

# Error correlation schemes for fully correlated quantum channels protecting both quantum and classical information

Chi-Kwong Li, Seth Lyles, Yiu-Tung Poon

## Abstract

We study efficient quantum error correction schemes for the fully correlated channel on an  $n$ -qubit system with error operators that assume the form  $\sigma_x^{\otimes n}, \sigma_y^{\otimes n}, \sigma_z^{\otimes n}$ . In particular, when  $n = 2k + 1$  is odd, we describe a quantum error correction scheme using one arbitrary qubit  $\sigma$  to protect the data state  $\rho$  in the  $2k$ -qubit system such that only  $3k$  CNOT gates (each with one control bit and one target bit) are needed to encode the  $n$ -qubits. The inverse operation of the CNOT gates will produce  $\tilde{\sigma} \otimes \rho$ , so a partial trace operation can recover  $\rho$ . When  $n = 2k + 2$  is even, we describe a hybrid quantum error correction scheme that protects a  $2k$ -qubit state  $\rho$  and 2 classical bits encoded as  $\sigma \in \{|ij\rangle\langle ij| : i, j \in \{0, 1\}\}$ ; the encoding can be done by  $3k + 2$  CNOT gates and a Hadamard gate on one qubit, and the inverse operation will be the decoding operation producing  $\sigma \otimes \rho$ . The scheme was implemented using Matlab, Mathematica and the IBM's quantum computing framework `qiskit`.

## 1 Introduction

In quantum information processing, information is stored and processed with a quantum system. In the mathematical setting, quantum states are represented as density matrices, i.e., complex positive semi-definite matrices with trace one. Denote by  $D_N$  the set of density matrices in the set  $M_N$  of  $N \times N$  complex matrices. A qubit will be represented as a matrix in  $D_2$ , and a quantum state of an  $n$ -qubit system will be a matrix in  $D_N$  with  $N = 2^n$ . A quantum channel on an  $n$ -qubit system is a trace preserving completely positive linear map  $\mathcal{E} : M_N \rightarrow M_N$  that admits the operator sum representation

$$\mathcal{E}(\rho) = \sum_{j=1}^r F_j \rho F_j^\dagger \quad \text{for all } \rho \in M_N,$$

where  $\sum_{j=1}^r F_j^\dagger F_j = I_N$ . The matrices  $F_1, \dots, F_r$  are sometimes called the error operators of the quantum channel  $\mathcal{E}$ , which is the source of the corruption of the quantum states corresponding to decoherence and other quantum effect on the quantum state  $\rho$ . To protect the information stored in the quantum state  $\rho$ , one can use quantum error correction schemes to encode the quantum state  $\rho$  with some auxiliary qubits  $\sigma$  so that one can recover the quantum state  $\rho$  after the encoded quantum state go through the quantum channel.

In [6], the authors considered the fully correlated channel  $\mathcal{E}$  on an  $n$ -qubit system with error operators that assume the form  $X_n = \sigma_x^{\otimes n}$ ,  $Y_n = \sigma_y^{\otimes n}$ ,  $Z_n = \sigma_z^{\otimes n}$ , where

$$\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad \sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad \sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix},$$

are the Pauli matrices. So, the quantum channel  $\mathcal{E} : M_N \rightarrow M_N$  with  $N = 2^n$  has the form

$$\mathcal{E}(\rho) = p_0\rho + p_1X_n\rho X_n^\dagger + p_2Y_n\rho Y_n^\dagger + p_3Z_n\rho Z_n^\dagger \quad \text{for all } \rho \in M_N, \quad (1)$$

where  $p_0, p_1, p_2, p_3$  are nonnegative numbers summing up to one. It was shown that if  $n$  is odd, one can use a single qubit  $\sigma \in D_2$  to protect an  $(n-1)$ -qubit data state; if  $n$  is even, one can use a two-qubit  $\sigma$  to protect  $(n-2)$ -qubit data state. Encoding and decoding can be performed without measurement. So, for such a fully correlated channel, if one would like to protect  $2k$ -qubit data states, only one additional qubit is needed, and using two additional qubits to protect the  $2k$ -qubit data state seems to be a waste of resources.

We will show that one can actually design a hybrid quantum correction scheme for an  $n = 2k+2$  qubit fully correlated channel to send  $2k$ -qubit quantum states together with two classical bits. The study of simultaneous transmission of both quantum and classical information over a quantum channel was initiated in [1] and followed up by other researchers, [2, 3, 4].

In this paper, we will present efficient error correction schemes for the fully correlated channel. When  $n = 2k+1$  is odd, we describe a quantum error correction scheme using one arbitrary qubit  $\sigma$  to protect the data state  $\rho$  in the  $2k$ -qubit system such that only  $3k$  CNOT gates (each has one control bit and one target bit) are needed to encode the  $n$ -qubits. The inverse operation of the CNOT gates will produce  $\tilde{\sigma} \otimes \rho$ , so a partial trace operation can recover  $\rho$ . When  $n = 2k+2$  is even, we describe a hybrid quantum error correction scheme that protects a state  $\rho$  in the  $2k$ -qubit system, and two classical bit encoded as  $\sigma \in \{|ij\rangle\langle ij| : i, j \in \{0, 1\}\}$ ; the encoding can be done by  $3k+2$  CNOT gates and a Hadamard gate on a qubit, and the inverse operation will be the decoding operation producing  $\sigma \otimes \rho$ . The program was implemented using the IBM's quantum computing framework `qiskit` [9].

