**Mitigating errors on logical qubits with stabilizer codes**

**Graeme Jacobson**

University of Illinois Urbana-Champaign

May 4, 2021 ………………………………………………………………………………………………………

**Present quantum computing simulators do not give the entire picture when it comes to the abilities and limitations of quantum computation. The hardware of real quantum computers is not a perfectly closed system, which can cause imperfections in computation, also known as errors. To mitigate qubit errors and their progression in a quantum circuit, multiple qubits can be encoded into one logical qubit and combined with parity measurements to check for errors without losing any information on the system. This along with tools such as group theory make up a class of error correcting codes called stabilizers that will stabilize the fidelity of states during computation.**

**I. Introduction**

Many times, when quantum information is talked about or studied, we are assuming an ideal system. However, this is not the case for real quantum systems that are influenced by unwanted interactions1 from the outside environment. These interactions are called noise and cause errors in quantum computation.

Noise in quantum computation can from two possible sources2. The first of these sources is coherent quantum errors, which comes from incorrect knowledge of system. This can lead to undesired uses of specific gate operations, flawing the design of an algorithm. The second source of noise is environmental decoherence. When systems are not perfectly closed off from an environment, errors can appear in computation, such as bit flip or phase flip errors1. Though work is being done to get quantum computers ever closer to being a perfectly closed system, work must also be done to compensate for these imperfections.

The focus of quantum error correction is to find efficient methods that can detect and correct errors, allowing for higher fidelity computation. Stabilizer codes are at the heart of quantum error correction using concepts such as group theory and parity measurements to find qubit errors and correct them without disturbing the state of the qubits. The two stabilizer codes that will be studied in this report are the five qubit stabilizer10 and the seven qubit stabilizer9. Five qubits are said to be the minimum number of qubits needed for error correcting, but it is not a CSS4 code (Calderbank, Shor, Steane) since *X* and *Z* gates must be mixed in its generators. The seven qubit code is much more transparent and said to be the easiest stabilizer code to work with.

The study of quantum error correction and fault tolerance contain many elements including topics like concatenation1, threshold theorom3, and universal sets of gates3. For the sake of this project, topics beyond parity measurements, basic single qubit error stabilizers, and error models will not be discussed.

**II. Stabilizer Formalism**

Qubits are typically expressed in the computational basis, spanned by the basis vectors |0⟩ and |1⟩. An error can occur on a single qubit with a probability *p*, and the probability of the qubit not receiving an error is 1-*p*. To mitigate the probability of an error, we encode a single qubit as multiple qubits, called a *logical qubit*1. For example, a logical qubit can be made from three single qubits.

|0⟩ →|000⟩

|1⟩ →|111⟩

Using a concept called *repetition code*6, the state of a simple majority of the qubits would be considered “correct” information, and others that do not agree are said to be wrong, or an error. This means that an error would go undetected if a simple majority of the qubits were truly wrong. The probability of error on a simple majority in a three qubit case is 3*p*2-2*p*3, which is much smaller than a single qubit error, providing more confidence in information processing. Logical qubits are a cornerstone of error correction and quantum computation. However, there are certain criteria that needs to be upheld to successfully implement an error correction scheme.

Eigenspaces describe how the states of qubits interact with operations and potential errors that appear in computation. Most operators that will be dealt with can either commute or anticommute with each other and qubits can be eigenstates of these operators with positive and negative eigenvalues. Say, that a qubit is a +1 eigenstate of an operator *S*. If another operator *T* anticommutes with *S*, that is *ST = -TS,* then the qubit will become a -1 eigenstate of *ST*. The goal of error correction code, or stabilizer code, is to contain operators that will anticommute with single qubit errors. This implies that for an error *Ei* acting on some state |ϕi⟩, the product of two errors, *EkϮEl­­­*,will anticommute with a stabilizer operator *Si*, following the criteria

