CPEN400Q - Quantum Computing

VGQEC

Variational Graphical Quantum Error Correction Codes

Group 8
Authors:
Kaushik Kolla, 74167503
Richard Nguyen, 74693235
Lina Bekken Babusiaux, 33299223

Date:
13th April 2025

Table of Contents

1 Introduction

1.1 Background theory

Quantum computing promises exponential speedups in solving certain classes of problems, but its practical realization is hindered by extreme sensitivity to noise. This poses a significant barrier to the development of large-scale, fault-tolerant quantum computers, as noise usually results in qubits losing their delicate quantum state. Unlike classical bits, which can be copied and refreshed, quantum states cannot be cloned (due to the no-cloning theorem) and are lost if disturbed.

Quantum noise can arise from various sources including thermal fluctuations, electromagnetic interference, imperfections in quantum gates, and interactions with the environment to name a few. In all cases, the final state of the quantum computer is not precisely what one expects. Fortunately, a beautiful technique known as Quantum Error Correction (QEC) is imperative to protect fragile quantum states, reverse the action of the noise, and recover as much of the encoded quantum information as possible.

Classical error correction uses redundancy to detect and correct bit flips, but quantum error correction must protect both bit and phase information while preserving superposition and entanglement. One of the simplest strategies is to introduce redundant information via the repetition code which detects and corrects only one type of error by majority voting. However, it requires significant physical resources and is unable to simultaneously correct bit-flip and phase-flip errors. More powerful error correction strategies like the [[5,1,3]] code [4], encodes one logical qubit into five physical qubits and is the smallest known QEC capable of correcting any arbitrary single-qubit error. This is achieved using a set of four stabilizer generators, built from carefully chosen Pauli operators to form a syndrome measurement circuit. Each possible single-qubit Pauli error produces a unique 4-bit syndrome, which can be mapped to a specific correction operation.

A modern approach to error correction that is quickly gaining popularity is the use of variational quantum circuits (VQC). This approach combines classical optimization with parameterized quantum circuits to tailor error correction strategies to specific noise models, expressed as trainable unitary circuits. These circuits are then optimized using a classical optimizer to maximize fidelity between the initial logical state and the recovered output.

Variational Graphical Quantum Error Correction (VGQEC) is one such hybrid approach that extends traditional stabilizer codes by inserting tunable gates and auxiliary qubits which can extract parity information without collapsing the logical state. Unlike hard-coded syndrome tables, VGQEC enables the code structure to evolve during optimization, potentially outperforming classical decoders on real noisy devices.

1.2 Variational Graphical Quantum Error Correction

A new framework from a paper called "Variational Graphical Quantum Error Correction (VGQEC) codes" uses a graphical language called "Quon Language" to represent and adapt error correction codes [1]. By embedding tunable parameters into graphical representations, the VGQEC approach can morph between known error-correcting codes to tailor performance for specific noise environments.

The main challenge is to physically implement the complex, parameter-dependent encoding and recovery maps associated with VGQEC on noisy intermediate-scale quantum (NISQ) era devices. The paper proposes a hybrid quantum-classical scheme using parameterized quantum circuits that take advantage of the original known encoding circuit for a base code, an additional parametrized circuit denoted as U_ε that encodes the modifications of the graphical structure G_ε shown in figure 1 (a). Similarly, the recovery map is built from a combination of the original known recovery scheme R_c and a parameterized circuit U_R shown in figure 1 (b). The auxiliary qubits are used to extend the degrees of freedom.

