Semidefinite Optimization for Model Specific Quantum Error Correction

James McGreivy

Massachusetts Institute of Technology (MIT)

mcgreivy@mit.edu

In this final project I describe a procedure to frame the design of a quantum error correcting code as a bi-convex optimization problem. I then implement a solver using Python, and explicitly solve for quantum error correcting codes which are optimal against error ensembles constructed from the amplitude damping, depolarizing, and dephasing noise channels. I also test this formalism in the case of one qubit embedded into a three qutrit Hilbert space, as well as in the case of concatenating two quantum error correcting codes. I find that if errors are able to be well characterized, these optimal codes can yield better error correcting performance – measured using average channel fidelity – than a generic Stabilizer code. This comes with a tradeoff, however, of being less robust against errors which were not expected.

1. Introduction

One of the primary barriers to the implementation of general purpose quantum computers is the effect of errors. Errors accumulate on stored quantum information due to unwanted interactions of qubits with the environment or eachother, as well as during program execution due to faulty quantum gates, faulty state preparation, and noisy measurements. (1).

Currently, the most influential quantum error correction (QEC) algorithms are stabilizer codes (2; 3). In a stabilizer code quantum information is encoded as logical states within the orthogonal subspaces of a larger quantum system, with the hope that errors will never accumulate enough to move a logical state out of its respective subspace. For an $[[n, k, d]]$ stabilizer code, k qubits are encoded into the orthogonal subspaces of an n qubit system in such a way as to prevent errors across at most d qubits from a sending a logical state between orthogonal subspaces. Stabilizer codes have the benefit of being intuitively understandable, have decades of theoretical development (4), and the projective measurements they use to correct errors are reasonable to physically implement.

In general stabilizer codes do not characterize a specific error model for a quantum device, instead promising to correct for any arbitrary error as long as they act on no more than d qubits - where d is the distance of the stabilizer code. While this makes stabilizer codes more architecturally agnostic[*], treating errors generically may not yield the most efficient algorithms. This is the central hypothesis behind using semidefinite optimization for model specific QEC – using the same number of qubits, a QEC code tailor-made to an exact error model can better correct for that error model than a generic stabilizer code (5).

In this final project I investigate the application of semidefinite optimization to model specific error correction, guided primarily by Kosut et al.'s 2009 paper on the topic (6). First, I will outline the process of casting quantum error correction as a semidefinite optimization problem, with some of my own modifications to the derivation. Then, I will show results from my own python implementations including the effectiveness of a one qubit into three qutrit QEC as well as an attempt to concatenate two convex optimized QECs together.

2. Theory

Quantum error correction involves the embedding of an n_L dimensional hilbert space, the logical space $\mathcal{H}^L$ on q_L qubits, into an $n_c > n_L$ dimensional code space $\mathcal{H}^C$ on q_c qubits. At its most fundamental level, QEC hopes to use extra degrees of freedom present in the code space in order to detect and correct for errors without destroying any logical quantum information. Since quantum channels describe the most general possible transformation of quantum information between hilbert spaces, the process of QEC can be represented via the following diagram:

$$\rho_{in} \longrightarrow \boxed{\mathcal{C}} \longrightarrow \rho_{enc} \longrightarrow \boxed{\mathcal{E}} \longrightarrow \hat{\rho}_{enc} \longrightarrow \boxed{\mathcal{D}} \longrightarrow \rho_{out}$$

Here ρ_{in} is the logical quantum information, represented by a density matrix, to be protected. $\mathcal{C} : \mathcal{H}^L \to \mathcal{H}^C$ is the *encoding* quantum channel which maps logical quantum information onto the encoding space. $\mathcal{E} : \mathcal{H}^L \to \mathcal{H}^L$ is the *error* channel representing whatever error processes corrupt quantum information in a given device. Finally, $\mathcal{D} : \mathcal{H}^L \to \mathcal{H}^C$ is the *decoding* channel which maps

[*]Some stabilizer codes, for example toplogical codes (3), do benefit more from particular architectures.

