Design of low-density-generator-matrix–based quantum codes for asymmetric quantum channels

Patricio Fuentes,[1,*] Josu Etxezarreta Martinez,[1,†] Pedro M. Crespo,[1,‡] and Javier Garcia-Frias[2,§]
[1]*Department of Biomedical Engineering and Sciences, Tecnun - University of Navarra, 20018 San Sebastian, Spain*
[2]*Department of Electrical and Computer Engineering, University of Delaware, Newark, Delaware 19716, USA*

(Received 11 September 2020; revised 1 February 2021; accepted 16 February 2021; published 26 February 2021)

Quantum low-density-generator-matrix (QLDGM) codes are known to exhibit great error correction capabilities, surpassing existing quantum low-density-parity-check (QLDPC) codes and other sparse-graph schemes over the depolarizing channel. Most of the research on QLDPC codes and quantum error correction (QEC) is conducted for the symmetric instance of the generic Pauli channel, which incurs bit flips, phase flips, or a combination of both with the same probability. However, due to the behavior of the materials they are built from, some quantum devices must be modelled using a different channel model capable of accurately representing asymmetric scenarios in which the likelihood of a phase flip is higher than that of a bit flip. In this work, we study the design of QLDGM CSS codes for such Pauli channels. We show how codes tailored to the depolarizing channel are not well suited to these asymmetric environments and we derive methods to aptly design QLDGM CSS codes for this paradigm.

DOI: 10.1103/PhysRevA.103.022617

I. INTRODUCTION

In the realm of classical communications, turbo codes [1] and low-density-parity-check (LDPC) codes [2–4], are known to exhibit capacity-approaching performance at a reasonable decoding computational complexity. Turbo codes offer great flexibility in terms of their block length and rate. The first quantum codes based on turbo codes appeared in Refs. [5,6], and have since been modified and improved [7–12]. Aside from their block length and rate flexibility being on par with that of turbo codes, the sparse nature of LDPC codes guarantees that their quantum equivalents will require small numbers of quantum interactions per qubit during the error correction procedure [13], avoiding additional quantum gate errors and facilitating fault-tolerant decoding. These traits make QLDPC codes especially well suited for quantum error correction.

Out of the existing types of LDPC codes, low-density-generator-matrix (LDGM) codes [14] provide a seamless manner for code design in the quantum domain. LDGM codes are a specific subset of LDPC codes whose generator matrices are also sparse, and thus their encoding complexity is similar to that of turbo codes, and much smaller than for standard LDPC codes. Given that LDGM codes form a special subclass of the LDPC code family, they can be decoded in the same manner and with the same complexity as any other LDPC code. LDGM codes have been extensively studied [3,14–16] and used in classical communications [17,18]. In Ref. [3], regular LDGM codes, which are a specific type of LDGM code whose parity check matrices have the same number of nonzero entries per row and the same amount of nonzero entries per column, were studied and shown to be asymptotically bad, displaying error floors that do not decrease with the block length. In Refs. [15,16], a concatenated LDGM scheme was shown to achieve performance similar to irregular LDPC codes at a very low encoding/decoding complexity.

The first Quantum LDPC codes were built by casting classical LDPC codes in the framework of stabilizer codes [19], which enabled the design of quantum codes from any arbitrary classical binary and quaternary codes. Later, in Ref. [20], design methodologies for classical LDPC codes where studied in the context of quantum error correction. In this work, numerous construction and decoding techniques along with their flaws and merits are analyzed. Among the discussed methods, the construction of QLDPC codes based on LDGM codes stands out because it maintains good performance at a reduced decoding complexity. This method was originally proposed in Refs. [21,22], where Calderbank-Steane-Shor (CSS) [23,24] quantum codes based on regular LDGM classical codes were shown to surpass the best quantum coding schemes of the time. The performance of these chemes was significantly improved in Refs. [25,26] by using a parallel concatenation of two regular LDGM codes. Later, the codes proposed in Refs. [27–30] were shown to outperform LDGM-based CSS quantum codes over the depolarizing channel. Recently, the non-CSS inspired LDGM-based stratagem proposed in Ref. [31] was shown to outperform all other CSS and non-CSS codes of similar complexity. These codes were also analyzed for the case of a misidentified depolarizing channel in Ref. [32].

Most of the research related to the performance of quantum error correcting codes considers the symmetric Pauli channel, generally referred to as the depolarizing channel, which incurs

*Corresponding author: pfuentesu@tecnun.es
†Corresponding author: jetxezarreta@tecnun.es
‡Corresponding author: pcrespo@tecnun.es
§Corresponding author: jgf@udel.edu

bit flips, phase flips, and bit-and-phase flips with the same probability. However, realistic quantum devices, given the nature of the materials used to construct them, often exhibit asymmetric behavior, where the probability of a phase flip taking place is orders of magnitude higher than the probability of a bit flip [12,33,34]. The behavior of these quantum devices is governed by the single-qubit relaxation time and the dephasing time of the device itself, the former sometimes being orders of magnitude larger than the latter. Generally, relaxation causes both bit flips and phase flips, while dephasing only leads to phase-flip errors. This difference in relaxation and dephasing times gives rise to the aforementioned asymmetric behavior, where bit-flip errors are much less likely to occur than phase flips, and it can be accurately modelled by the general Pauli channel [34–39]. Naturally, it stands to reason that the best QEC schemes for this asymmetric channel must somehow be able to exploit its asymmetry. In Ref. [40], the authors introduce an EXIT-chart based methodology to design quantum turbo codes (QTCs) specifically for the general Pauli channel. This work is extended in Ref. [12], where an online estimation protocol to decode QTCs over general Pauli channels is proposed. These results speak to the merit of constructing a coding scheme tailored to the asymmetric characteristics of the quantum channel in question, since performance of the QTCs varies depending on the degree of asymmetry of the channel.

In this paper, we study the performance of quantum CSS LDGM codes when they are applied over a Pauli channel. We show how although they are not the best known codes for the depolarizing channel, their simplicity allows for them to be almost seamlessly adapted to the general Pauli channel. Based on this result, we introduce a simple yet effective method to derive CSS QLDGM codes that perform well over channels with varying degrees of asymmetry. Such a strategy is necessary because CSS codes designed for the depolarizing channel perform poorly over its asymmetric counterpart. To the extent of our knowledge and at the time of writing, the research on designing quantum codes specifically for asymmetric quantum channels is quite limited [41], especially when compared to results regarding the depolarizing channel. Thus this work represents one of the first attempts at designing QLDPC codes specifically for asymmetric quantum channels.

The remainder of this paper is structured as follows. We commence with an overview of some important preliminary topics in Sec. II. We proceed by presenting the quantum channel model in Sec. III. In Sec. IV, we compare the performance of CSS QLDGM codes over the depolarizing channel to other state-of-the-art codes and explain why we have chosen them to build codes for the general Pauli channel. Additionally, we discuss and propose a method to modify CSS QLDGM codes so that they are capable of exploiting the asymmetry of the considered quantum channel. In Sec. V, we show the simulation results for our proposed design and compare them to other strategies that have been proposed in the literature. Section VI concludes the paper.

II. PRELIMINARIES

In this section, a brief review of the concepts, definitions, and notation used in this paper is provided.

A. Basic concepts

1. Quantum Information

The simplest quantum mechanical system and the basic unit in quantum information is known as the qubit. In the state vector formulation, it is denoted by $|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \in \mathcal{H}_2$; where $\alpha, \beta \in \mathbb{C}$, $|\alpha|^2 + |\beta|^2 = 1$ and $\mathcal{H}_2$ refers to the complex Hilbert space of dimension 2. Another formulation of quantum mechanics can be given in terms of the so-called density matrices ρ, which is useful in order to describe systems whose state is not completely known in state vector terms.

2. The Pauli group and the effective Pauli group

The *Pauli group* is a mathematical group of significant interest for quantum stabilizer codes. Let Π be the set of Pauli operators $\{I, X, Y, Z\}$, and $\Pi^{\otimes N} = \{I, X, Y, Z\}^{\otimes N}$ denote the set of N-fold tensor products of single-qubit Pauli operators. Then, $\Pi^{\otimes N}$ together with the possible overall factors $\pm 1, \pm i$ forms a group known as the *N-fold Pauli group* $\mathcal{G}_N$, defined as $\mathcal{G}_N = \{\beta_1 I, \beta_2 X, \beta_3 Y, \beta_4 Z\}^{\otimes N}$, where $\beta_k = \{\pm 1, \pm i\}$.

Now let $[A] = \{\beta A | \beta \in \mathcal{C}, |\beta| = 1\}$ be the equivalence class of matrices equal to A up to a phase factor. Then the set $[\bar{\mathcal{G}}_N] = [\Pi^{\otimes N}] = \{[I], [X], [Y], [Z]\}^{\otimes N}$ forms an Abelian group under the multiplication operation defined by $[A][B] = [AB]$. This group is called the *effective N-fold Pauli group*.

B. Stabilizer codes

Quantum stabilizer codes are a class of QECCs that can be efficiently designed based on existing classical codes. A stabilizer code $C(S)$ is defined by a set of operators S that generate an abelian subgroup of the N-fold Pauli group $\mathcal{G}_N$ under multiplication. The codespace defined by the stabilizer group is

$$C(S) = \left\{|\psi\rangle \in \mathcal{H}_2^{\otimes N} : S_i|\psi\rangle = |\psi\rangle, \forall i\right\},$$

i.e., the simultaneous $+1$-eigenspace defined by the elements of the stabilizer group S, where S_i denotes each generator of the stabilizer group.