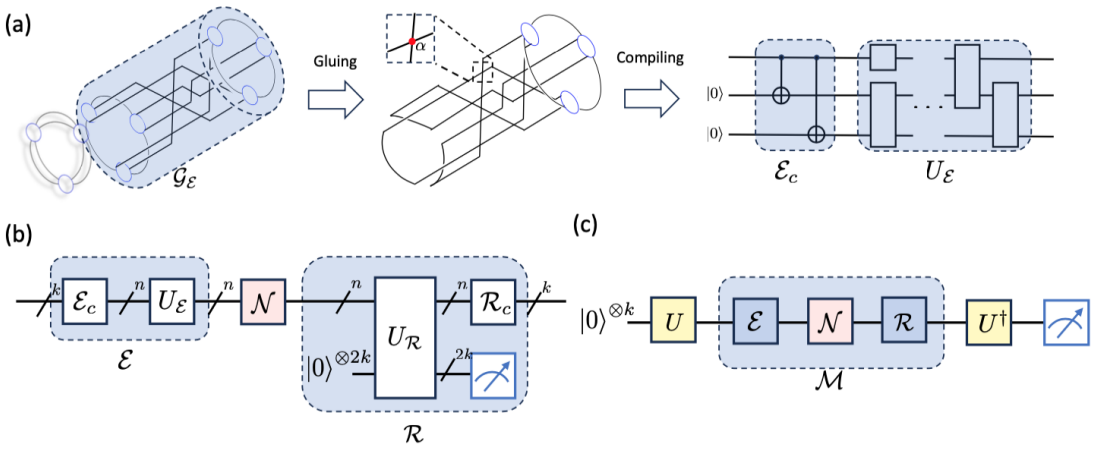


Figure 1: Hybrid Quantum-Classical Scheme Schematic: (a) Outlines how variable graphical structure G_ε is glued to existing Quon graph. Overall encoding map compiled from original encoding circuit ε_c and additional parameterized circuit U_ε. (b) Depicts how R is implemented with U_R together with original recovery circuit R_c using auxiliary qubits. (c) Shows fidelity sampling circuit in overall noise channel M is assessed using 2-design unitary U. Image taken from [1].

The overall performance, called fidelity, of the protected channel is estimated by running a fidelity sampling circuit shown in figure 1 (c). This circuit utilizes random unitaries taken from a 2-design. The measured outcome, the probability of obtaining an all-zero result, is used as feedback in a classical optimization loop to adjust the parameters. This hybrid approach makes the procedure scalable and adaptable to changing noise models without having to redesign error-correcting codes from scratch.

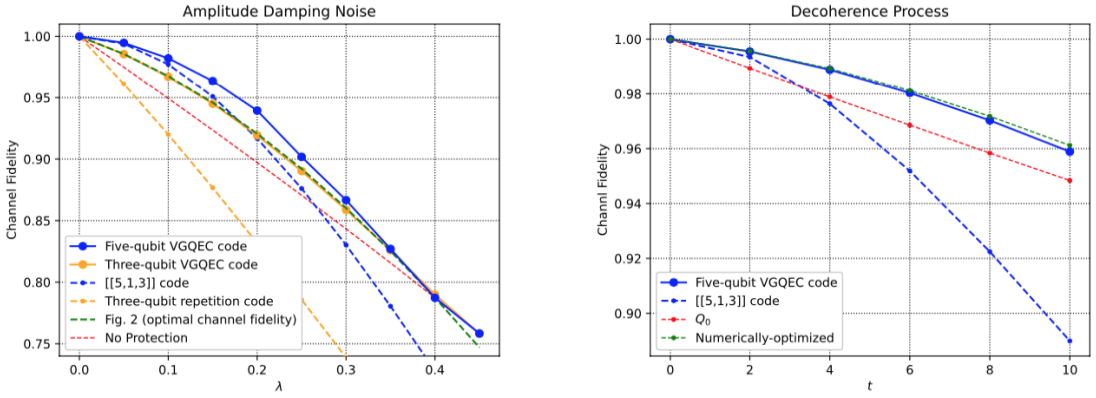


Figure 2: Numerical Performance of VGQEC Codes: (a) Compares channel fidelity of the three-qubit VGQEC code against other codes as a function of amplitude damping parameter λ. (b) Compares performace of five-qubit VGQEC code against other code under asymmetric noise model thermal relaxation error over time t. Image taken from [1].

The results show that VGQEC can significantly outperform non-adaptive codes under noise conditions shown in figure 2, but there are still some remaining challenges and future work discussed in the paper. Optimal strategies for embedding parameters into graphs remain an open problem. Although heuristics work well, a deeper theoretical understanding is needed. The construction of recovery maps still requires further research, especially to reduce computational costs. Future extensions might involve modifying the encoding maps to simplify the physical circuit implementation and reduce resource overhead.

2 Method

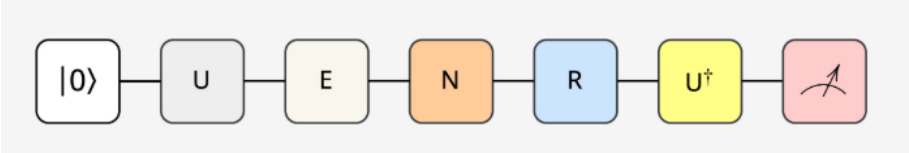


Figure 3: Illustration of our implemented circuit.

