Quantum error correction scheme for fully-correlated noise

Chi-Kwong Li · Yuqiao Li · Diane Christine Pelejo · Sage Stanish

Received: 16 March 2022 / Accepted: 30 May 2023 / Published online: 16 August 2023
© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2023

Abstract

This paper investigates quantum error correction schemes for fully-correlated noise channels on an \( n \)-qubit system, where error operators take the form \( W^\otimes n \), with \( W \) being an arbitrary \( 2 \times 2 \) unitary operator. In previous literature, a recursive quantum error correction scheme can be used to protect \( k \) qubits using \((k + 1)\)-qubit ancilla. We implement this scheme on 3-qubit and 5-qubit channels using the IBM quantum computers, where we uncover an error in the previous paper related to the decomposition of the encoding/decoding operator into elementary quantum gates. Here, we present a modified encoding/decoding operator that can be efficiently decomposed into (a) standard gates available in the qiskit library and (b) basic gates comprised of single-qubit gates and CNOT gates. Since IBM quantum computers perform relatively better with fewer basic gates, a more efficient decomposition gives more accurate results. Our experiments highlight the importance of an efficient decomposition for the encoding/decoding operators and demonstrate the effectiveness of our proposed schemes in correcting quantum errors. Furthermore, we explore a special type of channel with error operators of the form \( \sigma_x^\otimes n \), \( \sigma_y^\otimes n \) and \( \sigma_z^\otimes n \), where \( \sigma_x, \sigma_y, \sigma_z \) are the Pauli matrices. For these channels, we implement a hybrid quantum error correction scheme that protects both quantum and classical information using IBM’s quantum computers. We conduct experiments for \( n = 3, 4, 5 \) and show significant improvements compared to recent work.

Keywords Quantum error correction · IBM quantum · Qiskit · Quantum channels · Noise · Decoherence
1 Introduction

Quantum information science concerns the use of quantum systems as computational resources to store, communicate, and process information. One of the obstacles for building quantum computers is decoherence, which is a process caused by the coupling between a quantum system and its environment. A pure state, to be used as a computational resource, becomes a dirty mixed state due to decoherence, which makes the computational outcome unreliable. There are different strategies to fight against decoherence and quantum error correcting codes (QECC) is one of them. The main idea of QECC is to embed quantum information into a higher dimensional Hilbert space so that either

(i) the error acting on the physical qubits may be identified by introducing the error syndrome measurement qubits, so that the initial quantum information is recovered after applying appropriate corrections, or
(ii) the error operator acts only on a part of the Hilbert space, keeping the initial quantum information intact.

The second QECC scheme is often called the “error-avoiding” coding scheme. Decoherence free subspace (DFS) and noiseless subsystem (NS) are two popular examples of error-avoiding QECC schemes [1–14].

In this paper, we consider the second approach to deal with quantum channels in which all physical qubits involved in coding suffer from the same error operators with a certain probability distribution. Such a channel is what we refer to as a channel with fully-correlated noise. Mathematically, if we let $U(2)$ be the group of $2 \times 2$ unitary matrices, and $\mu$ be a probability measure on $U(2)$, then a channel with fully correlated noise can be represented as an operator $\Phi$ that transforms an $n$-qubit state $\hat{\rho}$ into

$$\Phi(\hat{\rho}) = \int W^{\otimes n} \hat{\rho} (W^{\otimes n})^\dagger d\mu(W). \quad (1)$$

As mentioned in [15], there are two relevant cases in which such error operators are in action; (i) when the size of a quantum computer is much smaller than the wavelength of the external disturbances, and (ii) when photonic qubits are sent one by one through an optical fiber with a fixed imperfection. In both cases, the qubits suffer from the same errors leading to decoherence. Another instance in which such encoding is useful is when Alice sends quantum information to Bob, possibly billions of light years away, without knowing which basis vectors Bob employs. In this case, mismatching of the basis vectors is common for all qubits and such mismatching is regarded as collective noise.

In [16], an explicit recursive implementation of encoding/decoding circuits for an arbitrary number $n$ of physical qubits was presented. Remarkably, the scheme depends only on the algebra generated by the error operators $W^{\otimes n}$ with $W \in U(2)$, but not affected by the probability measure $\mu(W)$. The study demonstrates that for an $n$-qubit channel with $n = 2k + 1$, the scheme can protect data encoded in $k$ logical qubits using the other $k + 1$ qubits. This leads to the asymptotic encoding rate of $1/2$. In [17], the maximum dimension $2^k$ of the error correction code corresponding to the subspace
in $\mathbb{C}^{2^n}$ immune to collective noise operators of the form $W^\otimes n$ was determined. The study shows that the encoding rate $k/n$ approaches 1 as the positive integer $n$ gets much larger. Nevertheless, as mentioned in [15], the recursive scheme in [16] is more practical compared to other schemes; for example, see [11, 18].

In this paper, we implement the recursive QECC scheme proposed in [19] for channels with fully-correlated noise using the IBM quantum computers and study the mathematical issues associated with the process. During our investigation, we identify an error in the decomposition of the encoding/decoding operator into elementary quantum gates as illustrated in the quantum circuits in [19]; see also [15]. We fix the error by finding a new encoding operator $U$ for the fully-correlated channels on 3 qubits, which is the base case for the recursive scheme. In particular, the matrix $U$ can be decomposed into a product of three CNOT gates, a single-qubit gate and two controlled-gates, which are standard gates available in qiskit—the Python library used to interface with the IBM quantum computers. We then implement the scheme using the IBM quantum computers for the fully-correlated channels on 3 qubits and 5 qubits. However, the numerical results vary among different quantum computers. It turns out that the IBM quantum computers would decompose the standard gates into more basic gates to run the program, and the decomposition varies on different machines on different runs. In response, we further decompose our encoding operator $U$ into a product of 6 CNOT gates and 8 single qubit gates before feeding the circuit into the IBM quantum computers. Intriguingly, qiskit has an internal algorithm that further adjusts the decomposition based on the specific machine being utilized.

