On the Influence of Numerical Representations in Quantum-Annealing-Based Linear Regression

Pedro Faria Albuquerque and Leonardo Tomazeli Duarte

***Abstract*— Quantum annealers are a class of quantum technologies that have attracted increasing attention due to their natural suitability for solving combinatorial optimization problems. Recent works reformulated the linear regression as a Quadratic Unconstrained Binary Optimization (QUBO) problem, enabling its implementation on quantum annealers and offering potential speed-ups for large datasets. However, this QUBO-based formulation requires the definition of a precision vector to represent the real-valued regression coefficients as integers variables, which introduces limitations related to the quantization accuracy. In this work, we investigate several numerical representations strategies for defining the precision vector. By performing a set of numerical experiments with synthetic datasets, we analyze the performance of these strategies in different configurations. Our results that the strategy usually known as conventional binary representation provides the best trade-off between performance and resource efficiency for quantum-assisted linear regression.**

***Keywords*— Quantum Annealing, QUBO, Linear Regression, Quantum Machine Learning.**

I. Introduction

Quantum computing is a fast-paced emergent technology that is expected to provide speed-ups for solving problems intractable to classical computers. This computational advantage stems from the unique properties of quantum bits (qubits) [1], which leverage quantum mechanical phenomena such as superposition and entanglement to process information more efficiently, with applications in different fields. For instance, in recent years, efforts have been made to adapt machine learning algorithms to quantum computing platforms, with the hope of achieving improvements in computational efficiency [2, 3].

Among the different quantum computing paradigms, quantum annealing is particularly well-suited for solving optimization problems. Since many machine learning tasks involve minimizing a well-defined cost or loss function, quantum annealers are naturally appealing platforms for implementing such models [4]. A key step in adapting machine learning algorithms to quantum annealers is formulating the optimization task as a Quadratic Unconstrained Binary Optimization (QUBO) problem [5]. In fact, quantum annealers can be viewed as specialized computers designed to solve QUBO problems. Several studies have already adapted machine learning tasks to the QUBO framework. In this work, we focus on the reformulation of linear regression as a QUBO problem, as proposed in [6].

Linear regression is a classical tool for predictive modeling and is particularly valuable in signal processing and related fields. It is fundamentally an optimization problem, and as shown in [6], it can be implemented on current quantum annealers through a QUBO-based reformulation. That study demonstrated a computational speed-up of up to 2.8 times on large datasets. However, one of the main challenges in implementing linear regression on quantum annealers lies in encoding of real-valued coefficients using binary variables.

To address the representation of continuous variables within a binary optimization framework, [6] introduced a *precision vector* to encode real-valued regression coefficients. Despite its usefulness, this approach is constrained by the limited number of qubits available on current quantum hardware, making the choice of the precision vector, and, consequently, of the numerical representation, particularly important. Although some studies have investigated numerical representation in QUBO formulations, such as in the context of solving systems of linear equations [7], research on this issue in the context of regression remains incipient.

In this work, we investigate five strategies for defining the precision vector. These strategies include the conventional binary representation (based on powers of two), linearly spaced values, constant values, and values sampled from uniform and normal distributions. We analyze the impact of each encoding strategy on the performance of the regression model under different experimental configurations. The remainder of the paper is organized as follows. In Section II, we introduce some basic concepts of quantum computing. Section III presents the formulation of linear regression as a QUBO problem. In Section IV, we describe the methodology used to compare the different encoding strategies. Sections V and VI present the numerical experiments and our conclusions, respectively.

II. Theoretical Background

We present a very brief introduction to some central aspects of quantum computing, with a particular focus on quantum annealing, which is central to the present work.