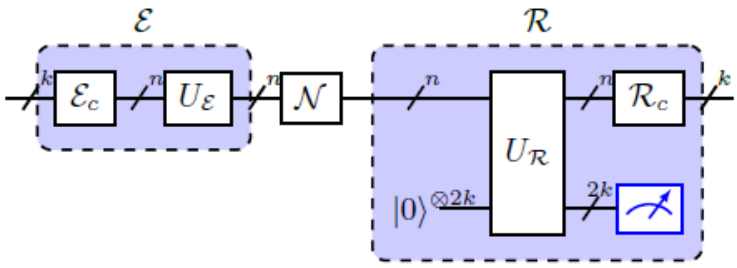


Figure 4: A more detailed illustration of the encoding and recovery part of the circuit. k is the number of logical qubits, and n the number of physical qubits. Image taken from [1].

2.1 Scope and Limitations

The implementation of the VGQEC framework is subject to a set of practical constraints and considerations that define the scope and limitations of this project.

We are implementing the 3-qubit VGQEC, with the original error correction code being the 3-qubit repetition code, and the 5-qubit VGQEC, with the original error correction code being the 5-qubit repetition code code. We only explore error correction on 1 logical qubit, Figures 3 and 4 give an overview of the circuit we implemented.

To compare the performance of our VGQEC with other error correction methods, we implemented the 3-qubit repetition code, the 5-qubit repetition code as well as a no protection circuit. The paper implements other error correction approaches like Qvector and Q0, however little to no information has been provided regarding these methods making implementation infeasible. The paper also discussed the Quon language and graphs used for the initial design, which we did not explore because these designs were not provided.

The paper delved into finding the optimal recovery channel by experimenting with different recovery circuits. We used their findings to implement the recovery VQC given in the paper. For optimization, we chose to use an existing Python library, PyTorch, as the paper does not outline in detail how they chose to implement or improve the optimizer.

A significant technical constraint is using a MacBook with an Apple M1 Pro chip for all simulation and optimization tasks due to budget limitations. Given the high computational cost of simulating quantum channels, such as using semi-definite programming for recovery map optimization, this restricts the project to small-scale codes, like the three-qubit and five-qubit VGQEC examples. The implementation uses Python with the PennyLane framework and other related libraries that support variational quantum circuit optimization and are compatible with the parameterized approach described in the paper.

The available two-month time frame is sufficient for reproducing selected results shown in figure 2. The paper notes that VGQEC codes require fewer parameters than methods like QVector, as they are based on known QEC codes rather than designed from scratch. This makes the optimization more manageable and the circuit structure more constrained, aligning with the project's scope.

However, implementing general-purpose VGQEC codes or large-scale noise adaptive codes is beyond the scope of this time frame and hardware setup. Furthermore, hardware-based or photonic verification, as discussed in the paper's experimental sections, is also beyond the scope of this simulation project.

2.2 Unitary

The unitary gate is added to test the circuit with different initial states as well as to avoid bias when measuring and calculating fidelity. The paper chooses a random operator from a 2-design each time they run the circuit. The 2-design quantum states set used for the simulation are:

$$\left\{ |0\rangle, \quad \frac{1}{\sqrt{3}}|0\rangle + \sqrt{\frac{2}{3}}|1\rangle, \frac{1}{\sqrt{3}}|0\rangle + \sqrt{\frac{2}{3}}e^{\frac{2\pi i}{3}}|1\rangle, \quad \frac{1}{\sqrt{3}}|0\rangle + \sqrt{\frac{2}{3}}e^{\frac{4\pi i}{3}}|1\rangle \right\}. \tag{1}$$

2.2.1 Rational and assumption

We used the same unitary operations, and assumed that since these are the ones they used for their simulation, it should work for our implementation too.

2.2.2 Implementation

We implemented the unitary operations shown in equation 1. Each time we run the circuit, a new random operation from out chosen unitary operation is applied.

2.3 Encoding

The encoding is made up of two parts as shown in figure 4, the first part is the encoding map ε_c, and the second is the variational quantum circuit U_ε. The encoding map ε_c is the original encoding for the 3-qubit repetition code (for 3-qubit VGQEC) and the original encoding for the 3-qubit repetition code (for 5-qubit VGQEC). The variational quantum circuits are given in the paper.

2.3.1 Rational and assumption

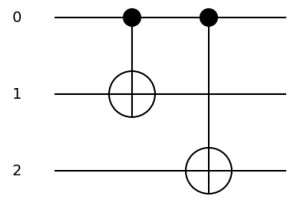
The 3-qubit VQGEC encoding map is given in the paper. For our 5-qubit VQGQEC we extended the 3-qubit encoding map for 5-qubit, assuming that it gives us the correct 5-qubit repetition code encoding map.