To deepen our understanding of the errors in the implementation process, we compare our numerical results on different quantum computers and their associated factors such as qubit connections, gate errors, decoherence time, etc. In Sect. 2, we will present and analyze these results.

In Sect. 3, we consider the special case of fully-correlated channels that use only the error operators $W^\otimes n$ for $W = I_2, \sigma_x, \sigma_y, \sigma_z$, where $\sigma_x, \sigma_y, \sigma_z$ are the Pauli matrices and $I_2$ is the $2 \times 2$ identity matrix. As mentioned before, the general recursive scheme is good for fully correlated quantum channels with any probability distribution $D\mu(U)$. In this special case, the probabilities are nonzero only for the four special choices of $W$, and thus it simplifies the error structure significantly. As a result, we can have a much more efficient quantum error correction scheme. In [20], a hybrid quantum error correction scheme was proposed for this special case, where a 1-qubit ancilla can be used to protect $2k$ qubits of quantum information, and a 2-qubit ancilla can be used to protect $2k$ qubits of quantum information and 2 bits of classical information. While the scheme was implemented on IBM quantum computers in [20], the results were not satisfactory for 4-qubit and 5-qubit channels. In our study, we present an improved implementation of the hybrid scheme with significantly better computational results. Furthermore, we analyze the error patterns of different quantum computers with different qubit connections, gate errors, decoherence time, etc.

Finally, in Sect. 4, we conclude this paper with a short summary of our work and a discussion of future research topics.
2 The recurrence scheme and the correction of previous error

Denote the space of all $N \times N$ complex matrices by $M_N$ and consider the fully-correlated channel $\Phi : M_{2^n} \rightarrow M_{2^n}$ defined in (1) with $n = 2k + 1$. In [16] (see also [15]), a recursive quantum error correction scheme was presented for protecting $k$-qubits. Suppose we wish to protect $k$-qubits of data, realized as a density matrix $\rho$ in $M_{2^k}$. First, we will embed the information into a higher-dimensional Hilbert space having initial state $\rho \otimes \sigma$ where $\sigma = |u\rangle\langle u|$ and $|u\rangle = |0\rangle^\otimes k \otimes |v_0\rangle$ is the product state of $k$ copies of the pure state $|0\rangle$ and one arbitrary qubit $|v_0\rangle$. Then, we apply a carefully-chosen encoding operator $E : M_{2^n} \rightarrow M_{2^n}$ such that the encoded state is equal to $E(\rho \otimes \sigma)$ and the decoded state of the system after going through the noisy channel is

$$E^{-1} \circ \Phi \circ E(\rho \otimes \sigma) = \rho \otimes \left(|0\rangle \langle 0|\right)^\otimes k \otimes |v_1\rangle\langle v_1|.$$

Finally, we can take a partial trace to recover the data state $\rho$.

It is worth noting that in some scenarios, error correction may need to be done multiple times, such as periodically when the data state $\rho$ is attacked by the correlation error regularly before the computation or transmission process is done. In such cases, there is no need to do the encoding and decoding multiple times, as doing so may cause additional errors. One only needs to let the encoded state $E(\rho \otimes \sigma)$ stay in the environment, going through $m$ rounds of error attack, and then apply the decoding scheme $E^{-1}$ to obtain the final decoded state

$$(E^{-1} \Phi^m E)(\rho \otimes \sigma) = (E^{-1} \Phi E)^m(\rho \otimes \sigma) = \rho \otimes \left(|0\rangle \langle 0|\right)^\otimes k \otimes |v_m\rangle\langle v_m|.$$

In the following, we will use the IBM quantum computers to implement the recursive quantum error correction schemes for the fully correlated channel for 3-qubits ($k = 1$) and 5-qubits ($k = 2$).

2.1 Three-qubit case

For convenience, we will reorder the positions of the data qubits and the ancilla qubits. The basic case is when $k = 1$, and the encoding operation is done by

$$\hat{E}(|0\rangle \langle 0| \otimes \rho \otimes |v_0\rangle\langle v_0|) = U \left(|0\rangle \langle 0| \otimes \rho \otimes |v_0\rangle\langle v_0|\right) U^\dagger,$$

where $\rho = |\psi\rangle\langle \psi|$ is the data qubit to be transmitted, $|v_0\rangle$ is an arbitrary qubit, and the unitary matrix $U$ is chosen such that for any $W \in U(2)$, we have

$$U^\dagger (W \otimes W \otimes W) U = \mu_W \left((I_2 \otimes W) \otimes F_W\right) = \mu_W \left(|0\rangle \langle 0| \otimes (I_2 \otimes W) + |1\rangle \langle 1| \otimes F_W\right),$$

(2)
Quantum error correction scheme for fully-correlated... Page 5 of 25

Fig. 1 (a) Shows the circuit diagram for the error correction scheme presented in Eq. (3) while figure (b) Is an example of a unitary matrix $U$ that satisfies Eq. (2).

In other words, one can use an arbitrary qubit $|v\rangle$ and the qubit $|0\rangle$ to protect a given quantum bit $|\psi\rangle$. This effect is illustrated in the circuit diagram in Fig. 1a. This scheme was utilized in [15] with the unitary matrix $U$ given in Fig. 1b.

To implement the quantum error correction scheme, it is necessary to decompose the encoding operator $U$ into elementary gates/operations that the IBM quantum computers can carry out. In [16], the matrix $U$ in Fig. 1b was supposed to have a simple realization by the circuit diagram in Fig. 2, where the part surrounded by the broken line is not needed if $|v\rangle = |0\rangle$, as claimed in the paper. However, we found that the product of the 6 elementary gates do not actually produce the matrix $U$ as asserted (see “Appendix 1”). With some effort, we identified a different decomposition of the unitary matrix in Fig. 1b to construct a circuit diagram for implementation. However, it requires 9 standard gates in the IBM quantum computer library including the use of a Toffoli gate, and the implementation of the scheme using the IBM quantum computers did not yield satisfactory results.