We state the results and prove them in the next section. Then we illustrate our schemes and depict the circuit diagrams. Furthermore, we implement and demonstrate our schemes using Matlab, Mathematica, and IBM's online quantum computer IBM Q 5 Yorktown. The last section is devoted to summary and discussions.

## 2 The quantum error and hybrid error correction schemes

In this section, we show that one can recursively construct the encoding matrix  $P_n \in M_{2^n}$  for the  $n$ -qubit fully correlated channel  $\mathcal{E}$  defined in 1. Moreover, we will show that  $P_n$  can be decomposed as simple CNOT gates (with one control qubit and one target qubit) and Hadamard gates (on one qubit). Once we find  $P_n$ , we can encode  $A \mapsto P_n A P_n^\dagger$  and decode  $B \mapsto P_n^\dagger B P_n$ . We will give a

full description of the construction, number of CNOT gates required, and the encoding / decoding procedures in Section 2.5. The circuit diagrams for encoding and decoding will be shown in Section 2.6.

We will depict an  $n$ -qubit vector state as  $|q_{n-1} \dots q_0\rangle$ . Let  $C_{ij}$  be the CNOT gate where the  $i$ th qubit controls the target  $j$ th qubit. For example, for  $(q_2, q_1, q_0) \in \{(0, 0, 0), (0, 0, 1), \dots, (1, 1, 1)\}$ ,

$$C_{02}|q_2q_1q_0\rangle = \begin{cases} |q_2 \oplus 1, q_1, q_0\rangle & \text{if } q_0 = 1, \\ |q_2q_1q_0\rangle & \text{otherwise.} \end{cases}$$

Denote  $H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \in M_2$  as the Hadamard gate. Also, we use  $e_1, \dots, e_n$  to denote the columns of  $I_n$ .

## 2.1 Two-qubit encoding/decoding operator

Let  $P_2 = C_{01}(I_2 \otimes H)C_{01}^t \in M_4$ , where  $C_{01}$  is the controlled not gate using the  $q_0$ -bit to control the  $q_1$ -bit for the two qubit state  $|q_1q_0\rangle$ . So,  $P_2 = C_{01}(I \otimes H)C_{01}$ , and in the matrix form  $C_{01} = Q = [e_1 \ e_4 \ e_3 \ e_2]$ . We readily verify the following.

**Proposition 2.1** *Let  $P_2 = C_{01}(I_2 \otimes H)C_{01}^t \in M_4$ . Then*

$$(P_2^\dagger X_2 P_2, P_2^\dagger Y_2 P_2, P_2^\dagger Z_2 P_2) = (D_X, D_Y, D_Z) \quad (2)$$

with  $D_X = \text{diag}(1, -1, 1, -1)$ ,  $D_Y = (-1, -1, 1, 1)$ ,  $D_Z = (1, -1, -1, 1)$ . Consequently,

$$P_2^\dagger(\mathcal{E}(P_2(\sigma)P_2^\dagger))P_2 = \sigma$$

whenever  $\sigma = |q_1q_0\rangle\langle q_1q_0|$  with  $|q_1q_0\rangle \in \{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$ .

## 2.2 Three-qubit encoding/decoding operator

**Proposition 2.2** *Let  $P_3 = C_{10}C_{02}C_{21} \in M_8$ , where  $C_{ij}$  use the  $|q_i\rangle$  to control the  $|q_j\rangle$  in  $|q_2q_1q_0\rangle$ . Then*

$$(P_3^\dagger X_3 P_3, P_3^\dagger Y_3 P_3, P_3^\dagger Z_3 P_3) = (X_1 \otimes I_4, -Y_1 \otimes I_4, Z_1 \otimes I_4).$$

Consequently, for any  $\sigma \in D_2$  and  $\rho \in D_4$ , we have

$$P_3^\dagger(\mathcal{E}(P_3(\sigma \otimes \rho)P_3^\dagger))P_3 = \tilde{\sigma} \otimes \rho,$$

where

$$\tilde{\sigma} = p_0\sigma + p_1X_1\sigma X_1^\dagger + p_2Y_1\sigma Y_1^\dagger + p_3Z_1\sigma Z_1^\dagger.$$

Moreover, if  $P_3$  is a product of  $m$  CNOT gates (each with one control bit and one target bit) on a 3-qubit system, then  $m \geq 3$ .

*Proof.* One readily verify the first two statements. For the last assertion, if we list the columns of  $I_8$  and  $P_3$  in binary form, we have

$$I_8 = [e_1 \ e_2 \ e_3 \ e_4 \ e_5 \ e_6 \ e_7 \ e_8] = [|000\rangle \ |001\rangle \ |010\rangle \ |011\rangle \ |100\rangle \ |101\rangle \ |110\rangle \ |111\rangle],$$

$$P_3 = [e_1 \ e_6 \ e_4 \ e_7 \ e_8 \ e_3 \ e_5 \ e_2] = [|000\rangle \ |101\rangle \ |011\rangle \ |110\rangle \ |111\rangle \ |010\rangle \ |100\rangle \ |001\rangle].$$

Since there are collectively 12 mismatched positions out of the 24 positions in the binary form of the 8 columns of the matrices  $I_8$  to  $P_3$ , and every CNOT gate will change 4 out of the 24 positions, we see that expressing  $P_3$  as the product of 3 CNOT gates is optimal.  $\square$