The paper did provide an encoding map for the 5-qubit VQGEC, but this was based on the original code being [5,1,3] and not 5-qubit repetition code which we ended up implementing. In section 4 we explained why we ended up not using [5,1,3] as original encoding and recovery.

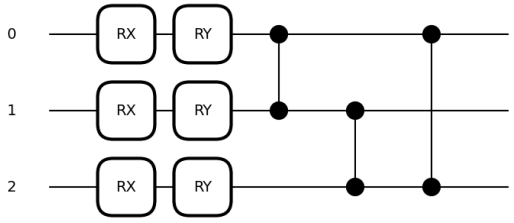
For the VQC it was not entirely clear which circuits we should use. In the paper we found one only U_ε with 3 wires, and only one with 5 wires, and therefore assumed these were the ones for the 3-VGQEC and 5-VGQEC circuits respectively. Additionally, the paper states that the 3-qubit VQC circuit should be run "L" times, but the value of L was not specified in the paper. They mentioned using L=2 for one of their implementation (Qvector), so we made the assumption to use the same value.

2.3.2 Implementation

The encoding circuits we implemented for 3-qubit VGQEC are 5-qubit VGQEC are illustrated in figures 5 and 6 respectively. The parameters of the rotations in the VQC are set by the optimizer (see subsection 2.7 for more information).

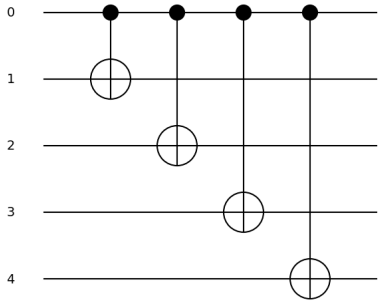


(a) Encoding map ε_c for 3-qubit VGQEC (same as the 3-qubit repetition code). Wire 0 has our logical qubit, and wire 1 and 2 has qubits in states $|0\rangle$.

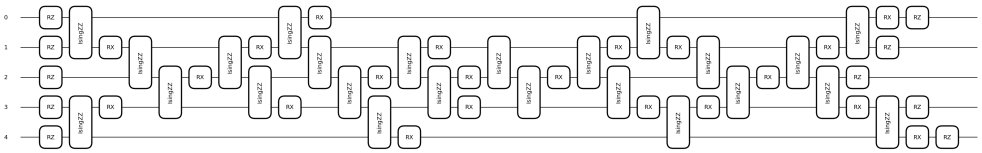


(b) Variational quantum circuit for 3-qubit VGQEC. The circuit is implemented L=2 times.

Figure 5: 3-qubit VGQEC encoding circuits.



(a) Encoding map ε_c for 5-qubit VGQEC (same as 5-qubit repetition code). Wire 0 has our logical qubit, and wires 1–4 have qubits in state $|0\rangle$.



(b) Variational quantum circuit for 5-qubit VGQEC.

Figure 6: 5-qubit VGQEC encoding circuits.

2.4 Noise

Simulating realistic noise is important in evaluating the effectiveness of quantum error correction methods in real-world situations. These models are well-suited for capturing the dominant noise mechanisms present in quantum hardware.

2.4.1 Rational and assumption

We assume that PennyLane's implementation of these channels aligns with the mathematical equations described in the VGQEC paper. Specifically, we rely on the assumption that the amplitude damping noise uses standard Kraus operators and that thermal relaxation is parametrized using T_1, T_2, and temperature values.

2.4.2 Implementation

Simulating the noise to reflect real-world situations has become relatively simple given PennyLane's inbuilt functions. We use the qml.AmplitudeDamping() and qml.ThermalRelaxationError() functions to introduce noise on each qubit of the encoded state.

2.5 Recovery

VGQEC introduces a two-staged recovery map that builds upon the structure of traditional codes by combining standard recovery maps with variationally optimized quantum circuits. The first component is the Variational Recovery map U_R, followed by the Original Recovery Map, R_c. The variational quantum circuit for 3-qubits was given in the paper (shown in figure 7 and consists of R_{ZZ}, R_X, R_Z gates.

The repetition code detects and corrects bit-flip errors using majority voting, however, its capability is limited to only one error type. The VGQEC framework enhances the system by introducing a variational recovery layer with trainable parameters that enables recovery from more general errors beyond its original capabilities.

2.5.1 Rational and assumption

Our design choices for recovery in VGQEC were guided by testing and the need for system-specific flexibility. While the framework allows for both variational and traditional recovery to be used together, we found that applying the original repetition code recovery for the 3-qubit VGQEC and the 5-qubit repetition often degraded performance. The system consistently achieved higher fidelity when relying solely on the variational layer. Therefore, in our final 3- and 5-qubit implementation, we removed the traditional recovery R_c step allowing U_R to dictate the full recovery.