$$U^\dagger(W \otimes W \otimes W)U\left(|0\rangle|\psi\rangle|v\rangle\right) = \mu_W\left[(I_2 \otimes W) \oplus F_W\right]\left(|0\rangle|\psi\rangle|v\rangle\right).$$
Here, $U$ is the matrix in (4) and $\alpha = 2 \arcsin(\sqrt{1/3})$

To get around the problem, we modify the encoding matrix $U$, satisfying Eq. (2), given by

$$U = \begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\
\sqrt{2/3} & 0 & 0 & 0 & \sqrt{1/3} & 0 & 0 & 0 \\
-\sqrt{1/6} & 0 & \sqrt{1/2} & 0 & \sqrt{1/3} & 0 & 0 & 0 \\
0 & \sqrt{1/6} & 0 & \sqrt{1/2} & 0 & -\sqrt{1/3} & 0 & 0 \\
-\sqrt{1/6} & 0 & -\sqrt{1/2} & 0 & \sqrt{1/3} & 0 & 0 & 0 \\
0 & \sqrt{1/6} & 0 & -\sqrt{1/2} & 0 & -\sqrt{1/3} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & \sqrt{2/3} & 0 & 0 & \sqrt{1/3} & 0 \\
0 & 0 & 0 & 0 & \sqrt{2/3} & 0 & 0 & 1
\end{pmatrix}.$$

(4)

It is worth noting that the modification of the unitary matrix was based on the fact that any unitary matrix $U$ with the first four columns equal to that of the unitary matrix in Fig. 1b will satisfy Eq. (2).

Now, for the matrix $U$ in (4), we have $U = P_1 P_2 P_3 Q_1 Q_2 Q_3$, where $Q_1$ is the single qubit gate $Q_1 = \sigma_z \otimes I_4$; while $Q_2$ and $Q_3$ are the controlled gates

$$Q_2 = \frac{1}{\sqrt{2}} \left( I_2 - I_2 \right) \oplus I_4 \quad Q_3 = \begin{pmatrix}
-\sqrt{1/3} & 0 & \sqrt{2/3} & 0 \\
0 & 1 & 0 & 0 \\
\sqrt{2/3} & 0 & \sqrt{1/3} & 0 \\
0 & 0 & 0 & 1
\end{pmatrix} \otimes I_2;$$

and $P_1, P_2$ and $P_3$ are the CNOT gates

$$P_1 = \begin{pmatrix}
0 & 0 & I_2 & 0 \\
0 & I_2 & 0 & 0 \\
I_2 & 0 & 0 & 0 \\
0 & 0 & 0 & I_2
\end{pmatrix}, \quad P_2 = I_4 \oplus (I_2 \otimes \sigma_x), \quad P_3 = I_2 \otimes \begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0
\end{pmatrix}.$$

This decomposition is illustrated in the circuit diagram in Fig. 3. (See “Appendix 2” for a Matlab script to verify this decomposition.) Moving forward, we will refer to this as the standard gates decomposition of $U$.

Using this decomposition, we implement the error correction scheme illustrated in Fig. 1a using six IBM quantum computers: ibmq_valencia, ibmq_santiago, ibmq_vigo, ibmq_5_yorktown, ibmq_ourense and ibmq_athens. The results are shown in Fig. 4a. Here, we set $W$ to be the Hadamard gate $H$ defined as:

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix}
1 & 1 \\
0 & 0
\end{pmatrix}.$$
Quantum error correction scheme for fully-correlated... Page 7 of 25 310

Fig. 4 (a) and (b) Show results obtained when implementing the QECC scheme in Fig. 1(a) on six IBM quantum computers (ibmq valencia, ibmq santiago, ibmq vigo, ibmq athens, ibmq ourense, ibmq 5 york-town). For Fig. 4(a) we feed qiskit the standard gates decomposition of $U$ illustrated in Fig. 3 and for Fig. 4(b), we feed qiskit the basic gates decomposition of $U$ illustrated in Fig. 5

![Figure 4](image)

![Figure 5](image)

**Fig. 5** Decomposition of the matrix $U$ in (4) as a product of 6 CNOT gates and 8 single qubit gates

and we measure only the data qubit (middle qubit). The results are quite satisfactory as shown in Fig. 4a. Specifically, the protected qubit is set to $|0\rangle$ and most outputs show a measurement of $|0\rangle$ higher than 80%. Furthermore, we also performed experiments using alternative choices of $W$ and obtained comparable results.

It turns out the IBM quantum computers will further decompose the standard gates into basic gates when they execute the encoding and decoding scheme. These decompositions will usually create many CNOT gates. (For example, see the circuit diagrams produced by ibmq_valencia in Fig. 19b of “Appendix 3”). In view of this, we try to improve the accuracy by decomposing $U$ into a product of 6 CNOT gates and 8 single qubit gates as shown in Fig. 5. (See “Appendix 2” for a Matlab script to verify this decomposition.) From here on out, we refer to this decomposition of $U$ as the **basic gates decomposition of $U$**.

As shown in Fig. 4b, we obtain improvements in all the machines except for ibmq_valencia when we use the basic gates decomposition. It is also worth noting that the IBM quantum computers will still make small changes before carrying out the user-specified decomposition of the scheme. (See Fig. 19d in “Appendix 3”).
2.2 Five-qubit case

As shown in [15, 16], one can then apply the error correction scheme recursively to protect two qubits of data using a 3-qubit ancilla having a product state $|0\rangle|0\rangle|v\rangle$, where $|v\rangle$ is an arbitrary qubit. The circuit diagram used in the error correction scheme is shown in Fig. 6. Recursively, we can protect $k$ data qubits using $k+1$ ancillary qubits having a product state $|0\rangle^\otimes k|v\rangle$, where $|v\rangle$ is an arbitrary qubit.

Setting $W = H$, $|\psi_1\rangle = |\psi_2\rangle = |0\rangle$, and using the decomposition in Fig. 6, we get the results shown in Fig. 7 upon measurement of the data qubits (second and fourth qubit positions). On the other hand, if we utilize the decomposition of $U$ given in Fig. 5, we obtain the results shown in Fig. 8. We observe that the results using the basic gates decomposition of $U$ in Fig. 5 are worse than those using the standard gates decomposition of $U$ in Fig. 3. This suggests that the internal algorithm used by the IBM computers to decompose standard gates into basic gates work well with the machines even though more CNOT gates are used.