⟨ϕi| *EkϮEl* |ϕj⟩ = 0 **(1)**

Group theory1 is used to define stabilizer groups which will interact with errors by either commuting or anticommuting. Generators1 are the minimum set of elements *g1*,…,*gl*, that can produce every element of a group *G*. The Pauli group4 contains elements *Pi* = ±{*I, iI, X, iX, Y, iY, Z, iZ*} where products of these elements either commute or anticommute. A stabilizer group can be written as a Abelian2 (all elements commute) subgroup of the Pauli group, which does not contain -*I*. For example, the stabilizer group for three qubits is given by the generators *S* = {*XXI, XIX*}. A stabilizer group must also follow the criteria that any error must either commute with a generator, meaning it is part of the stabilizer group, or anticommute with a generator, meaning it is an error that can be corrected. So, it must be true from **(1)** that if some Pauli error *P* anticommutes with a generator *Si*,

⟨ϕi|*P*|ϕj⟩ = 0 **(2)**

or if these errors are elements of the stabilizer group, then

⟨ϕi|*P*|ϕj⟩ = ⟨ϕi|*S*|ϕj⟩ = δij **(3)**

It follows from **(2)** and **(3)** that for some state |ϕ⟩ to be stabilized by a stabilizer operator, |ϕ⟩ must be a +1 eigenstate of *S.*

It is worthwhile to mention how errors are described in a stabilizer context. Single qubit errors can be written as a Hermitian linear combination8 of operators from the Pauli group. Typically, this linear combination will be written using the identity *I,* bit flip *X*, phase flip *Z*, and product of the bit flip and phase flip *XZ*, which is also the *Y* operator (with eigenvalues multiplied by *i*) as the basis operators.

*Ei* = *ei0I + ei1X + ei2Z + ei3XZ*  **(4)**

When these errors are measured, the error will collapse into one of the four basis errors, and a correction can be done on that part of the error. This process then continues down the stabilizer correcting more parts of this error from the total linear combination. However, an arbitrary error (random rotation of state) sometimes cannot be completely corrected, and the fidelity of the logical qubit will not remain at 1. This will become apparent when the fidelities of states are tested later in this paper. With that being said, the overall goal is to have higher fidelity states than without using a stabilizer.

**III. Design of Stabilizer Codes**

Now, that some mathematical properties and language for stabilizers has been laid out, a more wholistic view of the components that make up stabilizer codes can be mapped out.

Typically, a stabilizer code is labeled as a [*n, k, d*] code where *n* is the number of qubits being used to encode *k* logical qubits that can correct *t* errors. The maximum distance13 (between the state and the state effected by an error) of the code is *d =* 2*t* + 1. Stabilizer codes that can correct one single qubit error have a distance of 3, which will be reoccurring in the stabilizer definitions in Sect. IV. Even though *n* qubits of dimension 2*n* are being used in total, the stabilizers exist in a subspace of dimension 2*n-g* , where *g* is the number of generators in the stabilizer group. For example, the seven qubit stabilizer has 7 encoded qubits and 6 generators, meaning it exists in a 27-6 dimensional subspace, which is a single qubit.

The basic structure of a stabilizer contains four elements: an *encoding*2, generators, parity measurements, and *logical operators*5. Prior to starting a computation or algorithm, qubits will be encoded into a logical qubit using an encoding scheme specific to the number of qubits and defined generators being used. The purpose of this is to have logical qubits live in a +1 eigenspace of the generators, so they can effectively find errors and correct them. Different sources will slightly differ in their definitions of the generators for each stabilizer, but they will agree with some rotation. Once qubits are encoded, we can allow computation to take place in a noisy channel where error detection and correction will take place.

To detect errors, generators will now be placed on the logical qubits where the anticommutating interactions mentioned in sect. II will be utilized. Each generator will be connected to a certain parity measurement1. Parity measurements, also shown in Fig 3.1, can be made on ancilla qubits1 (qubits that are not part of the logical qubit for measurement purposes), informing if an error anticommutes with a generator, without measuring the states of the code qubits (qubits forming the logical qubit). These measurements come in the form of classical bits on a classical register called an error syndrome1, where the measurement results indicate whether an error has occurred, and where the error is (if one exists). This can be done by entangling6 the code qubits with the ancilla qubits. Like mentioned before, errors can turn qubit states into -1 eigenstates of stabilizers, and these measurements on the ancilla qubits will display this occurrence, warning that a correction needs to be made.