We assumed that the VQC for recovery was given in the paper (figure 7 was for the 3-qubit circuit). This is because figure 4 shows U_R should have $n + 2 \cdot k$ wires. The given VQC has 5 wires ($3 + 2 \cdot 1 = 5$), and therefore corresponds to the VQC for the 3-qubit recovery circuit. For the 5-qubit circuit, we assumed that extending the given recovery VQC with 2 additional wires by adding the entanglement gates between all the wires, in the same manner, gave us the correct VQC for the 5-qubit recovery.

From figure 4 we also see that the bottom two wires (or bottom $2 \cdot k$ wires) coming out of the VQC are measured, however, the paper does not explain what these measurements are used for, and if they are even used in the final recovery step. Therefore, we chose to utilize the bottom wires only for entanglement within the VQC.

An other assumption we made is the number of iterations we run the VQC which is specified

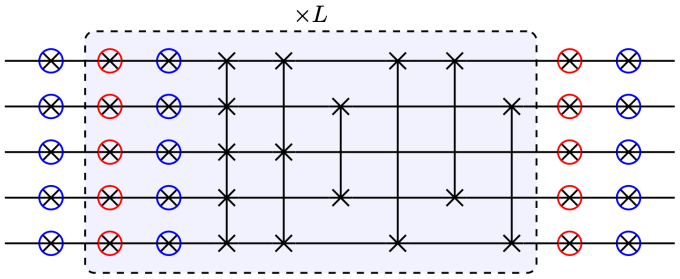


Figure 7: Variational Quantum Recovery Circuit given in the paper. Image taken from [1].

by L. The paper mentions they use $L = 3$ for numerical results, so we also set $L = 3$ for our implementation.

2.5.2 Implementation

For the 3- and 5-qubit recovery we implemented, as discussed, only the VQC circuit U_R and not any recovery map R_c. Figure 8 shows the complete variational recovery circuit we implemented for 3-qubit VGQEC and figure 9 shows the complete variational recovery circuit we implemented for 5-qubit VGQEC. The rotation parameters of all the gates in the VQC are set by the optimizer (see subsection 2.7 for more information).

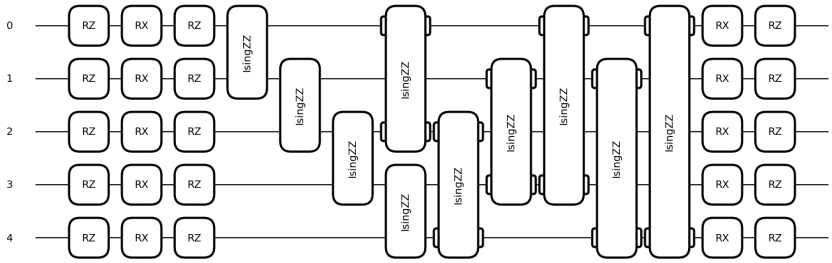


Figure 8: Implemented Variational Quantum Recovery Circuit for 3 Qubits.

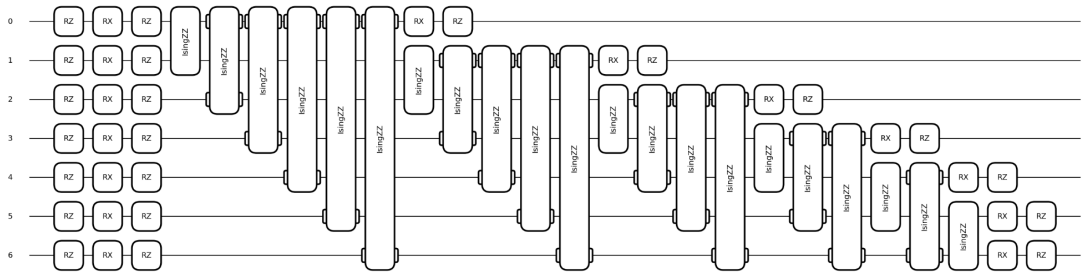


Figure 9: Implemented Variational Quantum Recovery Circuit for 5 Qubits.

2.6 Fidelity Calculation

Fidelity provides a practical way to estimate and understand the performance of a quantum channel. This is important for verifying that the VGQEC codes improve the preservation of quantum information under noisy conditions in the real world.

