Quantum Data Compression for Efficient Generation of Control Pulses

Daniel Volya and Prabhat Mishra
University of Florida, Gainesville, Florida, USA

ABSTRACT

In order to physically realize a robust quantum gate, a specifically tailored laser pulse needs to be derived via strategies such as quantum optimal control. Unfortunately, such strategies face exponential complexity with quantum system size and become infeasible even for moderate-sized quantum circuits. In this paper, we propose an automated framework for effective utilization of these quantum resources. Specifically, this paper makes three important contributions. First, we utilize an effective combination of register compression and dimensionality reduction to reduce the area of a quantum circuit. Next, due to the properties of an autoencoder, the compressed gates produced are robust even in the presence of noise. Finally, our proposed compression reduces the computation time of quantum control. Experimental evaluation using popular quantum algorithms demonstrates that our proposed approach can enable efficient generation of noise-resilient control pulses while state-of-the-art fails to handle large-scale quantum systems.

1 INTRODUCTION

Quantum technologies offer promising advantages over classical counterparts in a variety of tasks, including faster computation, secure communication, and high-precision sensors [1, 2]. However, they depend on quantum resources such as quantum coherence, which can be challenging to implement in practice [3, 4]. Due to the scarcity of quantum resources, strategies to minimize the use of these resources are vital for widespread adoption of quantum computing.

One promising strategy to optimize the usage of quantum resources is quantum optimal control (QOC). Generally, quantum computation and algorithms are described in terms of quantum circuits composed of discrete gates. In order to implement quantum gates, control pulses, which represent classical signals, such as specifically tailored laser pulses, need to be computed to optimize for fidelity and coherence. QOC techniques, such as GRadient Ascent Pulse Engineering (GRAPE) [5] uncover optimal pulses for a given system model and have been employed to robustly implement gates and algorithms [6]. However, the time complexity of these techniques scale exponentially with system size and become costly even for a couple of qudits.

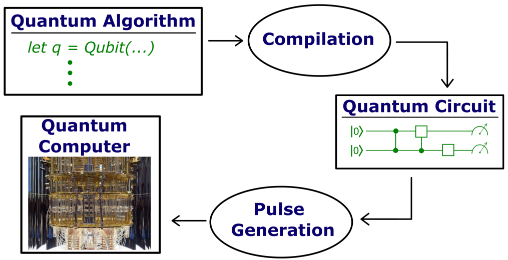


Figure 1: An overview of the general framework used to realize a quantum algorithm on a quantum computer.

Quantum compression has been used as a promising strategy to reduce the dimension of quantum gates. Since quantum gates are reversible, and therefore lossless, compression is achieved if the set of possible states does not span the full Hilbert space in the original encoding. Methods of compressing quantum data have been successfully used earlier [7, 8], but rely on particular assumptions about the properties of the quantum states. Alternatively, recent strategies propose the usage of autoencoders – utilizing machine learning to represent data in a lower dimension. Autoencoders do not rely on any prior assumptions of the types of quantum states. Instead of requiring a fixed structure of data, they can learn the structure based on the given dataset.

Figure 1 shows an overview of realizing a quantum algorithm on a quantum computer. A quantum algorithm, specified in terms of generic quantum operations, is first compiled to a lower-level quantum circuit that is tailored to device-specific constraints such as implementable quantum gates and gate connectivity. A pulse is computed for each of the implementable quantum gates, and is then sent to a quantum device to perform the computation. As shown in Figure 2, computation of control pulses using GRAPE [5] can be infeasible for larger gates or complex quantum algorithms. Therefore, state-of-the-art pulse generator (GRAPE) will significantly constrain the set of implementable quantum gates, which can lead to sub-optimal usage of quantum resources [9].

In this paper, we propose autoencoder-based compression to enable pulse exploration for larger quantum gates. The autoencoder can be trained on a few examples of qudit gates and subsequently tested with qudits from the same family. We use a classical machine learning algorithm, gradient descent, to optimize unitary transformations for compressing quantum states. This paper makes the following major contributions:

- Enables Hilbert space reduction using a combination of dimensionality reduction and register compression.
- Generation of training data by utilizing Clifford circuits, which implicitly contains noise-related information.

This work was partially supported by the NSF grant CCF-1908131.