Chart, box and whisker chart

Description automatically generated

Fig 3.16 – q0 and q1 are the code qubits and q2 is the ancilla qubit. The CNOT gates entangle the code qubits to the ancilla qubit, allowing for parity measurements. If q0 and q1 are the same state, q2 will measure |0⟩. If the code qubits are not the same, q2 will measure |1⟩.

Once ancilla measurements have made an error syndrome, logical operators can be placed after the generators in their correct locations. Logical operators act on logical qubits as single qubit operators would on a single qubit. Each generator comes typically with logical *X* and *Z* operators and each error syndrome corresponds to a single correction, telling the logical operators where to be placed to correct errors.

To briefly demonstrate the basic structure of stabilizer codes, let’s examine the three qubit bit flip code2. This code is not a great stabilizer code because it cannot correct arbitrary errors, only bit flip errors on a single qubit. A different encoding and set of generators are even needed to fix phase flip errors.

Starting with the logical encoding, also shown in Figure 3.2, two CNOT gates are used to replicate the state of q0 onto the other two qubits, creating one logical qubit.

Timeline

Description automatically generated with low confidence

Fig 3.2 – q0 represents the original state of the single qubit. CNOT gates move this state onto the other two qubits to create one logical qubit.

After the noisy channel takes effect in succession to the encoding, the generators *S* = {*XXI*, *XIX*} are applied to the logical qubit and entangled with the ancilla qubits where the error syndrome will be made. Logical operators can now be defined to make bit flip corrections based on the error syndrome. It is easy to see that *XXX*|000⟩ = |111⟩ and *XXX*|111⟩ = |000⟩, implying that *XXX* is a logical *X* operator. It is also implied that *ZII* is the logical *Z* operator for three qubits. The full stabilizer group definition is shown in Fig 3.4 and the full bit flip circuit can be found in Fig 3.5. The error syndromes and corresponding corrections are found in Fig 3.3, below.

|  |  |
| --- | --- |
| **Error Syndrome** | **Correction** |
| 00 | No error |
| 01 | Bit flip third qubit |
| 10 | Bit flip second qubit |
| 11 | Bit flip first qubit |

Fig 3.3 – Different error syndromes for the three qubit bit flip code with the corresponding location of the single error and the needed correction.

|  |  |
| --- | --- |
| **Name** | **Operators** |
| *g1* | *XXI* |
| *g2* | *XIX* |
| *ZL* | *ZII* |
| *XL* | *XXX* |

Fig 3.4 – The three qubit bit flip stabilizer group composed of two generators and two logical operators.

Chart

Description automatically generated

Fig 3.52 – The full circuit for the three qubit bit flip code. The encoding can be found on the left on the first three qubits followed by errors in the first set of brackets. Generators then entangle with the ancilla qubits where an error syndrome is made in the middle. A correction is then made in the second set of brackets.

**IV. Five and Seven Qubit Codes**

The two stabilizer codes that will be explored and tested in this project are the five qubit stabilizer and the Steane seven qubit stabilizer. For reasons that will be mentioned soon, the five qubit stabilizer is less transparent than the seven qubit stabilizer. This is because the seven qubit stabilizer is known as a CSS Code which will separate the *X* and *Z* gates in the stabilizer generators. However, the five qubit stabilizer is the minimum number of qubits needed to correct any arbitrary single qubit error. Again, these stabilizers will only work for errors that effect a single qubit.