A. Basics of Quantum Computing

In quantum computers the basic unit of computation is the qubit. It can be mathematically represented as a two-dimensional complex unit-norm vector on a Hilbert Space:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle,$$

Pedro Faria Albuquerque, Universidade Federal de Minas Gerais (UFMG), e-mail: pfalbuquerque@ufmg.br; Leonardo Tomazeli Duarte, Faculdade de Ciências Aplicada (FCA), Universidade Estadual de Campinas (Unicamp), e-mail: ltduarte@unicamp.br. This work was supported by FAPESP and the Academia Brasileira de Ciências (ABC) as part of the Aristides Pacheco Leão Program, Bolsa de Estímulo a Vocações Científicas (Process 2024/18269-0). This work was also supported by FAPESP under the Grant #2020/09838-0 (BI0S - Brazilian Institute of Data Science). L. T. Duarte thanks the National Council for Scientific and Technological Development (CNPq, Brazil) for the financial support.

where the pair $|0\rangle$ and $|1\rangle$ represents a orthonormal basis for the state space. Upon measurement, the qubits can be found in one of the two states, where $|\alpha|^2$ and $|\beta|^2$ represent probabilities associated with the state measured. Although after measurement the qubit is found in either 0 or 1, before measurement it is in a linear combination, or *superposition*, of states, one of the properties that provide advantages in quantum algorithms.

In quantum computing, there are two main paradigms of computation: the gate-based approach and adiabatic computation. Whereas the former uses unitary logic gates to evolve the state of the qubit to a desired final state – much like classical computers – the latter prepares the qubits in the initial ground state of a Hamiltonian (energy function) and lets an adiabatic time evolution drive the system to a final Hamiltonian, whose ground state encodes the solution to the problem. As discussed in the sequel, such a process can be associated with the resolution of an optimization problem.

B. Quantum Annealers

Quantum adiabatic computers do not yet exist; however quantum annealing is a closely related approach that already has experimental implementations. The difference is that in this approach the conditions for adiabatic evolution are not perfectly met. Nevertheless, it serves as a heuristic algorithm which can find the solution or grounds states close to the solution of the proposed problem [4]. Regarding currently available technology, quantum annealing is being developed by several companies, including the Canadian company D-Wave. Their experimental apparatus lets users define an optimization problem in the QUBO or Ising formulation, making it particularly well suited for binary combinatorial optimization problems [8].

In the annealing paradigm, the quantum computing process consists of initializing the quantum hardware in a known low-energy state of a Hamiltonian, denoted by $\mathcal{H}_i$. The goal is then to adiabatically, or slowly enough, change this initial Hamiltonian into a final Hamiltonian, $\mathcal{H}_f$, which encodes the solution to the desired problem. This process can be represented by the following system evolution process:

$$\mathcal{H}(t) = A(t)\mathcal{H}_i + B(t)\mathcal{H}_f, \tag{1}$$

where $t \in [0, T_{\text{max}}]$, and the boundary conditions are given by:

$$A(0) = 1, \quad A(T_{\text{max}}) = 0, \quad B(0) = 0, \quad B(T_{\text{max}}) = 1.$$

Essentially, during the annealing process, the contribution of the initial Hamiltonian decreases while that of the final Hamiltonian increases. The key aspect is that, if this transition is sufficiently adiabatic, the system will remain in the ground state, which means that can reach the ground state of $\mathcal{H}_f$. In other words, the process will (potentially) find the minimum of function $\mathcal{H}_f$. Another important point is that the final Hamiltonian $\mathcal{H}_f$ can be engineered, in quantum hardware, to represent an Ising model. This, in turn, allows one to encode a quadratic cost function defined over binary variables, without constraints. Therefore, this annealing scheme can serve to solve an optimization problem that belongs to the class of QUBO problems.