ASPDAC '23, January 16–19, 2023, Tokyo, Japan
© 2023 Association for Computing Machinery.
ACM ISBN 978-1-4503-9783-4/23/01...$15.00
https://doi.org/10.1145/3566097.3567927

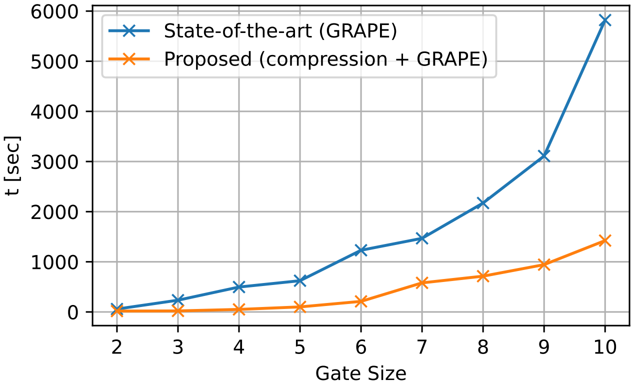


Figure 2: Pulse generation time increases exponentially with varying size using GPU-based GRAPE implementation, while pulse generation after compression is beneficial.

- Our proposed quantum compression leads to noise reduction via the inherent properties of autoencoders.
- Proposed compression leads to faster pulse generation for larger quantum gates or sequences of gates.

The remainder of the paper is organized as follows. Section 2 surveys related efforts. Section 3 describes our proposed quantum data compression framework. Section 4 presents the experimental results. Finally, Section 5 concludes the paper.

2 BACKGROUND AND RELATED WORK

2.1 Quantum Optimal Control

Qudits, the basic units of quantum computers, are experimentally realized using some underlying engineered technology. The engineered product, conceptually, exposes an effective Hamiltonian $H(t)$. Commonly, this Hamiltonian can be separated into two parts: a time-independent Hamlitonian H_d, and a set of Hamiltonians H_c with a controllable pulse $a(t)$, such that $H = H_d + \sum_i a_i(t)H_{ci}$. For example, a superconducting transmon has effectively two control Hamiltonians, which in the qubit case, can be approximated as the Pauli-matrices σ_x and σ_y. The field of quantum control addresses the underlying question of what pulses $a(t)$ to use in order to obtain a desired time evolution that will implement a given quantum gate.

A common strategy to achieve a desired U is to employ the optimal quantum control technique GRAPE [5]. Given a system's parameters H_d and set H_c, GRAPE will simulate Shrödinger's equation for an initial guess of the discretized pulses $a_i(t) = a_{ij}$, which will result in a unitary operator U_f. The error, F, between U and U_f is computed and the gradient $\nabla F(a_{ij})$ is approximated. Then, an optimizer attempts to follow the gradient and find a_{ij} that minimizes the error F. The computational cost is dependent on the simulation of a quantum system and the optimizer – both of which are dependent on the dimensions of the Hilbert space. Therefore, lowering the dimensionality of the Hilbert space will improve the computational cost of optimal quantum control techniques.

2.2 Autoencoders

Autoencoders are special type of neural-network. They are meant to take input data, compress it into a lower dimensionality representation – the bottleneck layer, also known as the latent space. Autoencoders then attempt to reconstruct the original data using only the latent-space representation. The output is compared directly to the input, meaning labeled data is unnecessary. Autoencoders have a wide variety of uses, including sound-generation, outlier detection, and dimensionality reduction. It is common to only use half of the autoencoder after training is complete. The first half is known as the encoder, and the second half is known as the decoder.

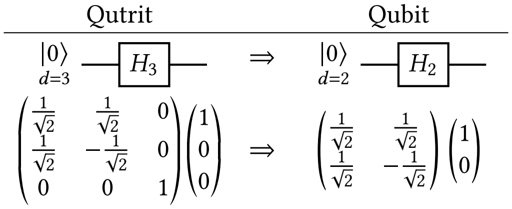

Qutrit
Qubit
$|0\rangle$
$d{=}3$
H_3
$\Rightarrow$
$|0\rangle$
$d{=}2$
H_2
$\begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix}1\\0\\0\end{pmatrix} \Rightarrow \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix}\begin{pmatrix}1\\0\end{pmatrix}$

Figure 3: Example qutrit gate (H_3) that can be trivially compressed into a qubit gate (H_2).

2.3 Related Work

Speeding up quantum optimal control algorithms, especially on real-time systems, is a growing research area. Methods such as automatic differentiation, and acceleration by general-purpose graphical processing units have been employed to speed up GRAPE calculations [10]. Strategies, such as Chopped RAndom Basis (CRAB) [11, 12], have been shown to be successful in targeting a constrained many-body quantum system by transforming functional minimization to a multi-variable function minimization. Alternative strategies, such as quantum control via deep reinforcement learning [13] have been introduced as a way to produce pulses model-free and simulation-free with greater accuracy. Compared to optimal control techniques, the learning based methods lack theoretical guarantees of an optimal solution, and are often limited by the dataset.

