{l=-m}^{m} a_{k,i}^2 2^{2l}\left(q_{i,l}^{+} + q_{i,l}^{-}\right) + \sum_{k=1}^{n}\sum_{i=1}^{n}\sum_{l_1<l_2} a_{k,i}^2 2^{l_1+l_2+1}\left(q_{i,l_1}^{+}q_{i,l_2}^{+} + q_{i,l_1}^{-}q_{i,l_2}^{-}\right) \tag{15}$$

Since x_i is a positive, zero, or negative number, we can constrain $q_{i,l_1}^{+} \times q_{i,l_2}^{-} = 0$ for the same i. We can get Eq. (13) from Eq. (12). Additionally, since $q^2 = q$, Eq. (15) can be finally obtained. Here, the front summation represents a linear term, and the second summation represents a quadratic term.

The QUBO form of the second term in Eq. (9) can be obtained as below:

$$\sum_{k=1}^{n}\sum_{i<j} 2a_{k,i}a_{k,j}x_ix_j \approx \sum_{k=1}^{n}\sum_{i<j} 2a_{k,i}a_{k,j}\left(\sum_{l=-m}^{m} 2^l q_{i,l}^{+} - \sum_{l=-m}^{m} 2^l q_{i,l}^{-}\right)\left(\sum_{l=-m}^{m} 2^l q_{j,l}^{+} - \sum_{l=-m}^{m} 2^l q_{j,l}^{-}\right) \tag{16}$$

$$= \sum_{k=1}^{n}\sum_{i<j}\sum_{l_1=-m}^{m}\sum_{l_2=-m}^{m} 2^{l_1+l_2+1}a_{k,i}a_{k,j}\left(q_{i,l_1}^{+}q_{j,l_2}^{+} + q_{i,l_1}^{-}q_{j,l_2}^{-} - q_{i,l_1}^{+}q_{j,l_2}^{-} - q_{i,l_1}^{-}q_{j,l_2}^{+}\right) \tag{17}$$

When we calculate the QUBO term, we use an upper triangular matrix. Therefore$q_{i,l}^{+}$, $q_{i,l}^{-}$, $q_{j,l}^{+}$ and $q_{j,l}^{-}$ have separate indices, and we can derive Eq. (17) from Eq. (16). This equation is part of the quadratic terms in our QUBO model.

The QUBO form of the third term in Eq. (9) can be obtained as below:

$$\sum_{k=1}^{n}\sum_{i=1}^{n}\left(-2a_{k,i}b_kx_i\right) \approx \sum_{k=1}^{n}\sum_{i=1}^{n}\sum_{l=-m}^{m} 2^{l+1}a_{k,i}b_k\left(q_{i,l}^{-} - q_{i,l}^{+}\right) \tag{18}$$

The above equation is part of the linear terms in our QUBO model.

The last term in Eq. (9) represents the minimum value when the vector $\vec{x}$ satisfies the linear equation in QUBO form. Therefore, our first QUBO model for linear systems is the summation of Eq. (15), Eq. (17), and Eq. (18). The pseudocode for QUBO model 1 is shown in

Algorithm 1.

QUBO model 2

To reduce the number of qubits to be used in Eq. (11), the new approximation of x_i was introduced as follows [5]:

$$x_i \approx -2^{m+1}q_i^- + \sum_{l=-m}^{m} 2^l q_{i,l}^+ \tag{19}$$

Since the coefficient of q_i^- represents the amount of translation of the range of the summation part in Eq. (19), any coefficient of q_i^- that can represent the range of x_i we want can be used. We select the coefficient of q_i^-, which represents the largest range that x_i can have.

To drive the QUBO model, we insert Eq. (19) into Eq. (9). We get another QUBO form of the first term in Eq. (9) as below:

$$\sum_{k=1}^{n}\sum_{i=1}^{n}\left(a_{k,i}x_i\right)^2 \approx \sum_{k=1}^{n}\sum_{i=1}^{n}\left\{a_{k,i}\left(-2^{m+1}q_i^- + \sum_{l=-m}^{m}2^l q_{i,l}^+\right)\right\}^2 \tag{20}$$

$$= \sum_{k=1}^{n}\sum_{i=1}^{n}a_{k,i}^2\left\{\left(-2^{m+1}q_i^-\right)^2 + \sum_{l=-m}^{m}\left(-2^{l+m+2}q_i^- q_{i,l}^+\right) + \left(\sum_{l=-m}^{m}2^l q_{i,l}^+\right)^2\right\} \tag{21}$$

$$= \sum_{k=1}^{n}\sum_{i=1}^{n}a_{k,i}^2\left\{2^{2m+2}\left(q_i^-\right)^2 + \sum_{l=-m}^{m}\left(-2^{l+m+2}q_i^- q_{i,l}^+\right) + \sum_{l=-m}^{m}2^{2l}\left(q_{i,l}^+\right)^2 + \sum_{l_1<l_2}2^{l_1+l_2+1}q_{i,l_1}^+ q_{i,l_2}^+\right\} \tag{22}$$

$$= \sum_{k=1}^{n}\sum_{i=1}^{n}a_{k,i}^2\left(2^{2m+2}q_i^- + \sum_{l=-m}^{m}2^{2l}q_{i,l}^+\right) + \sum_{k=1}^{n}\sum_{i=1}^{n}a_{k,i}^2\left(\sum_{l_1<l_2}2^{l_1+l_2+1}q_{i,l_1}^+ q_{i,l_2}^+ - \sum_{l=-m}^{m}2^{l+m+2}q_i^- q_{i,l}^+\right) \tag{23}$$