### 2.3 $n$ -qubit encoding / decoding operator for odd $n \geq 5$

**Proposition 2.3** *Let  $n = 2k + 1$  be an odd integer with  $k \geq 2$ , Let  $P_n = (I_4 \otimes P_{n-2})(P_3 \otimes I_{2^{n-3}})$ , which can be written as a product of  $3k$  CNOT gates (each as one control bit and one target bit). Then*

$$(P_n^\dagger X_n P_n, P_n^\dagger Y_n P_n, P_n^\dagger Z_n P_n) = (X_1 \otimes I_{2^{n-1}}, (-1)^k Y_1 \otimes I_{2^{n-1}}, Z_1 \otimes I_{2^{n-1}}).$$

Consequently, for any  $\sigma \in D_2$  and  $\rho \in D_{2^{n-1}}$ , we have

$$P_3^\dagger (\mathcal{E}(P_3(\sigma \otimes \rho)P_3))P_3 = \tilde{\sigma} \otimes \rho,$$

where

$$\tilde{\sigma} = p_0 \sigma + p_1 X_1 \sigma X_1^\dagger + p_2 Y_1 \sigma Y_1^\dagger + p_3 Z_1 \sigma Z_1^\dagger.$$

*Proof.* By Proposition 2.2,  $P_3$  is a product of 3 CNOT gates. By the recursive construction, when  $k$  increases 1 we need 3 more CNOT gates. So,  $P_n$  can be written as a product of  $3k$  CNOT gates.

The other assertions can be verified readily.  $\square$

### 2.4 $n$ -qubit encoding / operator for even $n \geq 4$

**Proposition 2.4** *Suppose  $n = 2k + 2$  for  $k \geq 1$ , and  $P_{n-1}$  is defined as in Proposition 2.3. Let  $P_n = (I_2 \otimes P_{n-1})(P_2 \otimes I_{2^{n-2}})$ , which is a product of  $3k + 2$  CNOT gates (each has one control bit and one target bit), and 1 Hadamard gate (on one qubit). Then*

$$(P_n^\dagger X_n P_n, P_n^\dagger Y_n P_n, P_n^\dagger Z_n P_n) = (D_X \otimes I_{2^{n-2}}, (-1)^k D_Y \otimes I_{2^{n-2}}, D_Z \otimes I_{2^{n-2}})$$

$D_X = \text{diag}(1, -1, 1, -1)$ ,  $D_Y = (-1, -1, 1, 1)$ ,  $D_Z = (1, -1, -1, 1)$ . Consequently,

$$P_n^\dagger (\mathcal{E}(P_n(\sigma \otimes \rho)P_n))P_n = \sigma \otimes \rho$$

whenever  $\rho \in D_{n-2}$  and  $\sigma = |q_1 q_0\rangle\langle q_1 q_0|$  with  $|q_1 q_0\rangle \in \{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$ .

*Proof.* By Proposition 2.1,  $P_2$  is a product of 2 CNOT gates and 1 Hadamard gate. By Proposition 2.3,  $P_{n-1}$  is a product of  $3k$  CNOT gates. So,  $P_n = (I_2 \otimes P_{n-1})(P_2 \otimes I_{2^{n-2}})$  is the product of  $3k + 2$  CNOT gates and a Hadamard gate.

The rest of the proposition can be verified readily.  $\square$

## 2.5 The encoding and decoding schemes

We summarize the results in Propositions 2.1 — 2.4 to the following.

**Theorem 2.5** *Let  $P_2, P_3$  and  $P_n$  be defined as in Sections 2.1 - 2.4.*

- (a) *Suppose  $n = 2k + 1 \geq 3$  is odd. Then  $P_n$  is a product of  $3k$  CNOT gates (each has 1 control and 1 target bit). One can encode an  $(n - 1)$ -qubit data state  $\rho$  using an arbitrary qubit  $\sigma$  by the encoding operator  $P_n$  so that*

$$\rho \mapsto P_n(\sigma \otimes \rho)P_n^\dagger.$$

*After the encoded state goes through the fully correlated channel  $\mathcal{E}$ , one can apply the operation  $B \mapsto P_n^\dagger B P_n$ . Then the encoded state  $P_n(\sigma \otimes \rho)P_n^\dagger$  becomes*

$$\begin{aligned} & p_0(\sigma \otimes \rho) + p_1 P_n^\dagger X_n P_n(\sigma \otimes \rho) P_n^\dagger X_n^\dagger P_n + \\ & p_2 P_n^\dagger Y_n P_n(\sigma \otimes \rho) P_n^\dagger Y_n^\dagger P_n + p_3 P_n^\dagger Z_n P_n(\sigma \otimes \rho) P_n^\dagger Z_n^\dagger P_n = \tilde{\sigma} \otimes \rho, \end{aligned}$$

where

$$\tilde{\sigma} = p_0 \sigma + p_1 X_1 \sigma X_1^\dagger + p_2 Y_1 \sigma Y_1^\dagger + p_3 Z_1 \sigma Z_1^\dagger.$$

Then, one can apply a partial trace  $\text{tr}_1(\tilde{\sigma} \otimes \rho)$  to recover the data state  $\rho$ .