A generator of a stabilizer code, or more generally any Pauli operator on N qubits, can be described in terms of its symplectic representation [42]. Using this representation, each element of the N-fold Pauli group can be written as a unique binary string of length $2N$, which is built by joining two separate binary strings[1] of length N. Individually, each of the length N binary strings will represent the presence of a Z or X operator on each of the N qubits. Considered jointly, the strings also represent I and Y operators. More explicitly, given an element of $\mathcal{G}_N$ represented by the length $2N$ string $U = (U_z|U_x)$, where U_z and U_x are length N strings, zero entries in the same position of both length N strings represent a single qubit I operator, a nonzero entry in U_z and a zero entry in the same position of U_x will represent a single qubit Z operator, a zero entry in U_z and a nonzero entry in the same

[1]Note that by doing so, the global phase is lost, and so the map is between the effective Pauli group and the binary field. Global phase has no observable consequences, so neglecting it makes good physical sense.

position of U_x will represent a single qubit X operator, and nonzero entries in the same position of both strings represent a single qubit Y operator. Applying this representation to the generators S_i of a stabilizer code enables the definition of the quantum parity check matrix (QPCM) for the code. The QPCM of a stabilizer code will be in the form $H_Q = (H_z|H_x)$, where row i of matrix H_Q is the symplectic representation of stabilizer generator S_i.

Using this QPCM notation, the requirement that stabilizer generators must commute can be re-expressed for the entire stabilizer code as

$$H_z \star H_x = \left(H_z H_x^T + H_x H_z^T\right)\text{mod}2 = 0\,, \tag{1}$$

where the $\star$ operator, known as the symplectic product, represents the operation itself. This expression, referred to as the *symplectic criterion*, is significant because it determines which existing classical codes can be used to design stabilizer codes.

In most quantum channels, decoherence is modelled by means of errors that belong to the N-fold Pauli group, which either commute or anticommute with each of the stabilizer generators S_i of a given stabilizer code $C(S)$ [42]. An error operator E can be described using the symplectic representation as the length $2N$ binary string e. If we write e as $(e_z|e_x)$, when multiplying e in terms of the symplectic product (mod2) by a row of the parity check matrix of a stabilizer code, 0 will be obtained if E and the generator associated to that row commute, whereas 1 will be obtained if they anticommute. As is shown in (2), multiplying this symplectic representation of the error operators by the quantum parity check matrix of a stabilizer code will yield the quantum syndrome s. We will later use this syndrome in the decoding process to estimate the error pattern e.

$$s = H_Q \star e = (H_z e_x + H_x e_z)\text{mod}2\,, \tag{2}$$

where $e = (e_z|e_x)$ is the symplectic representation of the error pattern, $H_Q = (H_z|H_x)$ is the quantum parity check matrix of a stabilizer code, and s represents the quantum syndrome.

C. CSS codes

Two binary classical LDPC codes can only be used to construct a stabilizer code if they satisfy the symplectic criterion (1). The first design strategy one could devise to construct stabilizer codes would be the random selection of pairs of classical LDPC codes. However, finding two LDPC codes of reasonable block size that satisfy (1) is highly unlikely. Calderbank-Shor-Steane (CSS) codes [23,24], provide a more efficient design strategy than random selection of classical codes. The quantum parity check matrix of these codes is written as

$$H_Q = (H_z|H_x) = \begin{pmatrix} H'_z & 0 \\ 0 & H'_x \end{pmatrix}, \tag{3}$$

where $H_z = \binom{H'_z}{0}$ and $H_x = \binom{0}{H'_x}$.

In this construction, H'_z and H'_x are the parity check matrices of two classical LDPC codes C_1 and C_2, respectively, where each matrix is used to correct either bit flips (H'_z) or phase flips (H'_x). The classical codes are chosen so that $C_2^\perp \subseteq C_1$, where $C_2^\perp$ is the dual of the classical LDPC code C_2. This design constraint, generally referred to as the *CSS condition*, reduces (1) to $(H'_z H'^T_x)\text{mod}2 = 0$.

D. Systematic classical LDGM codes

Systematic LDGM codes are useful, both in classical and quantum environments, because of the particular structure of their generator and parity check matrices. Let C be a systematic LDGM code. Then, its generator matrix $\tilde{G}$ and its parity check matrix $\tilde{H}$ can be written as

$$\tilde{G} = (\mathbb{I}\ P), \quad \tilde{H} = (P^T\ \mathbb{I}), \tag{4}$$

where $\mathbb{I}$ denotes the identity matrix, and $P = [p_{lm}]$ is a sparse matrix.

Because LDGM codes belong to the family of linear block codes, these matrices will satisfy $(\tilde{G}\tilde{H}^T)\text{mod}2 = (\tilde{H}\tilde{G}^T)\text{mod}2 = 0$. Those systematic LDGM codes in which the rows and columns of the PCM have degrees[2] X and Y, respectively, will be denoted as (X, Y) regular LDGM codes. Regular LDGM codes are known to be asymptotically bad [3], displaying error floors that do not decrease with the block length. However, in Ref. [43], codes built via the parallel concatenation of two regular LDGM codes[3] were shown to yield significant reduction in these error floors. The parallel concatenation of two regular LDGM codes with generator matrices $G_1 = [\mathbb{I}\ P_1]$ and $G_2 = [\mathbb{I}\ P_2]$, where P_1 and P_2 have degree distributions (y_1, y_1) and (y_2, z_2), is the irregular LDGM code with generator matrix $G = [\mathbb{I}\ P_1 P_2]$. We can represent such a scheme using the notation $P[(y_1, y_1); (y_2, z_2)]$. Generally, this concatenation is accomplished by using a high rate code G_2 that is able to reduce the error floor of G_1, while also causing negligible degradation of the original convergence threshold.

Classical LDPC decoding is performed by solving the equation $s = H_c e$, where s represents the received syndrome, H_c is the PCM of the code, and e is the error pattern we wish to recover. Given that LDGM codes are a specific subset of LDPC codes, they are decoded in exactly the same manner as generic LDPC codes. However, LDGM decoding can also be interpreted as a method to solve equation $c = Pu$, where c is the vector of parity bits generated at the encoder, P is the constituent sparse matrix of the LDGM generator matrix, and u is the information message we want to obtain. Thus the decoding algorithm for LDGM codes can be implemented by applying belief propagation (BP) [44] or the sum-product algorithm (SPA) [45] over the graph associated to the equation $c = Pu$.

III. QUANTUM CHANNEL MODEL

The effects quantum decoherence has on quantum information are usually described by means of quantum channels, $\mathcal{N}$.

[2]The degree of the columns is the number of nonzero entries per column of the PCM. The degree of the rows is given by the number of nonzero entries per row of the PCM. An LDGM code is said to be regular when all the rows of its PCM have the same number of nonzero entries, X, and so do its columns, Y.

[3]The parallel concatenation of regular LDGM codes is equivalent to an LDGM code with an irregular degree distribution.

A widely applied quantum channel model used to represent the decoherence effects suffered by quantum information described by a density matrix ρ, is the generic Pauli channel $\mathcal{N}_P$. The effect of the Pauli channel $\mathcal{N}_P$ upon an arbitrary quantum state is described by

$$\mathcal{N}_P(\rho) = (1 - p_x - p_y - p_z)\rho + p_x X\rho X + p_y Y\rho Y + p_z Z\rho Z.$$

A qubit then experiences a bit flip (X operator) with probability p_x, a phase flip (Z operator) with probability p_z or a combination of both (a Y operator) with probability p_y. When quantum states of N qubits are considered, the errors that take place belong to the N-fold Pauli group[4] $\mathcal{G}_N$. We define $p = p_x + p_y + p_z \leqslant 1$ as the gross flip probability of the generic Pauli channel.

A. Symmetric Pauli channel

In most of the work conducted on quantum error correction, the symmetric Pauli channel model, also known as the depolarizing channel, is considered [5,7,46]. This model is a specific instance of the Pauli channel in which the individual flip probabilities are all equal, i.e., $p_x = p_z = p_y$, and the channel is completely characterized by the gross flip probability p, commonly referred to as the depolarizing probability. When considering this symmetric instance of the Pauli channel, errors will act independently on each qubit causing an X, Z, or Y error with probability $p/3$ and leaving it unchanged with probability $(1 - p)$.

B. iid X/Z channel

In some scenarios, it is useful to employ a simpler channel model than the depolarizing channel. The *standard flipping channel* or *iid X/Z channel*, is introduced in Ref. [13], where Z and X errors are modelled as independent events identically distributed according to the flip probability f_m. This quantum channel model is analogous to two independent Binary Symmetric Channels (BSCs) with marginal bit-flip probability $f_m = 2p/3$, where the separate BSCs can be seen as Z and X error channels, respectively. Given that Y errors occur when both a phase flip and a bit flip happen to the same qubit, the simplified notion of the iid X/Z channel ignores any correlation that exists between X and Z errors in the depolarizing channel.

C. Realistic Pauli channel model

As has been mentioned throughout this paper, quantum devices are constructed employing specific materials that cause these devices to exhibit asymmetric behavior. This asymmetry is embodied by the probability of a bit flip p_x being orders of magnitude smaller than the probability of a phase flip p_z. To build a Pauli channel model that accurately represents this phenomenon the asymmetric relationship between p_x, p_y, p_z, and p must be established. This is achieved by introducing the parameter α, known as the channel's ratio of asymmetry [40], which arises due to the twirling of the channel [34]. This parameter represents the ratio of the phase-flip probability and the bit-flip probability as [47,48]

$$\alpha = \frac{p_z}{p_x} = 1 + 2\frac{e^{\frac{-t}{T_1}} - e^{\left(\frac{-t}{2T_1} - \frac{2t}{T_2}\right)}}{1 - e^{\frac{-t}{T_1}}}, \tag{5}$$

where T_1 is the relaxation time, T_2 represents the dephasing time, and t is the coherent operation duration of a physical quantum gate [49]. In Refs. [34,47] expressions for p_x, p_y, and p_z are given (6), where the bit-flip probability and bit-and-phase-flip probability can be considered to be equal.

$$p_x = p_y = \frac{1 - e^{\frac{-t}{T_1}}}{4}, \quad p_z = \frac{1}{2} - p_x - \frac{e^{\frac{-t}{T_2}}}{2}. \tag{6}$$

If the coherent operation duration t is assumed to be reasonably short, i.e., $t << T_1$ [47], then from (5) the ratio of the phase and bit-flip probabilities can be approximated by $\alpha \approx 2\frac{T_1}{T_2} - 1$. In consequence, this model allows us to completely determine the values of p_z, p_x, and p_y from α and p. Common values for the ratio of asymmetry are given in Refs. [12,34,40], with most materials used to build quantum devices having $\alpha = [10^2, 10^4, 10^6]$. Notice that, if we select $\alpha = 1$, we obtain the depolarizing channel model that is considered in most circumstances and that satisfies $p_x = p_y = p_z = \frac{p}{3}$. This last value of α exists for specific types of devices, hence the depolarizing channel can sometimes also provide a realistic representation of the behavior of a quantum machine.