a corrupted encoded state back onto the logical state space. Using the Kraus representation, these channels can be written as:

$$\begin{aligned} \mathcal{C}(\rho) &= \sum_c C_c \, \rho \, C_c^\dagger \quad \sum_c C_c^\dagger \, C_c = 1 \\ \mathcal{D}(\rho) &= \sum_d D_d \, \rho \, D_d^\dagger \quad \sum_d D_d^\dagger \, D_d = 1 \end{aligned} \tag{1}$$

In general logical states are not explicitly mapped down to the logical Hilbert space when their quantum information is error corrected. Instead, there is a recovery map $\mathcal{R} : \mathcal{H}^L \to \mathcal{H}^L$ consisting of a projective measurement followed by a unitary which corrects errors while keeping quantum information embedded within the code space. This recovery map can always be constructed from the encoding and decoding maps, however, simply by composition:

$$\mathcal{R}(\rho) = \sum_{c,d} C_c \, D_d \, \rho \, D_d^\dagger C_c^\dagger \tag{2}$$

The goal of quantum error correction is for ρ_{in} to be as close as possible to ρ_{out} for all possible ρ_{in}. Since these are density matrices, the notion of closeness can be expressed using fidelity. There are several ways to compute channel fidelity, such as the minimum fidelity over all mixed states ρ or the minimum fidelity over all pure states $|\Psi\rangle \langle\Psi|$. Kosut et al. instead uses average channel fidelity, which is the easiest to compute and numerically close to the previous two for small numbers of qubits (6):

$$f_{avg}(\mathcal{C}, \mathcal{E}, \mathcal{D}) = \frac{1}{n_L^2} \sum_k |\mathbf{Tr} \, D_d \, E_e \, C_c|^2 \tag{3}$$

This is beginning to have the structure of an optimization problem, where you are given an error model $\{E_e\}$ and the Kraus operators $\{C_c\}$ and $\{D_d\}$ should be chosen such that they maximize f_{avg}:

$$\begin{aligned} &\text{maximize} \;\; f_{avg}(\{C_c\}, \{E_e\}, \{D_d\}) \\ &\text{where} \;\; \sum_c C_c^\dagger \, C_c = 1 \;\; \text{and} \;\; \sum_d D_d^\dagger \, D_d = 1 \end{aligned} \tag{4}$$

However, this optimization problem is very difficult as written because the objective is a nonlinear function of each individual optimization variable. In addition, the constraints are quadratic and thus do not form a convex set. This problem can be simplified first by expanding the Kraus operators in the standard basis:

$$C_c = \sum_{i,l}^{n_L,n_c} C_{c,li} \left|l\right\rangle \left\langle i\right| \quad \text{and} \quad D_d = \sum_{i,l}^{n_L,n_c} D_{d,il} \left|i\right\rangle \left\langle l\right| \quad \text{and} \quad E_e = \sum_{l,m}^{n_c,n_c} E_{l,m} \left|l\right\rangle \left\langle m\right| \tag{5}$$

Substituting these into the optimization equation and simplifying yields the following:

$$\begin{aligned} \text{maximize} \ \ f_{avg} &= \frac{1}{n_L} \sum_{i,j}^{n_L} \sum_{l,m,\gamma,\sigma}^{n_c} (X_D)_{il,j\gamma} (X_C)_{mi,\sigma j} (X_E)_{lm,\gamma\sigma} \\ \text{where:} \ \ (X_D)_{il,j\gamma} &= \sum_r R_{r,il} \ R^*_{r,j\gamma} \ \ \text{and} \ \ \sum_i^{n_L} (X_D)_{il,i\gamma} = \delta_{l\gamma} \\ (X_C)_{mi,\sigma j} &= \sum_c C_{c,mi} \ C^*_{c,\sigma j} \ \ \text{and} \ \ \sum_m^{n_c} (X_C)_{mi,mj} = \delta_{ij} \\ (X_E)_{lm,\gamma\sigma} &= \sum_e E_{e,lm} \ E^*_{e,\gamma\sigma} \ \ \text{and} \ \ \sum_m^{n_C} (X_D)_{lm,l\sigma} = \delta_{m\sigma} \end{aligned} \tag{6}$$