Since $q^2 = q$, Eq. (23) can finally be obtained. The front summation represents linear terms, and the second summation represents quadratic terms. The QUBO form of the second term in Eq. (9) can be obtained as below:

$$\sum_{k=1}^{n}\sum_{i<j}2a_{k,i}a_{k,j}x_i x_j \approx \sum_{k=1}^{n}\sum_{i<j}2a_{k,i}a_{k,j}\left(-2^{m+1}q_i^- + \sum_{l=-m}^{m}2^l q_{i,l}^+\right)\left(-2^{m+1}q_j^- + \sum_{l=-m}^{m}2^l q_{j,l}^+\right) \tag{24}$$

$$\sum_{k=1}^{n}\sum_{i<j}a_{k,i}a_{k,j}\left\{2^{2m+3}q_i^- q_j^- - \sum_{l=-m}^{m}2^{l+m+2}\left(q_i^- q_{j,l}^+ + q_j^- q_{i,l}^+\right) + 2\left(\sum_{l=-m}^{m}2^l q_{i,l}^+\right)\left(\sum_{l=-m}^{m}2^l q_{j,l}^+\right)\right\} \tag{25}$$

Algorithm 1
Calculating the QUBO matrix for QUBO model 1.

```
Data input $A$, $\overrightarrow{b}$, and qubit numbers $2m$ for each element of $\overrightarrow{x}$
▷Linear Terms in Eqs. (15) and (18)
for $k = 1 : n$ do
    for $i = 1 : n$ do
        for $l = 1 : m$ do
            $Q(2m(i-1)+l, 2m(i-1)+l) \leftarrow 2^{2(l-1)}(A(k,i))^2 - 2^l A(k,i)b(k)$
            $Q(2m(i-1)+m+l, 2m(i-1)+m+l) \leftarrow 2^{2(l-1)}(A(k,i))^2 - 2^l A(k,i)b(k)$
▷Quadratic Terms in Eq. (15)
for $k = 1 : n$ do
    for $i = 1 : n$ do
        for $l_1 = 1 : m-1$ do
            for $l_2 = l_1+1 : m$ do
                $Q(2m(i-1)+l_1, 2m(i-1)+l_2) \leftarrow 2^{l_1+l_2-1}(A(k,i))^2$
                $Q(2m(i-1)+m+l_1, 2m(i-1)+m+l_2) \leftarrow 2^{l_1+l_2-1}(A(k,i))^2$
▷Quadratic Terms in Eq. (17)
for $k = 1 : n$ do
    for $i = 1 : n-1$ do
        for $j = i+1 : n$ do
            for $l_1 = 1 : m$ do
                for $l_2 = 1 : m$ do
                    $Q(2m(i-1)+l_1, 2m(j-1)+l_2) \leftarrow 2^{l_1+l_2-1}A(k,i)A(k,j)$
                    $Q(2m(i-1)+m+l_1, 2m(j-1)+m+l_2) \leftarrow 2^{l_1+l_2-1}A(k,i)A(k,j)$
                    $Q(2m(i-1)+m+l_1, 2m(j-1)+l_2) \leftarrow 2^{l_1+l_2-1}A(k,i)A(k,j)$
                    $Q(2m(i-1)+l_1, 2m(j-1)+m+l_2) \leftarrow 2^{l_1+l_2-1}A(k,i)A(k,j)$
```

$$= \sum_{k=1}^{n}\sum_{i<j} a_{k,i}a_{k,j}\left\{ 2^{2m+3}q_i^- q_j^- - \sum_{l=-m}^{m} 2^{l+m+2}\left(q_i^- q_{j,l}^+ + q_j^- q_{i,l}^+\right) + \sum_{l_1=-m}^{m}\sum_{l_2=-m}^{m} 2^{l_1+l_2+1} q_{i,l_1}^+ q_{j,l_2}^+ \right\} \tag{26}$$

Eq. (26) can be derived in a similar way to Eq. (17). This equation is part of the quadratic terms in our second QUBO model.

The QUBO form of the third term in Eq. (9) can be obtained as below:

$$\sum_{k=1}^{n}\sum_{i=1}^{n}(-2a_{k,i}b_k x_i) \approx \sum_{k=1}^{n}\sum_{i=1}^{n} 2^{m+2} a_{k,i}b_k q_i^- - \sum_{k=1}^{n}\sum_{i=1}^{n}\sum_{l=-m}^{m} 2^{l+1} a_{k,i} b_k q_{i,l}^+ \tag{27}$$

The last term in Eq. (9) represents the minimum value when the vector $\vec{x}$ satisfies the linear equation in QUBO form.

Therefore, our second QUBO model for linear systems is the summation of Eq. (23), Eq. (26), and Eq. (27). The pseudocode for QUBO model 2 can be derived similarly to Algorithm 1.

Result

We use QPU solvers [34] to test the QUBO model for the linear systems. d-Wave 2000Q and d-Wave's Advantage systems have similar rates of finding the lowest energy for the problems. If the number of logical qubits used in a linear system exceeds 30, the probability of finding the lowest energy is significantly reduced. So, we use less than 30 qubits in our example. For each example, we use less than 30 qubits and do 10,000 anneals to get the result. Algorithm 1 is a pseudo-code that creates a QUBO matrix for QUBO model 1. We denote x_i as the sum of qubits in Eq. (11). Each x_i consists of m qubits that can represent each positive or negative number. However, the qubits used to make the QUBO matrix must be expressed in the form used in Eq. (2).

In this paper, q^+ and q^- notations were used to simply display the calculation formula. Since quantum computers calculate based on the QUBO matrix, it is convenient to use consecutive numbers for the qubits used. Let the vector $\vec{x}$ have two elements, x_1 and x_2, with each x_i having 4 qubits. We can then represent $\vec{x}$ as a consecutive sequence of qubits as follows:

$$\vec{x} \approx c(q_1 + 2q_2 - q_3 - 2q_4,\ q_5 + 2q_6 - q_7 - 2q_8) \tag{28}$$

where c is a constant that varies depending on the range that x_i can represent. For example, if x_i changes from -3 to 3 with an integer solution, c becomes 1. If c is 0.5, the range of the x_i is halved and the accuracy is doubled. Algorithm 1 is a case where c is 1, and to change c, multiply each $a_{k,i}$ by c. The matrix for QUBO model 2 can be calculated in a similar way.