We conducted the same experiments a few more times and used a variety of error operators; readers can view the results of these additional experiments in “Appendix 4”. The Jupyter notebooks used to run these experiments are also available in the following Github repository:
To further understand the performance of our scheme on different IBM quantum computers, we compare the results in relation to factors such as gate errors, qubit connections, decoherence time of the machines.

3 Correlated channels with error operators $\sigma_x^{\otimes n}, \sigma_y^{\otimes n}, \sigma_z^{\otimes n}$

In [20], a hybrid quantum error correction scheme for fully-correlated channels with error operators $\sigma_x^{\otimes n}, \sigma_y^{\otimes n}, \sigma_z^{\otimes n}$ was implemented using the IBM quantum computers. Theoretically, this scheme allows the use of a single arbitrary ancilla to protect $n - 1$ data qubits if $n$ is odd, and use two classical bits to protect $n - 2$ data qubits and yet preserving the two classical bits. However, their numerical experiments using the IBM quantum computers failed to produce good results for $n = 4$ and $n = 5$. Here, we conducted additional experiments and discovered that we can indeed obtain reasonable results as shown in “Appendix 4”. Conceivably, the IBM quantum computers may have undergone improvements since our initial implementation. For ease of reference, we provide the circuit diagrams for the recursive scheme in Figs. 9, 10, 11 and 12.

Here $P_n$ denotes the encoding matrix for the $n$-qubit case. That is, $P_2 = \frac{1}{\sqrt{2}} (I_2 \otimes \sigma_z + \sigma_x \otimes \sigma_x)$ and $P_3$ is the permutation matrix such that for any $a, b, c \in \{0, 1\}$, $P_3|abc\rangle = |a \oplus c\rangle|a \oplus b\rangle|a \oplus b \oplus c\rangle$ or

$$P_3 \begin{bmatrix} v_{000} & v_{001} & v_{010} & v_{011} & v_{100} & v_{101} & v_{110} & v_{111} \end{bmatrix}^T = \begin{bmatrix} v_{000} & v_{111} & v_{101} & v_{010} & v_{110} & v_{001} & v_{011} & v_{110} \end{bmatrix}^T$$
Fig. 9 Hybrid QECC when $n = 2$ and $|q_0 q_1 \rangle \in \{00\}, \{01\}, \{10\}, \{11\}$
| Machine     | No. of CNOTS | No. of SQG | Run Time (sec) | Error     |
|-------------|--------------|------------|---------------|-----------|
| ibmq_london | 64           | 43         | 26.2          | 0.42181   |
| ibmq_london | 51           | 48         | 27            | 0.45883   |
| ibmq_london | 55           | 48         | 26.7          | 0.51326   |
| ibmq_london | 69           | 48         | 26.1          | 0.52089   |
| ibmq_london | 46           | 43         | 26.4          | 0.37490   |
| ibmq_london | 75           | 48         | 26.2          | 0.5896    |
| ibmq_london | 56           | 48         | 26.9          | 0.63888   |
| ibmq_london | 74           | 47         | 26.4          | 0.811697  |
| ibmq_london | 57           | 47         | 26.5          | 0.61646   |
| ibmq_london | 70           | 42         | 26.4          | 0.61826   |
| ibmq_london | 49           | 37         | 26.2          | 0.61963   |
| ibmq_london | 49           | 37         | 26.1          | 0.61057   |
| ibmq_london | 60           | 32         | 26.2          | 0.62988   |
| ibmq_london | 68           | 32         | 26.7          | 0.64388   |
| ibmq_london | 73           | 37         | 26.2          | 0.53774   |
| ibmq_london | 74           | 48         | 26.2          | 0.55885   |
| ibmq_london | 56           | 48         | 26.9          | 0.66662   |
| ibmq_london | 73           | 37         | 26.9          | 0.71187   |
| ibmq_london | 75           | 48         | 27.1          | 0.67859   |
| ibmq_london | 61           | 37         | 27.2          | 0.85847   |
| ibmq_london | 70           | 32         | 25.8          | 0.69874   |
| ibmq_london | 84           | 49         | 26.2          | 0.52064   |
| ibmq_london | 85           | 37         | 26.7          | 0.73737   |
| ibmq_london | 76           | 37         | 25.7          | 0.70727   |
| ibmq_london | 76           | 37         | 26.2          | 0.70141   |
| ibmq_london | 76           | 37         | 26.8          | 0.71545   |
| ibmq_london | 76           | 37         | 26.4          | 0.72277   |
| ibmq_london | 77           | 37         | 26.2          | 0.7206   |
| ibmq_london | 78           | 37         | 27.4          | 0.75379   |

| Machine     | No. of CNOTS | No. of SQG | Run Time (sec) | Error     |
|-------------|--------------|------------|---------------|-----------|
| ibmq_london | 96           | 48         | 28.5          | 0.46703   |
| ibmq_london | 53           | 37         | 27.7          | 0.48080   |
| ibmq_london | 59           | 37         | 28.3          | 0.50076   |
| ibmq_london | 54           | 38         | 28.8          | 0.58974   |
| ibmq_london | 84           | 37         | 26.2          | 0.50263   |
| ibmq_london | 66           | 37         | 27.6          | 0.59995   |
| ibmq_london | 55           | 37         | 29.5          | 0.8006   |
| ibmq_london | 78           | 48         | 26.6          | 0.5826    |
| ibmq_london | 57           | 37         | 26.9          | 0.5876    |
| ibmq_london | 73           | 37         | 26.1          | 0.59496   |
| ibmq_london | 70           | 37         | 29.9          | 0.60022   |
| ibmq_london | 70           | 37         | 25.8          | 0.60099   |
| ibmq_london | 72           | 38         | 28.8          | 0.61486   |
| ibmq_london | 57           | 37         | 26.7          | 0.61347   |
| ibmq_london | 70           | 32         | 27            | 0.61572   |
| ibmq_london | 66           | 37         | 28.1          | 0.61021   |
| ibmq_london | 56           | 37         | 23.5          | 0.62451   |
| ibmq_london | 61           | 32         | 27            | 0.63041   |
| ibmq_london | 73           | 48         | 27.2          | 0.63977   |
| ibmq_london | 55           | 37         | 29.2          | 0.64888   |
| ibmq_london | 79           | 37         | 27.5          | 0.64736   |
| ibmq_london | 79           | 44         | 28.9          | 0.64807   |
| ibmq_london | 76           | 37         | 27.4          | 0.64496   |
| ibmq_london | 78           | 37         | 28.8          | 0.65845   |
| ibmq_london | 74           | 37         | 28.8          | 0.66085   |
| ibmq_london | 76           | 37         | 26.8          | 0.67787   |
| ibmq_london | 74           | 37         | 26            | 0.84342   |
| ibmq_london | 77           | 37         | 26.2          | 0.79223   |
| ibmq_london | 74           | 37         | 26.1          | 0.79125   |
| ibmq_london | 85           | 37         | 27.9          | 0.71521   |
| ibmq_london | 57           | 37         | 29.1          | 0.71863   |