In short, the process of programming current quantum annealers consist of [9]:

1) Define a QUBO formulation for the optimization problem and convert it into a graph representation;
2) Map the graph onto the topology of the physical quantum hardware;
3) Set the parameters of the final Hamiltonian, which encodes the solution to the problem. This involves defining the interactions and coupling strengths between qubits;
4) Initialize the qubits in an equal superposition of states, a low-energy configuration that is easy to prepare;
5) Perform quantum annealing, where the system evolves from the initial Hamiltonian toward the final Hamiltonian according to an adiabatic schedule, represented by (1);
6) At the end of the annealing process, if successful, the system reaches a low-energy eigenstate of the final Hamiltonian, ideally corresponding to the global minimum of the energy function. A solution is then read out from the final qubit configuration as a binary string;
7) Since quantum annealing is a heuristic process, the procedure is repeated multiple times to obtain a distribution of candidate solutions.

D-Wave's largest quantum computer to date is the D-Wave Advantage quantum annealer, which has 5,640 qubits and 40,484 couplers. Despite this, the number of variables that can be used to solve practical problems remains limited. Based on the functioning of these machines and the constraints of current hardware, the problems that can be embedded into these technologies are still small compared to those we would like to solve. Therefore, a key challenge is to develop methods to formulate such problems in the most efficient way possible to take full advantage of the currently available architectures. This encompasses the challenges of recasting continuous optimization problems as integer ones.

III. Problem Formulation: quantum-assisted linear regression

A. Linear Regression and Notation

Let us begin by introducing the notation adopted here for formulating the linear regression problem:

- N: number of samples;
- d: number of features (excluding bias term);
- $D = d + 1$: number of features including the bias term;
- $\mathbf{y} \in \mathbb{R}^N$: vector of output values;
- $\mathbf{X} \in \mathbb{R}^{N \times D}$: data matrix including input values and bias term;
- $\mathbf{w} \in \mathbb{R}^D$: vector containing the regression coefficients.

Given a training set of N input-output pairs $(\mathbf{x}_i, y_i)$, the objective of linear regression is to define a linear model, parametrized by $\mathbf{w}$, such that $\hat{\mathbf{y}} = \mathbf{X}\mathbf{w}$ approximates the outputs $\mathbf{y}$. This task is formulated by means of the following optimization problem:

$$\min_{\mathbf{w} \in \mathbb{R}^D} J(\mathbf{w}) = \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2. \tag{2}$$

This cost function is a convex quadratic function and the solution to (2) admits an closed-form expression, given by:

$$\mathbf{w}^* = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}. \tag{3}$$

B. Quantum-Assisted Linear Regression

To implement linear regression in a quantum annealer, it becomes necessary to recast the optimization problem (2) as a QUBO problem. In mathematical terms, a QUBO problem can be written as:

$$\min_{\mathbf{z}\in\mathbb{B}^M} y = \mathbf{z}^T\mathbf{A}\mathbf{z} + \mathbf{z}^T\mathbf{b} \tag{4}$$

where $\mathbf{z} \in \mathbb{B}^M$ is a binary vector, and $\mathbf{A} \in \mathbb{R}^{M\times M}$ and $\mathbf{b} \in \mathbb{R}^M$ are parameters of the QUBO problem.

As discussed in [6], a simple expansion of the cost function (2) leads to:

$$\min_{\mathbf{w}\in\mathbb{R}^D} J(\mathbf{w}) = \mathbf{w}^T\mathbf{X}^T\mathbf{X}\mathbf{w} - 2\mathbf{w}^T\mathbf{X}^T\mathbf{y} + \mathbf{y}^T\mathbf{y}. \tag{5}$$

Note that $\mathbf{y}^T\mathbf{y}$ does not depend on $\mathbf{w}$ and, therefore, can be ignored in the optimization problem.

A last step to reformulate (2) as a QUBO problem is to define a numerical representation of each regression coefficient w_i as a binary vector. This can be done by introducing a precision vector $\mathbf{p} \in \mathbb{R}^K$, which maps binary variables to real values. For each regression coefficient w_i, we associate K binary variables $\hat{w}_{ik}$, so that:

$$w_i = \sum_{k=1}^{K} p_k \hat{w}_{ik}, \quad \forall i = 1, \ldots, D. \tag{6}$$

It is possible to write this representation in a matrix notation, so that:

$$\mathbf{w} = \mathscr{P}\hat{\mathbf{w}}, \tag{7}$$

where $\mathscr{P} = \mathbf{I}_D \otimes \mathbf{p}^T$, so $\mathscr{P} \in \mathbb{R}^{D\times KD}$.