Examples

Let's consider the following example:

$$\begin{pmatrix} 3 & -7 \\ 2 & 8 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} -16 \\ 2 \end{pmatrix} \tag{29}$$

where the range of integer x_i is $-3 \le x_i \le 3$. The first step to calculating Eq. (29) on a quantum computer is to convert it into energy function using Eq. (9). The energy function for $\vec{x}$ is shown in Fig. 1.

To find minimum energy, we represent x_i with a combination of qubits as follows:

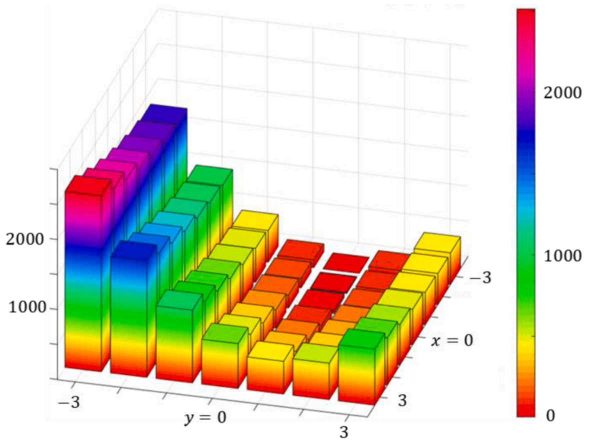

Fig. 1. Energy graph of $\| A\vec{x} - \vec{b} \|_2^2$.

$$\overrightarrow{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} q_1 + 2q_2 - q_3 - 2q_4 \\ q_5 + 2q_6 - q_7 - 2q_8 \end{pmatrix} \tag{30}$$

where q_i is binary and the range of integer x_i is $-3 \le x_i \le 3$. Since the given example uses a total of 8 qubits, by applying Algorithm 1, the 8×8 QUBO matrix Q_1 for QUBO model 1 can be obtained as follows:

$$Q_1 = \begin{pmatrix} 101 & 52 & 0 & 0 & -10 & -20 & 10 & 20 \\ & 228 & 0 & 0 & -20 & -40 & 20 & 40 \\ & & -75 & 52 & 10 & 20 & -10 & -20 \\ & & & -124 & 20 & 40 & -20 & -40 \\ & & & & -143 & 452 & 0 & 0 \\ & & & & & -60 & 0 & 0 \\ & & & & & & 369 & 452 \\ & & & & & & & 964 \end{pmatrix} \tag{31}$$

In a similar way, we represent x_i with a different combination of qubits as follows:

$$\overrightarrow{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} q_1 + 2q_2 - 4q_3 \\ q_4 + 2q_5 - 4q_6 \end{pmatrix} \tag{32}$$

The 6×6 QUBO matrix $Q_1^{'}$ for QUBO model 2 is as follows:

$$Q_1^{'} = \begin{pmatrix} 101 & 52 & -104 & -10 & -20 & 40 \\ & 228 & -208 & -20 & -40 & 80 \\ & & -144 & 40 & 80 & -160 \\ & & & -143 & 452 & -904 \\ & & & & -60 & -1808 \\ & & & & & 2832 \end{pmatrix} \tag{33}$$

The minimum energy required for two QUBO models to find the vector $\overrightarrow{x}$ is $b_1^2 + b_2^2 = -260$. We tested the two QUBO models of Eqs. (31) and (33) on the d-Wave system with 10,000 anneals. There are 5994 and 1610 occurrences of the lowest energy, out of the 10,000 anneals.

The second example is as follows:

$$\begin{pmatrix} 3 & -7 \\ 2 & 8 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 172 \\ -50 \end{pmatrix} \tag{34}$$

where the range of integer x_i is $-31 \le x_i \le 31$. Then we represent x_i with a combination of qubits as the following:

$$\overrightarrow{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} \sum_{k=1}^{5} 2^{k-1}(q_k - q_{k+5}) \\ \sum_{k=1}^{5} 2^{k-1}(q_{k+10} - q_{k+15}) \end{pmatrix} \tag{35}$$

where q_i is binary and the range of integer x_i is $-31 \le x_i \le 31$. Since the given example uses a total of 20 qubits, by applying Algorithm 1, the 20×20 QUBO matrix Q_2 for QUBO model 1 can be obtained as follows:

$$Q_2 = \begin{pmatrix} -819 & 52 & 104 & 208 & 416 & & 10 & 20 & 40 & 80 & 160 \\ & -1612 & 208 & 416 & 832 & & 20 & 40 & 80 & 160 & 320 \\ & & -3120 & 832 & 1664 & \cdots & 40 & 80 & 160 & 320 & 640 \\ & & & -5824 & 3328 & & 80 & 160 & 320 & 640 & 1280 \\ & & & & -9984 & & 160 & 320 & 640 & 1280 & 2560 \\ & & & & & \ddots & & & \vdots & & \\ & & & & & & -3095 & 452 & 904 & 1808 & 3616 \\ & & & & & & & -5964 & 1808 & 3616 & 7232 \\ & & & & & & & & -11024 & 7232 & 14464 \\ & & & & & & & & & -18432 & 28928 \\ & & & & & & & & & & -22400 \end{pmatrix} \tag{36}$$

In a similar way, we represent x_i with a different combination of qubits as follows:

$$\overrightarrow{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} \sum_{k=1}^{5} 2^{k-1} q_k - 2^5 q_6 \\ \sum_{k=1}^{5} 2^{k-1} q_{k+6} - 2^5 q_{12} \end{pmatrix} \tag{37}$$