Fig. 10 Hybrid QECC when n = 3, where |q2⟩ can be any qubit state.
Fig. 11 Hybrid QECC for odd $n$, where $|q_{n-1}\rangle$ can be any qubit state

Fig. 12 Hybrid QECC for even $n$ and $|q_{n-1}q_{n-2}\rangle \in \{|00\rangle, |10\rangle, |01\rangle, |11\rangle\}$

and for $k \geq 2$,

$$P_{2k} = (I_2 \otimes P_{2k-1})(P_2 \otimes I_{2^{k-2}})$$

and

$$P_{2k+1} = (I_4 \otimes P_{2k-1})(P_3 \otimes I_{2^{k-2}})$$

3.1 Experimental results using Qiskit

The Jupyter notebooks used to run these experiments are also available in the following Github repository:

https://github.com/dcpelejo/QECC/tree/main/XYZ%20hybrid%20experiments

3.1.1 Pure state, arbitrary state, and improvements

First, we implemented the QECC scheme for $n = 3$ as illustrated in Fig. 10. The results improved those in [20] as shown in Fig. 13a. Note that experiments on this section were done on ibmq_burlington.

Besides using the pure state $|0\rangle$ as the protection qubit, we may use any other qubit state. From Fig. 13b, we observe that the results using the protection qubit $|q_2\rangle = |0\rangle$ and using $|q_2\rangle = R_y(\frac{3\pi}{4})|0\rangle$ are fairly similar. Since the quantum channel only has error operator $W^{\otimes n}$ for $W = \{I_2, \sigma_x, \sigma_y, \sigma_z\}$, it is not surprising that we have a more effective scheme compared to the one in Sect. 2. Moreover, we only need to use one ancilla qubit to protect two qubits, and the experimental results are better than that of the general scheme which uses three ancilla qubits to protect two qubits as shown in Section 2.2.

Next, we implemented the scheme for $n$ qubits with $n = 4, 5$. Now, for the 4-qubit channel, we may use two pure classical ancillas or two arbitrary ancillas to protect two qubits. The experimental results are similar as shown in Fig. 14a, b. Note that if we want to protect two classical bits of information encoded in the ancillas as well, then we must use two pure classical ancillas. The result of experiments where the
Fig. 13  3-qubit QECC with circuit diagram illustrated in (10)

Fig. 14  4-qubit QECC with circuit diagram illustrated in (12)

protection ancillas $|q_3q_2\rangle$ are set to be $|01\rangle$, $|10\rangle$ or $|11\rangle$ can be found in “Appendix 5”.

Figure 15a, b show the 5-qubit scheme with pure classical and arbitrary state protection qubits respectively. In [20], the experimental results for 4 and 5-qubit schemes were not satisfactory. However, our new experiments produced significantly improved outcomes, as demonstrated in our current findings.

3.1.2 Errors across different IBM machines

We note that our implementation uses only CNOT gate and Hadamard gates, which are basic gates in the IBM quantum computers. We posited that the choice of whether to use arbitrary or pure classical state protection qubits will not impact our results. Therefore, in this subsection, we will focus on letting the protection qubits be any arbitrary state since, in practice, an arbitrary qubit is less expensive to prepare than pure classical ones.

For our numerical experiments in this subsection, we use 5 different IBM machines: ibmq_santiago, ibmq_vigo,ibmq_valencia, ibmq_ourense, and ibmq_yorktown. In [20], ibmq_yorktown does not yield satisfactory results.
Results of $\sigma_x \otimes \chi^3$, $\sigma_y \otimes \chi^3$, $\sigma_z \otimes \chi^3$ errors with arbitrary state protection in 5 IBM machines. However, we can see the improvement of this machine now, as it produces reasonable results using our implementation (Fig. 16).

We would like to note that since the two protection qubits are corrupted in the 4-qubit scheme, only the other two qubits are measured (Figs. 17, 18).

From the figures above, we can conclude that ibmq_ourense is the best machine for 3-qubit schemes and ibmq_santiago is best for 4 and 5 qubit schemes. Also, results from all machines match our predictions.
Fig. 17  Results of $\sigma_0^{\otimes 4}$, $\sigma_1^{\otimes 4}$, $\sigma_2^{\otimes 4}$ errors with arbitrary state protection in 5 IBM machines

Fig. 18  Results of $\sigma_0^{\otimes 5}$, $\sigma_1^{\otimes 5}$, $\sigma_2^{\otimes 5}$ errors with arbitrary state protection in 5 IBM machines
4 Conclusion and further research

In our study, we implemented a general recursive quantum error correction scheme for fully-correlated channels on \( n \)-qubits with error operators of the form \( W^{\otimes n} \) using different IBM quantum computers. This scheme was proposed in earlier papers, where an erroneous decomposition of the encoding operator was given for the 3-qubit channel. We modified the encoding operator so that it can be decomposed as the product of simple standard quantum gates which the IBM quantum computers can readily implement. We compared the errors on different IBM quantum computers, and tried to find out the key factors that will affect the accuracy of the results. Furthermore, we decomposed the encoding matrix as the product of basic gates, namely, CNOT gates and single unitary gates, and improved the results in all but one of the IBM quantum computers we used. Then, we implement the recurrence scheme for 5-qubit channels. It was somewhat surprising that better results were obtained by the standard gates decomposition instead of the basic gate decomposition despite the fact that much more CNOT gates were used in the former decomposition.