- (b) *Suppose  $n = 2k + 2 \geq 4$  is even. Then  $P_n$  is a product of  $3k + 2$  CNOT gates (each has 1 control and 1 target bit) with a Hadamard gate (on 1 qubit). One can encode an  $(n - 2)$ -qubit data state  $\rho$  and two classical bits by  $\sigma \in \{|ij\rangle\langle ij| : 1 \leq i, j \leq 2\}$  using the encoding operator  $P_n$  so that*

$$\rho \mapsto P_n(\sigma \otimes \rho)P_n^\dagger.$$

*After the encoded state goes through the fully correlated channel  $\mathcal{E}$ , one can apply the operation  $B \mapsto P_n^\dagger B P_n$ . Then the encoded state becomes*

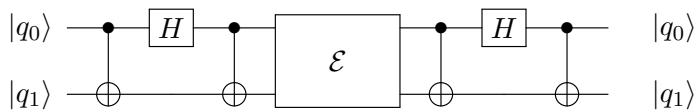
$$\begin{aligned} & p_0(\sigma \otimes \rho) + p_1 P_n^\dagger X_n P_n(\sigma \otimes \rho) P_n^\dagger X_n^\dagger P_n + \\ & p_2 P_n^\dagger Y_n P_n(\sigma \otimes \rho) P_n^\dagger Y_n^\dagger P_n + p_3 P_n^\dagger Z_n P_n(\sigma \otimes \rho) P_n^\dagger Z_n^\dagger P_n = \sigma \otimes \rho. \end{aligned}$$

One can apply a measurement to the first two qubits to obtain the two classical bits  $\sigma$ .

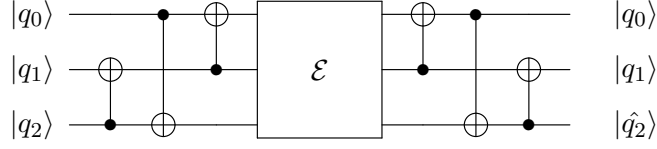
## 2.6 Circuit diagrams

We can set the  $n$  quantum state as  $|q_n \cdots q_1\rangle$ . Using our scheme, the circuit diagram are depicted in the following.

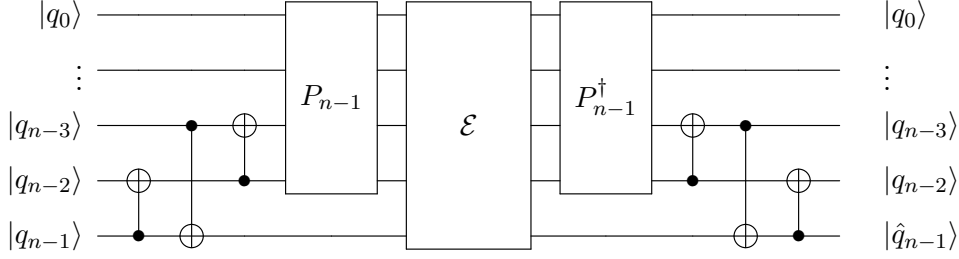
For  $n = 2$ , if  $|q_1 q_0\rangle \in \{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$ , then circuit diagram will be:



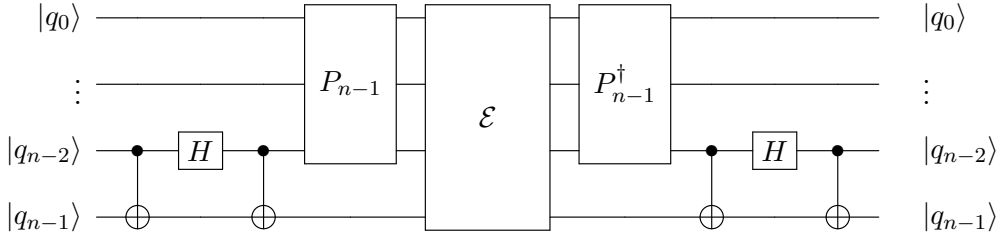
For  $n = 3$ ,



For odd  $n$ , the circuit diagram will be



For even  $n$ , if  $|q_n q_{n-1}\rangle \in \{|00\rangle, |10\rangle, |01\rangle, |11\rangle\}$ , then the circuit diagram will look like:



### 3 Conclusion and future work

We obtain an efficient error correction scheme for fully correlated quantum channels protecting both quantum and classical information. The scheme was implemented using Matlab, Mathematica and the IBM's quantum computing framework `qiskit`. We plan to extend the techniques to more general quantum channels such as the fully correlated quantum channels on  $n$ -qubits with general noise of the form  $U^{\otimes n}$ , where  $U \in M_2$  is unitary.

### Acknowledgment

Li is an affiliate member of the Institute for Quantum Computing, University of Waterloo. He is an honorary professor of Shanghai University. His research was supported by USA NSF grant DMS 1331021, Simons Foundation Grant 351047, and NNSF of China Grant 11571220.