The field of quantum data compression aims to optimize the usage of precious quantum resources, such as the ones in Noisy-Intermediate Scale Quantum (NISQ) computers. Methods based on genetic algorithms [14], or autoencoders for compression of qudit dimension [15] and gate size [16] have been experimentally and theoretically proposed as effective strategies for compression. Strategies such as parameterized quantum circuits [17] seek to limit the types of quantum gates but add a set of parameters. Our approach generalizes qubits to arbitrary quantum dimension (qudits) and works on any register size. Our approach is the first attempt in utilizing quantum compression of quantum registers.

3 QUANTUM DATA COMPRESSION

To mitigate the cost of quantum pulse generation, while maintaining the advantages of optimizing quantum resources, we propose a strategy which seeks to compress quantum gates – enabling faster pulse generation on a reduced space. Figure 4 visualizes our strategy that consists of three important steps. First, we transform the quantum circuit into an algebraic formulation. Next, we compress the design using dimensionality reduction as well as register compression. Finally, the control pulses are generated for the compressed gates. A major contribution of our work is to develop efficient quantum data compression. While dimensionality reduction transforms qudits to qunits, register compression reduces the number of qunits required to represent the functionality. This presents interesting choices to figure out which of the following is most beneficial: (i) dimensionality reduction only, (ii) register compression only, (iii) dimenstionality reduction followed by register compression, and (iv) register compression followed by dimensionality reduction.

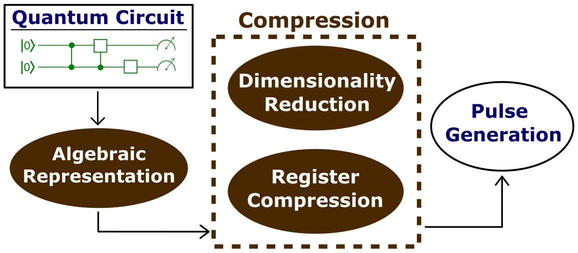


Figure 4: An overview of our proposed strategy of compressing quantum gates before pulse generation. It consists of three major tasks: algebraic representation, dimensionality reduction, and register compression.

In this section, we outline the representation of a quantum gate used as inputs and outputs for an autoencoder. We then describe the procedure for quantum register compression and gate dimensionality compression.

3.1 Representation of Quantum Circuits

We use the coordinate representation of a quantum gate in the basis spanned by the generalized Gell-Mann matrices. Particularly, a quantum operation represented as a $d \times d$ unitary matrix is expressed using $d^2 - 1$ coordinates of the $su(n)$ Lie algebra basis. Doing so comes with a few conveniences and advantages over working directly with unitary operators: the coordinates L_i are strictly real by choosing the appropriate $su(n)$ basis, the algebraic properties of these coordinates are well-defined, and the non-linear optimizer for pulse-finding can readily ignore the global phase. To represent a quantum gate U, we first map it to the special unitary matrix S. We then exclusively work with the Lie algebra representation of S, namely:

$$s = \log S \tag{1}$$

Example 1: Consider the matrix representations of H_3 as shown in Figure 3. Through Equation 1, the matrix H_3 is represented in the Lie algebra as:

$$s_3 = \begin{pmatrix} -0.587i & -1.111i & 0 \\ -1.111i & 1.634i & 0 \\ 0 & 0 & -1.047i \end{pmatrix}$$

As a coordinate in the $su(3)$ basis spanned by the generalized Gell-mann matrices, s_3 is

$$\vec{L}_3 = \begin{bmatrix} -1.111 & 0 & 0 & -1.111 & 0.907 & 0 & 0 & 0 \end{bmatrix}^T.$$

Similarly, $s_2 = \begin{pmatrix} -1.111i & -1.111i \\ -1.111i & 1.111i \end{pmatrix}$ with the coordinates $\vec{L}_2 = \begin{bmatrix} -1.111 & 0 & -1.111 \end{bmatrix}^T$. ■

The layout of our autoencoder is as follows. The input layer consists of $d^2 - 1$ nodes, representing a quantum gate in the $su(d)$ Lie algebra. The input layer is then fully connected to n^2 nodes, representing the $su(n)$ Lie algebra using $n^2 - 1$ nodes, and an extra 1-node that holds information about the compression between $su(d) \leftrightarrow su(n)$. The cost function includes the mean-squared error (MSE) between the output layer and the original real vector of the gate. Additionally, an optional cost function is included to ensure that the latent space of the autoencoder forms understandable vectors in the compressed $su(n)$ space. Namely, we may hand-compute the $su(n)$ compressed representation and include the MSE between the latent-space and the hand-computed result. This allows for a quick understanding of the encoded space.