The 12×12 QUBO matrix $Q_2^{'}$ for QUBO model 2 is as follows:

$$Q_2^{'} = \begin{pmatrix} -819 & 52 & 104 & 208 & 416 & -832 & -10 & -20 & -40 & -80 & -160 & 320 \\ & -1612 & 208 & 416 & 832 & -1664 & -20 & -40 & -80 & -160 & -320 & 640 \\ & & -3120 & 832 & 1664 & -3328 & -40 & -80 & -160 & -320 & -640 & 1280 \\ & & & -5824 & 3328 & -6656 & -80 & -160 & -320 & -640 & -1280 & 2560 \\ & & & & -9984 & -13312 & -160 & -320 & -640 & -1280 & -2560 & 5120 \\ & & & & & 39936 & 320 & 640 & 1280 & 2560 & 5120 & -10240 \\ & & & & & & 3321 & 452 & 904 & 1808 & 3616 & -7232 \\ & & & & & & & 6868 & 1808 & 3616 & 7232 & -14464 \\ & & & & & & & & 14640 & 7232 & 14464 & -28928 \\ & & & & & & & & & 32896 & 28928 & -57856 \\ & & & & & & & & & & 80256 & -115712 \\ & & & & & & & & & & & 13056 \end{pmatrix} \tag{38}$$

The minimum energy required for two QUBO models to find the vector $\overrightarrow{x}$ is $b_1^2 + b_2^2 = -32{,}084$. We tested the two QUBO models of Eqs. (36) and (38) on the d-Wave system with 10,000 anneals. There are 8 and 26 occurrences of the lowest energy, out of the 10,000 anneals.

The third example is as follows:

$$\begin{pmatrix} 3 & -7 & 2 & 8 \\ 2 & 8 & -4 & -6 \\ -2 & 5 & 1 & 1 \\ -5 & -3 & 6 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 28 \\ -16 \\ -8 \\ 1 \end{pmatrix} \tag{39}$$

The first qubit representation of x_i is as follows:

$$\overrightarrow{x} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} q_1 + 2q_2 - q_3 - 2q_4 \\ q_5 + 2q_6 - q_7 - 2q_8 \\ q_9 + 2q_{10} - q_{11} - 2q_{12} \\ q_{13} + 2q_{14} - q_{15} - 2q_{16} \end{pmatrix} \tag{40}$$

where q_i is binary and the range of integer x_i is $-3 \le x_i \le 3$. Since the given example uses a total of 16 qubits, by applying Algorithm 1, the 16×16 QUBO matrix Q_3 for QUBO model 1 can be obtained as follows:

$$Q_3 = \begin{pmatrix} -84 & 168 & 0 & 0 & 0 & & 136 & 10 & 20 & -10 & -20 \\ & -84 & 0 & 0 & 0 & & 272 & 20 & 40 & -20 & -40 \\ & & 168 & 168 & 0 & \cdots & -136 & -10 & -20 & 10 & 20 \\ & & & 420 & 0 & & -272 & -20 & -40 & 20 & 40 \\ & & & & 881 & & 236 & -204 & -408 & 204 & 408 \\ & & & & & \ddots & & & \vdots & & \\ & & & & & & 700 & -188 & -376 & 188 & 376 \\ & & & & & & & -524 & 408 & 0 & 0 \\ & & & & & & & & -844 & 0 & 0 \\ & & & & & & & & & 728 & 408 \\ & & & & & & & & & & 1660 \end{pmatrix} \tag{41}$$

In a similar way, the second qubit representation of x_i is as follows:

$$\overrightarrow{x} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} q_1 + 2q_2 - 4q_3 \\ q_4 + 2q_5 - 4q_6 \\ q_7 + 2q_8 - 4q_9 \\ q_{10} + 2q_{11} - 4q_{12} \end{pmatrix} \tag{42}$$

The second 12×12 QUBO matrix Q_3 for QUBO model 2 is as follows:

$$Q_3^{'} = \begin{pmatrix} -84 & 168 & -336 & 0 & 0 & 0 & -68 & -136 & 272 & 10 & 20 & -40 \\ & -84 & -672 & 0 & 0 & 0 & -136 & -272 & 544 & 20 & 40 & -80 \\ & & 1176 & 0 & 0 & 0 & 272 & 544 & -1088 & -40 & -40 & 160 \\ & & & 881 & 588 & -1176 & -118 & -236 & 472 & -204 & -408 & 816 \\ & & & & 2056 & -2352 & -236 & -472 & 944 & -408 & -816 & 1632 \\ & & & & & -584 & 472 & 944 & -1888 & 816 & 1632 & -3264 \\ & & & & & & -179 & 228 & -456 & 94 & 188 & -376 \\ & & & & & & & -244 & -912 & 188 & 376 & -752 \\ & & & & & & & & 1856 & -376 & -752 & 1504 \\ & & & & & & & & & -524 & 408 & -816 \\ & & & & & & & & & & -844 & -1632 \\ & & & & & & & & & & & 4136 \end{pmatrix} \tag{43}$$

The minimum energy required for two QUBO models to find the vector $\overrightarrow{x}$ is $b_1^2 + b_2^2 = -1105$. We tested the two QUBO models of Eqs. (41) and (43) on the d-Wave system with 10,000 anneals. There are 160 and 37 occurrences of the lowest energy, out of the 10,000 anneals.

When using Zephyr topology of d-Wave quantum processors, approximately $\frac{n(n+4)}{4}$ physical qubits are required to use n logical qubits with all connectivity [35,36]. Therefore, if n logical qubits are used to calculate a linear system, the number of possible cases increases to approximately $2^{\frac{n(n+4)}{4}}$. To find the global minimum energy, all physical qubits must satisfy an optimal state during the annealing process. This probability is similar to our experimental results, and the probability of finding a solution decreases as the number of logical qubits increases.