The five qubit stabilizer is a [5,1,3] code that contains four generators, listed in Fig 4.1, existing in a subspace of dimension 25-4 = 2. Note that the generators, though cyclic in nature, contain combinations of *X* and *Z* gates, making the code slightly more difficult to work with. A special encoding is needed for the logical qubit to work with these generators, which can be defined by **(5)**2 and **(6)** where *gi* is a generator. What is nice about this encoding is that its inverse can be applied after error correction to decode the logical qubit.

|0⟩*L* = ∏(*I*⊗5+ *gi*) |00000⟩ **(5)**

|1⟩*L* = ∏(*I*⊗5+ *gi*) |11111⟩ **(6)**

|  |  |
| --- | --- |
| **Name** | **Operators** |
| *g1* | *ZXXZI* |
| *g­2* | *ZIZXX* |
| *g3* | *XZIZX* |
| *g4* | *XXZIZ* |
| *ZL* | *ZZZZZ* |
| *XL* | *XXXXX* |

Fig 4.1 – Stabilizer generators for the five qubit stabilizer.

The error syndrome calculated from these generators are listed in Fig 4.2. Each measurement syndrome will tell whether an error was a bit flip, phase flip, or combination of the two (*Y* flip), the location of the error, and how to correct that error.

|  |  |
| --- | --- |
| **Classical Measurement** | **Error** |
| 0000 | No Error |
| 0001 | Bit Flip q1 |
| 0010 | Phase Flip q3 |
| 0011 | Bit Flip q0 |
| 0100 | Phase Flip q0 |
| 0101 | Phase Flip q2 |
| 0110 | Bit Flip q5 |
| 0111 | Y Flip q0 |
| 1000 | Bit Flip q2 |
| 1001 | Phase Flip q4 |
| 1010 | Phase Flip q1 |
| 1011 | Y Flip q1 |
| 1100 | Bit Flip q3 |
| 1101 | Y Flip q2 |
| 1110 | Y Flip q3 |
| 1111 | Y Flip q4 |

Fig 4.2 – Error syndromes for the five qubit stabilizer.

The seven qubit stabilizer, [7,1,3], is more optimal to deal with since *X* and *Z* operators can be separated in each generator. The circuit can even be separated into bit flip detection and phase flip detection portions, which is used in the simulation that will be discussed in Sect. V. It contains six generators shown in Fig 4.3, and in this case, can be encoded2 as |0⟩⊗7.

|  |  |
| --- | --- |
| **Name** | **Operators** |
| *g1* | *IIIXXXX* |
| *g2* | *XIXIXIX* |
| *g3* | *IXXIIXX* |
| *g4* | *IIIZZZZ* |
| *g5* | *ZIZIZIZ* |
| *g6* | *IZZIIZZ* |
| *ZL* | *ZZZZZZZ* |
| *XL* | *XXXXXXX* |

Fig 4.3 – Stabilizer generators for the seven qubit stabilizer.

Since the circuit can be separated into bit flip and phase flip components, only three ancilla qubits are needed. Also, since the generators for the bit and phase flips are the same, the error syndromes, shown in Fig 4.4, are the same for each case. Error correction is simpler and the only change that needs to be made is the use of the *X* or *Z* logical operators. In the case that there is a combination of these errors, the first part of the circuit will fix the bit flip component of the error, and the second part of the circuit will fix the phase flip component of the error.

|  |  |
| --- | --- |
| **Classical Measurement** | **Error** |
| 000 | No Error |
| 001 | q5 |
| 010 | q6 |
| 011 | q4 |
| 100 | q3 |
| 101 | q1 |
| 110 | q2 |
| 111 | q0 |

Fig 4.4 – Error syndrome for the seven qubit stabilizer.

**V. Experimental Setup**

The fidelity of a state is a good way to establish confidence that stabilizers are effective in correcting errors. Fidelity is a measurement of how good a state is on a scale of 0 to 1, where 1 is a perfect state. The calculation of a nonpure state can be defined as

|Tr √ √ρi ρf √ρi |2 **(7)**

where ρi is the density matrix of the initial state of the system, and ρf is the density matrix of the final state of the system. Comparing the fidelity of systems subject to stabilizers and those not subject to stabilizers should yield higher fidelity measurements for the former, since stabilizers are meant to increase state fidelity.