3.2 Qudit Pauli Groups and Clifford groups

A circuit that consists of only Clifford gates can be perfectly and efficiently simulated by a classical computer as given by the Gottesman-Knill theorem. In this section we summarize the notation for qudit Clifford groups.

The Pauli group for qubits is defined via the Pauli operators and identity: $\{I, \sigma_x, \sigma_y, \sigma_z\}$. To generalize the Pauli group to qudits, we need to define *clock* and *shift* operators

$$X = \sum_{j=0}^{d-1} |j\rangle \langle j+1|, \quad Z = \sum_{j=0}^{d-1} \omega^j |j\rangle \langle j| \tag{2}$$

where $\omega = e^{2\pi i/d}$ is the root of unity. The operators generalize σ_x and σ_z in the qubit case, and $X^d = Z^d = I$. For n-qudits, the operation acting on the i-th qubit is denoted with a subscript. A Pauli product is defined as

$$\omega^\lambda X^{\vec{x}} Z^{\vec{z}} = \omega^\lambda X_0^{x_0} Z_0^{z_0} \otimes X_1^{x_1} Z_1^{z_1} \otimes \cdots \otimes X_n^{x_n} Z_n^{z_n} \tag{3}$$

where λ is part of $\mathbb{R}_d$ and $\vec{x}$ and $\vec{z}$ are tuples of length n in $\mathbb{Z}_d^n$ – each element an integer mod d.

For a fixed n, the Pauli group $\mathcal{P}_n$ is defined by all possible Pauli products. For example, the Pauli group of a single qutrit ($n = 1$, $d = 3$) is given by $\omega^\lambda X^i Z^j = \omega^\lambda [I, Z, ZZ, XZ, XZZ, XXZ, XXZZ]$ for $i, j \in \mathbb{R}_3$, and will have a total of 21 elements due to the possible options for λ. A Clifford operation C acts on an element $p_1 \in \mathcal{P}_n$ such that under conjugation it returns another member of the Pauli group: $Cp_1C^\dagger = p_2$. All Pauli products are Clifford operations. A Clifford gate can be viewed by a tableau of its action on the generators X and Z[18].

3.3 Dimensionality Reduction

Qudit dimensionality compression seeks to take a d-dimensional qudit and compress it to a n-dimensional qunit where $d > n$. Doing so successfully yields advantages in two ways. First, as discussed in Section 2, the physical implementation of a quantum system generally yields a Hilbert space that is larger than desired. Being able to compress the physical Hilbert space closer to the desired size would decrease the computational cost for calculating control signals. Second, a quantum

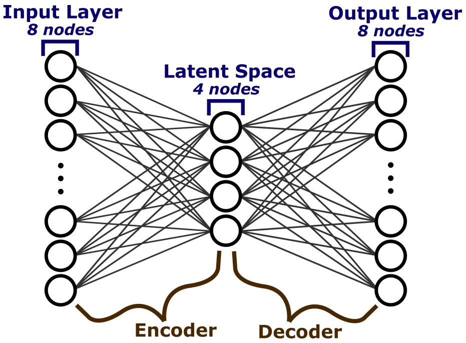


Figure 5: Example autoencoder layout for compressing a qutrit to a qubit. The input and output layers consist of 8 nodes, representing the qutrit gate in the $su(3)$ algebra. The latent space represents the encoding as a qubit gate in the $su(2)$ algebra, plus an extra component used for compression.

algorithm may be originally described in a qudit space, and without compression, may over-utilize the quantum resources.

