Quantum Principal Component Analysis for Financial Fraud Detection

Giacomo Lancellotti*‡, Alberto Guerrini*‡, Giacomo Ranieri†, Valeria Zaffaroni†, and Paolo Cremonesi*

*DEIB, Politecnico di Milano, Milano, Italy
{giacomo.lancellotti, alberto1.guerrini, paolo.cremonesi}@polimi.it
†Intesa Sanpaolo, Torino, Italy
{giacomo.ranieri, valeria.zaffaroni}@intesasanpaolo.com

***Abstract*—Among quantum machine learning applications anomaly detection has garnered significant attention due to its critical role in cybersecurity and financial fraud analysis. While prior research has proposed high-level quantum algorithms, most lack circuit-level implementations suitable for fault-tolerant quantum machines, limiting their practical viability. We present a complete quantum circuit design for an anomaly detection algorithm based on principal component analysis. We derive closed-form expressions for qubits count, circuit depth, gate count, T-depth, and T-count, of each quantum subroutine involved. We apply our implementation to financial fraud detection using a real-world dataset of approximately 10^6 samples with 20 features, and provide asymptotic resource estimates. Our analysis confirms logarithmic scaling in circuit depth with respect to the number of training samples and feature dimensions, achieving an exponential speed-up over the classical anomaly detection algorithm.**

***Index Terms*—anomaly detection, quantum machine learning, principal component analysis, fraud detection, resource analysis**

I. Introduction

Quantum Computing (QC) has emerged as a promising technology due to its potential to solve computational problems with exponential speed-ups over the best-known classical algorithms—particularly in fields such as physical simulation, cryptography, and artificial intelligence. Although current advancements in quantum hardware, error correction protocols, and quantum software are still at least a decade away from supporting utility-scale applications, it is crucial to estimate the resource requirements for such tasks and to develop suitable quantum algorithms accordingly. Recent research has begun exploring the application of QC to Machine Learning (ML). While early efforts primarily focused on accelerating specific subroutines, such as kernel mappings, more recent developments have introduced fault-tolerant quantum algorithms for complete ML workflows. These include Quantum Neural Networks [1], Quantum Support Vector Machines [2], and Quantum Principal Component Analysis [3]. One particularly impactful application domain for ML is Anomaly Detection (AD) [4], which encompasses a broad range of techniques for identifying patterns in data that deviate from expected or genuine behaviour. AD has found application in critical domains such as cybersecurity [5], finance [6], and healthcare [7], where the timely identification of anomalies is essential to ensure reliability, security, and the mitigation of adverse outcomes. However, the growing volume of data to be processed poses significant challenges for conventional computational methods, whereas QC has emerged as a promising alternative for addressing such data-intensive tasks [8]. Currently, fully quantum algorithms for AD targeting fault-tolerant quantum machines are primarily proposed at a high or theoretical level of abstraction. However, such formulations alone are insufficient to determine whether a quantum advantage can be realized on real-world datasets. When empirical evaluation is not feasible due to limited quantum hardware resources, a detailed resource complexity analysis becomes essential for assessing potential advantages. Consequently, resource requirements—including qubit count, circuit depth, and gate count—must be carefully evaluated to obtain reliable cost estimates that closely reflect the actual resource consumption.

This Work. To address the lack of fully quantum circuit-level implementations, we propose a complete quantum circuit design for an AD technique based on Principal Component Analysis (PCA), following the framework introduced in [9]. The method identifies anomalies in a dataset by computing their distance from the distribution of genuine data and projecting this distance vector onto the principal components, i.e., the directions of highest variance. We employ quantum Amplitude Estimation (AE), a subroutine capable of estimating the probability of measuring a target state with logarithmic scaling in the inverse estimation error [10]. All subcircuits are decomposed using the Clifford+T universal gate set and evaluated in terms of qubit count, circuit depth, and gate counts. Finally, we assess the fault-tolerant requirements to execute our implementation on a real-world financial anomaly detection task, using a dataset of real bank transactions, and provide concrete resource estimates.

Related Works. A comprehensive survey of AD techniques leveraging Quantum Machine Learning (QML) is presented in [11]. A practical application of a near-term AD method to financial fraud detection is explored in [12], demonstrating improved classification performance compared to classical baselines. A circuit-level implementation for the approximate extraction of principal components is proposed in [13], which

‡ These authors contributed equally.

includes a `Qiskit`-based realization and an analysis of asymptotic complexities. In contrast, existing PCA-based AD methods targeting fault-tolerant quantum devices are typically proposed from a theoretical perspective. These approaches often lack detailed circuit-level designs and evaluate the potential quantum advantage only in terms of asymptotic query complexity based on high-level oracle calls. Specifically, the algorithm introduced in [14] employs the swap test to compute inner products, achieving exponential speed-ups in both the number of samples and features over classical counterparts. However, this method assumes the access to quantum-encoded inputs, and therefore does not account for the overhead of loading classical data into quantum states. The method proposed in [9] addresses this issue by introducing a PCA-based quantum AD algorithm that uses AE and operates on classical data encoded digitally in quantum states. This approach achieves an asymptotic speed-up comparable to that of [14], while being more suitable for scenarios involving classical input data.

Contributions. We present a complete circuit-level implementation of a quantum AD approach targeting fault-tolerant quantum devices, based on the method described in [9]. Our implementation employs signed-value data representation and introduces new subcircuits for their manipulation. We demonstrate that our design achieves logarithmic depth and T-depth with respect to the number of training samples and features, thereby realizing the exponential quantum speed-up suggested by the asymptotic complexity analysis in [9]. We derive closed-form expressions for logical qubit requirements, depth, T-depth, gate count, and T-count for each subroutine in the circuit. In addition, we estimate the resources required to apply our method to a financial fraud detection task using a real-world dataset collected in 2024 by a financial institution. The complete implementation, developed in `Qiskit`, is publicly available at https://github.com/ague01/quantum-pca-fraud-detection.

II. Background

This section provides an overview of ML techniques commonly used to address AD tasks, with a particular focus on PCA-based methods. We then discuss how QC can be leveraged for ML applications by introducing fault-tolerant quantum subroutines relevant to this domain, along with their current limitations.