The simulation of noisy channels and fidelity tests were written completely in Python3 with IBM Qiskit6 packages. To take advantage of functions that calculate current states and fidelities of the system at various points in time, the Statevector simulator on IBM Aer was utilized. Data analysis was also done with Python3 using Matplotlib libraries to plot the fidelity comparisons.

An error model inspired by a group at Stanford University7 was used to test different errors on the raw qubits and stabilizers. This included functions that applied bit flip, phase flip, *XZ* (*Y)*, and arbitrary errors, which can be found in *error.py*. Note that a Hadamard was placed with the *Z* gate to allow it to take effect on the circuit. The arbitrary error was generated by a random combination of three gates from the group {*I,X,Y,Z*}. Each type of error was then applied to a random single qubit from the logical qubit with some associated probability.

A control group was used to test against circuits using a stabilizer. These were zero, five, and seven raw qubits all initialized to the state |0⟩. These can be found in *control\_qubit.py*. Alongside these raw qubits, the five and seven qubit stabilizers are tested following the set up explained in Sect. IV. In the stabilizer case, the generators are placed on the first qubits in the circuit where the operator on the far left is closest to the ancilla qubits found at the bottom of the circuit. These can be found in *five\_qubit\_stabilizer.py* and *seven\_qubit\_stabilizer.py*, respectfully.

For each type of error, a simulation will be running one hundred times for a variable probability of getting an error on a single qubit. The reason for one hundred trials is that it is enough to mitigate some randomness affecting the data, while staying within the current limitations of computing power. The probability of getting an error will start with 0% (all fidelities remain at 1), where one hundred shots will run, and go to 100% probability of an error with steps of 10%. The average fidelity of each probability step (raw and stabilizer) for the one hundred trials will be plotted against the probability of error for those trials. A function designed in *stabilizer\_main.py* takes the number of shots as the input, and runs this simulation for a specified error type, outputting a graph of fidelity vs. probability of error.

The expectation is that for each type of error, the stabilizers on average should have a higher fidelity than the circuits without stabilizers, meaning that stabilizers are effective in correcting single qubit errors in computation. It is expected that around a 25% probability of error, the stabilizers will become much less effective, and all the cases will trend towards a fidelity of 0.

**VI. Results of Fidelity Tests**

After running the simulation for each type of error, the overall results displayed that on average, the seven qubit stabilizer had the highest fidelity for a probability of error less than 30%. Beyond this threshold, the single raw qubit had the highest fidelity. It was also apparent that as the raw qubits increased in number, their overall fidelity decreased, portraying that the number of qubits is directly proportional to the significance of the effect of an error on the fidelity of a state. Surprisingly, the five qubit stabilizer had the lowest fidelity in each test, and potential reasons for this are discussed in this section. Finally, do notice that at 0% probability of error, all circuits remain at a fidelity of 1, confirming the correct initial and final states of the circuits.

First, looking at Fig 6.1, the five different circuits are exposed to a random single bit flip error. The five and seven raw qubits along with the five qubit stabilizer all dive down to around 0 fidelity around 10% probability of error, with the five raw qubits having a slightly less steep slope, indicating it is more stable. The five qubit case is most likely less steep due to randomness of the experiment since later tests do not indicate that the five qubit case is consistently a higher fidelity than the five qubit stabilizer. The seven qubit stabilizer remains the circuit with the highest fidelity of about 0.7, until around 25% probability of error where the single raw qubit becomes the highest fidelity circuit at around 0.5, where it stabilizes for the remainder of the simulation.

Chart, line chart

Description automatically generated

Fig 6.1 – The fidelities of raw qubits (single, five, and seven), the five qubit stabilizer, and the seven qubit stabilizer all exposed to a random single bit flip error with an increasing probability of finding an error.

Next, the phase flip error was simulated, and results are plotted in Fig 6.2. Here, the seven qubit stabilizer was not as effective, potentially due to the setup of the phase error, and fell sharply to a low fidelity at 15% probability of error. It grew above the other cases briefly at around 40% error probability, likely due to chance since each circuit, except for the single qubit, consistently decays as probability increases. The five and seven raw qubits and the five qubit stabilizer again fell quickly towards a fidelity of 0 around 10% probability of error. The single raw qubit remained the highest fidelity value the entire simulation, stabilizing again around 0.5.