2.6.1 Rational and assumption

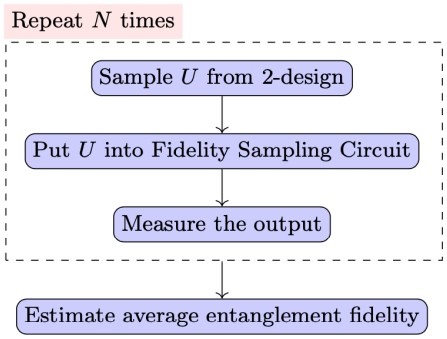


Figure 10: Average Entanglement Fidelity estimator. Image taken from [1].

In VGQEC, the fidelities measured are the channel fidelity and the average entanglement fidelity. According to the paper, the average entanglement fidelity is estimated by randomly sampling a unitary U from a 2-design and inputting U into the fidelity sampling circuit to obtain the circuit outcomes shown in figure 10. This process is repeated N times, and the estimated average entanglement fidelity is given by the probability of the all-zeros output from the fidelity sampling circuit. The relationship between the average entanglement fidelity $\overline{F_e}$ and the channel fidelity F_C is expressed by an equation [1],

$$\overline{F_e} = \frac{dF_C + 1}{d + 1} \tag{2}$$

, where d is the dimension of the system.

2.6.2 Implementation

To implement channel fidelity, a function takes the average entanglement fidelity and the dimension of the Hilbert space and compute equation 2. For the average entanglement fidelity, using qml.sample() yields binary measurement outcomes for each shot, but this approach is not differentiable when using backpropagation in PyTorch. PennyLane cannot compute gradients through non-deterministic sampling, which is necessary for training variational quantum circuits. Instead of sampling many shots and counting how often 0 is output, using the expected value via qml.expval() returns the expected probability of measuring the outcome $|0\rangle$. This method is fully differentiable and compatible with PyTorch and backpropagation.

2.7 Optimization

The optimization strategy is used to train VGQEC codes under specific quantum noise models. The primary goal is to maximize the average entanglement fidelity, which quantifies how well the VGQEC code preserves quantum information in the presence of noise.

2.7.1 Rational and assumption

The paper outlines the general steps of the optimization process for VGQEC codes, where the objective is to maximize the average entanglement fidelity. It states that the L-BFGS algorithm is used for optimization, which is well-suited for smooth objective functions and offers fast convergence. For each noise intensity level, the parameters are initialized independently, with initial values drawn from a normal distribution with mean 0 and standard deviation π. The encoding and recovery maps are parameterized variational circuits, denoted by U_ε and U_R, with optimized parameters $\vec{\alpha}_{\text{opt}}$ and $\vec{\beta}_{\text{opt}}$, respectively [1],

$$\mathcal{E}_{\text{out}} = U_{\varepsilon}(\vec{\alpha}_{\text{opt}}) \circ \mathcal{E}_c \tag{3}$$

$$\mathcal{R}_{\text{out}} = \mathcal{R}_c \circ U_R(\vec{\beta}_{\text{opt}}) \tag{4}$$

However, the paper does not specify the number of parameters used in these variational circuits or provide implementation details such as the optimization framework. Therefore, we assume that every gate in U_ε and U_R has a tunable parameter. Under this assumption, for the three-qubit and five-qubit VGQEC code, the circuit U_ε contains 12 and 55 parameters respectively, and the circuit U_R contains 75 and 46 parameters respectively, resulting in a total of 87 and 101 trainable parameters respectively.

2.7.2 Implementation

The implementation of a hybrid quantum-classical workflow optimizes the encoding and recovery circuits of the three-qubit and five-qubit VGQEC codes to adapt to a specific quantum noise model, such as amplitude damping and thermal relaxation. In this workflow, a differentiable quantum circuit, QNode, estimates the channel fidelity, while a classical optimizer, L-BFGS, updates the circuit parameters to maximize it using Pytorch [2, 3].

The QNode is a quantum function configured to use PennyLane's "default.mixed" simulator with the "torch" interface. It evaluates the probability of observing the all-zero outcome after the system passes through the noise channel protected by the VGQEC code, thereby approximating the average entanglement fidelity. The QNode receives a parameter vector $\vec{\alpha}$ and $\vec{\beta}$. To simulate the fidelity estimation circuit, a single-qubit unitary from a 2-design set is applied to qubit 0, followed by the encoding, noise model, recovery map, and inverse random unitary. The circuit returns the expectation value of the projector onto the $|0\rangle$ state.

The cost function evaluates the QNode over multiple pre-sampled 2-design unitaries and averages the resulting fidelities. Parameters are wrapped modulo 2π to ensure they remain within a valid rotation range. The negative mean fidelity is returned as the cost because PyTorch minimizes objectives, whereas the goal is to maximize fidelity.