A. Anomaly Detection with Principal Component Analysis

AD problems have been extensively addressed using ML techniques. Existing approaches in the literature include Support Vector Machines, Principal Component Classifiers, Random Forests, Deep Neural Networks, Hidden Markov Models, and Boltzmann Machines [1], [6], [15], [16]. Since anomalous data typically represents only a small fraction of the overall dataset and its characteristics are often unknown in advance, real-world datasets tend to be highly imbalanced toward genuine data [17]. As a result, unsupervised learning techniques are generally preferred for AD as they often outperform supervised methods in such settings [15]. Among unsupervised

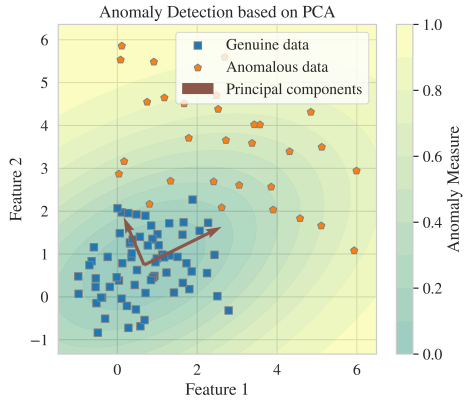

Fig. 1. Example of anomaly measures learned by the AD-PCA algorithm trained on genuine data (blue squares). Yellow areas indicate regions of higher anomaly. Arrow lengths are proportional to the explained variance.

ML techniques, the Anomaly Detection based on Principal Component Analysis (AD-PCA) algorithm, first proposed in [18], demonstrates robustness to noise and provides tighter decision boundaries compared to other unsupervised methods. It leverages the observation that anomalies tend to deviate from the main variance directions of the training data when measured by their distance from the sample mean. Specifically, given a training set composed of M genuine data vectors $\boldsymbol{x}^i \in \mathbb{R}^d$ with d real-valued features, the sample mean vector $\boldsymbol{\mu} \in \mathbb{R}^d$ is defined as the average of all feature vectors in the dataset, i.e. $\boldsymbol{\mu} = \frac{1}{M}\sum_{i=1}^{M} \boldsymbol{x}^i$. The sample covariance matrix $\boldsymbol{C} \in \mathbb{R}^{d\times d}$ captures the pairwise relationships between features, with each entry representing the covariance between two features across all samples. It is defined as $\boldsymbol{C} = \frac{1}{M-1}\sum_{i=1}^{M}(\boldsymbol{x^i} - \boldsymbol{\mu})(\boldsymbol{x^i} - \boldsymbol{\mu})^T$. An example of the AD-PCA algorithm applied to two-dimensional data is illustrated in Fig. 1. The principal components, i.e. the eigenvectors of the covariance matrix $\boldsymbol{C}$, are shown as arrows representing the main directions of variance in the dataset and originating from the centre of the genuine data $\boldsymbol{\mu}$. The anomaly measure for a new data point $\boldsymbol{x^0} \in \mathbb{R}^d$ is computed as the difference between two quantities: (1) the squared norm of the distance vector between $\boldsymbol{x^0}$ and the sample mean $\boldsymbol{\mu}$, and (2) the squared norm of the projection of this distance vector onto the principal components of the dataset. As presented in Fig. 1, this measure is lower for points that lie closer to the centre and align with the principal components. Consequently, anomalous points can be detected by setting a threshold δ tailored for the specific application, such that an input is flagged as anomalous if $p(\boldsymbol{x^0}) \geq \delta$. Although this formulation is based on a linear kernel, it can be extended to non-linear settings using kernel functions, as demonstrated in [18].

B. Quantum Machine Learning Basics

Quantum Machine Learning (QML) investigates how quantum computers can be leveraged to accelerate or improve the performance of ML tasks [8], [19]. A major challenge in practical QML implementations is the efficient encoding of

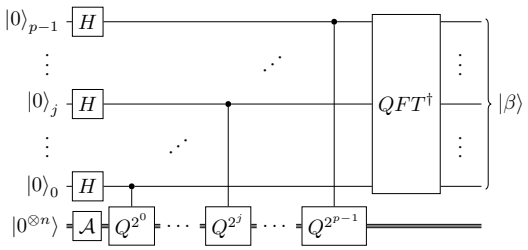

Fig. 2. Amplitude Estimation circuit from [10] with p ancilla qubits, performed on a n-qubit register. Where $\mathcal{A}$ is the state-preparation circuit, $Q = -\mathcal{A}S_0\mathcal{A}^\dagger S_{\psi_0}$, $QFT^\dagger$ is the inverse Quantum Fourier Transform, and β is the output of the procedure, basis-encoded in in the ancilla qubits.

classical data into quantum states, a process known as quantum data loading. Given a classical input $x \in \mathbb{N}$, several encoding strategies have been proposed, each with trade-offs in terms of resource requirements. In *amplitude encoding*, the components of a normalized classical vector are mapped to the amplitudes of a quantum state. In the simplest case of a scalar, the latter can be encoded as $|\psi(x)\rangle = \frac{1}{\sqrt{C}}(x\,|0\rangle + \sqrt{1-x^2}\,|1\rangle)$, where C is a normalization constant set to the maximum value x can take. This method allows exponential compression of data but generally requires deep circuits for the value encoding and introduces additional complexity in the implementation of arithmetic circuits. In *basis encoding*, data is stored directly in the computational basis states of qubits. Considering the binary representation $x = x_1 \dots x_n = \sum_{j=1}^{n} x_j 2^{n-j}$, where $x_j \in \{0,1\}$, it is represented as $|x\rangle = |x_1 \dots x_n\rangle \in \{|0\rangle, |1\rangle\}^{\otimes n}$. This encoding requires a number of gate and qubits which is linear in the bit-length of the input, however it allows for easier manipulation of encoded data. In *phase encoding*, classical values are embedded in the relative phases of a quantum state. For instance, the state $|\psi(x)\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle + e^{ix}\,|1\rangle\right)$, encodes the information value x in the complex phase.

As in classical computation, many quantum algorithms require efficient access to stored data. For this purpose, the quantum analogue of random-access memory—*Quantum Random Access Memory* (QRAM)—can be employed. Similar to its classical counterpart, QRAM consists of M memory cells $[x_1, x_2, \dots, x_M]$, indexed by $1 \le i \le M$, where each cell stores a value $x_i \in \mathbb{N}$ that can be queried in superposition. A QRAM query corresponds to the unitary transformation $|i\rangle\,|0^{\otimes n}\rangle \mapsto |i\rangle\,|x_i\rangle$, where the index is encoded in a $\lceil \log_2 M \rceil$-qubit quantum register $|i\rangle$, and it may also address n-dimensional data matrices. Although no physical QRAM devices have been realized to date [20], various quantum subroutines can simulate QRAM-like behavior. Efficient QRAM implementations and data structures—such as the *Bucket-Brigade* architecture [21], the *KP-Tree* structure [22], and the *Select-Swap* technique [23]—enable the simulation of QRAM within a quantum circuit, incurring only a polynomial overhead in terms of gate depth requirements.