Chart, line chart

Description automatically generated

Fig 6.2 – The fidelities of raw qubits (single, five, and seven), the five qubit stabilizer and seven qubit stabilizers all exposed to a random phase flip error, which also contains a Hadamard gate to allow the error to affect the states. The fidelity of the states is plotted against an increasing probability error.

Moving onto Fig 6.3, the random single bit and phase flip error (*Y* error) is placed in each circuit. A similar trend to the bit flip case is seen here. The five and seven raw qubits along with the five qubit stabilizer all going to a fidelity of 0 at 10% probability of error. The seven qubit stabilizer remains the highest fidelity case around a fidelity of 0.8 until the single raw qubit becomes the highest fidelity case around 15% probability of error, where it again stabilizes at 0.5 fidelity. However, the seven qubit stabilizer and single raw qubit remain very close in fidelity until around 30% probability of error in which the single qubit overtakes the seven qubit stabilizer.

Chart, line chart

Description automatically generated

Fig 6.3 – The fidelities of raw qubits (single, five, and seven), the five qubit stabilizer, and the seven qubit stabilizer all exposed to a random single bit and phase flip error (Y error) with an increasing probability of finding an error.

Finally, looking at the arbitrary error case in Fig 6.4, the previous trends are apparent once again. The seven raw qubits and the five qubit stabilizer all dive to a fidelity of 0 around 10% probability of error. This time, the five raw qubits take a dive to 0 around 20% probability of error, but this could be due to the random nature of the experiment. The seven qubit stabilizer remains the highest fidelity case of around 0.7 until around 50% probability of error where again the single raw qubit takes over at a fidelity of 0.5.

Chart, line chart

Description automatically generated

Fig 6.4 – The fidelities of raw qubits (single, five, and seven), the five qubit stabilizer, and the seven qubit stabilizer all exposed to a random single arbitrary error with an increasing probability of finding an error.

**VII. Analysis and Conclusions**

Overall, the prior expectations were not completely met with the simulation results. The seven qubit stabilizer did consistently outperform other circuits, but the five qubit stabilizer was unsuccessful in correcting errors, and the single raw qubit consistently remained around 0.5 fidelity, even with 100% probability of error.

As previously expected, the seven qubit stabilizer was remotely successful in correcting errors. It remained at a high fidelity until around 25-30% probability of error and then slowly trended towards 0 fidelity. This confirms that the seven qubit stabilizer is effective in correcting errors up to a significant probability of error. Most hardware today will have probabilities of error much lower than 25%, which makes this stabilizer optimal to use over single or multiple raw qubits. One downside to the approach used for the seven qubit stabilizer, here, is that it is effective for circuits initialized to ∏ |0⟩⊗I, which most circuits are not. This leads one to believe that a better encoding needs to be implemented, so that an arbitrary single qubit state could effectively be used in the seven qubit stabilizer. Using an encoding like the three qubit case (superposition and CNOT gates) caused the stabilizer to fail at correcting errors, so more research would need to be focused into this encoding.

On the other hand, the five qubit stabilizer was ineffective in correcting single qubit errors. On average, the five qubit stabilizer became ineffective around 10% probability of error and was just as effective as the raw qubit cases. There could be several reasons why this stabilizer failed such as discrepancies with the logical encoding and the stabilizer generators. Such a discrepancy can have qubits not under the influence of an error be a -1 eigenstate of the stabilizer group, meaning errors would have been detected where there were none, propagating wrong states to the point of fidelity measurement. A more thorough analysis of the encoding and decoding could shed light on any discrepancies with the stabilizer generators. Other encoding methods could also be tested such as rotations to the generators and logical qubits.