In the optimization loop, parameters are initialized from a normal distribution with mean 0 and standard deviation π, consistent with the initialization strategy described in the paper. The optimization uses PyTorch's L-BFGS optimizer with a closure function required by L-BFGS to recompute the objective and its gradient at each step. The loop iteratively updates the parameters and records the cost at each step. After optimization completes, the final optimized parameters are $\vec{\alpha}_{\text{opt}}$ for the encoding circuit and $\vec{\beta}_{\text{opt}}$ for the recovery map and therefore the optimized average entanglement fidelity is $F(\vec{\alpha}_{\text{opt}}, \vec{\beta}_{\text{opt}})$.

3 Results

To be able to see how well our implemented VGQEC code performs, we ran the same code, with the same noise but different error correction code. In figure 11 we can see our 3-qubit VGQEC plot compared to no protection and 3-qubit repetition code, and in figure 12 we can see the 5-qubit VGQEC compared to no protection and 5-qubit repetition code.

3.1 3-qubit VGQEC result

First we can see that the no protection performs better than the 3-qubit repetition code, which makes sense since repetition code should only work for bit flips, and not amplitude damping.

Our 3-qubit VGQEC performs better than the no protection for all λ values. We can also see that at 0 noise the channel fidelity is at 1, and that as λ increases, the channel fidelity decreases. This is the expected behavior as more noise makes error recovery more difficult. With this we can conclude that our code does improve the error recovery.

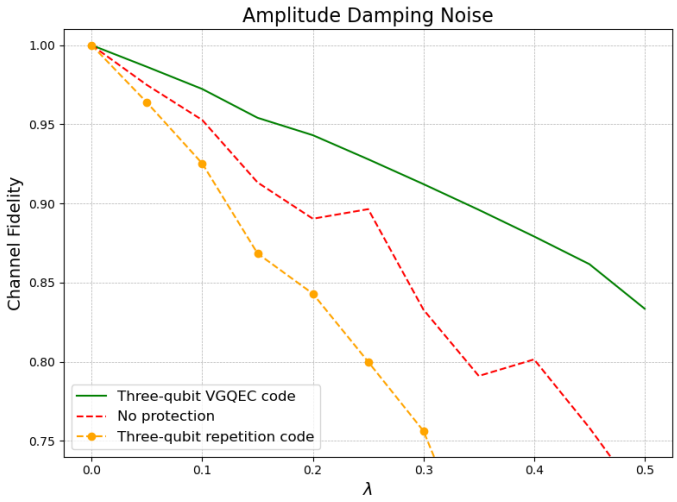


Figure 11: Graph of the channel fidelity when amplitude damping is applied with different λ values. The channel fidelity is showed for no protection, three qubit repetition code and three qubit VGQEC.

3.2 5-qubit VGQEC result

Again we can see that the no protection performs better than the 5-qubit repetition code.

Our 5-qubit VGQEC however does not perform better than no protection for most λ values. As explained in section 1.2, the encoding variational quantum circuit for the 5-qubit VGQEC was generated by compiling the Quon diagram attached to the GE flexible graph specifically for the [5,1,3] and not the 5 qubit repetition code as the original error correction method. The encoding VQC we used was therefore not the correct circuit to use to optimize the 5-qubit VGQEC based of 5-qubit repetition.

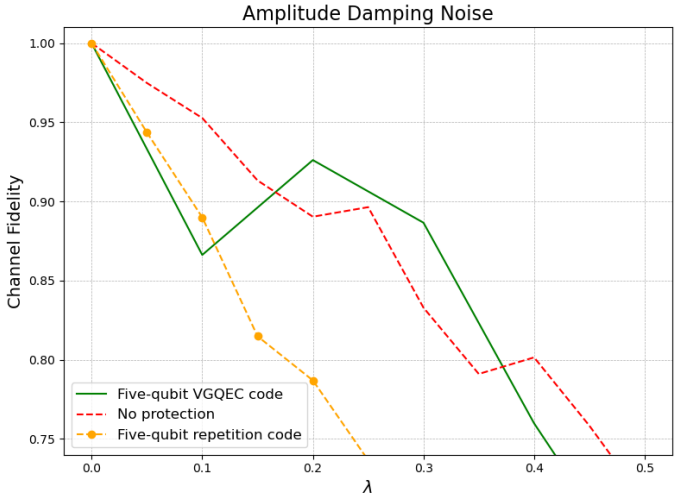


Figure 12: Graph of the channel fidelity when amplitude damping is applied with different λ values. The channel fidelity is showed for no protection, five qubit repetition code and five qubit VGQEC.