A relevant subroutine that finds applications in QML to retrieve amplitude-encoded values is the *Amplitude Estimation* (AE) routine [10]. Given a quantum state $|\psi\rangle = \sqrt{\alpha}\,|\psi_0\rangle + \sqrt{1-\alpha}\,|\psi_1\rangle$, AE estimates the value of α assuming the ability to perform phase reflections around $|\psi_0\rangle$–often referred to as the good state. The original AE algorithm is based on Quantum Phase Estimation and, to obtain a δ-approximation of α, it requires $p = \log_2(\delta^{-1})$ additional qubits and a total of $\sum_{i=0}^{p-1} 2^i = 2^p - 1 \sim \mathcal{O}(\delta^{-1})$ applications of the controlled Grover operator Q as illustrated in Fig. 2. The operator is defined as $Q = -\mathcal{A}S_0\mathcal{A}^\dagger S_{\psi_0}$, where $\mathcal{A}$ is a state-preparation circuit for $|\psi\rangle$, i.e. $|\psi\rangle = \mathcal{A}\,|0\rangle$, S_0 is the phase reflection about the $|0\rangle$ state, and S_{ψ_0} is the reflection about the good-state $|\psi_0\rangle$. A generic reflection about a state $|\phi\rangle$ is defined as $S_\phi = I - 2\,|\phi\rangle\langle\phi|$. Finally, the probability α is then recovered as $\alpha = \sin^2(\pi\beta)$, where β is the basis-encoded output of the AE circuit. This step can be performed either classically, after measuring the value β, or within a quantum setting using dedicated subroutines [24].

QML techniques can be employed to effectively accelerate the AD-PCA algorithm, as initially proposed in [14]. A recent approach introduced in [25] and refined in [9], assumes to access in superposition the training data $\boldsymbol{x^i}$ and the new point $\boldsymbol{x^0}$ via QRAM. It leverages AE to compute inner products between vectors in parallel across dataset dimensions, leading to an exponential asymptotic speed-up in query complexity compared to the classical AD-PCA.

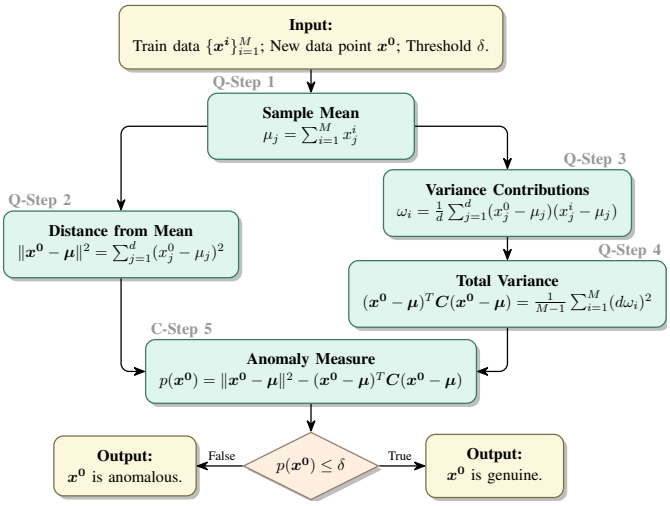

Fig. 3. High-level workflow for the proposed AD-QPCA algorithm where each step is represented by the output value it produces and can be either quantum (Q) or classically (C) performed.

III. Quantum Anomaly Detection Algorithm

In this section, we present our Anomaly Detection based on Quantum Principal Component Analysis (AD-QPCA) algorithm. A high-level overview with the main quantum and classical steps is shown in Fig. 3. Except for Step 5, where the subtraction is handled classically, Steps 1–4 involve a summation over either the features or the data points, which can be reformulated as an inner product, as shown in [9]. Provided that AE enables efficient computation of inner products between amplitude-encoded vectors in superposition over their

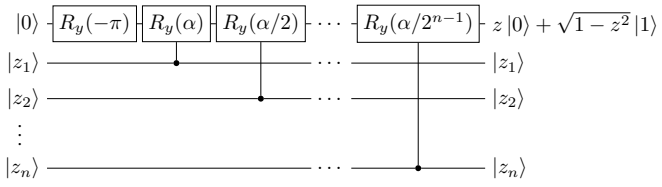

Fig. 4. Circuit for the Basis-to-Amplitude (BtA) conversion, where the input z is basis-encoded in a n-qubit register and α is a real parameter.

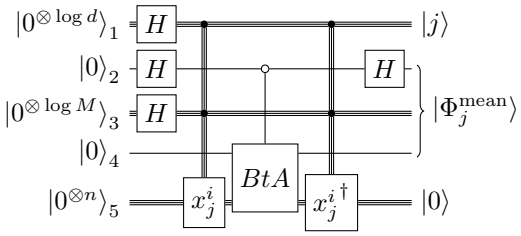

Fig. 5. Circuit preparing the state $|\Phi_j^{\text{mean}}\rangle$ having the sample mean value μ_j encoded in the amplitude of the good-state, for each feature j in superposition.

components, each quantum step in the algorithm leverages an AE circuit to achieve a quantum speed-up. The design of quantum subcircuits on which AE operates is detailed below.

(Step 1) Sample mean features. In this step, we compute the mean of each feature across all data points in the dataset. We assume the input data to be stored in QRAM using basis encoding, which allows logarithmic-resource access to individual entries [21]. Let x_j^i denote the subcircuit used to access the j-th feature of the i-th training data point in QRAM. Since the dataset is accessed in superposition over the index i, the mean computation is naturally supported through amplitude encoding combined with AE, which offers a more resource-efficient approach than summing values explicitly through basis encoding and adder subroutines.

To achieve this, we perform a controlled basis-to-amplitude (BtA) conversion using a subcircuit whose approximate implementation was first proposed in [26]. This conversion relies on a sequence of controlled rotations about the y-axis and maps an input $z \in [0,1]$, basis-encoded in n-qubits, to its amplitude representation as follows: $|z\rangle\,|0\rangle \mapsto |z\rangle\,(z\,|0\rangle + \sqrt{1-z^2}\,|1\rangle)$. The second register is initialized in the $|0\rangle$ state and holds the amplitude-encoded value z after the transformation. A key distinction between the conversion implemented in [26] and the one required in this work lies in the target state: here, the value y must be encoded in the amplitude of the $|0\rangle$ state. This is realized by applying an additional $R_y(-\pi)$ rotation, since $\sin y = \cos(y - \frac{\pi}{2})$. The construction of the BtA subcircuit used in our implementation is shown in Fig. 4. Although more accurate methods to implement such mapping exist, they typically involve resource-intensive arithmetic operations such as exponentiation [27].