We also implemented a hybrid quantum error correction scheme for the subclass of fully-correlated channels where the error operators have the special form \( \sigma_x^{\otimes n}, \sigma_y^{\otimes n}, \sigma_z^{\otimes n} \) for the \( n \)-qubit channels when \( n = 4, 5 \). The scheme was implemented and good results were obtained, which covered cases a previous paper failed to handle.

There are a number of future research directions worth pursuing and we suggest a few of them in the following.

1. Implement the QECC schemes in our study for \( n \)-qubit channels for higher \( n \) by improving our schemes or finding better quantum computers.
2. Implement other quantum error correction schemes on IBM or other quantum computers.
3. In our study, we performed numerical experiments using different IBM quantum computers aiming to test how the architecture (e.g. network connections between the qubit nodes, range of approximate gate or measurement errors) of the quantum computers may perform differently, and to examine if some general beliefs in quantum computing are valid in practice. For instance, in general, one would believe that the use of more CNOT gates will cause more errors. In our case, we manually found a decomposition of our encoding and decoding operators using as few CNOT gates as possible. However, different IBM quantum computers may provide different decompositions, often with more CNOT gates. Yet, the accuracy is comparable or even better than. So, it is of interest to examine the issues of how to optimize the performance of numerical implementations of quantum algorithms in connection to the hardware.
4. While our study focused on comparing different IBM quantum computers, as pointed out by the referee, it would be interesting to perform experiments of our scheme on other quantum computing platforms such as Origin Quantum.

Acknowledgements We thank Wenxuan Ding, Mikio Nakahara, Yiu-Tung Poon, and Yiyang Zhang for some discussion. We also thank the referee for many helpful suggestions. CK Li is an affiliate member of the Institute for Quantum Computing. His research was supported by the Simons Foundation Grant 851334.
Data Availability The datasets generated during and/or analysed during the current study are available in the GitHub repository https://github.com/dcpelejo/QECC. We acknowledge the use of IBM Quantum services for this work. The views expressed are those of the authors, and do not reflect the official policy or position of IBM or the IBM Quantum team.

Declarations

Conflict of interest The authors certify that they have no affiliations with or involvement in any organization or entity with any financial interest or nonfinancial interest in the subject matter or materials discussed in this manuscript.

Appendix 1: The circuit decomposition in Fig. 2 does not produce the unitary matrix in Fig. 1b

We can construct the simple gates corresponding to \( Y_\theta, Y_{\pi/4}, I_2 \otimes I_2 \otimes \sigma_z \), and the three control gates, as \( U_1, U_2, \ldots, U_6 \) as follows.

\[
\begin{align*}
 a_1 &= \frac{[1 - \sqrt{2}]}{\sqrt{3}}'; \\
 a_2 &= \frac{[\sqrt{2}, 1]}{\sqrt{3}}'; \\
 b_1 &= \frac{[1 -1]}{\sqrt{2}}'; \\
 b_2 &= \frac{[1 1]}{\sqrt{2}}'; \\
 e_0 &= \frac{[1 0]}{'}; \\
 e_1 &= \frac{[0 1]}{'};
\end{align*}
\]

\[\begin{align*}
 U_1 &= \text{kron}(e_0, \text{kron}(e_0, e_0)), \text{kron}(e_0, \text{kron}(e_0, e_1)), \text{kron}(a_1, \text{kron}(e_1, e_0)), \\
 & \quad \text{kron}(a_1, \text{kron}(e_1, e_1)), \text{kron}(e_1, \text{kron}(e_0, e_0)), \text{kron}(e_1, \text{kron}(e_0, e_1)), \\
 & \quad \text{kron}(a_2, \text{kron}(e_1, e_0)), \text{kron}(a_2, \text{kron}(e_1, e_1)); \\
 U_2 &= \text{kron}(e_0, \text{kron}(b_1, e_0)), \text{kron}(e_0, \text{kron}(b_1, e_1)), \text{kron}(e_0, \text{kron}(b_2, e_0)), \\
 & \quad \text{kron}(e_0, \text{kron}(b_2, e_1)), \text{kron}(e_1, \text{kron}(e_0, e_0)), \text{kron}(e_1, \text{kron}(e_0, e_1)), \\
 & \quad \text{kron}(e_1, \text{kron}(e_1, e_0)), \text{kron}(e_1, \text{kron}(e_1, e_1)); \\
 Z &= \text{kron}(e_0, e_0); \\
 U_3 &= \text{kron}(e_0, e_0); \\
 U_4 &= \text{kron}(e_0, e_0); \\
 U_5 &= \text{kron}(e_0, e_0); \\
 U_6 &= \text{kron}(e_0, e_0);
\end{align*}\]

\[
\begin{align*}
 U_6^* U_5^* U_4^* U_3^* U_2^* U_1 &= \\
 \begin{bmatrix}
 1 & 0 & 0 & 0 & 0 & -1.0000 \\
 0.7071 & 0.4082 & 0 & 0 & 0 & 0.5774 \\
 -0.7071 & 0.4082 & 0 & 0 & 0 & 0.5774 \\
 0 & 0 & 0.8165 & 0 & 0 & 0 & -0.5774 \\
 0 & 0 & -0.8165 & 0 & 0 & 0 & 0.5774 \\
 0 & 0.7071 & 0 & -0.4082 & 0 & 0 & 0 & -0.5774 \\
 0 & -0.7071 & 0 & -0.4082 & 0 & 0 & 0 & -0.5774 \\
 0 & 0 & 0 & 1.0000 & 0 & 0 & 0 & 0
\end{bmatrix}
\end{align*}
\]

\[
[\sqrt{2/3}, \sqrt{1/3}, \sqrt{1/6}, 1/\sqrt{2}] = [0.8165, 0.5774, 0.4082, 0.7071]
\]
So, we see that $U \neq U_6 U_5 U_4 U_3 U_2 U_1$.