Major cost for calculating the QUBO matrix

The most computationally intensive part of our first QUBO form is Eq. (17), one of the quadratic terms. We want to discuss Eq. (17) below. By computing Eq. (17), we calculate variables k, i and j. Variable k can vary from 1 to n, and i and j represent the (i,j) components of the upper triangular matrices without diagonal components, so the total amount of calculation can be expressed as follows:

$$\textit{Major computing cost for } \sum_{i<j} a_{k,i}a_{k,j} = 2n - 3 \tag{44}$$

Eq. (27) can be calculated by adding $n - 2$ times to the variable i. After that, it is calculated by multiplying the variable j by $n - 1$ times. $l_1 + l_2$ can have a value from $-2m$ to $2m$. Additionally, the product of $a_{k,i}a_{k,j}$ and $2^{l_1+l_2+1}$ can be calculated by addition or multiplication. Therefore, the total amount of calculation is as follows:

$$\textit{Major computing cost for } (i,j,l_1,l_2) = 2n + 8m - 1 \tag{45}$$

$$\textit{Major computing cost for } (i,j,k,l_1,l_2) = n(2n + 8m - 1) \tag{46}$$

Eqs. (15) and (18) can be calculated in a similar way. To make the QUBO model solve the linear equation, the total number of flops is $O(n^2)$. Conventional QR decomposition, LU decomposition, and SVD have $O(n^3)$ flops. Compared to these, the computation required to create the QUBO model is $O(n^2)$, so it looks like quantum computing will be more advanced when the size of the matrix expands. The biggest advantage of the QUBO model is that it allows parallel computation. In Eq. (17), the most computationally intensive variable to parallelize is i. The total parallel computation on i is about $\log_2 n$. If we perform n parallelization on variable k and then $\frac{n}{2}$ parallelization on variable i again, we can calculate the total calculation in about $O(\log_2 n)$. Parallel computation using quantum computers for existing large matrices will have a huge effect that reduces the total calculation time.

Discussion

QUBO model 1 has a major disadvantage as it uses approximately twice as many qubits as QUBO model 2. However, we mainly discussed the results of QUBO model 1 to explain the important results for the QUBO constraint. To solve the system of linear equations, each element x_i of $\overrightarrow{x}$ will be either positive or negative numbers, or by zero. When Eq. (11) is used for the element x_i, the result can be expressed as combinations of elements in $q^+$and q^-. However, a solution x_i can be made with either $\sum 2^l q_{i,l}^+$ or $-\sum 2^l q_{i,l}^-$. Because of this property, we can assume this problem to be a constrained problem for certain quadratic terms. To make certain quadratic terms equal to 0, it is common to add the coefficients of the terms to a large value. However, in this paper, we propose a new method of adding different values to each term so that it is set to 0 when the coefficient of each term is negative. We set each coefficient of $q_{i,l_1}^+ q_{i,l_2}^-$ to zero for each x_i. We call this method the coefficient matching algorithm in constraint problems. In the d-Wave systems, this is the most efficient way to find the minimum. You can check the actual results of various results using this method through the GitHub address listed in the Appendix. Finally, when an element of a matrix has a complex number, x_i must also be expressed as a binary complex number, and the number of logical qubits is doubled.

To calculate the QUBO model, d-Wave provides both the QPU solver and the hybrid solver [37]. Our experimental results show that d-Wave 2000Q and Advantage convey similar performance. d-Wave 2000Q and Advantage can use up to 64 and 177 logical qubits, respectively [38,39]. Those values decide the size of the QUBO matrix. This means that as the size of matrix A increases, the number of qubits each x_i can use decreases. d-Wave Advantage can use 177 all-connected logical qubits. However, this does not mean that a 177 $\times$ 177 QUBO matrix can always be embedded in the Advantage system. This makes finding the lowest energy much more difficult as the size of the QUBO matrix increases. We believe that in order to solve this problem efficiently, an algorithm that changes the part of the QUBO matrix to zero or a method for calculating a linear system using a small number of qubits is needed. Another alternative is to use a hybrid solver instead of the QPU solver. The hybrid solver can use 1000,000 fully connected qubits. We tested a 100 $\times$ 100 QUBO matrix with this solver. We didn't do many tests because each test takes about 3 s. For several tests, this solver always found the lowest energy. Due to the limitations of QAOA in gate model quantum computers, we used up to 20 logical qubits [40]. To calculate the linear system, QUBO models 1 and 2 were applied using the simulator provided by the Qiskit library [41]. We applied Eq. (4) to each QUBO model to convert it to the Ising model for the minimization model, and then multiplied each variable by -1 to use the Ising model for the maximization model. The Ising model for QUBO model 1 used 8, 12, 16, and 24 logical qubits, respectively. For model 2, QAOA was applied in the simulator using 6, 9, 12, and 16 logical qubits. For all cases, the quantum annealer and Qiskit simulator were able to find the global optimal energy. Additionally, we checked the number of cases for finding the global minimum energy for 1000 anneals in the quantum annealer, d-Wave 2000Q. The result is shown in Table 1.

Our algorithm was able to find the global minimum energy even when including constraints [42]. We attach the Python code we used as a supplementary file. Readers can easily check additional results by converting the number of qubits used for the accuracy of the matrix and variable x. As a result of applying QAOA to each case, the maximum value was accurately found to two decimal places, and all solutions were found [43]. The d-Wave quantum annealer also has similar accuracy to the Qiskit simulator and calculated the global minimum energy of the Ising model for each QUBO model [44]. When using 20 qubits, the Qiskit simulator had a circuit depth of 38, but the IBM quantum computer had about 490 [45]. It's evident that optimization calculations yield promising results when each QUBO model incorporates a linear system solution for up to 20 logical qubits. Our quantum linear system algorithm is versatile, with potential applications in CT image reconstruction. By employing a hybrid solver with 10,000 logical qubits, we can assess the results and error rates [46].