IV. QLDGM CSS CODE DESIGN FOR ASYMMETRIC CHANNELS

In this section, as is done in Ref. [31], we introduce the design technique for symmetric CSS QLDGM codes of Refs. [25,26]. Following this, we justify why we have chosen symmetric CSS QLDGM codes as the basis to design QEC strategies for more realistic quantum channels by comparing their performance with other QLDPC codes. Having provided the necessary context, we then propose a scheme that optimizes these CSS codes for the asymmetric Pauli channel introduced in Sec. III C.

A. Symmetric QLDGM CSS codes

The first intuition to derive the QPCM of a QLDGM CSS code would be to select any classical LDGM code with parity check and generator matrices $\tilde{H}$ and $\tilde{G}$, and set $H'_z = \tilde{H}$ and $H'_x = \tilde{G}$ in (3), since the property $(\tilde{G}\tilde{H}^T)\text{mod}2 = (\tilde{H}\tilde{G}^T)\text{mod}2 = 0$ would ensure the fulfillment of the symplectic criterion. However, this results in a QPCM H_Q of size $N \times 2N$, which cannot be used for encoding purposes as it would produce a code of $R_Q = 0$. In order to build a valid quantum code, the number of rows in H_Q must be reduced, while ensuring that the *CSS condition* is fulfilled. in Ref. [21], the authors successfully reduce the number of rows of the generator and parity check matrices of a classical LDGM code via linear row operations while showing that the *CSS condition* is kept. They achieve this by applying the following theorem.

Theorem 1. Given the generator and parity check matrices of a systematic LDGM code (4), define $H_{m_1 \times N} =$

[4]Given that the global phase has no observable consequence, the instances of considered errors will be the elements of $\bar{\mathcal{G}}_N$.

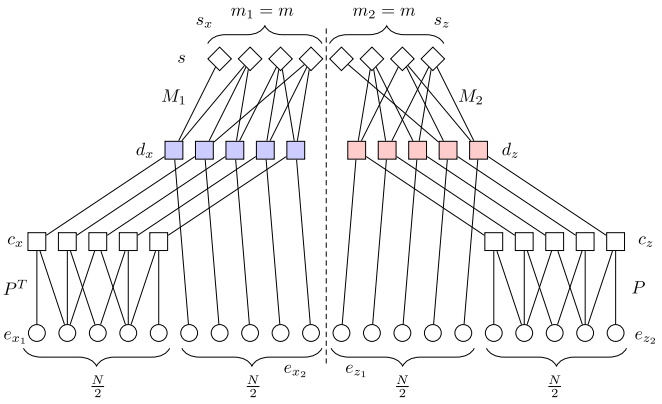

FIG. 1. Decoding graph for a symmetric QLDGM CSS scheme. The dashed line is included to emphasize the separation of the two constituent subgraphs. The leftmost subgraph decodes the X errors while the one on the right decodes the Z errors. We have assumed that $m_1 = m_2 = \frac{m}{2}$, $n_1 = n_2 = \frac{N}{2}$, and $m = m_1 + m_2$.

$[M_1]_{m_1\times n_1}[\tilde{H}]_{n_1\times N}$ and $G_{m_2\times N} = [M_2]_{m_2\times n_2}[\tilde{G}]_{n_2\times N}$, where $n_1 + n_2 = N$ and M_1 and M_2 are low-density full-rank binary matrices whose number of rows satisfy $m_1 < n_1$ and $m_2 < n_2$, respectively. Then, the quantum PCM shown in (7), obtained by setting $H'_z = H$ and $H'_x = G$ in (3), is the quantum PCM of an LDGM-based CSS code with rate $R_Q = \frac{N-m_1-m_2}{N}$.

$$H_Q = (H_z|H_x) = \begin{pmatrix} H & 0 \\ 0 & G \end{pmatrix} = \begin{pmatrix} M_1\tilde{H} & 0 \\ 0 & M_2\tilde{G} \end{pmatrix}. \quad (7)$$

Quantum CSS LDGM codes are decoded by applying the SPA over the factor graph defined by the QPCM in (7). The derivation of this factor graph is performed as in Ref. [21], by splitting the symplectic representation of the error pattern into two parts, $e = (e_z|e_x)$, and relating it to the syndrome via a two step process.[5] As a result of the structure of CSS codes, the syndrome can also be split into two parts, $s = (s_x|s_y)$, where each part of the syndrome contains information regarding either bit or phase flips and is directly related to either e_x or e_z. In the following, we illustrate this derivation for e_x, the part of the symplectic representation of the error sequence related to the X operators. The procedure for e_z is identical but using G instead of H in (8).

$$s_x = He_x = M_1\tilde{H}e_x = M_1[P^T\ \mathbb{I}]e_x. \quad (8)$$

If we now split the symplectic representation of the error pattern of the X operators into $e_x = (e_{x_1}\ e_{x_2})^T$, we can write

$$d_x = [P^T\ \mathbb{I}]e_x = [P^T\ \mathbb{I}]_{n_1\times N}\begin{pmatrix} e_{x_1} \\ e_{x_2} \end{pmatrix}_{N\times 1}$$
$$= P^T_{n_1\times n_2}[e_{x_1}]_{n_2\times 1} + [e_{x_2}]_{n_1\times 1}. \quad (9)$$

We then relate d_x to the syndrome as

$$s_{x_{m_1\times 1}} = M_{1_{m_1\times n_1}}d_{n_1\times 1}. \quad (10)$$

The factor graph shown in Fig. 1 is obtained based on expressions (9) and (10), as well as their equivalents when using e_z and G in (8).

[5]The syndrome is obtained as shown in (2).

Upon closer examination of the QPCM in (7), it is easy to see that decoding for the H and G matrices can be done separately. This is also visible in Fig. 1, where the leftmost subgraph is associated to the decoding of matrix H (e_x or X containing operators) and the rightmost subgraph is associated to the decoding of matrix G (e_z or Z containing operators). Separate decoding of these matrices is made possible by the nature of CSS constructions, which, as was mentioned earlier, results in syndrome nodes containing information only of either X or Z operators. This is reflected on the factor graph by the fact that each type of s node (s_x or s_z nodes) connects to either a d_x or a d_z node. The graph is constructed by setting $m_1 = m_2$, which ensures that the same number of syndrome nodes is used to decode the X and Z operators: $\frac{m}{2}$ s nodes are connected strictly to d_x nodes and the other $\frac{m}{2}$ s nodes only connect to d_z nodes. This equal distribution of syndrome nodes for each type of error operator is logical given that the code is designed for the depolarizing channel, where X, Y, and Z errors are equally likely. Asymmetric channels like the Pauli channel model of Sec. III C present a different quandary, in which, as will be shown later, using more syndrome nodes to decode more likely error operators and less syndrome nodes to decode less likely error operators can have a positive impact on performance.

The matrix multiplications used to perform the linear row operations on $\tilde{H}$ and $\tilde{G}$ (7) generate a middle layer, represented by the c and d nodes, in both decoding subgraphs of Fig. 1. This new layer hampers the decoding algorithm, especially during the initial decoding iterations, since *a priori* information regarding the aforementioned middle layer nodes is not available. in Ref. [21], the authors circumvent this lack of information by using the so-called *doping* technique of [50]. This method introduces degree-1 syndrome nodes into the decoding graph. These degree-1 nodes, which we will refer to as s_A nodes, send exact[6] information to the d nodes they are connected to. The transmission of correct syndrome information from the s_A nodes to the d nodes represents a passing down of accurate knowledge to lower layers of the factor graph that should make up for the lack of information regarding c and d nodes during initial decoding iterations. This should have a positive impact on decoding performance and ultimately push the entire process in the right direction. The degree-1 syndrome vertices are embodied within the M_1 and M_2 matrices as rows with a single nonzero entry, which corresponds to the edge that connects a given s_A node to a d node. The other rows of matrices M_1 and M_2, which correspond to the rest of the s nodes, have as many nonzero entries as required to guarantee the regularity[7] of the d nodes and the necessary number of s_A nodes. This results in matrices M_1 and M_2 having a special degree distribution which is described by means of the notation $(y; 1, x)$ and the parameter t, where y

[6]These messages are exact because, as required by the SPA update rule and the fact that s_A nodes are degree-1 nodes, the messages they send are strictly dependent on the syndrome information available at each s_A node.

[7]Regularity in this context implies that all the d nodes have the same degree, i.e., that they are all connected to the same number of s nodes.

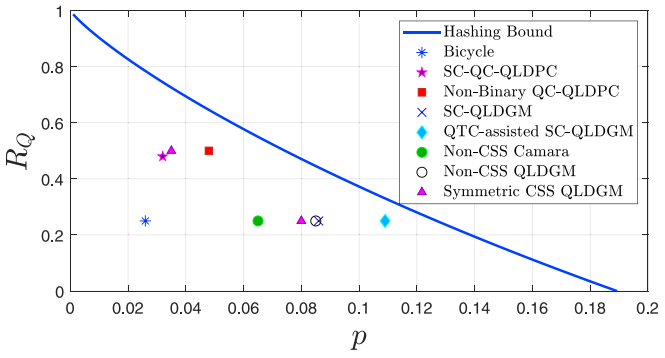

FIG. 2. Achievable coding rate at a WER of 10^{-3} for different types of QLDPC codes.

represents the degree of the d nodes, t is the number of syndrome nodes that are forced to have degree 1 (they become s_A nodes), and x represents the degree of the remaining syndrome nodes, referred to as s_B nodes.