## References

- [1] I. Devetak and P. W. Shor, The capacity of a quantum channel for simultaneous transmission of classical and quantum information, *Communications in Mathematical Physics*, 256, no. 2, pp. 287303, 2005.
- [2] M.H. Hsieh and M.M. Wilde, Entanglement-assisted communication of classical and quantum information, *IEEE Transactions on Information Theory*, 56, no. 9, 4682–4704, 2010.
- [3] M.H. Hsieh and M.M. Wilde, Trading classical communication, quantum communication, and entanglement in quantum Shannon theory, *IEEE Transactions on Information Theory*, vol. 56, no. 9, 4705–4730, 2010.
- [4] M. Grassl, S. Lu, and B. Zeng, Codes for simultaneous transmission of quantum and classical information, 2017 IEEE International Symposium on Information Theory (ISIT), 1718–1722, 2017.
- [5] E. Knill, R. Laflamme, and L. Viola, Theory of Quantum Error Correction for General Noise, *Physical Review Letters* 84, 2525, 2000.
- [6] C.K. Li, M. Nakahara, Y.T. Poon, N.S. Sze and H. Tomita, Efficient Quantum Error Correction for Fully Correlated Noise, *Phys. Lett. A*, 375:3255-3258 (2011).
- [7] M.A. Nielsen and I.L. Chuang, *Quantum Computation and Quantum Information*, Cambridge University Press, Cambridge, 2000.
- [8] J. Yard, Simultaneous classical-quantum capacities of quantum multiple access channels, Ph.D. dissertation, Electr. Eng. Dept., Stanford Univ., Stanford, CA, 2005.
- [9] G. Aleksandrowicz, T. Alexander, P. Barkoutsos, L. Bello, Y. BenHaim, D. Bucher, F. J. Cabrera-Hernandez, J. Carballo-Franquis, A. Chen, C.-F. Chen, J. M. Chow, A. D. Corcoles-Gonzales, A. J. Cross, A. Cross, J. Cruz-Benito, C. Culver, S. D. L. P. Gonzalez, E. D. L. Torre, D. Ding, E. Dumitrescu, I. Duran, P. Eendebak, M. Everitt, I. F. Sertage, A. Frisch, A. Fuhrer, J. Gambetta, B. G. Gago, J. Gomez-Mosquera, D. Greenberg, I. Hamamura, V. Havlicek, J. Hellmers, . Herok, H. Horii, S. Hu, T. Imamichi, T. Itoko, A. Javadi-Abhari, N. Kanazawa, A. Karazeev, K. Krsulich, P. Liu, Y. Luh, Y. Maeng, M. Marques, F. J. Martin-Fernandez, D. T. McClure, D. McKay, S. Meesala, A. Mezzacapo, N. Moll, D. M. Rodriguez, G. Nannicini, P. Nation, P. Ollitrault, L. J. ORiordan, H. Paik, J. Perez, A. Phan, M. Pistoia, V. Prutyanov, M. Reuter, J. Rice, A. R. Davila, R. H. P. Rudy, M. Ryu, N. Sathaye, C. Schnabel, E. Schoute, K. Setia, Y. Shi, A. Silva, Y. Siraichi, S. Sivaram, J. A. Smolin, M. Soeken, H. Takahashi, I. Tavernelli, C. Taylor, P. Taylour, K. Trabing, M. Treinish, W. Turner, D. Vogt-Lee, C. Vuillot, J. A. Wildstrom, J. Wilson, E. Winston, C. Wood, S. Wood, S. Worner, I. Y. Akhalwaya, and C. Zoufal, Qiskit: An open-source framework for quantum computing, 2019.

(C.K. Li) Department of Mathematics, College of William & Mary, Williamsburg, VA 23185, USA.  
Email: ckli@math.wm.edu

(S. Lyles) Department of Mathematics, College of William & Mary, Williamsburg, VA 23185, USA.  
Email: smlyles@email.wm.edu

(Y.T. Poon) Department of Mathematics, Iowa State University, Ames, IA 50011, USA. Email:  
ytpoon@iastate.edu



Below we present the formulation and emperical verification of the encoding and decoding schemes. All code is available here:

<https://github.com/slyles1001/QECC>

## A Matlab results

We write a Matlab program to generate the matrices  $X_n, Y_n, Z_n, P_n$ , etc., and demonstrate our quantum error correction scheme described in Section 2.

We briefly describe our program in the following. For an integer  $n > 1$ , the following commands will generate the encoding matrix  $P_n$ :

```
if mod(n,2) == 1
    P = eye(8); P3 = P(:, [1,6,4,7,8,3,5,2]); Pn = P3;
    k = (n-1)/2;
    for j = 2:k
        Pn = kron(eye(4),Pn)*kron(P3,eye(2^(2*j-2)));
    end
else
    H = [1 1; 1 -1]/sqrt(2); C01 = [1 0 0 0; 0 0 0 1; 0 0 1 0; 0 1 0 0];
    P2 = C01*kron(eye(2),H)*C01;
    if n == 2
        Pn = P2;
    else
        P = eye(8); P3 = P(:, [1,6,4,7,8,3,5,2]); Pn = P3; k = (n-2)/2;
        for j = 2:k
            Pn = kron(eye(4),Pn)*kron(P3,eye(2^(2*j-2)));
        end
        Pn = kron(eye(2),Pn)*kron(P2,eye(2^n/4));
    end
end
```

Then one can test the encoding and decoding schemes. First, set up the error operators  $X_n, Y_n, Z_n$ .

```
%% Set up the error operators for the channel
X = [0 1; 1 0]; Y = [0 -i; i 0]; Z = [1 0; 0 -1];
Xn=X; Yn = Y; Zn = Z;
for j = 2:n
    Xn = kron(X,Xn); Yn = kron(Y,Yn); Zn = kron(Z,Zn);
end
```