A new quantum optimization algorithm for linear systems can find a solution through global energy optimization when the vector $\vec{x}$ of the quantum state can represent the solution. If the condition number of the matrix is low, the energy for axe takes a paraboloid form as shown in Fig. 2A. On the other hand, if the condition number of the matrix is high, it takes the form of a parabola with a nearly similar cross section, as shown in Fig. 2C. Fig. 2B and D show contour lines for Fig. 2A and C, respectively. The weakness of the proposed algorithm is that when the condition number is large and the vector $\vec{x}$ cannot represent the solution, it is difficult to find the approximate solution we want. As shown in Fig. 2D, let us assume that the vector x of the quantum state can represent the yellow and green points simultaneously and that the actual solution is around the origin. Our QUBO model points to the yellow dot closer to the center line as having the lowest energy. This is because there is very little energy difference with respect to the center line passing through the solution. If one or more elements of a vector $\vec{x}$ have irrational numbers and the condition number is very high, it is difficult for our proposed QUBO model to have a good approximation. Furthermore, ongoing research is exploring scenarios where linear systems have either an infinite number of solutions or none at all.

Conclusion

This research provides a method for formulating the QUBO model that makes it computationally possible on quantum computers and quantum annealers for linear systems. Traditional computation models for linear systems require $O(n^3)$ computations, but the QUBO model can efficiently reduce the computation difficulty down to $O(n)$ through parallel computing. This approach enables the efficient computation of large matrices that were previously challenging to compute. In this paper, two QUBO models for linear systems are presented. Both QUBO models were computed using the QPU solver and hybrid solver provided by d-Wave. Additionally, results for each QUBO model are demonstrated using the simulator from Qiskit. The new method is particularly effective when $\vec{x}$ represents a superposition state with multiple values containing the solution. However, a major drawback of the new algorithm occurs when $\vec{x}$ does not contain the solution. Theoretically, the position of $\vec{x}$ indicates the global minimum energy, and efficiency is excellent when the eccentricity of the energy contour approaches zero. On the other hand, when the eccentricity approaches 1, the global minimum energy and approximate position for the solution may differ. This problem can be diminished when x_i in the superimposed state represents a narrow range, but a more efficient method is needed to allow a broader range. The new algorithm applies to all linear systems; however, there needs to be a more efficient method for finding solutions when there are infinitely many solutions for the linear system.

Efficiently solving linear systems is crucial in both engineering and scientific fields. Recently, a method for reconstructing CT

Table 1
Number of occurrences of finding the global minimum energy for 1000 anneals in d-Wave 2000Q.

QUBO model 1 # Qubits	# Occurrences	QUBO model 2 # Qubits	# Occurrences
8	828	6	172
12	12	9	10
16	72	12	5
24	1	16	0

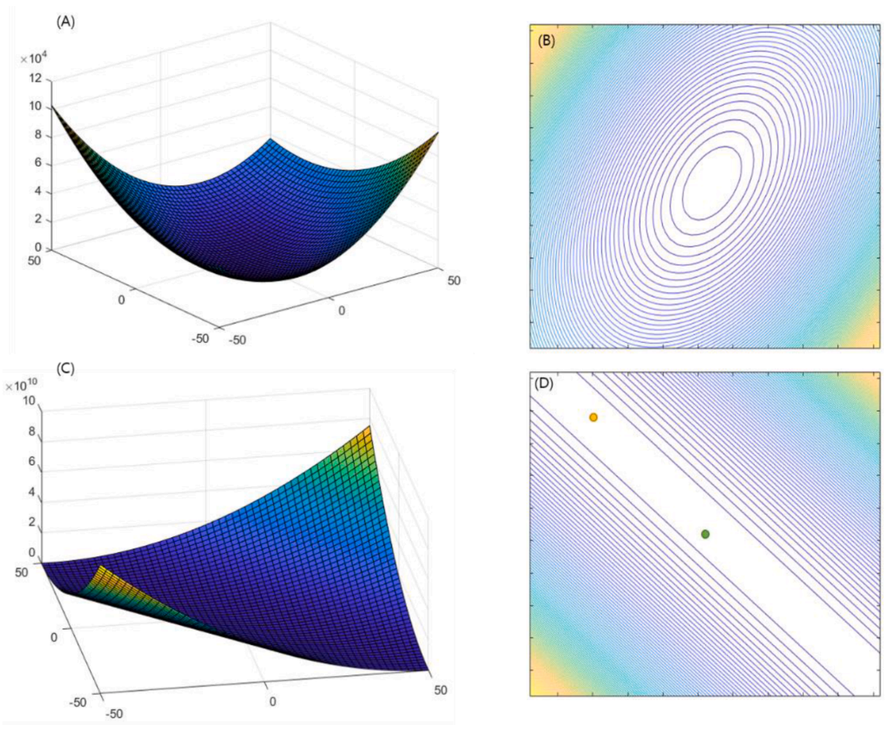

Fig. 2. Energy graph of $\| A\overrightarrow{y} - \overrightarrow{b} \|_2^2$ for a two-dimensional vector $\overrightarrow{y}$. The solution vector for the linear system $A\overrightarrow{x} = \overrightarrow{b}$ lies near the center. (A) Energy graph for a matrix with low condition number 1.77. (B) Contour lines for the matrix in (A). (C) Energy graph for a matrix with high condition number 8.00×10^6. (D) Contour lines of the matrix in (C).

images using the QUBO model for linear systems has been proposed [46]. This approach proves to be both fast and accurate as it leverages a novel algorithm. The QUBO model used in CT image reconstruction can find solutions even with an insufficient number of equations. Furthermore, it can find solutions that lead to an overall minimum energy, making it a robust approach to reducing the noise in the data even when $\overrightarrow{b}$ contains the noise. The progress of quantum computing, suggests a significant advancement in the diagnostic use of medical imaging. We hope that the utilization of quantum optimization algorithms for linear systems will continue to find its use and application in various fields in the future.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Kyungtaek Jun reports statistical analysis was provided by QTomo. Kyungtaek Jun reports a relationship with QTomo that includes: non-financial support.