The QUBO formulation of the linear regression problem [6] can be obtained by substituting (7) into (5), which leads to the following cost function:

$$\min_{\hat{\mathbf{w}}\in\mathbb{B}^{KD}} J(\hat{\mathbf{w}}) = \hat{\mathbf{w}}^T\mathscr{P}^T\mathbf{X}^T\mathbf{X}\mathscr{P}\hat{\mathbf{w}} - 2\hat{\mathbf{w}}^T\mathscr{P}^T\mathbf{X}^T\mathbf{y}. \tag{8}$$

This matches the standard QUBO formulation of (4), with:

$$\mathbf{z} = \hat{\mathbf{w}}, \quad \mathbf{A} = \mathscr{P}^T\mathbf{X}^T\mathbf{X}\mathscr{P}, \quad \mathbf{b} = -2\mathscr{P}^T\mathbf{X}^T\mathbf{y}.$$

It is worth mentioning that each coefficient w_i takes 2^K distinct values when $p_k \geq 0$, which is assumed in this work. This discretization procedure restricts the precision of $\mathbf{w}$ for small K, potentially degrading the regression quality. Therefore, careful selection of the precision vector $\mathbf{p}$ is critical and will be analyzed in depth in the next section.

IV. Methodology

A. Strategies for Defining the Precision Vector

The precision vector $\mathbf{p}$ has dimension K, so it is necessary KD qubits to represent all the features. Given the limited number of qubits currently available in quantum computers, a trade-off must be made between the number of bits used for precision and the number of features. In [6], the authors chose to use $K = 2$ to test how the linear regression problem scales. The usefulness of quantum annealing for linear regression, however, depends on the future availability of a larger number of qubits and, consequently, the ability to accurately estimate real-valued coefficients. Currently, we are still limited to a few qubits. Therefore, it is important to investigate the best ways to formulate the precision vector, both to improve performance on current problems and to prepare for scenarios where more qubits become available. To explore this issue, we test different numerical representations of the precision vector for various values of K.

More precisely, in our experiments, we consider the following strategies to define the precision vector [7]:

- Conventional binary representation :
$$p_k = 2^{-k}, \quad k = 0, 1, \ldots, K.$$
- Representation with constant values:
$$p_k = \frac{1}{K}.$$
- Random sampling from uniform distribution:
$$p_k \sim \mathcal{U}(0, 1).$$
- Random sampling from normal distribution:
$$p_k \sim \mathcal{N}(1/2, 0).$$

B. Experimental description

The primary objective of our experiments is to evaluate the impact of different precision vectors on the performance of quantum-assisted linear regression (QALR). To achieve this, we generate synthetic datasets and assess how various representations affect the quantization process and the resulting regression error. More precisely, we build synthetic datasets $\mathbf{X}$ consisting of 10^3 samples, with the number of features set to $D = 2$. Although QALR is expected to show advantages primarily for larger feature spaces [6], we empirically observe that the typical behavior of the performance is not significantly affected by an increase in the number of feature dimensions. Therefore, this limited range for D suffices to evaluate how different lengths of the precision vector impact performance.

Feature values were generated from samples drawn from a standard normal distribution. In contrast to [6], which uses fixed regression coefficients that are always exactly representable by the precision vector, we randomly generate real-valued regression weights w uniformly in the interval $[0, 1]$. The target vector $\mathbf{y}$ is then computed as

$$\mathbf{y} = \mathbf{X}\mathbf{w} + \epsilon\mathbf{n},$$

where $n_i \sim \mathcal{N}(0, 1)$ represents Gaussian noise. We conduct a series of experiments for different combinations of the following parameters: number of features D, precision vector length K, and noise level ϵ. For each configuration, the experiment was executed 50 times, and we report the average results.