X_C, X_D, and X_E are called the *process matrices*, and are fully specified by the Kraus operators of each quantum channel. While the objective is now a convex function with respect to each individual process matrix, the quadratic constraints are still an issue. Kosut et al. proposes a *convex relaxation* of the problem, where the quadratic constraints ensuring decomposability into Kraus operators are relaxed to the convex constraint of positive semidefiniteness (6):

$$\begin{aligned} \text{maximize} \ \ f_{avg} &= \frac{1}{n_L} \sum_{i,j}^{n_L} \sum_{l,m,\gamma,\sigma}^{n_c} (X_D)_{il,j\gamma} (X_C)_{mi,\sigma j} (X_E)_{lm,\gamma\sigma} \\ \text{where:} \ \ (X_D)_{il,j\gamma} &\geq 0 \ \ \text{and} \ \ \sum_i^{n_L} (X_D)_{il,i\gamma} = \delta_{l\gamma} \\ (X_C)_{mi,\sigma j} &\geq 0 \ \ \text{and} \ \ \sum_m^{n_c} (X_C)_{mi,mj} = \delta_{ij} \\ (X_E)_{lm,\gamma\sigma} &\geq 0 \ \ \text{and} \ \ \sum_m^{n_C} (X_D)_{lm,l\sigma} = \delta_{m\sigma} \end{aligned} \tag{7}$$

This relaxation is justified by noting that any positive semidefinite matrix can be decomposed into the product of a matrix with its hermitian conjugate:

$$(X)_{il,j\gamma} = \sum_k K^\dagger_{il,k} \ K_{k,j\gamma} = \sum_k K^*_{k,il} \ K_{k,j\gamma} \tag{8}$$

Thus, a Kraus representation $\{K_k\}$(note that Kraus representations are not unique) can be recovered from a process matrix X via singular value decomposition:

$$\text{if} \quad (X)_{il,j\gamma} = \sum_s \sigma_s U_{il,s} V^\dagger_{s,j\gamma} \quad \text{then} \quad K_k = \sqrt{\sigma_k} \sum_{j\gamma} V_{j\gamma,k} \left|j\right\rangle \left\langle\gamma\right| \tag{9}$$

We now have all of the machinery necessary to solve for an optimal encoding map $\mathcal{C}$ and decoding map $\mathcal{D}$, given $\mathcal{E}$. Note that the objective is only convex in *either* X_D or X_C individually, and so this problem is solvable via biconvex iteration using a semidefinite programming solver. Because of the biconvex iteration, when an iteration has converged we can only know for sure that we have found a local optimum (6).

3. Results

For this project I implemented a solver using Pytorch (7) and the CVXPY (8; 9) semidefinite optimization framework in Python. This solver constructs a convex optimized QEC code given any single-qubit error model, where the single-qubit error model is then applied indepedently to each qubit in the encoding space. To account for the fact that the optimization only returns local minima, several optimizations are performed with only the highest fidelity model kept for analysis. The complete code for this project can be found at:

https://github.com/JamesMcGreivy/convex-quantum-error-correction.

3.1. Effectiveness on 1-qubit error channels

To begin, I compared the effectiveness of this algorithm to a stabilizer code with an equivalent number of qubits. I chose the five-qubit stabilizer code due to it being the smallest stabilizer code to protect against any arbitrary single qubit error. For my error models, I constructed them from three simple single-qubit error channels – the amplitude damping channel, the depolarizing channel, and the dephasing channel. The intensity of each of these error channels is tuned via the parameters λ_{damping}, $\lambda_{\text{depolarizing}}$, and $\lambda_{\text{dephasing}}$. Each error model was constructed as an ensemble of these three channels plus the identity channel (no error), and the optimization was performed simultaneously across the entire ensemble.