Data availability

Data sharing not applicable to this article as no datasets were generated or analyzed during the current study.

Acknowledgments

This research was supported by the quantum computing technology development program of the National Research Foundation of Korea (NRF) funded by the Korean government (Ministry of Science and ICT (MSIT)) (no. 2020M3H3A111036513). The authors would like to thank Seungjoon Noh for their assistance. We would like to thank Editor and Reviewers for taking the time and effort necessary to review the manuscript. We sincerely appreciate all valuable comments and suggestions, which helped us to improve the quality of the manuscript.

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.rico.2024.100380.

Appendix A

We introduce two Python code generators for linear systems available on the d-Wave 2000Q and Advantage. The code we provide is not optimized for the amount of computations. Our code is an early version with an emphasis on readability. d-Wave 2000Q has 2048 qubits and can make all connected 64 qubits. Therefore, if you use d-Wave 2000Q, the maximum size of the QUBO matrix is 64. d-Wave Advantage has more than 5000 qubits and 35,000 couples with 15 couplers per qubit. This system can solve QUBO matrices up to 177 × 177. The code below can find the integer solution. You can change the range of x_i by adjusting the number of qubits used in each x_i.

Additional results: "https://github.com/ktfriends/Numerical_Quantum_Computing"

A.1. QUBO model 1

Our generator code can be used for matrices of any size and can be used in d-Wave by adjusting the size of the matrix and number of qubits. The example uses QUBO model 1 for a quadratic square matrix.

See "Appendix A. Supplementary data: MMC S1"

If you want to use the tabu simulator or hybrid solver of the d-Wave system to solve the linear system, you can use the following samplers:

i) tabu simulator

from tabu import TabuSampler sampleset = TabuSample().sample_qubo()

ii) D-Wave's hybrid solver

from dwave.system import LeapHybridSampler sampler = LeapHybridSampler()

The Lowest Energy from d-Wave Quantum Annealer (3000 anneals):

Sample(sample={'q01': 1, 'q02': 1, 'q03': 0, 'q04': 1, 'q05': 1, 'q06': 0, 'q07': 0, 'q08': 0, 'q09': 0, 'q10': 0, 'q11': 0, 'q12': 0, 'q13': 0, 'q14': 0, 'q15': 0, 'q16': 1, 'q17': 0, 'q18': 1, 'q19': 1, 'q20': 0}, energy=−32,084.0, num_occurrences=12, chain_break_fraction=0.0)

A.2. QUBO model 2

Our generator code can be used for matrices of any size and can be used in DWave by adjusting the size of the matrix and number of qubits. The example uses QUBO model 2 for a quadratic square matrix.

See "Appendix A. Supplementary data: MMC S2"

The Lowest Energy from d-Wave Quantum Annealer (3000 anneals):

Sample(sample={'q01': 1, 'q02': 1, 'q03': 0, 'q04': 1, 'q05': 1, 'q06': 0, 'q07': 1, 'q08': 1, 'q09': 0, 'q10': 0, 'q11': 1, 'q12': 1}, energy=−32,084.0, num_occurrences=6, chain_break_fraction=0.0)

References

[1] Boixo S, Rønnow TF, Isakov SV. Evidence for quantum annealing with more than one hundred qubits. Nat Phys 2014;10:218–24. https://doi.org/10.1038/nphys2900.
[2] Kadowaki T, Nishimori H. Quantum annealing in the transverse ising model. Phys Rev E 1998;58:5355–63. https://doi.org/10.1103/PhysRevE.58.5355.
[3] Harrow AW, Hassidim A, Lloyd S. Quantum algorithm for linear systems of equations. Phys Rev Lett 2009;103(15):150502.
[4] Aaronson S. Read the fine print. Nat Phys 2015;11(4):291–3.
[5] Borle A, Lomonaco SJ. Analyzing the quantum annealing approach for solving linear least squares problems. WALCOM: Algorith Comput 2019;11355:289–301. https://doi.org/10.1007/978-3-030-10564-8.
[6] Chang TH, Lux TC, Tipirneni SS. Least-squares solutions to polynomial systems of equations with quantum annealing. Quant Inf Process 2019;18(12):1–17.
[7] Date P, Potok T. Adiabatic quantum linear regression. Sci rep 2021;11(1):21905.
[8] Lee DQ. Numerically efficient methods for solving least squares problems. 2012.
[9] Adachi SH, Henderson MP. Application of quantum annealing to training of deep neural networks. 2015.
[10] B O, R B, Perdomo-Ortiz Aea. Bayesian network structure learning using quantum annealing. Eur Phys J Spec Top 2015;224:163–88.
[11] Daniel O, Vesselinov VV, Alexandrov BS, Alexandrov LB. Nonnegative/binary matrix factorization with a D-Wave quantum annealer. PLoS One 2018;13(12):1–12. https://doi.org/10.1371/journal.pone.0206653.
[12] Date P, Arthur D, Pusey-Nazzaro L. Qubo formulations for training machine learning models. Sci Rep 2021;11(1):1–10.
[13] Arthur D, et al. Balanced k-means clustering on an adiabatic quantum computer. Quant Inf Process 2021;20(9):1–30.
[14] Pusey-Nazzaro, L. Adiabatic quantum optimization fails to solve the knapsack problem. 2020; *arXiv preprint* arXiv:2008.07456.
[15] Jun K, et al. Solving linear systems by quadratic unconstrained binary optimization on D-Wave quantum annealing device. Quantum information science, sensing, and computation XIII, 11726. SPIE; 2021. p. 41–7.