The circuit for the preparation of the state $|\Phi_j^{\text{mean}}\rangle$ required to compute the sample mean, is shown in Fig. 5. The BtA gate, open-controlled on qubit 2, conditionally encodes the values x_j^i in the amplitude of ancilla qubit 4, in superposition over indexes (i,j). After uncomputing ancillary registers, the mean components are encoded in registers 2 to 4 as

$$|\Phi_j^{\text{mean}}\rangle = \frac{1}{\sqrt{2}}\left(\sqrt{1+\mu_j}\,|\Phi_{j,0}^{\text{mean}}\rangle + \sqrt{1-\mu_j}\,|\Phi_{j,1}^{\text{mean}}\rangle\right),$$

in superposition over all indexes j, encoded in register 1. We then compute the probability values $\alpha_j = \frac{1+\mu_j}{2}$ from the $|\Phi_j^{\text{mean}}\rangle$ state by applying the AE routine. Being the good-state $|\Phi_{j,0}^{\text{mean}}\rangle$ characterized by qubit 2 in state $|0\rangle$, the reflection operator required for AE is defined as $S = I - 2\,|0\rangle_2\,\langle 0|_2$ and can be implemented by applying the gate sequence XZX on this qubit. To recover the true mean components from the AE output, the post-processing $\mu_j = 2\alpha_j - 1$ is required. Finally, the mean values μ_j are assumed to be stored in QRAM for their reuse in subsequent steps.

(Step 2) Distance from sample mean. This step computes the squared Euclidean distance between the new data point and the sample mean. As in Step 1, we assume that both the new data point and the mean vector are stored in QRAM and can be queried in superposition over the features j. After data loading, we compute the difference between the data point and the mean, i.e. $x_j^0 - \mu_j$, using quantum arithmetic in basis encoding, to prepare the value to be amplitude-encoded in superposition, enabling for the AE-based summation. However, while arithmetic subroutines in basis encoding naturally handle negative values through two's complement representation [28], amplitude encoding typically favours sign-magnitude representation, since negative values can be expressed directly via negative amplitudes. It follows that attempting to sum values encoded in two's complement by amplitude encoding and AE (as described in Step 1) can yield incorrect results.

To this end we define the $Dist$ gate, presented in Fig. 6, computing the difference between two non-negative values $x, y \geq 0$ and providing the output still in basis encoding but in sign-magnitude notation, i.e. $|y\rangle_1\,|0\rangle_2\,|x\rangle_3 \mapsto ||y-x|\rangle_1\,|\text{sign}(y-x)\rangle_2\,|x\rangle_3$, where inputs x and y are basis-encoded in registers 1 and 3, respectively, and qubit 2 stores the sign being $|1\rangle$ when $y - x < 0$, $|0\rangle$ otherwise. The circuit for this unitary is realized by first applying a subtraction gate (Sub) [28], followed by a controlled sign-flip gate ($SignFlip$) that inverts the sign of the result in two's complement notation, when it is negative. Specifically, the circuit for the $SignFlip$ gate is constructed by applying a bitwise NOT to all qubits using X gates and adding one—performed in-place by a QFT-based incrementer [29].

We then transform the computed distance from basis encoding to amplitude encoding using the BtA subcircuit, which maps the distance value to the amplitude of an ancilla qubit, thereby enabling the application of the AE subroutine. All qubits used for arithmetic and data loading are then uncomputed, resulting in the final quantum state:

$$|\Phi^{\text{dist}}\rangle = \frac{1}{\sqrt{d}}\sum_{j=1}^{d}|j\rangle\left((x_j^0 - \mu_j)\,|0\rangle + \sqrt{1-(x_j^0-\mu_j)^2}\,|1\rangle\right),$$

where register 1 encodes the feature index $|j\rangle$, and qubit 2 encodes in its amplitude the j-th component of the distance

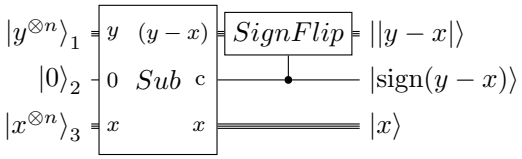

Fig. 6. Circuit for the $Dist$ gate, where inputs $x, y \geq 0$ are basis-encoded in n-qubits, Sub is the subtraction circuit with carry-out qubit c, and $SignFlip$ computes the inversion of the sign in two's complement notation.

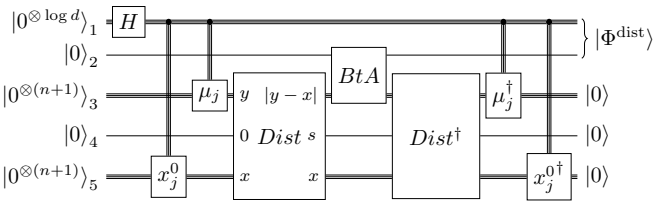

Fig. 7. Circuit preparing the state $|\Phi^{\text{dist}}\rangle$ having the squared distance norm value $\|\boldsymbol{x^0} - \boldsymbol{\mu}\|^2$ encoded in the good-state amplitude.

vector. The circuit preparing the state $|\Phi^{\text{dist}}\rangle$ is shown in Fig. 7. The squared Euclidean distance is computed by applying AE to $|\Phi^{\text{dist}}\rangle$ yielding the probability α of measuring the good-state. This state is identified by qubit 2 being in the $|0\rangle$ state, thus the reflection S is implemented as in Step 1. Finally, the classical post-processing $\|\boldsymbol{x^0} - \boldsymbol{\mu}\|^2 = d\alpha$ is applied to the AE output to retrieve the squared Euclidean distance. Since this value is not needed for further quantum computation, it is measured and stored classically for use in later steps.