Figure **1** demonstrates the fidelity of the convex optimized QEC code for two cases. In the left column the convex optimization was performed with $\lambda_{\text{damping}} = \lambda_{\text{depolarizing}} = \lambda_{\text{dephasing}} = 0.25$,

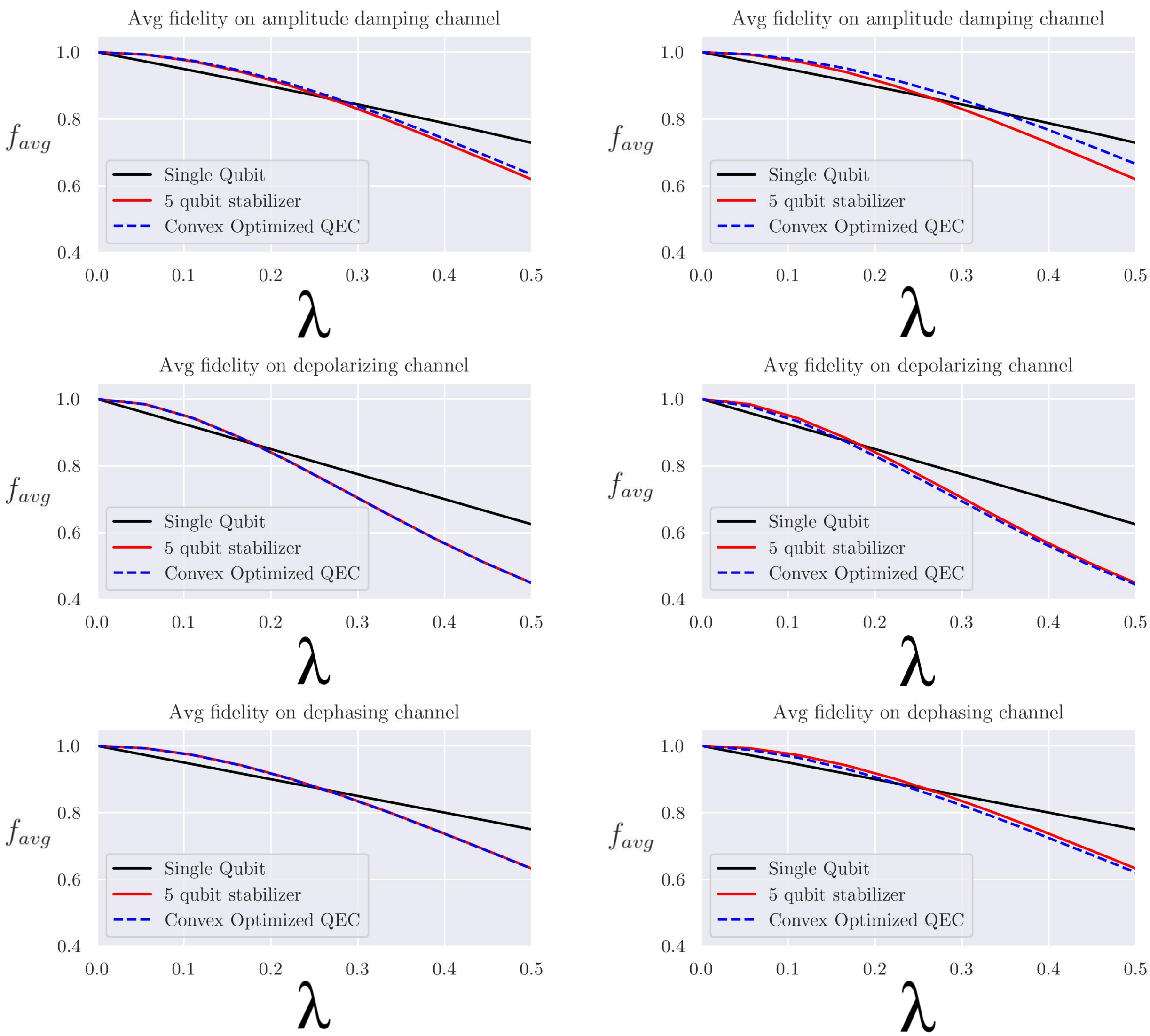

Fig. 1: Fidelity of the single qubit, 5 qubit stabilizer, and 5 qubit convex optimized QEC against the (top) amplitude damping (middle) depolarizing and (bottom) dephasing channels, with the intensity of the error channel λ varied. The (Left) Convex QEC was trained using an error ensemble with λ_{damping}=0.25, $\lambda_{\text{depolarizing}}$=0.25, and $\lambda_{\text{dephasing}}$=0.25 (Right) Convex QEC was trained using an ensemble with λ_{damping}=0.3, $\lambda_{\text{depolarizing}}$=0.1, and $\lambda_{\text{dephasing}}$=0.1