The evaluation of each experiment is based on the mean squared error (MSE) between the observed and predicted outputs, that is and the signal-to-noise ratio (SNR). The MSE is defined as:

$$\mathrm{MSE} = \frac{1}{N}\|\mathbf{X}\mathbf{w}_Q - \mathbf{y}\|_2^2, \tag{9}$$

where $\mathbf{w}_Q$ is the quantized solution obtained by solving the QUBO problem expressed in (8). This metric provides a standard measure of the regression accuracy. To assess the effect of quantization noise, we compute the signal-to-noise ratio (SNR) given by:

$$\mathrm{SNR} = 10\log_{10}\left(\frac{\sigma_Y^2}{\sigma_e^2}\right), \tag{10}$$

where σ_Y^2 is the variance of $\mathbf{y}$, and σ_e^2 is the variance of the quantization error, calculated as $\mathbf{X}\mathbf{w}_Q - \mathbf{X}\mathbf{w}$. Both metrics are computed for each trial, and their average is used to assess the performance of each parameter configuration.

We have not yet been able to run these tests on D-Wave's quantum annealers. Therefore, we consider exact search optimization in all cases we analyzed. Naturally, results on quantum hardware may differ, as quantum annealing does not always guarantee finding the global minimum of the Hamiltonian due to hardware noise and annealing schedule limitations. Some solutions may therefore not be found. Nevertheless, the optimization problem is the same in both classical and quantum settings, they are both solving a QUBO problem.

V. Results

To evaluate the different precision vectors strategies in the context of QUBO-assisted linear regression, we conducted a series of experiments using synthetic data with $D = 2$ features and $N = 1000$ samples. The tests were performed by varying precision vector lengths $K = 2$ up to $K = 11$, and two levels of Gaussian noise given by $\varepsilon = 0.5$ and $\varepsilon = 1.0$.

We provide the results obtained for classical linear regression, which solves equation 3 (we shall refer to this solution as classical solver). Unlike the QUBO-based solver, the classical method does not require quantization and serves as a benchmark: the SNR from the classical model establishes an upper bound on quantization quality, while its MSE represents a lower bound on regression error. It is important to note that this work does not aim to demonstrate the superiority of quantum solutions over classical ones—particularly given the unavailability of quantum annealers. Rather, our objective is to investigate the behavior of different precision vectors within the QUBO framework, using classical results as a reference point to assess each representation.

Figure 1 shows the SNR values for different precision vector representations as K increases. For both noise levels, a good SNR is typically above 25 dB, while values below that indicate dominant quantization noise. In our case, the classical solver yields SNRs of approximately 45 dB for $\varepsilon = 0.5$ and 40 dB for $\varepsilon = 1.0$, which serve as practical upper bounds. Figure 2 presents the corresponding MSE values, where the classical solution provides lower bounds of approximately zero (for $\varepsilon = 0.5$) and one (for $\varepsilon = 1.0$).

We first analyze the case with $\varepsilon = 0.5$. Across all representations, the SNR improves as K increases, which aligns with expectations since greater quantization resolution yields better signal fidelity. The constant precision vector shows the slowest improvement, while binary, normal, and uniform representations eventually reach or closely approach the classical upper bound. Among these, the binary representation is the most efficient, achieving the upper bound at $K = 7$, thereby requiring fewer qubits to reach high-quality results.

The MSE values in Figure 2 reflect similar trends. The binary representation reaches the lower bound more rapidly than the others, though all representations eventually converge to it. This highlights that even if MSE is low, significant differences may remain in the precision of the estimated weights, a gap effectively captured by the SNR metric.