(Step 3) Variance contributions. This step computes the contribution of each data point to the projection of the total training set variance onto the direction defined by the difference between the new data point $\boldsymbol{x^0}$ and the sample mean $\boldsymbol{\mu}$. Specifically, we compute the inner product $\omega_i = \frac{1}{d}\sum_{j=1}^{d}(x_j^0 - \mu_j)(x_j^i - \mu_j)$ for each of the M samples in superposition over index i. As in the previous step, we assume that the components of the samples x_j^i, the mean μ_j, and the new data point x_j^0 are all stored in QRAM and are accessible in superposition over the paired indices (i, j). Two ancillary qubits are reserved to handle the sign bits of the two distance measures. We first compute $|x_j^0 - \mu_j|$ and $|x_j^i - \mu_j|$ in basis encoding, along with their corresponding sign bits, using two applications of the $Dist$ gate, as described in Step 2. The resulting values are then multiplied using a quantum multiplier gate (Mul), such as the one proposed in [30], which stores the output in a $2n$-qubit register to prevent overflow issues. Next, we encode the product into the amplitude of qubit 2 using the BtA subcircuit. To preserve the sign information of the two computed distances, we apply two multi-controlled Z gates—each conditioned on the opposite sign bits—which flip the phase of the amplitude-encoded product when it is negative. This ensures that negative contributions are properly considered in the final summation. After uncomputing all temporary registers used for arithmetic and data loading, the resulting state in registers 2–4 is given by:

$$|\Phi_i^{\text{cov}}\rangle = \frac{1}{\sqrt{2}}\left(\sqrt{1+\omega_i}\,|\Phi_{i,0}^{\text{cov}}\rangle + \sqrt{1-\omega_i}\,|\Phi_{i,1}^{\text{cov}}\rangle\right),$$

while the sample index i is encoded in the first register, in superposition. The circuit that prepares the state $|\Phi_i^{\text{cov}}\rangle$ is shown in Fig. 8. To extract the final values ω_i, we follow the approach from Step 1 by applying AE to the states $|\Phi_i^{\text{cov}}\rangle$ obtaining the probabilities α_i (in superposition over index i), and then post-processing the results using $\omega_i = (2\alpha_i - 1)$. Finally, the resulting ω_i values are stored in QRAM for their use in the subsequent step.

(Step 4) Total variance along $(\boldsymbol{x^0} - \boldsymbol{\mu})$. This step computes the total variance of the training dataset projected along the direction defined by the difference vector $(\boldsymbol{x^0} - \boldsymbol{\mu})$. This quantity is obtained as the sum of the M output values of Step 3, i.e. $(\boldsymbol{x^0} - \boldsymbol{\mu})^T\boldsymbol{C}(\boldsymbol{x^0} - \boldsymbol{\mu}) = \frac{1}{M-1}\sum_{i=1}^{M}\omega_i^2$, that are assumed to be stored in QRAM and accessed in superposition over the sample index i. To prepare the required amplitude-encoded state, we apply a BtA gate that maps each basis-encoded value ω_i in superposition into the amplitude of an ancilla qubit. This produces

$$|\Phi^{\text{cov}}\rangle = \frac{1}{\sqrt{M}}\sum_{i=1}^{M}|i\rangle\left(\omega_i\,|0\rangle + \sqrt{1-\omega_i^2}\,|1\rangle\right),$$

where register 1 encodes the index i and register 2 the variance components, in superposition. The circuit to prepare the state $|\Phi^{\text{cov}}\rangle$ is equivalent to the one for Step 2 in Fig. 7, where the superposition is over M values instead of d, and there is no need for the $Dist$ gate or other algebraic circuits. The AE procedure is applied to the state $|\Phi^{\text{cov}}\rangle$ to extract the squared variance projection value. As in previous steps, the good-state is defined by register 2 being in the $|0\rangle$ state, allowing the reflection operator to be equivalently constructed. Finally, the AE output α is measured and classically post-processed to obtain the normalized total variance along the distance vector between $\boldsymbol{x^0}$ and $\boldsymbol{\mu}$, using $(\boldsymbol{x^0} - \boldsymbol{\mu})^T\boldsymbol{C}(\boldsymbol{x^0} - \boldsymbol{\mu}) = \frac{M}{M-1}\alpha$. The value is stored classically as it directly contributes to the final anomaly score.

(Step 5) Anomaly measure. The final step computes the anomaly score $p(\boldsymbol{x^0}) = \|\boldsymbol{x^0} - \boldsymbol{\mu}\|^2 - (\boldsymbol{x^0} - \boldsymbol{\mu})^T\boldsymbol{C}(\boldsymbol{x^0} - \boldsymbol{\mu})$, which quantifies the likelihood that the new data point $\boldsymbol{x^0}$ is an anomaly. This value is obtained by subtracting the total variance of the training dataset projected along the direction of $\boldsymbol{x^0} - \boldsymbol{\mu}$ (computed in Step 4) from the squared norm of the distance $\boldsymbol{x^0} - \boldsymbol{\mu}$ (computed in Step 2). Since both quantities are already measured and stored classically, this step can be performed as a classical operation with complexity $\mathcal{O}(1)$, therefore it requires no additional quantum computation. Finally, based on the value of $p(\boldsymbol{x^0})$ and a user-defined threshold δ—which can be arbitrarily chosen depending on the properties of the dataset under consideration and on operational requirements—the AD-QPCA algorithm classifies the new data point $\boldsymbol{x^0}$ as *genuine* if $p(\boldsymbol{x^0}) \leq \delta$, and as *anomalous* otherwise.

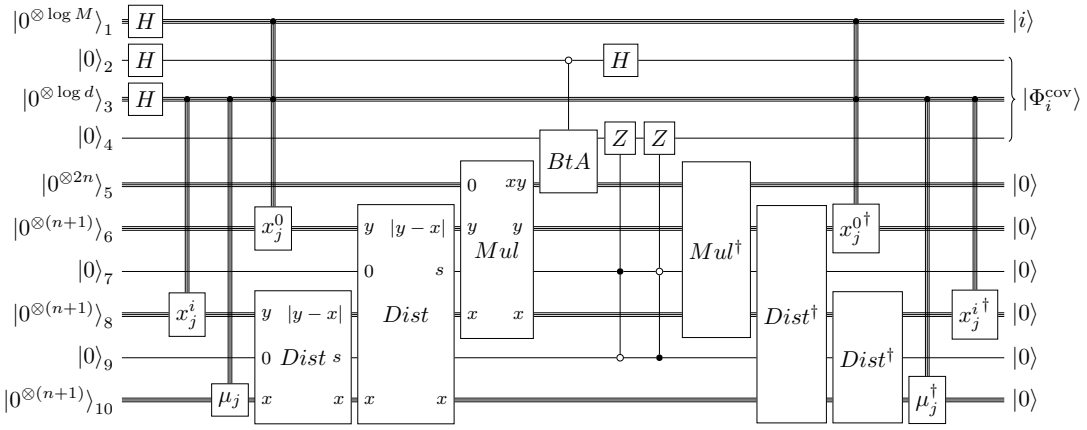

Fig. 8. Circuit preparing the state $|\Phi_i^{\text{cov}}\rangle$ having the variance contribution ω_i encoded in the amplitude of the good-state, for each sample i in superposition. Registers 6, 8, and 10 include an additional qubit to encode the sign for internal computation.