Example 2: Consider an example where a quantum gate H_3 operates on a qutrit, as shown in Figure 3. H_3 only operates on the $|0\rangle$ and $|1\rangle$ states, while leaving $|2\rangle$ constant. Namely, a non-trivial operation occurs only in a qubit subspace. Hence, we can compress without any loss the H_3 qutrit gate to a qubit gate H_2. It should be noted, that in general, the subspaces may not be immediately apparent – an encoding may be formed by a superposition of states $|0\rangle$, $|1\rangle$, and $|2\rangle$ and compressed to $|0\rangle$, $|1\rangle$. Moreover, noise introduces additional ambiguity: for example, H_3 may operate non-trivially on $|2\rangle$ in the presence of noise. Figure 5 shows the layout of an autoencoder to automatically compress the qutrit gate to a qubit gate. Autoencoders can successfully compress arbitrary gates, even in the presence of noise. ■

3.4 Register Compression

Qudit register compression aims to reduce the number of qudits required for a quantum operation. Namely, given a quantum gate U that operates on n-qudits, qudit register compression seeks to create a gate U' that operates on m-qudits, where $m < n$. Successful compression allows for a quantum algorithm to use fewer qudits, enabling faster pulse generation. Additionally, this can open doors to alternative usage of the spare qudits, e.g. for quantum error correction.

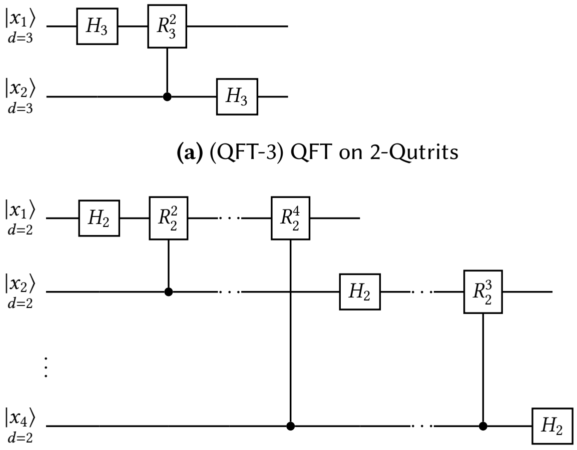
Example 3: Consider two qutrits operated on by a $3^2 \times 3^2$ quantum gate. Namely, the gate is represented as a vector $\vec{L}$ in the $su(3^2)$-algebra, containing $(3^2)^2 - 1$ elements. To only use one qutrit (to invoke register compression) the quantum gate must be represented as a 3×3 quantum gate – a vector in $su(3)$-algebra. In essence, register compression is a special case of dimensionality reduction where the encoding size is a lower power of the original size. ■

3.5 Clifford-based Model Training

Generation of a suitable training dataset is a non-trivial problem. On the one hand, a simulation of a quantum circuit needs to be performed to acquire the true noise-less result. On the other hand, quantum circuits composed of many qubits with significant circuit depth are infeasible to simulate on a classical computer. Our approach is to generate a training dataset by using quantum circuits that are largely composed of Clifford gates, and hence are efficient to simulate on a classical computer. A straight-forward approach is to choose a set of random state $\{|\psi_i\rangle\}$, generated by appropriate quantum circuits [19]. A subset of gates in the circuit are replaced with Clifford gates that are close in distance to the original gates. The circuits are then efficiently simulated on a classical computer to generate a noiseless dataset[20]. In additional, the same circuits are executed on a quantum computer to generate a noisy dataset. As a result, the dataset $L = \{P_i^{\text{exact}}, P_i^{\text{noisy}}\}$ represents the exact result from simulation in conjunction with the actual results from the quantum computer. The dataset is used to train the autoencoder where the ideal gate/circuit is given as input and the final noisy result (after tomography) is used to calculate the error of the autoencoder during training.

4 EXPERIMENTS

This section is organized as follows. First, we describe the utilized quantum algorithms (benchmarks) and evaluation framework (superconducting Transmon in IBMQ). Next, we present our experimental results in terms of improvement in area and computation (pulse generation) time.



(a) (QFT-3) QFT on 2-Qutrits

(b) (QFT-2) QFT on 4-Qubits

Figure 6: QFT circuits for (a) qutrits and (b) qubits.