3.3 Comparing to the paper's results

In figure 2 we can see the papers result after simulating the 3- and 5-qubit VGQEC and other error recovery methods for comparison.

Compared to our result (figure 11), we can see that we get a similar 3-qubit repetition code and no protection code (ours fluctuates a bit more, but it has the same trends).

Comparing our 3-qubit VGQEC and the paper's 3-qubit VGQEC graph, we can see that we get a similar overall shape, with a gradually descending channel fidelity as λ increases. Up until $\lambda = 0.2$ the channel fidelity values are very similiar. After this point, the graph shows that our implementation performs a little better than the paper's. For example, at $\lambda = 0.4$, the paper has a channel fidelity value of approximately 0.79, while ours is at approximately 0.87. We can not say for sure why that is, but as discussed throughout the paper, we have had to do a lot of assumption because of missing information in the paper. This is probably why we do not get the exact same values.

For our 5-qubit VGQEC, the result of our implementation is much worse than the papers. This is what we expected as the paper used [5,1,3] encoding and recovery map, whilst we used 5-qubit repetition code. Our result is worse first of all because [5,1,3] is a better error correction model for amplitude damping than repetition code, but also because the encoding VQC that we used was meant for [5,1,3] original code and not 5-qubit repetition.

4 Discussion

4.1 Challenges and reproducibility

Reproducing the results of the paper proved to be more challenging than anticipated due to limited transparency of critical information.

Our main challenge, that made the 5-qubit VGQEC hard to implement and reproduce, was the lack of recovery map for [5,1,3]. The paper used [5,1,3] as their original encoding and recovery for the 5-qubit VGQEC. We did attempt that as well. We started by using the [5,1,3] encoding map that they gave us, and implemented our own [5,1,3] recovery map [5]. The given encoding map, and our implemented recovery for [5,1,3] did not work together. We realized after awhile that the output of the encoding circuit did not match what we expected for the input of our recovery circuit and that might have been the error. After a lot of trial and error, we concluded that we would not be able to reproduce the 5-qubit VGQEC with [5,1,3] due to the lack of information in the paper. We ended up doing the 5-qubit VGQEC with the original encoding being 5-qubit repetition code. This means we did not manage to reproduce the same 5-qubit VGQEC as the paper.

Since we implemented the 5-qubit VGQEC based on repetition and not [5,1,3], we only used amplitude damping for the noise, and not thermal relaxation.

Furthermore, the number of trainable parameters for optimization in the variational circuit is not disclosed, which is a key factor that affects both performance and computational cost and prevents an accurate assessment of the model's complexity. a different number go parameters to optimize can easily lead to different result, making it harder to replicate the papers results. The paper also lacks information on the optimization algorithms employed, further complicating the replication of the training procedure.

The authors also omit important details about the role of auxiliary wires in the recovery variational quantum circuits as discussed in our assumptions for recovery. This caused challenges for our reproducibility, as we did not know weather these auxiliary qubit are giving useful information for the last recovery part of our circuit.

The paper did also not provide implementation details for other error correction strategies used as

comparison such as Qvector and repetition codes, making it difficult to replicate the exact graphs provided in the paper.

Collectively, all these challenges posed a significant challenge, some more than other, in accurately reproducing the papers results.

References

[1] Yuguo Shao, Yong-Chang Li, Fuchuan Wei, Hao Zhan, Ben Wang, Zhaohui Wei, Lijian Zhang, and Zhengwei Liu. *Variational Graphical Quantum Error Correction Codes*, Mar 12, 2025.

[2] PyTorch Team, "LBFGS — PyTorch Documentation," PyTorch, 2025. [Online]. Available: https://pytorch.org/docs/stable/generated/torch.optim.LBFGS.html *LBFGS optimizer*, Mar 15, 2025.

[3] PennyLane Team, "How to choose your device," PennyLane Blog, Jan. 12, 2023. [Online]. Available: https://pennylane.ai/blog/2023/01/how-to-choose-your-device *Device selection guide*, Mar. 15, 2025.

[4] M. Newman and K. Satzinger, "Making quantum error correction work," Google Research Blog, Dec. 9, 2024. [Online]. Available: https://research.google/blog/making-quantum-error-correction-work/ *513 code*, Mar. 12, 2025.

[5] J. Preskill, "Chapter 7: Quantum Error Correction," Lecture Notes for Physics 229: Quantum Information and Computation, California Institute of Technology, Pasadena, CA, USA, 1999. [Online]. Available: http://theory.caltech.edu/ preskill/ph229/notes/chap7.pdf *513 syndrome*, Mar. 12, 2025.