Given the particular structure of the M_1 and M_2 matrices, and the number of different types of nodes that are present in the factor graph shown in Fig. 1, the sum-product decoding of these quantum LDGM CSS codes becomes relatively nuanced. In Ref. [26], a technique known as discretized density evolution (DDE) [4] is applied to optimize quantum LDGM CSS codes, which also provides a complete description of how the decoding process unfolds over the graph shown in Fig. 1.

1. Performance comparison with other QLDPC codes

Based on the discussion given in this section, it is easy to see that designing LDGM-based CSS quantum codes is not a complex task. In fact, all that is needed to build the factor graphs of such codes is a set of matrices comprised of the generator and parity check matrices of a classical LDGM code, $\tilde{G}$ and $\tilde{H}$, and the matrices M_1 and M_2 given in theorem 1. This set of matrices will define code parameters such as the rate R_Q or distance of the code, and given how straightforward it is to modify these matrices, it will be relatively simple to adapt the construction of QLDGM CSS codes to different requirements to those of the depolarizing channel. As will be shown in the following section, this comes in handy when designing codes for the asymmetric Pauli channel.

Nonetheless, given that CSS QLDGM codes were originally designed to operate over the depolarizing channel, it is important to analyze how symmetric LDGM-based CSS codes perform over the depolarizing channel when compared to other existing QLDPC codes. As is done in Ref. [20], in Fig. 2, we show the highest possible coding rate at which various QLDPC codes that have been proposed in the literature can achieve a word error rate of 10^{-3} over the depolarizing channel. In this figure, we compare the performance of two symmetric CSS QLDGM codes based on the structure $M(3;1,11.04)$ and $P[(8,8);(8,160)]$ with blocklength $n = 19014$ and rates $R_Q = \frac{1}{2}$ and $R_Q = \frac{1}{4}$ to the following codes:

(1) The $K = 32$ bicycle code with block length $n = 19014$ and rate $R_Q = \frac{1}{4}$ proposed in Ref. [13].

(2) The Spatially Coupled (SC) Quasi-Cyclic (QC) QLDPC code of rate $R_Q = 0.49$ and blocklength $n = 181000$ given in Ref. [51].

(3) The nonbinary QC-QLDPC GF(2^{10}) code of rate $R_Q = \frac{1}{2}$ and block length $n = 20560$ proposed by Kasai et al. in Refs. [27,28].

(4) The SC-QLDGM code of $R_Q = \frac{1}{4}$ and block length $n = 76800$ proposed in Ref. [29].

(5) The QTC-assisted SC-QLDGM code of $R_Q = \frac{1}{4}$ and block length $n = 821760$ of [30].

(6) The non-CSS concatenated code (code C) of $R_Q = \frac{1}{4}$ and block length $n = 138240$ of [46].

(7) The $t = 5000$, $q = 500$, $M(3;1,11.04)$, $P[(8,8);(8,160)]$ non-CSS QLDGM code of $R_Q = \frac{1}{4}$ and block length $n = 19014$ of [31].

As can be seen in Fig. 2, at both of the considered rates, the symmetric CSS QLDGM code is outperformed by some of the other codes. At a rate of $R_Q = \frac{1}{4}$, the symmetric CSS QLDGM code is beaten by the non-CSS implementation of Ref. [31], the SC-QLDGM code of [29], and the QTC-assisted SC-QLDGM code of Ref. [30]. However, this comes as no surprise, since all three of these codes take a symmetric CSS QLDGM code as their starting point and then modify it (by changing the factor graph or combining them with a QTC) with the purpose of improving performance. For a rate of $R_Q = \frac{1}{2}$, the symmetric CSS QLDGM code is once again outperformed by the nonbinary QC-QLDPC GF(2^{10}) code of [27,28]. We can further expand this comparison by looking at the distance of each code to the Hashing bound. This can be done as in Ref. [31], by computing

$$\delta = 10\log_{10}\left(\frac{p^*}{p}\right), \tag{11}$$

where p^* is the noise limit of the depolarizing channel for a specific quantum rate R_Q, and p is the highest depolarizing probability at which the code in question can operate with a WER of 10^{-3}.

At a rate of $R_Q = \frac{1}{2}$, the QC-QLDPC code of [27,28] is $\delta_{\text{QC-QLDPC}} = 1.9$ dB away from the Hashing bound. At this same rate, the distance for the symmetric CSS QLDGM code is $\delta_{\text{CSS-QLDGM}} = 2.86$ dB. Based on these values, it is clear that when $R_Q = \frac{1}{2}$, QC-QLDPC codes significantly outperform CSS QLDGM codes. In contrast, at a rate of $R_Q = \frac{1}{4}$, the symmetric CSS QLDGM code is $\delta_{\text{CSS-QLDGM}} = 1.95$ dB away from the Hashing bound. This means that the performance of CSS QLDGM codes is better at a lower rate, which may entail that a different optimization strategy must be adopted to improve the performance of higher rate CSS QLDGM codes, similar to what is done classically in Ref. [52]. Once more, the non-CSS and the SC-QLDGM codes of Refs. [29,31], which exhibit approximately the same distance to the Hashing Bound $\delta_{\text{non-CSS}} \approx \delta_{\text{SC-QLDGM}} = 1.69$ dB, outperform the CSS QLDGM code. However, at this rate, the difference in performance is notably less significant than when $R_Q = \frac{1}{2}$.

At this point, it should be noted that these improvements in performance come at the expense of the complexity of the error correction schemes. This is especially true for the best known code for $R_Q = \frac{1}{4}$, the QTC-assisted SC-QLDGM code of Ref. [30], which requires a QTC and a large block length in order to get closer to the Hashing bound. Herein lies the main appeal of CSS QLDGM codes, because although they are slightly worse than the state-of-the art codes

at $R_Q = \frac{1}{4}$, the simplicity with which their design parameters can be manipulated allows for them to be seamlessly adapted to different channels. Doing so for the other codes included in this discussion is a much more difficult task, since their increased complexity does not allow direct modifications like those permitted by CSS QLDGM codes. For this reason, and knowing that performance of CSS QLDGM codes over the depolarizing channel is acceptable at $R_Q = \frac{1}{4}$, we use these codes in the following section as the basis to design asymmetric CSS QLDGM codes for a more realistic Pauli channel model.

B. Asymmetric QLDGM CSS codes

Over Pauli channels that model practical quantum devices, a phase flip is generally much more likely to occur than a bit flip. Therefore it is reasonable to assume that for a quantum error correction scheme to be optimal for this type of channel, it must be capable of appropriately exploiting the channel's asymmetry. To be more precise, attaining the best performance over such channels requires a more complex strategy (by modifying the code construction) than just decoding as is done over the depolarizing channel with the only difference that the flip probabilities will no longer be $\frac{2p}{3}$. This means that decoding a symmetric QLDGM CSS scheme by feeding the a priori bit and phase error probabilities $f_m^1 = p_x$ and $f_m^2 = p_z$ of an asymmetric channel to the corresponding bit and phase error decoders, while certainly an improvement to decoding over an asymmetric channel based on the mismatched probability of the original iid channel $f_m = \frac{2p}{3}$, will not result in noticeable performance improvements. This is shown in the following section, which also portrays the significant improvement yielded by asymmetric CSS schemes that tailor specifically to the Pauli channel model for asymmetry.

The QLDGM CSS scheme introduced in the previous subsection can be adapted to an asymmetric channel by increasing the number of syndrome nodes used to decode the Z operators and decreasing the number of syndrome nodes used to decode the X operators. In this way, the decoder can take advantage of the channel's asymmetric behavior and improve the performance of the error correcting scheme. The factor graph of a QLDGM CSS code tailored to the Pauli channel model for asymmetry is shown in Fig. 3.

Despite how intuitive the idea appears, it is worth discussing why utilizing more syndrome nodes to decode Z operators and less syndrome nodes to decode X operators is beneficial when the considered channel is asymmetric. The asymmetry-integrating Pauli channel model causes phase flips (Z errors) with much higher probability than bit flips or bit-and-phase flips (X and Y errors, respectively). Thus the symplectic error representation[8] of a pattern induced by this asymmetric channel will have a much higher number of nonzero elements in its e_z string than in its e_x string. In contrast, when e represents an error induced by a depolarizing channel, e_x and e_z will, on average, have the same number of nonzero entries. This presence of similar amounts of X and Z errors in error sequences produced by the depolarizing channel is the very reason why the decoding graph of Fig. 1 uses the same amount of syndrome information to decode the e_x and e_z nodes. However, such a graph will more than likely not be optimal for an asymmetric scenario in which the distribution of nonzero entries over the length N constituent strings of the symplectic representation of an error pattern is not equal like for the depolarizing channel.

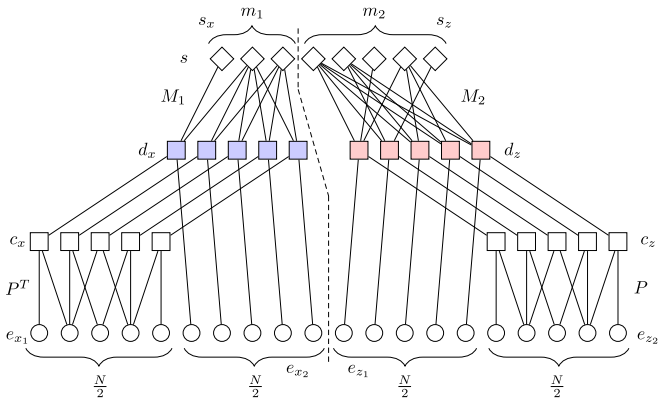

FIG. 3. Decoding graph for an asymmetric QLDGM CSS scheme. The dashed line is included to emphasize the separation of the two constituent subgraphs. The leftmost subgraph decodes the X errors while the one on the right decodes the Z errors. We have assumed that $n_1 = n_2 = \frac{N}{2}$ and $m = m_1 + m_2$.