TABLE I
CLOSED FORM RESOURCE COUNTS FOR BASIC GATES, WITH n THE SIZE OF INPUT DATA REGISTER AND T_C^{Rz} THE T GATE COUNT FOR THE R_Z.

Gate	N_Q	C	D	T_C	T_D	Refs
CCX	7	23	7	7	1	[31]
CCZ	7	25	9	7	1	–
C^cX	c+5	92(c-1)	28(c-2)	28(c-2)	4(c-2)	[32]
R_z	1	$2T_C^{Rz}$	$2T_C^{Rz}$	$10+4\log(1/\epsilon)$	T_C^{Rz}	[33]
R_y	1	$4+2T_C^{Rz}$	$4+2T_C^{Rz}$	T_C^{Rz}	T_C^{Rz}	[34]
CR_y	2	$54+4T_C^{Rz}$	$10+4T_C^{Rz}$	$2T_C^{Rz}$	$2T_C^{Rz}$	[32]
QFT	n	$27n^2$-26n+ $2n(n-1)T_C^{Rz}$	$(4n-2)T_C^{Rz}$	$n(n-1)T_C^{Rz}$	$(2n-1)T_C^{Rz}$	[35]
Add	2n+2	53n-25	14n-2	14n-7	2n-1	[28]
Mul	6n+1	$95n^2$-119n-23	$23n^2$+11n-7	$21n^2$+21n-7	$3n^2$+2n-1	[30]
Sign-Flip	n	$54n^2$-55n+ $2n(n+1)T_C^{Rz}$	$(8n-2)T_C^{Rz}$	$n(2n-1)T_C^{Rz}$	$(8n-2)T_C^{Rz}$	–

IV. EXPERIMENTAL EVALUATION

This section presents an evaluation of the proposed AD-QPCA algorithm, assessing the realization costs using the Clifford+T gate set. Furthermore, we estimate the quantum resources required to process a real-world financial fraud dataset, and compare the performance of our algorithm to the classical counterpart.

A. Resource Analysis

Our resource analysis aims to validate the exponential quantum speed-up, with respect to the classical AD-PCA algorithm, claimed by the asymptotic query complexity analysis in [9]. The proposed subcircuits have been implemented to assess their correctness using the `Qiskit 1.4` SDK on a machine with an Intel-Core i7-13700H CPU and 16GB of RAM. The presented resource analysis relies on the universal Clifford+T gates set decomposition, reporting the number of logical qubits N_Q, gate count C, gate depth D, T gate count T_C, and T gate depth T_D, for each of the state-preparation circuits in Sec. III.

Resource counts are provided in closed-form notation for subcircuits up to the scale of individual state-preparation circuits. For larger composite circuits, we instead report asymptotic resource metrics, as they provide a clearer picture of the overall algorithm's scaling behaviour—particularly when multiple controlled subcircuits are combined, as in the AE procedure. Closed-form resource counts for basic gates used in our implementation such as multi-controlled gates, rotational gates, and arithmetic gates, are reported in Tab. I and grounded on the considerations below.

For the CCX gate, we adopt a decomposition strategy that minimizes circuit depth at the cost of using 4 ancillary qubits [31], assuming ancillas can be reused across subsequent gates. For controlled gates with more than two control qubits, the decomposition follows the method proposed in [32]. A generic single-qubit rotation around the z-axis, $R_z(\theta)$, can be approximated up to an error ϵ using Clifford+T circuits, with the T-count and Hadamard count estimated via the Matsumoto-Amano normal form [33], [36]. The $R_y(\theta)$ rotation can be implemented as $R_y(\theta) = SHR_z(\theta)HS^\dagger$ [34], allowing resource estimates for $R_y(\theta)$ to be directly derived from those of $R_z(\theta)$. The n-qubit QFT gate is implemented using n Hadamard gates and $\frac{n(n-1)}{2}$ controlled-R_z gates [35]. The addition/subtraction circuit is implemented using the ripple-carry adder [28], while multiplication is performed using the shift-and-add algorithm, whose quantum circuit involves multiple controlled-add operations. A T-count-optimized implementation of the controlled-add gate is presented in [30]. The n-qubit BtA encoding circuit is implemented using a sequence of n CR_y gates followed by a single R_y gate. Its controlled version requires n CCR_y gates and one CR_y gate. Note that the CCR_y gates can share the same ancilla qubits, as they are applied sequentially. The resource counts for the $Dist$ gate are obtained by summing the cost of the subtraction circuit, a single layer of n X gates, and the cost of a constant increment operation. Tab. II reports the closed-form expressions for the cost of the state-preparation circuits presented in Sec. III for each step of the algorithm.

TABLE II

CLOSED-FORM RESOURCE COUNTS FOR THE STATE-PREPARATION CIRCUITS IN TERMS OF NUMBER OF QUBITS N_Q GATE COUNT C, GATE DEPTH D, T GATE COUNT T_C AND DEPTH T_D. WITH M TRAINING SAMPLES, d FEATURES, n QUBIT ENCODING, AND A TOLERATED ERROR ϵ FOR $R(\theta)$.

Step:	1	2	3	4
N_Q	$n + \log(Md) + 2$	$2n + \log(d) + 3$	$5n + \log(Md) + 7$	$n + \log(M) + 1$
C	$16n^2 \log(1/\epsilon) + 94n^2 + 4n - 24$	$188n^2 + 102n + (32n^2 + 16n + 16)\log(1/\epsilon) + 45$	$566n^2 + 1050n + (64n^2 + 144n + 80)\log(1/\epsilon) + 346$	$94n + (16n + 16)\log(1/\epsilon) + 95$
D	$64n + (8n + 8)\log(1/\epsilon) + 51$	$260n + (96n + 40)\log(1/\epsilon) + 89$	$46n^2 + 638n + (168n + 200)\log(1/\epsilon) + 663$	$30n + (16n + 24)\log(1/\epsilon) + 49$
T_C	$34n + (8n + 8)\log(1/\epsilon) + 20$	$40n^2 + 48n + (160n^2 + 8n + 4)\log(1/\epsilon) - 4$	$122n^2 + 376n + (320n^2 + 648n + 328)\log(1/\epsilon) + 212$	$20n + (8n + 4)\log(1/\epsilon) + 10$
T_D	$22n + (8n + 8)\log(1/\epsilon) + 20$	$104n + (40n + 4)\log(1/\epsilon) + 8$	$6n^2 + 206n + (72n + 72)\log(1/\epsilon) + 194$	$20n + (8n + 4)\log(1/\epsilon) + 10$

B. Financial Fraud Dataset Resource Estimation

We provide an estimate of the resources required by the AD-QPCA algorithm for a utility-scale fraud detection task. A real-world financial dataset containing bank transfers from January to September 2024 has been utilized. After preprocessing and data cleaning, the resulting dataset comprises approximately $M \approx 11 \times 10^6$ samples and $d = 20$ features. These features include timestamps, transaction amounts, textual descriptions provided by the sender, and the fingerprint of the device used to confirm the transfer. Approximately 2% of the transactions are labelled as fraudulent, indicating a significant class imbalance—a common characteristic in AD tasks.