Suppose  $n = 2k + 1$  is odd. The following commands check

$$(P_n^\dagger X_n P_n, P_n^\dagger Y_n P_n, P_n^\dagger Z_n P_n) = (X \otimes I, (-1)^k Y \otimes I, Z \otimes I). \quad (3)$$

The output 0,0,0 will confirm the equality.

```
%%% Check (Pn'XnPn, Pn'YnPn, Pn'ZnPn)
II = eye(2^(n-1)); norm(Pn'*XnPn - kron(X,II)),
norm(Pn'*YnPn - (-1)^k*kron(Y,II)), norm(Pn'*ZnPn - kron(Z,II))
```

Next, we verify the error correction scheme for random input  $\sigma \otimes \rho$  with  $\sigma \in D_2$  and  $\rho \in D_4$ . The output 0,0,0 will confirm the scheme works.

```
%%% Generate random S in D_2, S in D_{2k}
S = rand(2,2) + i*rand(2,2) - rand(1,1)*(1+i)*eye(2); S = S*S'; S = S/trace(S);
K = 2^(n-1);
R = rand(K,K) + i*rand(K,K) - rand(1,1)*(1+i)*eye(K); R = R*R'; R = R/trace(R);
% Encode kron(S,R) and compared with the decoded state for each error operator.
A = kron(S,R); AA = Pn*A*Pn';
norm(Pn'*Xn*AA*XnPn - kron(X*S*X',R)), norm(Pn'*Yn*AA*YnPn - kron(Y*S*Y',R))
norm(Pn'*Zn*AA*ZnPn - kron(Z*S*Z',R))
```

Suppose  $n$  is even. The following commands check

$$(P_n^\dagger X_n P_n, P_n^\dagger Y_n P_n, P_n^\dagger Z_n P_n) = (D_Z \otimes I_{2^{n-2}}, D_Y \otimes I_{2^{n-2}}, D_Z \otimes I_{2^{n-2}}). \quad (4)$$

The output 0,0,0 will confirm the equality.

```
Dx = diag([1 -1 1 -1]); Dy = diag([-1 -1 1 1]); Dz = diag([1 -1 -1 1]);
II = eye(2^n/4); norm(Pn'*XnPn - kron(Dx,II));
norm(Pn'*YnPn - (-1)^k*kron(Dy,II)), norm(Pn'*ZnPn - kron(Dz,II))
```

Then we verify our error correction scheme that for any  $\sigma \in \{|00\rangle\langle 00|, |01\rangle\langle 01|, |10\rangle\langle 10|, |11\rangle\langle 11|\}$  and  $\rho \in D_{2^{n-2}}$ , the encoding and decoding yield  $\sigma \otimes \rho$ . Again, the output 0,0,0 will confirm the scheme works.

```
%%% Set up the classical bits in D_4, and arbitrary qubits in D_{2k}
K = 2^(n-2);
R = rand(K,K) + i*rand(K,K) - rand(1,1)*(1+i)*eye(K); R = R*R'; R = R/trace(R);
b0 = [1 0; 0 0]; b1 = [0 0; 0 1];
b00 = kron(b0,b0); b01 = kron(b0,b1); b10 = kron(b1,b0); b11 = kron(b1,b1);
%
S = b00; A = kron(S,R); AA = Pn*A*Pn'; norm(Pn'*Xn*AA*XnPn - kron(S,R)),
norm(Pn'*Yn*AA*YnPn - kron(S,R)), norm(Pn'*Zn*AA*ZnPn - kron(S,R))
```

```

%
S = b01; A = kron(S,R); AA = Pn*A*Pn'; norm(Pn'*Xn*AA*Xn*Pn - kron(S,R)),
norm(Pn'*Yn*AA*Yn*Pn - kron(S,R)), norm(Pn'*Zn*AA*Zn*Pn - kron(S,R))
%
S = b10; A = kron(S,R); AA = Pn*A*Pn'; norm(Pn'*Xn*AA*Xn*Pn - kron(S,R)),
norm(Pn'*Yn*AA*Yn*Pn - kron(S,R)), norm(Pn'*Zn*AA*Zn*Pn - kron(S,R))
%
S = b11; A = kron(S,R); AA = Pn*A*Pn'; norm(Pn'*Xn*AA*Xn*Pn - kron(S,R)),
norm(Pn'*Yn*AA*Yn*Pn - kron(S,R)), norm(Pn'*Zn*AA*Zn*Pn - kron(S,R))

```

**Remark A.1** Note that when the dimension is high, the matrices  $P_n, X_n$ , etc. are too big to display. So, we just use the command `norm(Pn'*Zn*AA*Zn*Pn - kron(S,R))` to compute the norm of the matrix  $P_n^\dagger Z_n P_n (\sigma \otimes \rho) P_n^\dagger Z_n P_n - \sigma \otimes \rho$  to confirm that it gives zero (up to machine error). In fact, in the odd case, when we do not use the Hadamard gate to do encoding and decoding, the norm values of the relevant matrices are exactly 0; in the even case, when the Hadamard gate is used (once in encoding and once in decoding), the norm value of matrices will yield a number at the order of the machine error.

## B Mathematica results

We write a Mathematica program to generate the matrices  $X_n, Y_n, Z_n, P_n$ , etc., and demonstrate our quantum error correction scheme described in Section 2.