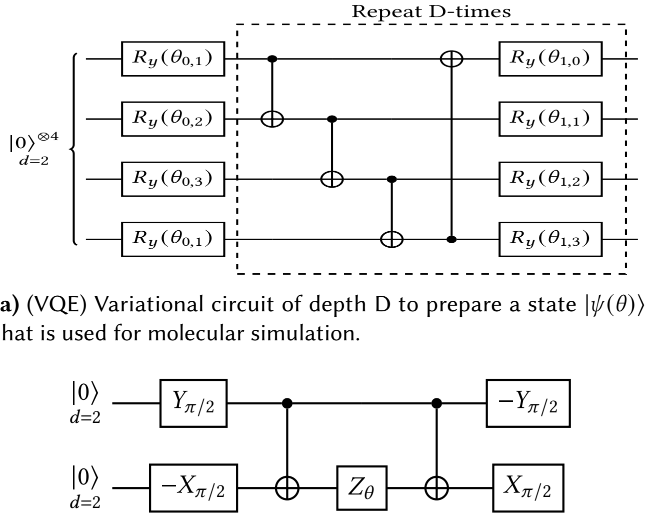
4.1 Quantum Algorithms

We use the quantum Fourier transform and the quantum variational eigensolver as benchmarks for our approach. These two algorithms are fundamental, and serve as subroutines for a variety of other quantum algorithms.

4.1.1 *Quantum Fourier Transform.*

The quantum Fourier transform (QFT) algorithm is at the heart of many quantum algorithms including Quantum Phase Estimation and Shor's Algorithm. The net result of QFT maps the computational basis $\{|0\rangle, |1\rangle, \ldots, |n-1\rangle\}$ to

$$QFT\,|x\rangle \rightarrow \frac{1}{\sqrt{d^n}} \sum_{k=0}^{d^n-1} e^{2\pi ixk/d^n}\,|k\rangle \qquad (4)$$



(a) (VQE) Variational circuit of depth D to prepare a state $|\psi(\theta)\rangle$ that is used for molecular simulation.

(b) (VQE-OPT) An optimized variational circuit to prepare a state $|\psi(\theta)\rangle$ for a constrained class of molecular simulation.

Figure 7: Parametrized quantum circuits for hydrogen-hydrogen VQE algorithm.

where d is the dimension of qudits, and n is the number of qudits. Here, the integer x is expanded in base-d.

Example 4: Let $x = 8$. Then with qutrits, $|x\rangle$ is written as $|22\rangle$, while with qubits $|x\rangle$ is written as $|1000\rangle$. In general, the differences in the number of qudits required scales as

$$\left(\left\lfloor\frac{1}{\log 2}\right\rfloor - \left\lfloor\frac{1}{\log 3}\right\rfloor\right)\lfloor\log(x)\rfloor \approx O(\log x) \tag{5}$$

Figure 6 shows the quantum circuits to perform QFT using qutrits and qubits for $0 \leq x \leq 8$. A guaranteed lossless compression can be achieved by constraining the possible ranges of x. For example, $0 \leq x \leq 7$ would allow Figure 6b to only require 3 qubits rather than 4. ■

4.1.2 Variational Quantum Eigensolver. The Variational Quantum Eigensolver (VQE) is a heuristic-driven algorithm targeting NISQ devices. The core task is to solve for the ground state of any molecular Hamiltonian $\hat{H}$ by preparing a parametrized wave function ansatz $|\psi(\theta)\rangle$ on a quantum computer and adopt classical optimization methods to adjust the parameters θ to minimize the expectation value $\langle\psi(\theta)|\,\hat{H}\,|\psi(\theta)\rangle$.

Example 5: Let the Hamiltonian $\hat{H}$ represent two hydrogen atoms with a given interatomic distance in the fermion basis. The quantum circuit needs to generate a form of $|\psi(\theta)\rangle$ that include the minimal solution for $\langle\psi(\theta)|\,\hat{H}\,|\psi(\theta)\rangle$. One method is to represent states in the Hartree-Fock basis, resulting in a circuit that generates full-entanglement of 4-qubits, as shown in Figure 7a. An optimized version is shown in Figure 7b. ■

4.2 Superconducting Transmon

In this section, we first provide a brief overview of modeling transmon qubits. Next, we fit a model for a superconducting transmon. The drift and control Hamiltonians from the fitted model are used for optimal quantum control. A transmon has energy levels $E_0, E_1, \ldots, E_N$ corresponding to states $|0\rangle, |1\rangle, \ldots, |E_N\rangle$. More importantly, transition energies can be directly obtained from experiments and are generally expressed in terms of frequency (proportional to 2π): $w_{01} \propto E_1 - E_0$; $w_{12} \propto E_2 - E_1$; and so on. By knowing the transition frequencies, the parameters of the transmon – the control and drift Hamiltonians – can be sufficiently approximated. The accuracy of the approximated Hamiltonians depends on the number of transition frequencies used in the approximation – the more the better.