**Appendix 2: Matlab scripts to verify the circuit decompositions of the matrix $U$**

**Decomposition in Fig. 3**

\[
U = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\
\sqrt{2/3} & 0 & 0 & 0 & \sqrt{1/3} & 0 & 0 & 0 \\
-\sqrt{1/6} & 0 & \sqrt{1/2} & 0 & \sqrt{1/3} & 0 & 0 & 0 \\
0 & \sqrt{1/6} & 0 & \sqrt{1/2} & 0 & -\sqrt{1/3} & 0 & 0 \\
-\sqrt{1/6} & 0 & -\sqrt{1/2} & 0 & \sqrt{1/3} & 0 & 0 & 0 \\
0 & \sqrt{2/3} & 0 & 0 & 0 & -\sqrt{1/3} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}
\]

\[
E_0 = \begin{bmatrix} 1 & 0 \\
0 & 0 \end{bmatrix}; \quad E_1 = \begin{bmatrix} 0 & 0 \\
0 & 1 \end{bmatrix}; \quad Z = \begin{bmatrix} 1 & 0 \\
0 & -1 \end{bmatrix}; \quad X = \begin{bmatrix} 0 & 1 \\
1 & 0 \end{bmatrix};
\]

\[
P_1 = \text{kron}(\text{eye}(2), \text{kron}(E_1, \text{eye}(2))) + \text{kron}(X, \text{kron}(E_0, \text{eye}(2)));
\]

\[
P_2 = \text{kron}(E_1, \text{kron}(\text{eye}(2), X)) + \text{kron}(E_0, \text{eye}(4));
\]

\[
P_3 = \text{kron}(\text{eye}(2), \text{kron}(X, E_1)) + \text{kron}(\text{eye}(2), \text{kron}(\text{eye}(2), E_0));
\]

\[
Q_1 = \text{kron}(\text{eye}(4), Z);
\]

\[
A_2 = \sqrt{1/2} \begin{bmatrix} 1 & -1 \\
1 & 1 \end{bmatrix};
\]

\[
Q_2 = \text{kron}(E_0, \text{kron}(A_2, \text{eye}(2))) + \text{kron}(E_1, \text{eye}(4));
\]

\[
A_3 = \begin{bmatrix} -\sqrt{1/3} & \sqrt{2/3} \\
\sqrt{2/3} & \sqrt{1/3} \end{bmatrix};
\]

\[
Q_3 = \text{kron}(A_3, \text{kron}(E_0, \text{eye}(2))) + \text{kron}(\text{eye}(2), \text{kron}(E_1, \text{eye}(2)));
\]

\[
P_1 \ast P_2 \ast P_3 \ast Q_1 \ast Q_2 \ast Q_3 \ast U
\]

**Decomposition in Fig. 5**

\[
a = \pi/8;
\]

\[
t = \text{asin}(\sqrt{1/3})/2;
\]

\[
A = \begin{bmatrix} \cos(a) & -\sin(a) \\
\sin(a) & \cos(a) \end{bmatrix};
\]

\[
B = \begin{bmatrix} \cos(t) & -\sin(t) \\
\sin(t) & \cos(t) \end{bmatrix};
\]

\[
S_1 = \text{kron}(B', \text{eye}(4));
\]

\[
S_2 = \text{kron}(X \ast B, \text{eye}(4));
\]

\[
S_3 = \text{kron}(\text{eye}(2), \text{kron}(A, \text{eye}(2)));
\]

\[
S_4 = \text{kron}(\text{eye}(2), \text{kron}(A', \text{eye}(2)));
\]

\[
X_1 = \text{kron}(\text{eye}(2), \text{kron}(X, \text{eye}(2)));
\]

\[
X_2 = \text{kron}(X, \text{eye}(4));
\]

\[
Z_0 = \text{kron}(\text{eye}(4), Z);
\]

\[
C_{12} = \text{kron}(X, \text{kron}(E_1, \text{eye}(2))) + \text{kron}(\text{eye}(2), \text{kron}(E_0, \text{eye}(2)));
\]

\[
C_{21} = \text{kron}(E_1, \text{kron}(X, \text{eye}(2))) + \text{kron}(E_0, \text{kron}(\text{eye}(2), \text{eye}(2)));
\]

\[
C_{01} = \text{kron}(\text{eye}(2), \text{kron}(X, E_1)) + \text{kron}(\text{eye}(2), \text{kron}(\text{eye}(2), E_0));
\]

\[
C_{20} = \text{kron}(E_1, \text{kron}(\text{eye}(2), X)) + \text{kron}(E_0, \text{kron}(\text{eye}(2), \text{eye}(2)));
\]

\[
X_1 \ast C_{12} \ast C_{20} \ast C_{01} \ast Z_0 \ast S_4 \ast X_2 \ast C_{21} \ast S_3 \ast C_{21} \ast S_2 \ast C_{12} \ast S_1 \ast X_1 \ast U
\]
Appendix 3: Circuits generated by IBMQ

Here we demonstrate how the IBM quantum machines may process the same user-input circuit differently for two separate runs.

See Fig. 19.

(a) user-input circuit diagram to implement QECC scheme

(b) Two different circuits generated by the transpiler for ibmq_valencia given the input circuit in (a).

(c) user-input circuit diagram to implement QECC scheme

(d) Two different circuits generated by the transpiler for ibmq_valencia given the input circuit in (c).

Fig. 19 (a) and (c) are user-input circuits fed to the IBM quantum machines using the qiskit library. The input circuits are processed further by the machines and two runs may be processed differently as illustrated in figure (b) and (d).