We briefly describe our program in the following. We begin by setting up the CNOT gates, the Hadamard matrix, the Pauli matrices,  $D_X, D_Y, D_Z$  and  $P_2$  by the following commands:

```

CNOT[n0_,h0_,k0_] := Module[{n=n0,h=h0,k=k0}, U=IdentityMatrix[2^n];
Do[cindex=IntegerDigits[i-1,2,n];
If[cindex[[n-h]]==1,cindex[[n-k]]=Mod[cindex[[n-k]]+1,2]];
s=Sum[cindex[[r]]*2^(n-r),{r,1,n}]+1;
U[[i]]=Table[KroneckerDelta[s,j],{j,1,2^n},{i,1,2^n}];U]

H={{1,1},{1,-1}}/Sqrt[2];
x={{0,1},{1,0}};
y={{0,-I},{I,0}};
z={{1,0},{0,-1}};
DX = DiagonalMatrix[{1, -1, 1, -1}];
DY = DiagonalMatrix[{-1, -1, 1, 1}];
DZ = DiagonalMatrix[{1, -1, -1, 1}];
P2=CNOT[2,0,1].KroneckerProduct[IdentityMatrix[2],H].Transpose[CNOT[2,0,1]];

```

Then we define  $X_n$ ,  $Y_n$  and  $Z_n$  recursively:

```
X[n_]:=X[n]=KroneckerProduct[X[n-1],x]; X[1]=x;
Y[n_]:=Y[n]=KroneckerProduct[Y[n-1],y]; Y[1]=y;
Z[n_]:=Z[n]=KroneckerProduct[Z[n-1],z]; Z[1]=z;
```

Then we define  $P_n$  recursively by setting  $P_{2k+1} = Q[k]$  and  $P_{2k} = R[k]$ .

```
Q[k_]:=Q[k]=KroneckerProduct[IdentityMatrix[4],Q[k-1]].
KroneckerProduct[Q[1],IdentityMatrix[2^(2k-2)]];
Q[1]=CNOT[3,1,0].CNOT[3,0,2].CNOT[3,2,1];
```

```
R[k_]:=KroneckerProduct[IdentityMatrix[2],Q[k-1]].
KroneckerProduct[P2,IdentityMatrix[2^(2k-2)]];
```

Then the validity of the formula

$$(P_{2k+1}^t X_{2k+1} P_{2k+1}, P_{2k+1}^t Y_{2k+1} P_{2k+1}, P_{2k+1}^t Z_{2k+1} P_{2k+1}) = (X_1 \otimes I_{2^{2k}}, (-1)^k Y_1 \otimes I_{2^{2k}}, Z_1 \otimes I_{2^{2k}}).$$

can be checked by calculating  $\|P_n^t X_n P_n - X_1 \otimes I_{2^{2k}}\|$ ,  $\|P_n^t Y_n P_n - (-1)^k Y_1 \otimes I_{2^{2k}}\|$ ,  $\|P_n^t Z_n P_n - Z_1 \otimes I_{2^{2k}}\|$  with the corresponding functions:

```
Norm[Transpose[Q[3]].X[7].Q[3] - KroneckerProduct[x,IdentityMatrix[2^6]]]
Norm[Transpose[Q[3]].Y[7].Q[3] - (-1)^3*KroneckerProduct[y,IdentityMatrix[2^6]]]
Norm[Transpose[Q[3]].Z[7].Q[3] - KroneckerProduct[z,IdentityMatrix[2^6]]]
```

Similarly, we can check the formula

$$(P_{2k}^t X_{2k} P_{2k}, P_{2k}^t Y_{2k} P_{2k}, P_{2k}^t Z_{2k} P_{2k}) = (D_X \otimes I_{2^{2k-2}}, (-1)^{k-1} D_Y \otimes I_{2^{2k-2}}, D_Z \otimes I_{2^{2k-2}})$$

by calculating (for  $k = 3$ ) respectively as follows:

```
Norm[Transpose[R[3]].X[6].R[3] - KroneckerProduct[DX,IdentityMatrix[2^4]]]
Norm[Transpose[R[3]].Y[6].R[3] - (-1)^2*KroneckerProduct[DY,IdentityMatrix[2^4]]]
Norm[Transpose[R[3]].Z[6].R[3] - KroneckerProduct[DZ,IdentityMatrix[2^4]]]
```

We can also check that for all  $n \geq 3$ ,  $\|P_n^\dagger \mathcal{E}(P_n(\sigma \otimes \rho) P_n^\dagger) P_n - \tilde{\sigma} \otimes \rho\| = 0$ .

**Remark B.1** It is interesting to note that in Mathematica, the computation of the norm of the matrix  $P_n^\dagger \mathcal{E}(P_n(\sigma \otimes \rho) P_n^\dagger) P_n - \tilde{\sigma} \otimes \rho$  is always exactly 0 even if the Hadamard gate is used in the even case. Here  $\tilde{\sigma} = p_0 \sigma + p_1 X \sigma X^\dagger + p_2 Y \sigma Y^\dagger + p_3 Z \sigma Z^\dagger \in D_2$  if  $n$  is odd and  $\sigma = \tilde{\sigma} \in D_4$  is one of the 4 classical binary bits if  $n$  is even. This is due to Mathematica being an algebraic solver versus Matlab and Python.

## C Python results

Similar to the other methods, we can verify the results in Python. This is convenient because IBM has provided the `qiskit` framework, and the encoding matrices can be extracted from the circuit build.