To simulate the QRAM, we employ the parallel Bucket Brigade implementation [21], which constitutes a practical approach compatible with current quantum hardware, which, for an $M \times d$ training data matrix with n-qubit entries, requires $8Mdn$ logical qubits, $123Mdn - 2n\log(Md) - 156n$ gates, total circuit depth of $16\log(Md) - 5$, T-count of $21Mdn - 28n$, and T-depth of $2\log(Md)$. As explained in Sec. II, the AE algorithm provides a δ-approximated result with $\mathcal{O}(\delta^{-1})$ sequential application of the controlled oracle Q and $\lceil \log_2 \delta^{-1} \rceil$ ancilla qubits. We assume without loss of generality that, at each step, the tolerated errors δ and ϵ are at most the discretization error $2^{-(n+1)}$ arising from the encoding of values in n qubits. It follows that AE requires $n + 1$ additional qubits and $\mathcal{O}(2^n)$ executions of the state-preparation circuit for each step, and the tolerated error for the approximation of arbitrary rotations is set s.t. $\log_2(\frac{1}{\epsilon}) = n + 1$ holds. We highlight that the dependence on n does not arise from the dimensionality of the dataset, but rather from the specific data encoding technique employed by the quantum algorithm. Consequently, once n is fixed to achieve a given discretization error, it contributes only as a multiplicative constant to the overall computational complexity.

Under the above considerations and results in Tab. II, the application of the AD-QPCA algorithm to the aforementioned dataset, assuming $n = 8$, requires fault-tolerant quantum hardware capable of executing a circuit with $\mathcal{O}(1.41 \times 10^{10})$ logical qubits, $\mathcal{O}(1.16 \times 10^{14})$ total gate count, $\mathcal{O}(5.80 \times 10^6)$ total depth, $\mathcal{O}(1.99 \times 10^{13})$ T-count, and $\mathcal{O}(2.26 \times 10^6)$ T-depth.

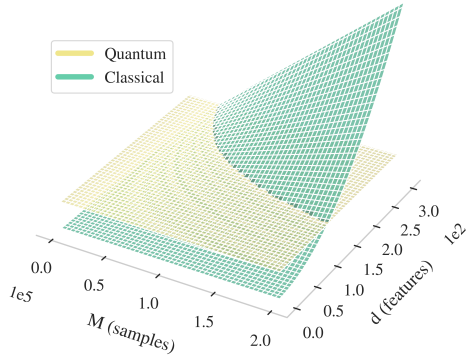

Fig. 9. Asymptotic complexity comparison of the AD-PCA algorithm execution times, between its quantum (circuit depth) and classical implementation, assuming a quantum encoding in n=8 qubits.

For varying problem dimensions, we compare the asymptotic depth of the complete quantum circuit—considered as the quantum running time [37]—to the algorithmic complexity of the classical AD-PCA algorithm [18], which requires $\mathcal{O}(md^2)$ operations. These trends are shown in Fig. 9. The logarithmic scaling of the quantum implementation exhibits an exponential advantage starting from training datasets with approximately 2×10^5 samples and 100 features, or equivalent dataset sizes. However, in addition to these results, one must account for the overhead introduced by quantum compilation and error correction protocols—estimated to increase the cost by at least two orders of magnitude [21]—to approximate the actual execution time on fault-tolerant quantum hardware.

Notably, Tab. II shows that, for state-preparation circuits, the dependence of resource requirements on the dataset dimensions (M, d) arises mainly from the QRAM implementation. The only exception is a logarithmic dependence on the number of qubits, which is required for encoding the indices. Indeed, once the data is loaded, the information is processed in superposition over both indices i and j leading to a dependence only on the approximation and discretization errors. This observation also holds for circuits composed upon the presented state-preparation circuits, as AE operates within the

same superposition regime. We point out that improvements in QRAM design and dedicated hardware devices could further lower the resource requirements [38].

V. Conclusion

We have presented a complete circuit-level design, targeting Clifford+T quantum gate set, of an Anomaly Detection algorithm based on Principal Component Analysis. Our resource analysis shows that the circuit depth scales logarithmically with both the number of samples and features in the input dataset, demonstrating an exponential quantum speed-up over the classical counterpart—assuming efficient data access in superposition via QRAM. We provide a resource estimate based on a real-world bank transaction dataset, representative of a utility-scale fraud detection task. Future research could focus on designing efficient QRAM access schemes tailored for superposition-based queries of the training data matrix. Additionally, extending the framework to integrate kernel methods represents a valuable future direction.

Acknowledgment

We would like to thank all the people involved within the Fraud Detection initiative founded by the *Italian Centro Nazionale di Ricerca in HPC, Big Data e Quantum computing – SPOKE 10*, in particular Riccardo Crupi, Gianbiagio Curato and Davide Corbelletto from Intesa Sanpaolo for their support. The views and opinions expressed are those of the authors and do not necessarily reflect the views of Intesa Sanpaolo, its affiliates or its employees.

References

[1] N. Innan *et al.*, "Financial fraud detection using quantum graph neural networks," *Quantum Machine Intelligence*, vol. 6, no. 1, 2024. doi: 10.1007/s42484-024-00143-6

[2] P. Rebentrost *et al.*, "Quantum Support Vector Machine for Big Data Classification," *Physical Review Letters*, vol. 113, no. 13, 2014. doi: 10.1103/PhysRevLett.113.130503

[3] S. Lloyd *et al.*, "Quantum principal component analysis," *Nature Physics*, vol. 10, no. 9, 2014. doi: 10.1038/nphys3029

[4] A. B. Nassif *et al.*, "Machine Learning for Anomaly Detection: A Systematic Review," *IEEE Access*, vol. 9, 2021. doi: 10.1109/ACCESS.2021.3083060

[5] A. Bellante *et al.*, "Evaluating the potential of quantum machine learning in cybersecurity: A case-study on PCA-based intrusion detection systems," *Computers & Security*, vol. 154, 2025. doi: 10.1016/j.cose.2025.104341

[6] A. Ali *et al.*, "Financial Fraud Detection Based on Machine Learning: A Systematic Literature Review," *Applied Sciences*, vol. 12, no. 19, 2022. doi: 10.3390/app12199637

[7] M. Tabassum *et al.*, "Anomaly-based threat detection in smart health using machine learning," *BMC Medical Informatics and Decision Making*, vol. 24, no. 1, 2024. doi: 10.1186/s12911-024-02760-4

[8] J. Biamonte *et al.*, "Quantum Machine Learning," *Nature*, vol. 549, no. 7671, 2017. doi: 10.1038/nature23474

[9] M. Guo *et al.*, "Quantum algorithms for anomaly detection using amplitude estimation," *Physica A: Statistical Mechanics and its Applications*, vol. 604, 2022. doi: 10.1016/j.physa.2022.127936

[10] G. Brassard *et al.*, "Quantum amplitude amplification and estimation," in *Contemporary Mathematics*, S. J. Lomonaco *et al.*, Eds. American Mathematical Society, 2002, vol. 305.