For the case with $\varepsilon = 1.0$, those trends remain largely consistent. The binary, normal, and uniform representations again outperform the constant vector. The binary representation reaches the upper SNR bound at $K = 5$, and achieves the MSE lower bound as early as $K = 3$, further reinforcing its efficiency. While stochastic representations (uniform and normal) perform better under this higher noise setting, they still do not surpass the binary scheme in terms of overall efficiency or performance.

The obtained results demonstrate that the choice of precision vector representation significantly impacts the quality of linear regression solutions within the QUBO formulation. In particular, the binary representation provides the best trade-off between performance and resource requirements, achieving optimal SNR and MSE values with fewer bits. For the problem under study, a minimum length of $K = 7$ for the binary precision vector appears necessary to ensure sufficiently accurate quantization.

One limitation of our experiments lies in the restricted range of regression coefficients, which were constrained to the interval $[0, 1]$. This simplification was necessary due to computational limitations, particularly when solving large QUBOs classically.

VI. Conclusions

Quantum computing is rapidly advancing and is expected to become an important technology for solving machine learning problems. Quantum linear regression, when formulated as a QUBO problem, has demonstrated advantages over its classical counterpart. However, its use in real-world applications remains limited by current hardware constraints, particularly in representing coefficients with sufficient precision. In this work, we investigate different number representations for the precision vector (a structure used to quantize linear regression coefficients). Our results show that, under current limitations, the binary representation consistently outperforms constant, uniform, and normally distributed alternatives. These findings suggest that the binary representation is the most promising choice for future studies on quantum-assisted linear regression.

There is still a long way to go before quantum-assisted regression can be effectively applied to large-scale problems, and there remains substantial room for improvement in this initial line of research. For example, a prior study [10] tested this problem with $D = 88$ features using $K = 2$, resulting in

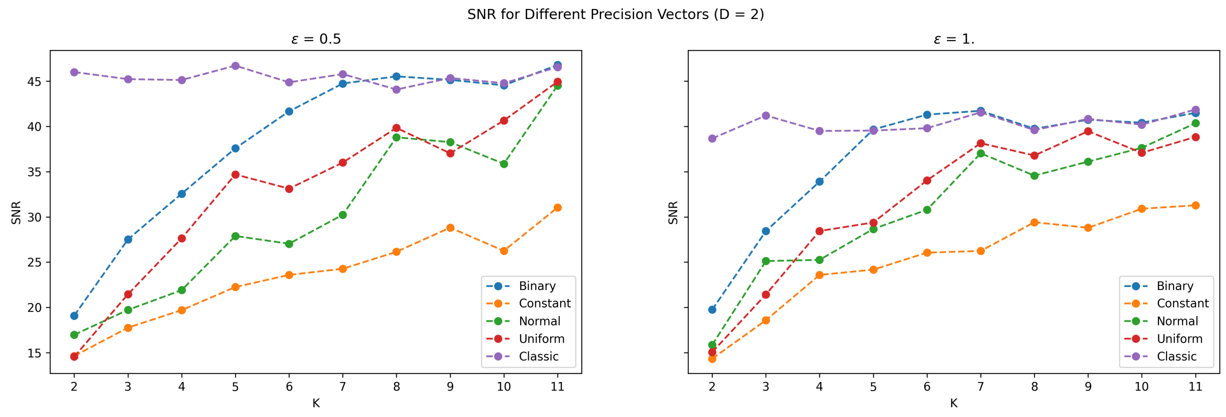

Fig. 1. SNR for different precision vectors with fixed number of samples and features ($N = 1000$, $D = 2$). ε denotes the standard deviation of the Gaussian noise, with $\varepsilon = 0.5$ and $\varepsilon = 1.0$.