We find the first three transition frequencies, $\omega_{01}, \omega_{12}, \omega_{23}$ of a superconducting transmon via Rabi experiments and Ramsey experiments. The measured transitions are then used to fit E_C, E_J and n_g of a transmon Hamiltonian:

$$H = 4E_C(\hat{n} - n_g)^2 - E_J \cos\hat{\phi}. \tag{6}$$

Due to the system being a transmon, n_g is close to 0. An initial approximation is made by using the first two transitions frequencies, namely via

$$E_C = \omega_{01} - \omega_{12} \tag{7}$$

$$\omega_{01} = \sqrt{8E_J E_C} - E_C \Rightarrow E_J = \frac{(\omega_{01} + E_C)^2}{8E_C} \tag{8}$$

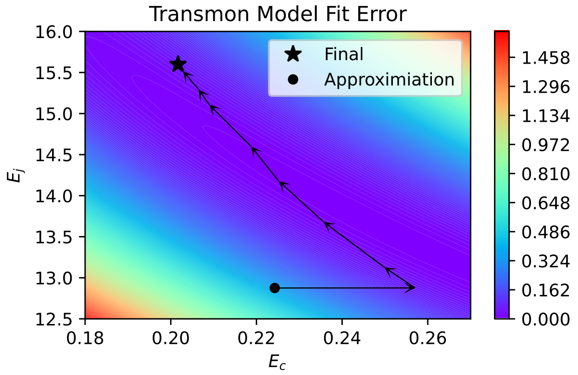


Figure 8: Error landscape with respect to E_C and E_J

The initial approximation is then improved by optimizing for the eigenvalues with respect to parameters E_C and E_J as shown in Figure 8.

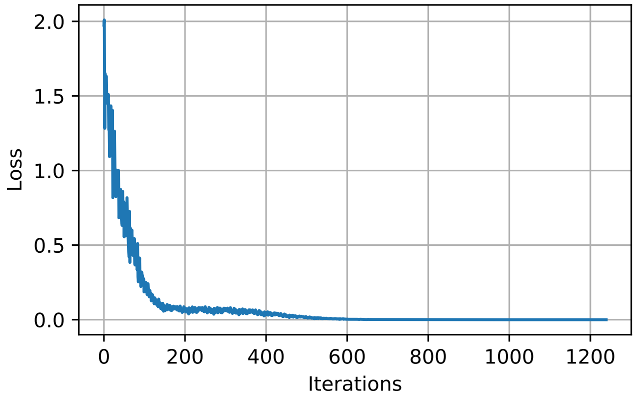


Figure 9: Autoencoder training loss for qutrit to qubit compression. Loss of zero indicates lossless compression.

4.3 Improvement in Area

The area is computed as a product of (a) the total number of gates, (b) the size of the quantum register, and (c) the dimensionality of the qudits. In our case, the QFT and VQE algorithms use qudits with dimensionality of either two or three. Figure 9 shows the loss function of the autoencoder, where each iteration starts with a random state, then takes the measurement results from a quantum gate acting on the random state. In other words, the loss function goes to zero, indicating that the quantum gate is compressible.The total improvement factor of the area, by using our approach, is shown in Figure 10. The compression ratio is computed between the original and compressed version of the quantum circuit. Compression was not possible for the optimized version of VQE (VQE-opt) due to circuit already being hand-optimized for Hydrogen-Hydrogen molecular simulation. Our proposed compression provided more than 2.5 times reduction in area for the original VQE.

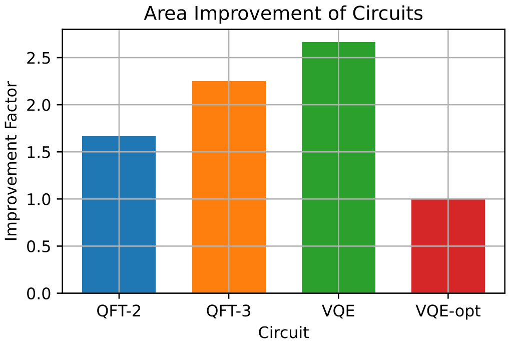


Figure 10: Improvement in area (number of gates and qudit dimension) for varying circuits. Our approch can lead to significant area reduction (up to 2.5 times for VQE).

4.4 Improvement in Pulse Generation Time

The computation time for pulse generation and compression is taken for each circuit and compared against the uncompressed version. Figure 11 shows the improvement factor in time, taken as a ratio between original and compressed versions. Since the measurements are based on realistic and fixed gates, the improvements are weaker than those shown in Figure 2 which used random gates. In this case, the pulses for each fixed gate are computed separately, which yields a linear scaling in total time. The largest improvement was obtained by compressing qutrits into qubits – which resulted in a factor of seven improvement in time, as shown for QFT-3.