[11] S. Corli *et al.*, "Quantum machine learning algorithms for anomaly detection: A review," *Future Generation Computer Systems*, vol. 166, 2025. doi: https://doi.org/10.1016/j.future.2024.107632

[12] A. Tudisco *et al.*, "Evaluating the Computational Advantages of the Variational Quantum Circuit Model in Financial Fraud Detection," *IEEE Access*, vol. 12, 2024. doi: 10.1109/ACCESS.2024.3432312

[13] E. Dri *et al.*, "Towards An End-To-End Approach For Quantum Principal Component Analysis," in *IEEE International Conference on Quantum Computing and Engineering (QCE)*, 2023. doi: 10.1109/QCE57702.2023.10175

[14] N. Liu *et al.*, "Quantum machine learning for quantum anomaly detection," *Physical Review A*, vol. 97, no. 4, 2018. doi: 10.1103/PhysRevA.97.042315

[15] Yufeng Kou *et al.*, "Survey of fraud detection techniques," in *IEEE International Conference on Networking, Sensing and Control*, vol. 2, 2004. doi: 10.1109/ICNSC.2004.1297040

[16] N. Innan *et al.*, "Financial Fraud Detection: A Comparative Study of Quantum Machine Learning Models," *International Journal of Quantum Information*, vol. 22, no. 02, 2024. doi: 10.1142/S0219749923500442

[17] J. Stein *et al.*, "Exploring Unsupervised Anomaly Detection with Quantum Boltzmann Machines in Fraud Detection," in *Proceedings of the 16th International Conference on Agents and Artificial Intelligence*, 2024. doi: 10.5220/0012326100003636

[18] H. Hoffmann, "Kernel PCA for novelty detection," *Pattern Recognition*, vol. 40, no. 3, 2007. doi: 10.1016/j.patcog.2006.07.009

[19] M. Schuld *et al.*, *Machine Learning with Quantum Computers*, ser. Quantum Science and Technology. Springer International Publishing, 2021.

[20] D. Weiss *et al.*, "Quantum Random Access Memory Architectures Using 3D Superconducting Cavities," *PRX Quantum*, vol. 5, no. 2, 2024. doi: 10.1103/PRXQuantum.5.020312

[21] O. di Matteo *et al.*, "Fault-Tolerant Resource Estimation of Quantum Random-Access Memories," *IEEE Transactions on Quantum Engineering*, vol. 1, 2020. doi: 10.1109/TQE.2020.2965803

[22] I. Kerenidis *et al.*, "Quantum Recommendation Systems," *LIPIcs, Volume 67, ITCS 2017*, vol. 67, 2017. doi: 10.4230/LIPICS.ITCS.2017.49

[23] G. H. Low *et al.*, "Trading T gates for dirty qubits in state preparation and unitary synthesis," *Quantum*, vol. 8, 2024. doi: 10.22331/q-2024-06-17-1375

[24] S. S. Zhou *et al.*, "Quantum Fourier transform in computational basis," *Quantum Information Processing*, vol. 16, no. 3, p. 82, 2017. doi: 10.1007/s11128-017-1515-0

[25] J.-M. Liang *et al.*, "Quantum anomaly detection with density estimation and multivariate Gaussian distribution," *Physical Review A*, vol. 99, no. 5, 2019. doi: 10.1103/PhysRevA.99.052310

[26] B. Duan *et al.*, "Efficient quantum circuit for singular-value thresholding," *Physical Review A*, vol. 98, no. 1, 2018. doi: 10.1103/PhysRevA.98.012308

[27] I. Cong *et al.*, "Quantum discriminant analysis for dimensionality reduction and classification," *New Journal of Physics*, vol. 18, no. 7, 2016. doi: 10.1088/1367-2630/18/7/073011

[28] S. A. Cuccaro *et al.* (2004) A new quantum ripple-carry addition circuit. doi: 10.48550/arXiv.quant-ph/0410184

[29] A. Shakeel, "Efficient and scalable quantum walk algorithms via the quantum Fourier transform," *Quantum Information Processing*, vol. 19, no. 9, 2020. doi: 10.1007/s11128-020-02834-y

[30] E. Munoz-Coreas *et al.*, "Quantum Circuit Design of a T-count Optimized Integer Multiplier," *IEEE Transactions on Computers*, vol. 68, no. 5, 2019. doi: 10.1109/TC.2018.2882774

[31] P. Selinger, "Quantum circuits of T-depth one," *Physical Review A*, vol. 87, no. 4, 2013. doi: 10.1103/PhysRevA.87.042302

[32] A. Barenco *et al.*, "Elementary gates for quantum computation," *Physical Review A*, vol. 52, no. 5, 1995. doi: 10.1103/PhysRevA.52.3457

[33] P. Selinger. (2014) Efficient Clifford+T approximation of single-qubit operators. doi: 10.48550/arXiv.1212.6253

[34] H. Sayginel *et al.*, "A fault-tolerant variational quantum algorithm with limited T-depth," *Quantum Science and Technology*, vol. 9, no. 1, 2024. doi: 10.1088/2058-9565/ad0571

[35] M. A. Nielsen *et al.*, *Quantum computation and quantum information*, 10th ed. Cambridge university press, 2010. ISBN 978-1-107-00217-3

[36] B. Giles *et al.* (2019) Remarks on Matsumoto and Amano's normal form for single-qubit Clifford+T operators. doi: 10.48550/arXiv.1312.6584

[37] T. G. de Brugiere *et al.*, "Reducing the Depth of Linear Reversible Quantum Circuits," *IEEE Transactions on Quantum Engineering*, vol. 2, 2021. doi: 10.1109/TQE.2021.3091648

[38] F. Cesa *et al.* (2025) Fast and Error-Correctable Quantum RAM. doi: 10.48550/arXiv.2503.19172