Let $e^{\text{dep}} = (e_z^{\text{dep}}|e_x^{\text{dep}})$ and $e^{\text{asym}} = (e_z^{\text{asym}}|e_x^{\text{asym}})$ denote the symplectic representations of two error patterns induced by a depolarizing channel and a Pauli channel that models a realistic quantum device, respectively. Let us also define the operator $\sigma(a)$, which computes the number of nonzero entries in a binary string a. Finally, assume that the asymmetry coefficient of the assymetric Pauli channel in question satisfies $\alpha \geqslant 10^2$. For the same value of p, $\sigma(e^{\text{dep}}) \approx \sigma(e^{\text{asym}})$. However, while $\sigma(e_z^{\text{dep}}) \approx \sigma(e_x^{\text{dep}})$, the same does not occur for the asymmetric channel, $\sigma(e_z^{\text{asym}}) \ggg \sigma(e_x^{\text{asym}})$. Additionally, $\sigma(e_z^{\text{asym}}) \gg \sigma(e_z^{\text{dep}})$ and $\sigma(e_x^{\text{asym}}) \ll \sigma(e_x^{\text{dep}})$. In consequence, it is quite obvious that a decoder tasked with decoding error patterns induced by a general Pauli channel will benefit from an uneven decoding graph in which more syndrome information is employed to decode e_z and less syndrome nodes are utilized to decode e_x.

The design of such a decoding graph, an example of which is shown in Fig. 3, gives rise to a whole new set of questions. The first and most significant one is how can the optimum values for m_1 and m_2 be determined, where m_1 and m_2 denote the number of syndrome nodes used to decode the X and Z operators, respectively. It is evident that $m_2 > m_1$, with said difference growing larger as the asymmetry of the channel increases. Ideally, we would like to devise a mathematical formulation from which the values of m_2 and m_1 that yield the best possible performance could be obtained.

Another important matter, of which little insight is possessed, is which configuration of the M_1 and M_2 matrices will yield the best results. However, establishing which values of

[8]Recal that the symplectic representation of an error pattern $e = (e_z|e_x)$ is a length $2N$ binary string where e_z and e_x are length N binary strings that describe the presence of phase flips and bit flips with nonzero entries, respectively.

$(y; 1, x)$ and t for each of these matrices is optimal, further augments the complexity of the asymmetric design procedure when compared to the symmetric scenario. This increase in complexity is caused by the fact that exploiting the asymmetry of the channel requires $M_1 \neq M_2$, which theoretically allows for myriads of different configurations in terms of the values chosen for m_i, y_i, x_i, and t_i, where $i = 1, 2$. Finding the optimum configuration through a brute force search requires such a plethora of simulations that the issue becomes computationally intractable.

In order to simplify our search for these matrices, we recover the design methodology of [25,26,53] used to construct symmetric CSS codes. In this work, the construction procedure of the M_1 and M_2 matrices is simplified by the fact that, since the codes are designed for the symmetric Pauli channel, $\frac{m}{2} = m_1 = m_2$ holds and $M_1 = M_2 = M$. This means that instead of building two different matrices, the same matrix M is used to define the upper layers of both CSS subgraphs.

The methodology to construct $[M(y; 1, x)]_{\frac{m}{2} \times \frac{N}{2}}$ begins by defining the values of m, the total number of syndrome nodes of the decoding graph, and N, the block length of the code. N is chosen to be sufficiently large so as to ensure the code will possess good error correcting capabilities, while m is selected to guarantee that the code has the desired quantum rate, which for these CSS QLDGM codes is given by $R_Q = \frac{N-m}{N}$ [31]. Once again, since these codes are built for the symmetric Pauli channel, $\frac{m}{2}$ syndrome nodes are assigned to each CSS subgraph. Following this, y is set as a natural number to make sure that the d nodes of the CSS subgraphs have the same number of edges, and the number of s_A nodes is chosen as $t \leqslant \frac{m}{2}$. Finally, x is obtained from the following equation:

$$\left(\frac{m}{2} - t\right)x + t = y\frac{N}{2}. \tag{12}$$

In Refs. [25,26,31,53], where $R_Q = \frac{1}{4}$ codes with $N = 19014$ and $m = 14262$ are considered, the configurations of $[y, x, t]$ that achieved the best performance were [3,8.72,4161] and [3,11.04,5000]. These configurations were also shown to be slightly dependant on the characteristics of the underlying parallel-concatenated classical LDGM code.

Finding the combination of the parameters involved in (12) that produces the code with best possible performance is no easy task. Nonetheless, as is done in Refs. [25,26,31], certain assumptions can be made in order to reduce the complexity of this endeavour. To begin with, we know that N and m are fixed in order to define the desired rate of the code. If N is appropriately chosen (it is large enough to guarantee good error correction potential of the code), this simplifies matters and reduces the number of parameters from (12) that must be studied. Now, recall that y must be set as a natural number to ensure the regularity of the d nodes. In Refs. [25,26,31] results showed that only a single value of this parameter yielded positive outcomes[9], $y = 3$. Given that the codes introduced in this paper are based on the structures proposed in the aforementioned work, it is reasonable to adopt the same value for y in our constructions. In consequence, this results in N, m, and y being fixed to specific values, implying that the only parameters in Eq. (12) that can actually be modified are t, the number of s_A nodes, and x, the degree of the s_B nodes. Several insights regarding the value of these parameters can be obtained by analyzing our previous discussion and the aforementioned equation.

(1) For large values of x, the reliability of the messages transmitted by the s_B nodes in the decoding process is significantly reduced. This occurs because when nodes have many edges in SPA-based decoding, the messages that are considered in the computations of each of these nodes are numerous enough to have an "averaging" effect and reduce the impact of any one given message. Naturally, this should hinder the overall performance of the code.

(2) As the values of x grow, given that the RHS of (12) is fixed, the value of t will also be larger. Note that this is intuitive: the more degree-1 syndrome nodes that there are (the larger the value of t), the larger the degree of the remaining s_B nodes (the larger the value of x) will be because the degree y of the lower layer d nodes must remain the same, which can only be guaranteed by adding more edges to the s_B nodes. Growth in the value of t should have a positive impact on performance, as having more s_A nodes in the decoding graph means that more "perfect" information from these degree-1 syndrome nodes will be transmitted to the lower layer nodes in the initial decoding iterations.

Against this backdrop, it seems likely that the optimum values of x and t will be dictated by a trade-off between these two effects and that the choice of x will be inherently linked to the choice of t.

As was mentioned previously, this search for the best combination of t and x is further complicated by the fact that our goal is to design asymmetric CSS codes. This is more difficult than building CSS QLDGM codes for the depolarizing channel because the matrices M_1 and M_2 must now be different in order to exploit the asymmetry of the channel. Because more syndrome nodes are used to decode phase flips and less syndrome nodes are used to decode bit flips, $m_1 \neq m_2$, the x and t parameters corresponding to each of these matrices must now be optimized. Therefore, to appropriately design LDGM-based CSS codes for the Pauli channel model of Sec. III C, we have to adapt Eq. (12) into the new verison shown below

$$\left(\frac{m_i}{2} - t_i\right)x_i + t_i = y_i\frac{N}{2}, \tag{13}$$

where $i = 1, 2$. Essentially, two designs have to be optimized instead of one: we must now find the configurations of $M_{m_1 \times \frac{N}{2}}(y_1; 1, x_1)$ and t_1 and $M_{m_2 \times \frac{N}{2}}(y_2; 1, x_2)$ and t_2 that yield the CSS codes with the best performance. Recall that the rate of the scheme will remain fixed since the sum $m = m_1 + m_2$ does not change. The demands of this process are discussed in the following section, where we show how in reality, most of the parameter optimization is only needed for one of these matrices.

[9]Choosing $y = 2$ resulted in too little syndrome information being propagated throughout the graph and applying $y = 4$ resulted in worse and slower decoding due to the large amount of messages exchanged over the graph.

V. SIMULATIONS

In this section, we study the performance of the proposed asymmetric CSS scheme over the Pauli channel model for asymmetry. First, we perform simulations to analyze the behavior of a variety of asymmetric schemes over a Pauli channel with a specific degree of asymmetry and compare these results to the performance of a symmetric CSS code when it is applied over that same channel. Based on these results, we proceed by studying those schemes that yield the best performance and narrowing down the search for the optimum size and configuration of the M_1 and M_2 matrices. Then, we propose a methodology to design asymmetric CSS QLDGM codes based on the asymmetry coefficient of the channel. Finally, we compare the performance of the proposed asymmetric schemes to the theoretical limits of the Pauli channel and study how they measure up against other codes that are found in the literature.

A. Performance over the asymmetric Pauli channel

Realistic Pauli channel models for quantum devices induces phase flips (Z errors) with much higher probability than bit flips (X errors) [34]. Asymmetric CSS QLDGM schemes can exploit this phenomenon by utilizing more syndrome nodes to decode Z errors and employing less syndrome information to decode X errors. Given that the only design guideline we possess to begin this analysis is that m_2 should be larger than m_1, we start by fixing the asymmetry coefficient of the channel to $\alpha = 10^2$, and simulating different configurations of the proposed asymmetric CSS scheme. For comparison purposes, we also simulate a symmetric CSS code [25,26] over the Pauli channel with $\alpha = 10^2$. We select the value $\alpha = 10^2$ because it is the smallest out of the set of realistic values for the asymmetric coefficient provided in Refs. [34,40]. Performance of the proposed schemes for channels with other degrees of asymmetry is studied in the last part of this section.

For our simulations, we build codes of quantum rate $R_Q = \frac{1}{4}$ and block length $N = 19014$ that encode $k = 4752$ qubits into N qubits. The pseudorandom matrix P of the underlying LDGM code has size 9507×9507, has the same degree distribution as its transpose P^T, and corresponds to a rate $\frac{1}{2}$ classical irregular LDGM code. The generator matrices of the irregular LDGM code, $\tilde{G}$ and $\tilde{H}$, have size 9507×19014. The irregular LDGM code is designed via the parallel concatenation of two regular LDGM codes.[10] As is done in Ref. [31], we use the particular concatenation $P[(8, 8)(3, 60)]$ because of its relatively small number of degrees, which reduces simulation time substantially. Once the optimum M_1 and M_2 configuration has been found, a parallel concatenated LDGM code of larger degrees can be used to improve performance. Figure 4 shows the performance of the simulated schemes and Table I outlines the details of each specific design. The results are depicted using the qubit error rate (QBER) and the WER. QBER represents the fraction of physical qubits that experience an error, while WER is the fraction of blocks that have at least one physical qubit error.