[16] Childs AM, Kothari R, Somma RD. Quantum algorithm for systems of linear equations with exponentially improved dependence on precision. SIAM J Comput 2017;46(6):1920–50.
[17] Ambainis A. Variable time amplitude amplification and quantum algorithms for linear algebra problems. STACS'12 (29th symposium on theoretical aspects of computer science), 14. LIPIcs; 2012. p. 636–47.
[18] Clader BD, Jacobs BC, Sprouse CR. Preconditioned quantum linear system algorithm. Phys Rev Lett 2013;110(25):250504.
[19] Rebentrost P, Schuld M, Wossnig L, Petruccione F, Lloyd S. Quantum gradient descent and Newton's method for constrained polynomial optimization. New J Phys 2019;21(7):073023.
[20] Ciliberto C, Herbster M, Ialongo AD, Pontil M, Rocchetto A, Severini S, Wossnig L. Quantum machine learning: a classical perspective. Proceed Roy Soc A: Math, Phys Eng Sci 2018;474(2209):20170551.
[21] Childs AM. Equation solving by simulation. Nat Phys 2009;5(12):861.
[22] Aaronson S. Read the fine print. Nat Phys 2015;11(4):291–3.
[23] Rebentrost P, Mohseni M, Lloyd S. Quantum support vector machine for big data classification. Phys Rev Lett 2014;113(13):130503.
[24] Schuld M, Sinayskiy I, Petruccione F. Prediction by linear regression on a quantum computer. Phys Rev A 2016;94(2):022342.
[25] Wiebe N, Braun D, Lloyd S. Quantum algorithm for data fitting. Phys Rev Lett 2012;109(5):050505.
[26] Wiebe, N., & Granade, C. Can small quantum systems learn?. 2015. *arXiv preprint* arXiv:1512.03145.
[27] Wiebe N, Granade C, Cory DG. Quantum bootstrapping via compressed quantum Hamiltonian learning. New J Phys 2015;17(2):022005.
[28] Wiebe N, Kapoor A, Svore KM. Quantum nearest-neighbor algorithms for machine learning. Quant inform Comput 2015;15(3–4):318–58.
[29] Wiebe, N., Kapoor, A., & Svore, K. M. Quantum deep learning. 2014; *arXiv preprint* arXiv:1412.3489.
[30] Kapoor A, Wiebe N, Svore K. Quantum perceptron models. Advances in neural information processing systems, 29; 2016.
[31] Kartsaklis, D., Lewis, M., & Rimell, L. Proceedings of the 2016 Workshop on Semantic Spaces at the Intersection of NLP, Physics and Cognitive Science. 2016; *arXiv preprint* arXiv:1608.01018.
[32] Zhao Z, Fitzsimons JK, Fitzsimons JF. Quantum-assisted Gaussian process regression. Phys Rev A 2019;99(5):052331.
[33] O'Malley D, Vesselinov VV. Toq. jl: a high-level programming language for D-Wave machines based on julia. In: 2016 IEEE High Performance Extreme Computing Conference (HPEC). IEEE; 2016. p. 1–7.
[34] Willsch D, Willsch M, Gonzalez Calaza CD, Jin F, De Raedt H, Svensson M, Michielsen K. Benchmarking Advantage and D-Wave 2000Q quantum annealers with exact cover problems. Quant Inf Process 2022;21(4):141.
[35] Jun K, Lee H. HUBO and QUBO models for prime factorization. Sci Rep 2023;13:10080.
[36] Jiang S, et al. Quantum annealing for prime factorization. Sci Rep 2018;8:17667.
[37] McGeoch C, Farre P, Bernoudy W. D-Wave hybrid solver service+ advantage: technology update. Tech. Rep. 2020.
[38] Choi V. Minor-embedding in adiabatic quantum computation: II. minoruniversal graph design. Quant Inf Process 2011;10(3):343–53.
[39] McGeoch C, Farré P. The advantage system: performance update. D-Wave technical report series. 2021. Retrieved from: https://www.dwavesys.com/media/qdmlgsu1/14.
[40] Basso J, Gamarnik D, Mei S, Zhou L. Performance and limitations of the QAOA at constant levels on large sparse hypergraphs and spin glass models. In: 2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS). IEEE; 2022. p. 335–43.
[41] Farhi, E., Goldstone, J., & Gutmann, S. A quantum approximate optimization algorithm. 2014; arXiv preprint arXiv:1411.4028.
[42] Lee H, Noh S, Jun K. Effective QUBO modeling for solving linear systems on D-Wave quantum annealing device. Quantum information science, sensing, and computation xiv, 12093. SPIE; 2022. p. 138–45.
[43] Breugem WPA. second-order accurate immersed boundary method for fully resolved simulations of particle-laden flows. J Comput Phys 2012;231(13):4469–98.
[44] Zaborniak T, de Sousa R. Benchmarking Hamiltonian noise in the D-Wave quantum annealer. IEEE Trans. Quant Eng. 2021;2:1–6.
[45] Park SW, Lee H, Kim BC, Woo Y, Jun K. Circuit depth reduction algorithm for qubo and ising models in gate-model quantum computers. In: 2021 International Conference on Information and Communication Technology Convergence (ICTC). IEEE; 2021. p. 1357–62.
[46] Jun K. A highly accurate quantum optimization algorithm for CT image reconstruction based on sinogram patterns. Sci Rep 2023;13(1):14407.