meaning there is no dominant error mode. For this case, there is no signifigant improvement of the convex optimized QEC code over the 5 qubit stabilizer code. A slight improvement can be seen in fidelity against the amplitude damping channel, however this is offset by a small decrease in fidelity against the depolarizing and dephasing channels and is a result of the particular local minima found during optimization. This demonstrates (but does not prove) that in the case of no dominant error mode the five qubit stabilizer code is optimal, in the sense that no code exists which gives better fidelity using the same number of qubits. As discussed in the introduction, this is to be expected since stabilizer codes are designed to correct for any arbitrary error.

In the right column the convex optimization was performed with $\lambda_{\text{damping}} = 0.3$ and $\lambda_{\text{depolarizing}} = \lambda_{\text{dephasing}} = 0.1$. Here the optimization learns a QEC which can significantly outperform the 5 qubit stabilizer code when correcting for amplitude damping, however this comes with the tradeoff of decreased fidelity on the depolarizing and dephasing errors. Thus, if we had mischaracterized the noise of our device and over-estimated the effects of the amplitude damping channel we would get less QEC fidelity than expected.

3.2. Learning to embed a qubit into a qutrit's Hilbert space

For my next experiment, I tested the effectiveness of a convex optimized QEC code when embedding one qubit into the 9-dimensional Hilbert space of three qutrits (a quantum object with three orthogonal states). For my error models, I once against constructed them as a combination of the amplitude damping, depolarizing, and dephasing channels. However, these channels require a re-definition for the case of a qutrit. I chose the following as a natural extension:

$$
\begin{aligned}
&\text{Amplitude Damping:} \\
&K_0 = |0\rangle\langle 0| + \sqrt{1-\lambda}(|1\rangle\langle 1| + |2\rangle\langle 2|) \\
&K_1 = \sqrt{\lambda}\,|0\rangle\langle 1| \quad K_2 = \sqrt{\lambda}\,|0\rangle\langle 2| \\
&\text{Depolarizing:} \\
&K_0 = \sqrt{1-\lambda}I \quad K_1 = \sqrt{\lambda/8}Y \\
&K_2 = \sqrt{\lambda/8}Z \quad K_3 = \sqrt{\lambda/8}YY \\
&K_4 = \sqrt{\lambda/8}YZ \quad K_5 = \sqrt{\lambda/8}YYZ \\
&K_6 = \sqrt{\lambda/8}YZZ \quad K_7 = \sqrt{\lambda/8}YYZZ \quad K_8 = \sqrt{\lambda/8}ZZ \\
&\text{Dephasing:} \\
&K_0 = \sqrt{1-\lambda}I \quad K_1 = \sqrt{\lambda}\,|0\rangle\langle 0| \\
&K_2 = \sqrt{\lambda}\,|1\rangle\langle 1| \quad K_3 = \sqrt{\lambda}\,|2\rangle\langle 2|
\end{aligned}
\tag{10}
$$

For the amplitude damping channel it is assumed that transitions between the $|1\rangle$ and $|2\rangle$ states are forbidden, for example by selection rules in an atomic system.

Figure **2** plots the average fidelity of a 1-qubit-into-3-qutrit convex optimized code vs a 1-qubit-into-5-qubit convex optimized and 1-qubit-into-5-qubit stabilizer code. In the figure, the 3 qutrit code seems to achieve better fidelity than both the 5 qubit convex optimized code and the 5 qubit stabilizer code. This is not a 1-1 comparison, however, because the amplitude damping channel of a qutrit is different from the amplitude damping channel of a qubit. In addition, the optimized qutrit code does not take into account that transitions between the $|1\rangle$ and $|2\rangle$ states are forbidden when constructing a decoding channel.