5 CONCLUSION

Quantum optimal control is an effective strategy to optimize the usage of quantum resources. Specifically for NISQ systems, it is beneficial to control quantum computers at a continuous pulse level, rather than imposing the constraint of a discrete set of operators. Unfortunately, methods in quantum optimal control scale exponentially with the quantum state-space size. We have proposed a strategy to compress quantum data via a classical autoencoder, effectively reducing the state-space, which allows for faster pulse generation in the encoded space. The compression results in significant reduction in both the dimensionality of qudits and the number of qudits in a quantum register. This work opens a pathway to bringing the benefits of quantum optimal control techniques to real-time devices as well as larger quantum systems.

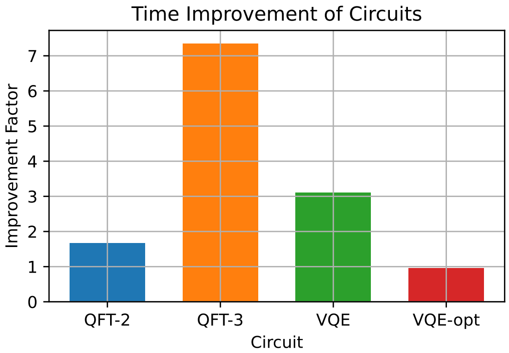


Figure 11: Improvement in pulse computation time for different circuits (up to seven times for QFT-3).

REFERENCES

[1] Alwin Zulehner and Robert Wille. *Introducing Design Automation for Quantum Computing*. Springer, 2020.

[2] Daniel Volya and Prabhat Mishra. Quantum spectral clustering of mixed graphs. In *Design Automation Conference (DAC)*, pages 463–468, 2021.

[3] Daniel Volya and Prabhat Mishra. Impact of noise on quantum algorithms in noisy intermediate-scale quantum systems. In *ICCD*, 2020.

[4] Daniel Volya and Prabhat Mishra. Modeling of noisy quantum circuits using random matrix theory. In *ICCD*, 2022.

[5] Navin Khaneja et al. Optimal control of coupled spin dynamics: Design of NMR pulse sequences by gradient ascent algorithms. *JMR*, 2005.

[6] Eric Holland et al. Optimal control for the quantum simulation of nuclear dynamics. *Physical Review A*, 101(6), June 2020.

[7] Lee Rozema et al. Quantum Data Compression of a Qubit Ensemble. *Physical Review Letters*, 113(16), October 2014.

[8] Lucie Bartuskova et al. Optical implementation of the encoding of two qubits to a single qutrit. *Physical Review A*, 74(2), August 2006.

[9] Pranav Gokhale et al. Optimized Quantum Compilation for Near-Term Algorithms with OpenPulse. *arXiv:2004.11205*, May 2020.

[10] Mohamed Abdelhafez et al. Gradient-based optimal control of open quantum systems using quantum trajectories and automatic differentiation. *Physical Review A*, 99(5):052327, May 2019.

[11] Patrick Doria et al. Optimal Control Technique for Many-Body Quantum Dynamics. *Physical Review Letters*, 106(19), May 2011.

[12] Tommaso Caneva et al. Chopped random-basis quantum optimization. *Physical Review A*, 84(2), August 2011.

[13] Murphy Niu et al. Universal quantum control through deep reinforcement learning. *npj Quantum Information*, 5(1), April 2019.

[14] L. Lamata et al. Quantum autoencoders via quantum adders with genetic algorithms. *Quantum Science and Technology*, 4(1), October 2018.

[15] Alex Pepper et al. Experimental Realization of a Quantum Autoencoder: The Compression of Qutrits via Machine Learning. *PRL*, 122(6), 2019.

[16] Jonathan Romero et al. Quantum autoencoders for efficient compression of quantum data. *Quantum Science and Technology*, 2017.

[17] Marcello Benedetti et al. Parameterized quantum circuits as machine learning models. *Quantum Science and Technology*, November 2019.

[18] Scott Aaronson and Daniel Gottesman. Improved Simulation of Stabilizer Circuits. *Physical Review A*, 70(5), November 2004.

[19] Piotr Czarnik et al. Error mitigation with Clifford quantum-circuit data. *Quantum*, 5, November 2021.

[20] Sergey Bravyi et al. Simulation of quantum circuits by low-rank stabilizer decompositions. *Quantum*, 3:181, 2019.