Appendix 4: Results from the 5-qubit QECC implementation on the IBM quantum computers

In this appendix, we present more experimental results for the implementation of the 5-qubit QECC presented in Sect. 2.2. Each experiment is run three times in the IBM quantum computers ibmq_valencia, ibmq_santiago, ibmq_vigo, ibmq_5_yorktown, ibmq_ourense and ibmq_athens. The leftmost histograms show the best (least error or highest probability for $|0\rangle$) of the three
Fig. 20 Using the standard gate decomposition of $U$ and $W = H$

Fig. 21 Using the basic gate decomposition of $U$ and $W = H$

Fig. 22 Using the standard gate decomposition of $U$ and $W = X$

Fig. 23 Using the basic gate decomposition of $U$ and $W = X$

Fig. 24 Using the standard gate decomposition of $U$ and $W = Y$

runs, while the rightmost histogram shows the worst of the three runs (Figs. 20, 21, 22, 23, 24, 25, 26, 27, 28, 29).
Fig. 25 Using the basic gate decomposition of $U$ and $W = Y$

Fig. 26 Using the standard gate decomposition of $U$ and $W = Z$

Fig. 27 Using the basic gate decomposition of $U$ and $W = Z$

Fig. 28 Using the standard gate decomposition of $U$ and $W = I$

Fig. 29 Using the basic gate decomposition of $U$ and $W = I$

Appendix 5

The following illustrate the results obtained in implementing the 4-qubit QECC illustrated in Eq. (12) using $|q_3q_2\rangle \in \{01, 10, 11\}$ and the IBM machines ibmq_santiago and ibmq_athens (Figs. 30, 31, 32).
Fig. 30 (a) and (b) illustrate the results obtained in implementing the 4-qubit QECC illustrated in Eq. (12) using $|q_3q_2q_1q_0\rangle = |0100\rangle$.

(a) using ibmq_santiago and $|q_3q_2q_1q_0\rangle = |0100\rangle$

(b) using ibmq_athens and $|q_3q_2q_1q_0\rangle = |0100\rangle$
Fig. 31 (a) and (b) illustrate the results obtained in implementing the 4-qubit QECC illustrated in Eq. (12) using $|q_3q_2\rangle = |10\rangle$. 

(a) using ibmq_santiago and $|q_3q_2q_1q_0\rangle = |1000\rangle$

(b) using ibmq_athens and $|q_3q_2q_1q_0\rangle = |1000\rangle$
Fig. 32 (a) and (b) illustrate the results obtained in implementing the 4-qubit QECC illustrated in equation (12) using $|q_3 q_2 q_1 q_0\rangle = |1100\rangle$.

References

1. Zanardi, P., Rasetti, M.: Noiseless quantum codes. Phys. Rev. Lett. 79, 3306 (1997)
2. Zanardi, P., Rasetti, M.: Error avoiding quantum codes. Mod. Phys. Lett. B 11, 1085 (1997)
3. Zanardi, P.: Dissipation and decoherence in a quantum register. Phys. Rev. A 57, 3276 (1998)
4. Lidar, D.A., Chuang, I.L., Whaley, K.B.: Decoherence-free subspaces for quantum computation. Phys. Rev. Lett. 81, 2594 (1998)
5. Kempe, J., Bacon, D., Lidar, D.A., Whaley, K.B.: Theory of decoherence-free fault-tolerant universal quantum computation. Phys. Rev. A 63, 042307 (2001)
6. Lidar, D.A., Bacon, D., Kempe, J., Whaley, K.B.: Protecting quantum information encoded in decoherence-free states against exchange errors. Phys. Rev. A 61, 052307 (2000)
7. Lidar, D.A., Bacon, D., Kempe, J., Whaley, K.B.: Decoherence-free subspaces for multiple-qubit errors: I: characterization. Phys. Review A 63, 022306 (2001)
8. Choi, M.-D., Kribs, D.W.: Method to find quantum noiseless subsystems. Phys. Rev. Lett. 96, 050501 (2006)
9. Knill, E., Laflamme, R., Viola, L.: Theory of quantum error correction for general noise. Phys. Rev. Lett. 84, 2525 (2000)
10. Fortunato, E.M., Viola, L., Pravia, M.A., Knill, E., Laflamme, R., Havel, T.F., Cory, D.G.: Exploring noiseless subsystems via nuclear magnetic resonance. Phys. Rev. A 67, 062303 (2003)
11. Viola, L., Fortunato, E.M., Pravia, M.A., Knill, E., Laflamme, R., Cory, D.G.: Experimental realization of noiseless subsystems for quantum information processing. Science 293, 2059 (2001)
12. Lidar, D.A., Brun, T.A.: Quantum Error Correction. Cambridge University Press (2013)
13. Kondo, Y., Bagnasco, C., Nakahara, M.: Implementation of a simple operator-quantum-error-correction scheme. Phys. Rev. A 88, 022314 (2013)
14. Byrd, M.S.: Implications of qudit superselection rules for the theory of decoherence-free subsystems. Phys. Rev. A 73, 032330 (2006)
15. Gungordu, U., Li, C.K., Nakahara, M., Poon, Y.T., Sze, N.S.: Recursive encoding and decoding of the noiseless subsystem for qudits. Phys. Rev. A 89, 042301 (2014)
16. Li, C.K., Nakahara, M., Poon, Y.T., Sze, N.S., Tomita, H.: Recursive encoding and decoding of noiseless subsystem and decoherence free subspace. Phys. Rev. A 84, 044301 (2011)
17. Li, C.-K., Nakahara, M., Poon, Y.-T., Sze, N.-S.: Maximal error correction rates for collective rotation channels on qudits. Quantum Inf. Process. 14, 4039–4055 (2015)
18. Yang, C.-P., Gea-Banacloche, J.: Three-qubit quantum error-correction scheme for collective decoherence. Phys. Rev. A 63, 022311 (2001)
19. Li, C.K., Nakahara, M., Poon, Y.T., Sze, N.K., Tomita, H.: Efficient quantum error correction for fully correlated noise. Phys. Lett. A 375, 3255–3258 (2011)
20. Li, C.K., Lyles, S., Poon, Y.T.: Error correction schemes for fully correlated quantum channels protecting both quantum and classical information. Quant. Inf. Process. 19(5), 153 (2020)

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.