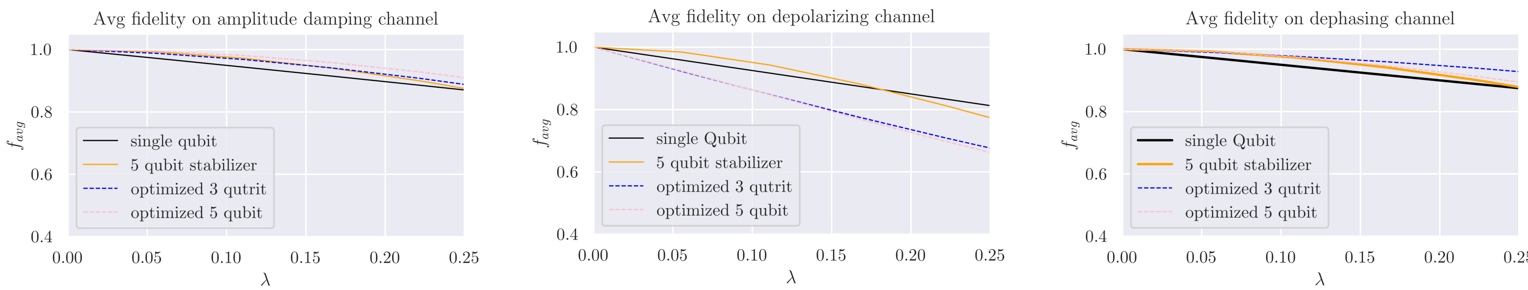

Fig. 2: Average fidelity vs channel intensity for a single qubit, the 5 qubit stabilizer code, a 5 qubit convex optimized code, and a 3 qutrit convex optimized code. The codes were evaluated against the amplitude damping channel, the depolarizing channel, and the dephasing channel. The convex optimized codes were trained using only the amplitude damping channel. Note that the amplitude damping channel is defined differently for a qubit and a qutrit, so this plot does not show a 1-1 comparison.

3.3. Code concatenation

The 9 qubit Shor code is built by concatenating a 3 qubit bitflip and a 3 qubit phase flip code. Separately the bitflip and phase flip codes are only able to correct for a specific kind of error, however through concatenation the 9 qubit Shor code can correct for arbitrary 1 qubit errors. Inspired by this, we can consider concatenating convex optimized QEC codes in a similar way. The concatenation of two codes can be written as:

$$\{C_a^A\} \otimes \{C_b^B\} = \{(C_b^B)^{\otimes q_c}\, C_a^A\} \quad \{D_a^A\} \otimes \{D_b^B\} = \{D_a^A\, (D_b^B)^{\otimes q_c}\} \tag{11}$$

Figure **3** compares the average fidelity vs channel intensity of a 9-qubit code, constructed by concatenating two 3 qubit convex optimized codes, to the 9 qubit Shor stabilizer code. Each individual 3 qubit code was optimized with $\lambda_{\text{damping}} = \lambda_{\text{depolarizing}} = \lambda_{\text{dephasing}} = 0.25$, so there is no dominant error mode. The fidelity of the concatenated codes on the amplitude damping channel comes close to that of the Shor code. The fidelity against the dephasing and depolarizing channels are not shown, due to their larger number of Kraus operators causing my laptop to run out of memory during training.