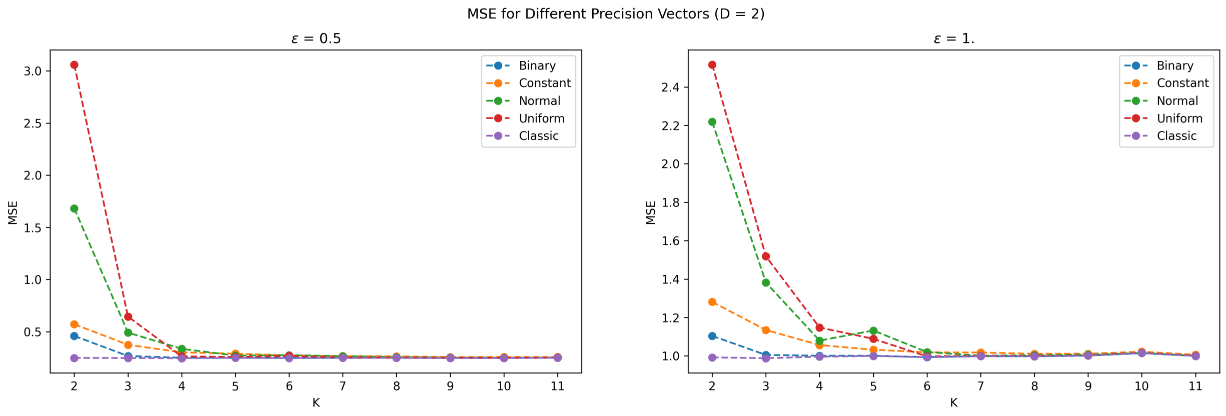

Fig. 2. MSE for different precision vectors with fixed number of samples and features ($N = 1000$, $D = 2$). ε denotes the standard deviation of the Gaussian noise, with $\varepsilon = 0.5$ and $\varepsilon = 1.0$.

$K \cdot D = 176$ binary variables. After minor embedding onto quantum hardware, this expanded to 2,939 physical qubits — currently among the largest instances tested. Using this as a reference, we estimate that employing a binary representation with $K = 7$ would restrict the model to approximately $D = 25$ features, given current quantum hardware limitations. This highlights the significant gap between the demands of high-fidelity numerical representations and the current capabilities of quantum annealers. Nonetheless, as quantum hardware continues to advance, the precision requirements identified in this work may serve as a benchmark.

References

[1] Jack D. Hidary. *Quantum Computing: An Applied Approach*. Springer International Publishing, 2019.

[2] Maria Schuld and Francesco Petruccione. *Machine Learning with Quantum Computers*. Springer International Publishing, 2021.

[3] Jacob Biamonte et al. “Quantum machine learning”. In: *Nature* 549.7671 (2017), pp. 195–202.

[4] Tameem Albash and Daniel A. Lidar. “Adiabatic quantum computation”. In: *Reviews of Modern Physics* 90.1 (Jan. 2018), p. 015002.

[5] Fred Glover, Gary Kochenberger, and Yu Du. “Quantum Bridge Analytics I: a tutorial on formulating and using QUBO models”. In: *4OR* 4 (Nov. 2019), pp. 335–371.

[6] Prasanna Date and Thomas Potok. “Adiabatic quantum linear regression”. In: *Scientific Reports* 11.1 (Nov. 2021), p. 21905.

[7] Katsuhiro Endo et al. “Novel real number representations in Ising machines and performance evaluation: Combinatorial random number sum and constant division”. In: *PLOS ONE* 19.6 (June 2024), e0304594.

[8] Catherine C. McGeoch et al. “Practical Annealing-Based Quantum Computing”. In: *Computer* 52.6 (June 2019), pp. 38–46.

[9] Sheir Yarkoni et al. “Quantum annealing for industry applications: introduction and review”. In: *Reports on Progress in Physics* 85.10 (Sept. 2022), p. 104001.

[10] Costantino Carugno, Maurizio Ferrari Dacrema, and Paolo Cremonesi. “Adaptive Learning for Quantum Linear Regression”. In: *2024 IEEE International Conference on Quantum Computing and Engineering (QCE)*. IEEE, Sept. 2024, pp. 1595–1599.