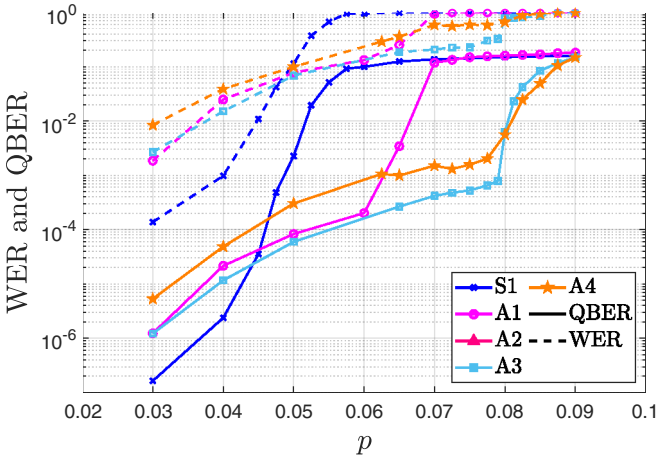

FIG. 4. Simulated QBER for different CSS QLDGM schemes. p represents the gross flip probability of the Pauli channel with asymmetry coefficient $\alpha = 10^2$.

As was expected, the results shown in Fig. 4 portray how the asymmetric CSS schemes outperform the symmetric CSS code over the Pauli channel with $\alpha = 10^2$. This can be appreciated by observing how the QBER and WER curves of the symmetric CSS code (code S1) enter the waterfall region and experience significant degradation at a substantially lower value of p than the asymmetric codes.

The simulation outcomes depicted in Fig. 4 also serve to provide insight regarding the performance determining factors of our asymmetric schemes. For instance, defining an increasingly unbalanced configuration of the upper layer of the decoding graph by selecting larger values of m_2 and decreasing the values of m_1 appears to have a positive impact on our construction. This is reflected by the betterment in QBER/WER results of the codes of Table I as m_2 grows. However, the performance curves of code A4, which has the largest value of m_2, exhibit a higher error floor than all the other simulated codes. This may occur because selecting such a large value for m_2 reduces the number of syndrome nodes leftover to decode the X operators to such an extent that the corresponding decoder, despite the low probability of bit flips over the channel, sees an inevitable increase in its error rate. At the same time, it may also be that for such a small value of m_1, the choice of $[t_1, x_1, y_1]$ is so critical that inappropriate

TABLE I. Parameter values and configurations of the CSS codes simulated over an Pauli channel with $\alpha = 10^2$. The results of these simulations are shown in Fig. 4.

Code Type	No.	$[m_1, t_1, x_1, y_1]$	$[m_2, t_2, x_2, y_2]$
Symmetric	S1	[7131, 4161, 8.22, 3]	[7131, 4161, 8.22, 3]
Asymmetric	A1	[6262, 3000, 7.82, 3]	[8000, 5100, 6.33, 3]
Asymmetric	A2	[4262, 2487, 14.66, 3]	[10000, 5835, 5.44, 3]
Asymmetric	A3	[3262, 1500, 12.49, 3]	[11000, 8497, 8, 3]
Asymmetric	A4	[1262, 750, 54.24, 3]	[13000, 9507, 6.9, 3]

[10]The parallel concatenation of two regular LDGM codes results in a new LDGM code with an irregular degree distribution that outperforms the original regular codes that make it up.

selection of these values degrades performance significantly. Thus, although large values of m_2 provide more syndrome information to decode the Z operators and improve the ability of the code to correct phase flips, they come at the expense of using less information to correct bit flips (values of m_1 that are too low), which results in increased error floors and worse overall performance if the bit-flip decoder is not correctly designed.

Another aspect of the proposed asymmetric CSS scheme that is integral to its performance and which was discussed in the previous section, is the relationship between the degree of the s_B nodes of the CSS decoding subgraphs, denoted by x_i, and the number of degree-1 syndrome nodes t_i, where $i = 1, 2$. The impact of this relationship, and specifically its aforementioned trade-off nature, is significant, as it ties into the selection of m_i. For the symmetric CSS schemes of Refs. [25,26] the best performance was obtained for codes that utilized $M_1 = M_2 = M(3; 1, 11.02)_{7131\times9507}$ and $t = 5000$. Let us assume that the optimal degree of x for matrix M of a symmetric CSS code will still be optimal for the M_1 and M_2 matrices used to build each of the subgraphs of an asymmetric CSS code. In reality, achieving a configuration where $x_i = 11.02$ for a scenario in which $M_1 \neq M_2$ will not always be possible. If we revisit Eq. (13) we can understand why this happens. Once the quantum rate of the code has been selected, aside from t_i, the only other parameter we can modify is y_i. Recall, that in Refs. [25,26], results showed that only a single value of this parameter yielded good outcomes, $y = 3$. Thus, since N, m_i, and y_i are fixed and t_i is bounded[11] by $\frac{N}{2}$, it will not always be possible to build matrices that have $x_i = 11.02$.

We illustrate this with an example: Introducing $N = 19014$, $y_2 = 3$, $m_2 = 11500$ and the maximum possible value of $t_2 = 9507$ into (13), we obtain $x_2 = 9.54$. Since t_2 cannot be increased further, a scheme with this parameter configuration will have a maximum s_B node degree of $x_2 = 9.54$. In consequence, it becomes apparent that our choice of m_1 and m_2 also affects the values of x_1 and x_2. Although this may seem overwhelming with regard to the design of M_1 and M_2, it is actually a positive outcome, since if we can show that performance of the asymmetric schemes is optimized for a specific value of x_i, the design procedure can be reduced to finding the values of m_i and t_i that produce this particular value of x_i.

The last detail worthy of mention related to this first analysis is that, aside from code A4, whose decoding errors are overwhelmingly caused by X operators, the entirety of the decoding errors of all the other asymmetric codes are caused by phase flips. Although not surprising given the nature of an asymmetric Pauli channel, it would be ideal to design an asymmetric scheme in which errors are equally distributed, as is the case over the depolarizing channel. In other words, we would like the X and Z operator decoders of our asymmetric CSS codes to fail with similar rates, instead of all the decoding errors being attributed to one of them.

[11]The parameter t_i can theoretically be as large as m_i. However, since each decoding subgraph only has $\frac{N}{2}$ d nodes, it is not logical to choose $t_i > \frac{N}{2}$ since with $t_i = \frac{N}{2}$ all the d nodes are already d_a nodes (they receive perfect syndrome information).

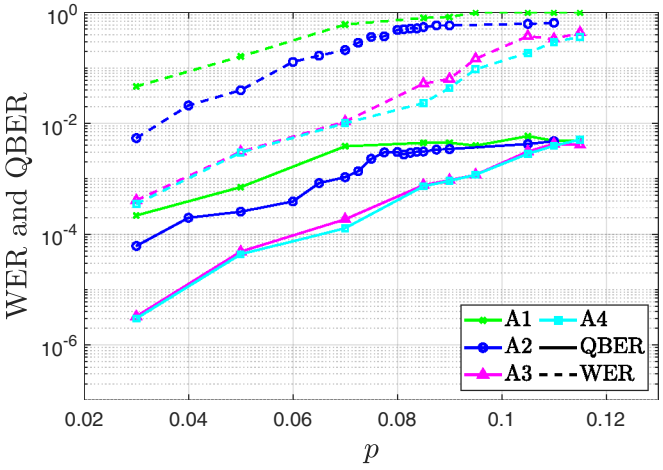

FIG. 5. Simulated QBER for different X operator decoders. p represents the gross flip probability of the Pauli channel with asymmetry coefficient $\alpha = 10^2$.

1. Impact of CSS decoding

In most cases, CSS codes are decoded separately by executing the SPA over each of the subgraphs of the overall CSS factor graph. Over the depolarizing channel, given the equal likelihood of X and Z error events, the error contributions of each individual decoder to the overall code are essentially identical. Over the asymmetric channel, however, maintenance of this separate decoding policy results in each of the CSS decoders impinging on the error correcting capabilities of the overall CSS code in a different manner.

The decoder for the X operators can cause an increment in the error floor of the code if the values of m_1 and $[y_1, x_1, t_1]$ are not chosen correctly. For sufficiently large values of $m_1 < \frac{N}{2}$, due to the low likelihood of X errors, the bit-flip decoder performs well regardless of the values of $[y_1, x_1, t_1]$, but when m_1 becomes too small, performance of the decoder is only acceptable if $[y_1, x_1, t_1]$ are chosen appropriately. This can be seen in Fig. 5, where the QBER/WER curves of X operator decoders with $m_1 = 1262$ and different configurations of $[y_1, x_1, t_1]$ are shown. The complete characteristics of these X decoders are detailed in Table II.

As is shown in Fig. 5, performance of the X decoders varies significantly depending on the values of $[y_1, x_1, t_1]$. Decoders A1 and A2 are substantially worse than decoders A3 and A4.

TABLE II. Parameter values and configurations of X operator decoders of asymmetric CSS codes simulated over a Pauli channel with $\alpha = 10^2$. The results of these simulations are shown in Fig. 5.

Decoder	m_1	t_1	x_1	y_1
A1	1262	900	76.3	3
A2	1262	750	54.2	3
A3	1262	300	29.3	3
A4	1262	100	24.4	3

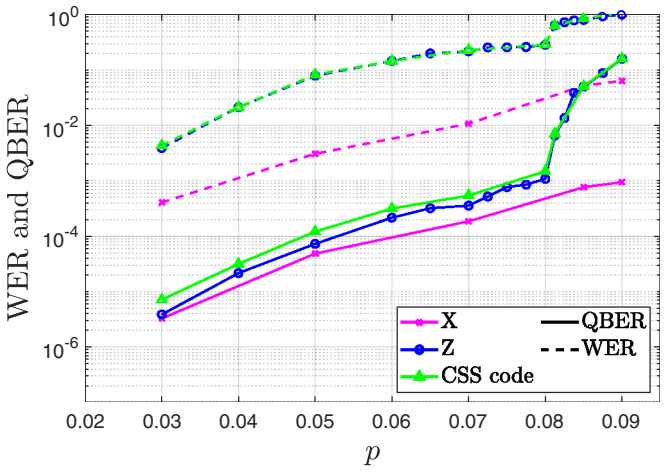

FIG. 6. Simulated QBER for the constituent decoders of a CSS code. p represents the gross flip probability of the Pauli channel with asymmetry coefficient $\alpha = 10^2$.