In contrast to the matrix operations of sections A and B, the recursive scheme operates on the quantum circuit itself. For convenience, the imports have been omitted.

To initialize a quantum circuit, we first create qubits using `qr = QuantumRegister(n)` and classical bits for measurement with `cr = ClassicalRegister(n)`. The quantum circuit is constructed by the `qc = QuantumCircuit(qr, qc)`. To verify functionality, we can apply arbitrary unitary operations to initialize the state to a given vector. Then, the encoding scheme is defined recursively:

```
def build_circ(qc, q, E):
    n = q[-1][1] + 1 # number of qubits

    def err(e, base=False):
        d = {'X':qc.x, 'Y':qc.y, 'Z':qc.z, 'I':lambda x: None} # apply errors
        d[e](q[n-1])
        if base: # if we're at q3
            d[e](q[0])
            d[e](q[1])

    if n % 2 == 0: # qn is even
        qc.cx(q[n-2], q[n-1]) # encode with P_2
        qc.h(q[n-2])
        qc.cx(q[n-2], q[n-1])
        build_circ(qc, q[:-1], E) #recurse
        err(E)
        qc.cx(q[n-2], q[n-1]) # decode with P_2^T
        qc.h(q[n-2])
        qc.cx(q[n-2], q[n-1])
    else: # it's odd
        qc.cx(q[n-1], q[n-2]) # encode with P_3
        qc.cx(q[n-3], q[n-1])
        qc.cx(q[n-2], q[n-3])
        if n == 3: # base case
            err(E, True)
        else:
            build_circ(qc, q[:-1], E) #recurse
```

```

err(E)
qc.cx(q[n-2], q[n-3]) # decode with P_3^T
qc.cx(q[n-3], q[n-1])
qc.cx(q[n-1], q[n-2])
return()

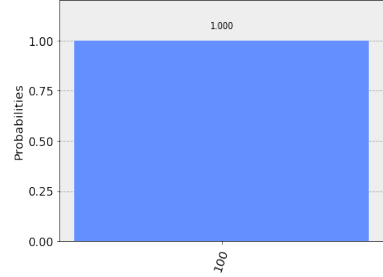
```

### C.1 Using IBM's simulated quantum computer

We can use the IBM's `qasm` quantum computer simulator to validate our scheme. The results are identical to the above two sections, so we omit them here. All code is available at the github link at the beginning of the appendix.

For example, when  $n = 3$ , we can input  $|q_2q_1q_0\rangle = |000\rangle$ , and use  $I_8, X_3, Y_3, Z_n$  as the error operators, respectively. Or we can use the error operators "blindly". The simulation always give us the perfect statistics. For example, using input  $|000\rangle$  and error operator  $X_3$ , the procedure in the following circuit diagram (IMB version of the one in Section 2.6) changes the state  $|000\rangle$  to  $|100\rangle$ , which is given by the statistics of the IBM's quantum computer simulator.

Circuit diagram here.



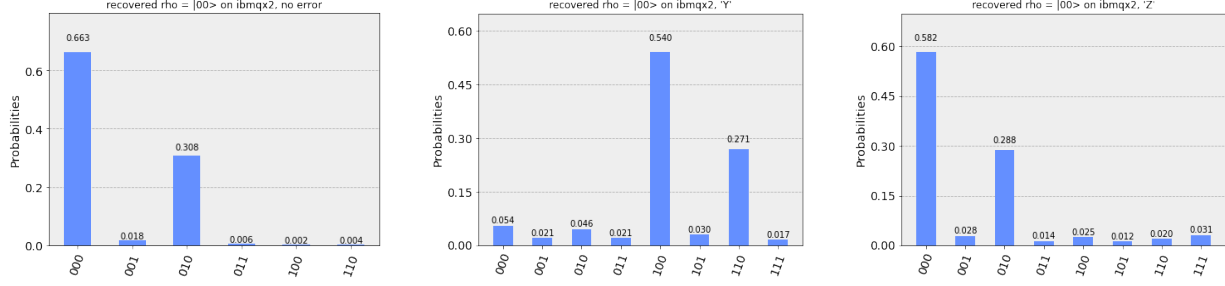
One may consider using  $|q_2\rangle = U|0\rangle$  and  $|q_1q_0\rangle = V|00\rangle$  for unitary  $U \in M_2, V \in M_4$ , and apply our scheme. One should get perfect decoding of  $|q_1q_0\rangle = V|00\rangle$ .

One may do simulation and get the statistics for  $n = 4, 5, 6$ , etc.

### C.2 Using IBM's quantum computer

We can use the IBM's online quantum computer (with noise) to implement our scheme.

For  $n = 3$ , we input  $|000\rangle$  and use the error operators  $X_3, Y_3, Z_3$  in three different experiments using the following circuit (IMB version of the one in Section 2.6). The expected results states should be  $|100\rangle, |100\rangle, |000\rangle$ , respectively. We get the following statistics of the experiments. The expected measured state always has the highest probability. It is interesting to note that  $q_1$  always attracts more noise.



One may consider using  $|q_2\rangle = U|0\rangle$  and  $|q_1q_0\rangle = V|00\rangle$  for unitary  $U \in M_2, V \in M_4$ , and apply our scheme. One should high probability of decoded state:  $|q_1q_0\rangle = V|00\rangle$ .