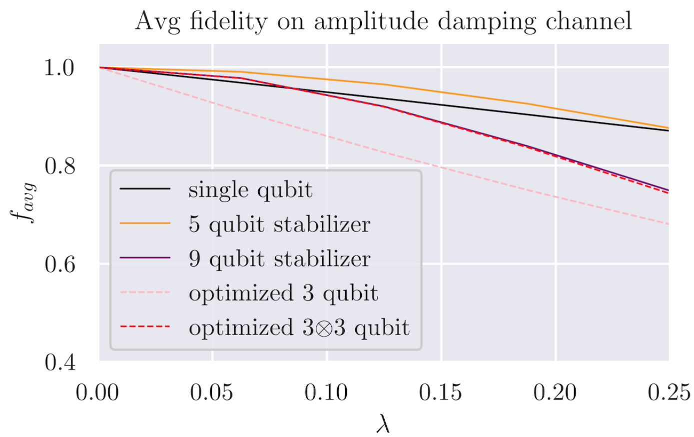

Fig. 3: Average fidelity vs channel intensity for the amplitude damping channel, the depolarizing channel, and the dephasing channel. The red line shows the concatenation of two 3 qubit convex optimized codes (shown in pink). The 3 qubit convex optimized codes were trained on a combination of the amplitude damping channel and the dephasing channel. The average fidelity very closely matches that of the 9 qubit Shor code, which is also the concatenation of two codes. The fidelity against the dephasing and depolarizing channels are not shown, due to their larger number of Kraus operators causing my laptop to run out of memory during training.

Although it is interesting that the independently trained concatenated codes are able to achieve similar fidelity to the Shor code, it would be much more interesting if the codes could be trained in such a way that they took into account that they were part of a concatenated code. This is a possible direction for future projects to investigate.

4. Conclusions

As demonstrated in this paper, the convex optimization formalism outlined by Kosut et al. (6) can construct effective quantum error correcting codes given a specific error model. However, this formalism suffers from two key limitations when applied to an actual quantum computer. Quantum error correcting codes constructed via optimization can overfit the given error model, making the codes less robust to errors which were not anticipated for and included in the training model. Additionally, the optimized encoding, decoding, and recovery channels do not take into consideration the implementation difficulty of a given quantum channel.

Quantum computers generally include a discrete set of native gate operations, from which all other possible gate operations can be approximated to an arbitrary precision. The encoding, decoding, and recovery channels found via optimization are not written in terms of these gate operations, however, and could require an extremely deep circuit to effectively approximate. Since each of these native operations have a nonzero error rate, the depth of the circuits needed to implement these optimized quantum error correcting codes could offset the gains in fidelity when compared to a simpler to implement stabilizer code.

I believe that code concatenation may be an effective way to construct convex optimized QEC codes while taking into account implementation compelxity, however I did not have enough time to fully explore this through my final project. When implementing multi-qubit operations on a quantum computer, as needed by quantum error correction, it is generally easier to perform operations which involve a smaller number of qubits. This could be incentivized for by choosing a quantum error correcting code for the outermost layer of the concatenated code which is easiest to implement, and then training for an optimal innermost code given some well-characterized error channel. This way, the more difficult to implement convex optimized code acts on a smaller number of qubits within each inner code block, and is thus easier to implement.

Thus, for future work on this topic I would choose to investigate the possibility of building code concatenation into the optimization scheme, so that an easy-to-implement code could be provided as the outermost code and a more complex innermost code could be optimized for.

References

[1] W. Cai, Y. Ma, W. Wang, C.-L. Zou, and L. Sun, "Bosonic quantum error correction codes in superconducting quantum circuits," *Fundamental Research*, vol. 1, no. 1, pp. 50–67, 2021.

[2] D. Gottesman, *Stabilizer codes and quantum error correction*. California Institute of Technology, 1997.

[3] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland, "Surface codes: Towards practical large-scale quantum computation," *Physical Review A*, vol. 86, no. 3, p. 032324, 2012.

[4] P. W. Shor, "Scheme for reducing decoherence in quantum computer memory," *Phys. Rev. A*, vol. 52, pp. R2493–R2496, Oct 1995.

[5] A. S. Fletcher, P. W. Shor, and M. Z. Win, "Structured near-optimal channel-adapted quantum error correction," *Physical Review A*, vol. 77, Jan. 2008.

[6] R. L. Kosut and D. A. Lidar, "Quantum error correction via convex optimization," *Quantum Information Processing*, vol. 8, p. 443–459, July 2009.

[7] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019.

[8] S. Diamond and S. Boyd, "CVXPY: A Python-embedded modeling language for convex optimization," *Journal of Machine Learning Research*, vol. 17, no. 83, pp. 1–5, 2016.

[9] A. Agrawal, R. Verschueren, S. Diamond, and S. Boyd, "A rewriting system for convex optimization problems," *Journal of Control and Decision*, vol. 5, no. 1, pp. 42–60, 2018.