Another flaw in the simulation is that ancilla qubits were included in the state fidelities. The Statevector simulator was essential to this simulation to read the current state of the system at various points, but no method to either only include certain qubits in a statevector, or trace over certain qubits was not found in Qiskit documentation. This could create a negative impact on the true effectiveness of the five qubit stabilizer since ancillas are not actually part of the state we desire to do computation on. Further research into the Statevector simulator and tracing out ancilla qubits could solve this issue.

One last potential factor that could be negatively affecting the five qubit stabilizer are the number of gates required to encode logical qubits and build generators. When errors are placed on these encodings, there many single and two qubit gates in which errors can propagate through the circuit. The implementation of the seven qubit stabilizer uses significantly less gates than the five qubit stabilizer, which may speak for why the five qubit stabilizer is much less successful.

Though the five and seven raw qubit circuits fell quickly to a fidelity of 0, which was expected, the single raw qubit did not. It would be expected that a single raw qubit would fall the quickest to 0 since a single error would have the largest effect on the fidelity of the state. It is expected to have a single qubit measure a fidelity of about 0.5 during the beginning of the simulation since half the time, the state will have a fidelity of 1, and the rest of the time, it will have a fidelity of 0. A possible reason that the single qubit remains around 0.5 for the entirety of the test could be a flaw in the set up of the test. If an error propagates on the single qubit and another error is placed afterwards, this could reverse the effect of the error, essentially correcting the error. This flaw could also have impact on the testing of the stabilizers and negatively impact the effectiveness of the stabilizers. Placing measurements of the state at different points in time during the test could prevent a propagation like this that would reverse errors. It was tested placing a measurement at the initialization of the circuit and at the start of each trial, but both placements had a similar effect on the output graphs.

Overall, results show that the seven qubit stabilizer is effective, and the five qubit stabilizer is not effective. These could be due to flaws in the experimental set up, limitations of the quantum framework, and overall designs of the stabilizers. Through more careful reworking of the simulation itself, and research of the encoding of a five qubit stabilizer for future tests, the ideal result that both stabilizers are consistently more effective than raw qubits may be more likely to be observed.

The GitHub repository of the simulation and error model can be found at <https://github.com/gwjacobson/QuantumErrorCorrection.git>.

1. Nielson, M., Chuang, I. Quantum Computation and Quantum Information. *Cambridge (UK): Cambridge University Press.* **10,** 426-474 (2016).

2. Devitt, S., Munro, W., Nemoto, K. Quantum Error Correction for Beginners. *Quant-ph.* (2013).

3. Gottesman, D. Stabilizer Codes and Quantum Error Correction. *California Institute of Technology (US): Thesis.* (2004).

4. Bacon, D. CSE 599d – Quantum Computing Stabilizer Quantum Error Correction Codes. *University of Washington (US).*

5. Wilde, M. On the logical operators of quantum codes. *Quant-ph.* **0903.5256v1** (2009).

6. Abraham, H., Agarwal, R., … , Zoufal Christa. Qiskit: An Open-source Framework for Quantum Computing. *IBM Quantum.* **5.1**, (2019).

7. Chandak, S., Mardia, J., Tolunay, M. Implementation and analysis of stabilizer codes in pyQuil. *Stanford University: Project*. (2019).

8. Dankert, C. Efficient Simulation of Random Quantum States and Operators. *University of Waterloo: quant-ph.* **0512217v2** (2005).

9. Steane, A.M. Error Correcting Codes in Quantum Theory. *Phy. Rev. Let.* **77,** 793-797 (1996).

10. Laflamme, R., Miquel, C., Paz, J., Zurek, W. Perfect Quantum Error Correction Code. *quant-ph.* **9602019v1** (1996).

11. Knill, E. Quantum computing with realistically noisy devices. *Nature Publish. Group.* **434,** 39-44 (2005).

12. Katabarwa, A., Geller, M. Logical error rate in the Pauli twirling approximation. *Scientific Reports.* **5:14670,** 1-6 (2015).

13. Raussendorf, R. Key ideas in quantum error correction. *Phil. Trans. of the Royal Society.* **370,** 4541-4565 (2012).