They mainly differ in the values of the parameters x_1 and t_1, which are much larger for decoders A1 and A2 than for decoders A3 and A4. This implies that the performance of X decoders with smaller values of m_1 will be better when lower values of x_1 are chosen by using less degree-1 syndrome nodes in the decoding graph, i.e., selecting smaller values for t_1. This is logical, since as was mentioned in the previous section, despite the fact that selecting larger values of t_1 will increase the amount of exact information transmitted from the upper layer nodes during initial iterations, it will also make the degree of the s_B nodes so large that the impact of these "perfect" messages might be mitigated and message passing may not operate successfully. For a decoder with a small value of m_1 ($m_1 = 1262$), if $[y_1, x_1, t_1]$ are chosen correctly, performance of the X decoder is excellent, with its QBER/WER curves increasing in a quasi-linear fashion as functions of p (codes A3 and A4 of Fig. 5). The performance curves of such aptly built X decoders can sometimes be orders of magnitude below those of the corresponding Z decoder. On the contrary, if the selected configuration of $[y_1, x_1, t_1]$ yields a value of x_1 that is too large, performance of the decoder will be degraded enough to cause an increase in the error floor of the overall CSS code.

The Z operator decoder faces the daunting task of correcting the much more frequent Z errors. Considering the previous discussion regarding the X operator decoders, if we assume that we have an appropriately designed X operator decoder, we hypothesize that the performance of the CSS code as a whole will be majorly determined by the quality of its phase-flip decoder. To evaluate this hypothesis we study the performance curves shown in Fig. 6, which correspond to the Z decoder of code A4 of Table I, the X decoder A3 of Table II, and the CSS code that arises when using these two decoders simultaneously.[12]

Figure 6 shows how the error contribution of the X decoder to the QBER/WER curves of the overall CSS code is almost negligible when compared to the Z decoder. This occurs because the X decoder is correctly designed (the design parameters m_1 and $[y_1, x_1, t_1]$ have been selected appropriately), contrary to the X decoder of code A4 in Fig. 4. Additionally, the results of Fig. 6 show that the performance curves of the CSS code and its Z operator decoder are very similar, especially at the decoding threshold.[13] Two factors play a role in defining when the CSS code reaches its decoding threshold. The first, which will be discussed later on in this section, is the block length of the code itself. The second, is the design of the individual CSS decoders of the code. The abrupt increase in the error rate for the Z decoder (both for the WER and the QBER) while the X decoder maintains performance at the error floor, proves that for our proposed scheme, the decoding threshold is essentially defined by the Z operator decoder. Therefore this confirms our postulation that the performance of an asymmetric CSS code, if it is aptly designed (i.e., values of m_1 and $[y_1, x_1, t_1]$), will be determined by the error correcting capabilities of its phase-flip decoder.

In short, this implies that the design of the best possible decoder for the proposed asymmetric CSS scheme can be approached through the separate optimization of its constituent Z and X decoders. This can be done by conducting simulations of different configurations of $[y_2, x_2, t_2]$ and m_2 that allow a sufficiently large value of m_1 or configuration of $[y_1, x_1, t_1]$ for which the bit-flip decoder exhibits few errors. In this manner, even if an equal distribution of X and Z errors is not obtained, we avoid the increased error-floor associated to bad X operator decoders while optimum performance (when the waterfall region is entered) is achieved. This also serves to simplify matters, since by having shown that the performance of these asymmetric CSS codes is overwhelmingly determined by the quality of the phase flip decoder (assuming the bit-flip decoder is aptly built), we can now focus only on optimizing the parameter configuration of a single decoder. With this goal in mind, in the sequel we simulate different configurations of m_2 and $[y_2, x_2, t_2]$ and determine which one results in the best performance.

B. Optimization of the Z decoder

In the previous subsection we showed that the performance of these asymmetric CSS codes is determined by the behavior of its constituent decoders. A faulty X operator decoder can degrade performance by raising the error floor of the code, while its decoding threshold can change depending on the quality of the Z operator decoder. Having previously established that if the X operator decoder is designed appropriately code performance is completely determined by the Z operator decoder, we now conduct simulations for various Z operator decoders in an attempt to discover the optimum values of m_2 and $[y_2, x_2, t_2]$.

[12]The performance curves of the CSS code in Fig. 6 have been simulated. Nonetheless, summing the WER/QBER of each constituent CSS decoder (the X and Z operator decoders) is also a valid method to obtain the performance curves of the overall code.

[13]For an error correcting code, the decoding threshold or waterfall region is the region where a sharp drop in the code error rate that stabilizes at what is known as the error floor of the code takes place.

TABLE III. Parameter values and configurations of Z operator decoders of asymmetric CSS codes simulated over a Pauli channel with $\alpha = 10^2$. The results of these simulations are shown in Fig. 7.

Decoder	m_2	t_2	x_2	y_2
Z1	12262	8198	5	3
Z2	12262	9010	6	3
Z3	12262	9507	6.9	3
Z4	11000	8497	8	3
Z5	11000	8810	9	3
Z6	11000	9060	10	3

TABLE IV. Parameter values and configurations of Z operator decoders of asymmetric CSS codes simulated over a Pauli channel with $\alpha = 10^2$. The results of these simulations are shown in Fig. 8.

Decoder	m_2	t_2	x_2	y_2
Z7	10410	9507	21	3
Z8	10600	8972	12	3
Z9	11232	9507	11	3
Z10	12262	9507	6.9	3
Z11	12676	9507	6	3
Z12	13262	9507	5	3

Given the flexibility the design of these Z error decoders allows, it is important to provide structure to the simulation process. Earlier in this paper we mentioned that if x_2 could be shown to be a good indicator for the performance of the overall scheme, the design process could be reduced to simply finding the parameter configuration that would yield the optimum value of x_2. The value of x_2 is representative of its relationship with the parameter t_2, given that growth or reduction in one of these parameters will have the same effect on the other. In fact, when all the other parameters are fixed, the only way we have to modify the value of x_2 is by changing t_2, hence, t_2 will play a critical role in this process. At the same time, the parameter m_2 is intricately related to the value of x_2, which means that the relationship between these parameters will also play a part in the performance of the Z decoder. In an attempt to verify how good a performance indicator x_2 is and the nature of the relationship between this parameter and m_2, we test the Z decoder schemes detailed in Table III. These decoders differ in unit increments of x_2 while the value of m_2 is maintained equal as long as it is permitted by the design process. Recall that for a specific value of m_2, given that $t_2 \leqslant \frac{N}{2}$, there is a maximum value of x_2 that can be obtained. The performance curves of these decoders are portrayed in Fig. 7.

The results shown in Fig. 7 shed light on the relationship between m_2 and x_2, as well as how these parameters are linked to the performance of the decoder. For starters, consider decoders Z1, Z2, and Z3. All three of them have the same value of m_2, but by selecting larger values of t_2, each scheme attains a higher value of x_2, with $x_2 = 6.9$–7 representing the largest possible[14] value of the parameter for $m_2 = 12262$. The performance curves of these decoders show how the waterfall region is entered for subsequently higher values of p (the decoding threshold improves) as x_2 grows. For instance, code Z1 which has $x_2 = 5$, enters the waterfall region at roughly $p = 0.081$. In contrast, code Z3 which has $x_2 \approx 7$, enters the waterfall region at approximately $p = 0.085$. The same trend of performance improvement as x_2 becomes larger can be observed by looking at decoders Z4, Z5 and Z6. All three decoders have the same value of m_2 with the performance curves of decoder Z6, which has the largest value of x_2, being slightly better than those of its counterparts. In consequence, this outcome proves that for a given value of m_2 the decoder that will attain the best performance will be the one for which the value of x_2 is maximized. Notice that maximizing x_2 also means maximizing t_2, which, in this particular instance means that the negative effects associated to having larger degree s_B nodes (x_2 is maximized) are outweighed by the positive impact of having the largest possible amount of s_A nodes (t_2 is maximized). This is the exact opposite of what happened in the previous subsection when studying the bit flip decoder, where maximizing the parameter x yielded worse results.

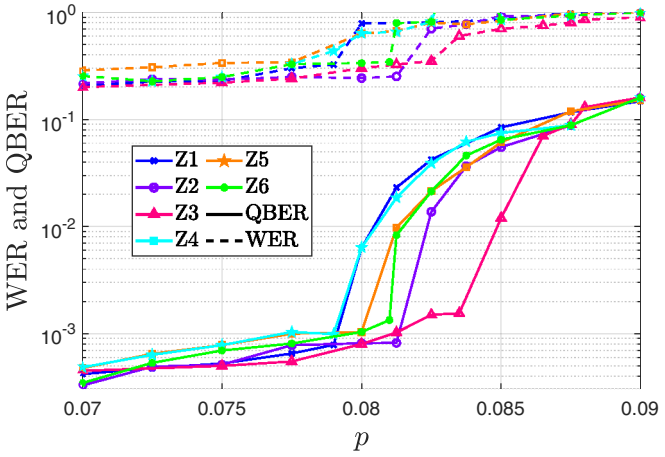

FIG. 7. Simulated QBER for different CSS QLDGM schemes. p represents the gross flip probability of the Pauli channel with asymmetry coefficient $\alpha = 10^2$.

Let us now compare the decoders of Table III in terms of their value of m_2. The results of Fig. 7 show that the larger the value of m_2 the better the performance of the decoder. In fact, even though decoder Z6 has the highest value of x_2, it is outperformed by both decoders Z2 and Z3. These results present a new conundrum to which we must now give answer: Which decoders will perform better, those with larger values of m_2 and lower values of x_2 or those with larger values of x_2 and smaller values of m_2? This notion is analogously formulated as: how is the trade-off relationship between x_2 and t_2 affected by the value of m_2?. To analyze these questions, we simulate the Z decoders shown in Table IV.

Figure 8 portrays the simulation results for the Z decoders of Table IV. Performance is similar for all the simulated decoders, with decoder Z10 having the best decoding threshold. In terms of the relationship between x_2 and m_2, the curves

[14]The parameter t_2 must fulfill $t_2 \leqslant \frac{N}{2}$.

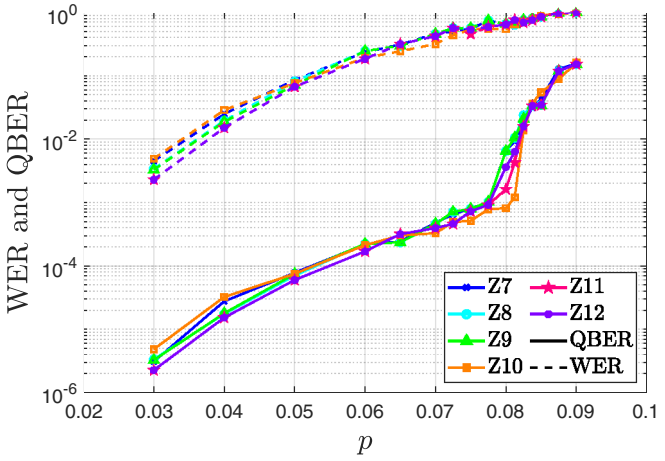

FIG. 8. Simulated QBER and WER for the decoders of Table IV. p represents the gross flip probability of the Pauli channel with asymmetry coefficient $\alpha = 10^2$.

shown in Fig. 8 show that, up to a certain point, better performance is obtained by maximizing m_2 over x_2. In other words, increasing the number of syndrome nodes used to decode Z errors is more important, within certain limits, than trying to obtain the largest value of x_2. This is reflected by decoder Z10, which has $m_2 = 12262$, outperforming decoders Z11 and Z12, which have $m_2 = 12676$ and $m_2 = 13262$. Therefore the best performance of the proposed scheme over the asymmetric channel with $\alpha = 10^2$ is obtained by setting $m_2 = 12262$ and maximizing t_2 ($t_2 = 9507$ in this case) so that the largest possible value of x_2 is obtained for the selected m_2 value. This outcome tells us that increasing m_2 beyond a certain value has a negative impact on performance, despite the maximization of t_2 and m_2. A plausible cause for this is that when $m_2 > 12262$, there is an increased number of degree x_2 s_B nodes and a reduced percentage of degree-1 s_A nodes (since t_2 is bounded). A smaller percentage of s_A nodes means that a lower amount of perfect information will be propagated during initial decoding interations, which when also considering the increased number of s_B nodes, explains the degradation in the performance of the message passing decoding algorithm for the $m_2 > 12262$ schemes. Nonetheless, the performance of all the decoders shown in Fig. 8 differs by such a small margin that we can confidently state the following. The best or near-best configuration of the proposed asymmetric CSS schemes is obtained by selecting $m_2 = \beta m$, where $\beta \in (0, 1)$, that allows a sufficiently large value of $m_1 = (1 - \beta)m$ for the X operator decoder to function well. In terms of doping, t_2 should be maximized as $t_2 = \frac{N}{2}$, and $t_1 = 0.3m_2$.

For $\alpha = 10^2$, from the decoders of Table IV we can ascertain that setting $0.73 \leqslant \beta \leqslant 0.9$ results in good performance, provided that x_2 is maximized by setting $t_2 = \frac{N}{2}$ once m_2 is chosen. We expect the value of the parameter β to vary with the degree of asymmetry of the channel, becoming larger as α grows. In terms of the value of t_1, the results of Fig. 5 show that setting 30% of the m_1 nodes to be degree-1 syndrome nodes produces the best results. We test the validity of these statements for channels with larger degrees of asymmetry in the final subsection of this chapter. Prior to doing so, we show how the error floor of the CSS codes can be reduced by increasing the degrees of the underlying classical LDGM code, as well as showing how the decoding threshold of our schemes can be improved by selecting larger values for the block length.

1. Error floor reduction

As was mentioned in the introduction to this work, the proposed CSS scheme is based on an underlying classical irregular LDGM code constructed through the parallel concatenation of two regular LDGM codes. The motivation behind such a construction is that the error floor of a single regular LDGM code can be substantially reduced when it is parallel-concatenated with another regular LDGM code of much higher degree. This results in an irregular configuration in which using a second LDGM code with larger degrees serves to lower the error floor of the whole scheme. However, as has been observed in Ref. [31], the use of larger degrees in the second code of the concatenation may result in a worse decoding threshold. To study these phenomena we conduct simulations in which the optimum configuration of the CSS decoders devised for the Pauli channel with $\alpha = 10^2$ is employed: $[m_1, t_1, x_1, y_1] = [2000, 700, 11.03, 3]$ and $[m_2, t_2, x_2, y_2] = [12262, 9507, 6.9, 3]$. As is done in Ref. [31], we utilize the irregular LDGM codes described by the concatenations $P[(8, 8); (3, 60)]$, $P[(8, 8); (5, 100)]$, and $P[(8, 8); (8, 160)]$. The simulation results are shown in Fig. 9, where the best symmetric scheme of [25,26] is included for comparison purposes.

It is easy to see from Fig. 9(a) how increasing the degrees of the underlying irregular LDGM code lowers the overall error floor of the CSS code. Moreover, these results also show that as the degrees of the second regular LDGM code used in the parallel concatenation become larger, in accordance with what has been shown throughout the literature, the decoding threshold of the scheme (more visible in terms of the QBER) begins to deteriorate. Despite the slight worsening of the decoding threshold, the error floor yielded by the CSS code that uses $P[(8, 8); (8, 160)]$ (the irregular LDGM code with the largest degrees) is orders of magnitude better than for the other simulated concatenations, both in terms of the WER and the QBER. Hence, as occurs for quantum LDGM-based CSS codes designed for the depolarizing channel, using irregular LDGM codes of larger degrees is also a valuable technique to improve the performance of codes designed for the general Pauli channel.

The results included in Fig. 9(b) serve to showcase the improvements provided by designing CSS codes specifically for the Pauli channel with $\alpha = 10^2$. In this subfigure, the performance curves of the best symmetric CSS QLDGM scheme in the literature are compared to those of our best asymmetric CSS QLDGM code. Both constructions are based on the parallel concatenation described by $P[(8, 8); (8, 160)]$ and both have block length $N = 19014$. Consider the decoding threshold of the symmetric CSS scheme: the code enters the waterfall region at approximately $p_{\text{sym}} = 0.0525$. The asymmetric scheme enters the waterfall region at $p_{\text{asym}} = 0.08$, which when compared to the symmetric code, is equivalent to an improvement of approximately 41%.

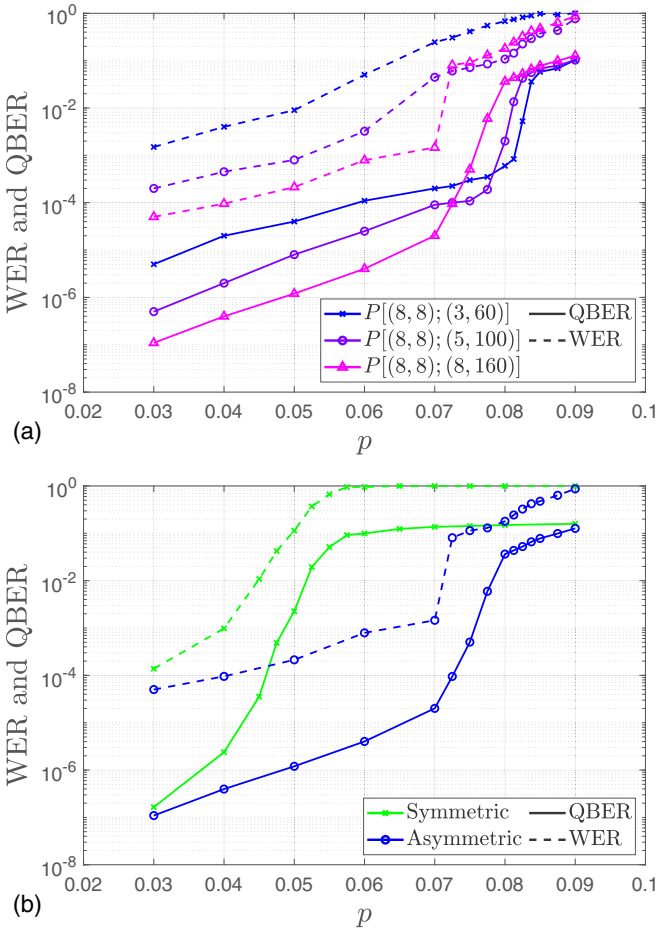

FIG. 9. (a) Simulated QBER and WER for asymmetric CSS schemes with $[m_1, t_1, x_1, y_1] = [12262, 9507, 6.9, 3]$, $[m_2, t_2, x_2, y_2] = [2000, 700, 11.03, 3]$. Different degrees of the underlying irregular LDGM code have been tested. p represents the gross flip probability of the Pauli channel with asymmetry coefficient $\alpha = 10^2$.(b) Simulated QBER and WER for an asymmetric CSS scheme with $[m_1, t_1, x_1, y_1] = [12262, 9507, 6.9, 3]$, $[m_2, t_2, x_2, y_2] = [2000, 700, 11.03, 3]$, and a symmetric CSS scheme of the same blocklength ($N = 19014$). The degree of the underlying LDGM code is the same for both schemes $P[(8, 8); (8, 160)]$. p represents the gross flip probability of the Pauli channel with asymmetry coefficient $\alpha = 10^2$.

2. Decoding threshold improvements

LDGM codes have significant performance losses due to the use of finite block lengths [53], which is why increasing the value of N may result in slight performance improvements. We close out this subsection by showing how an augmentation of the block length of the scheme results in an improvement of its decoding threshold. For this purpose we compare the code with $[m_1, t_1, x_1, y_1] = [2000, 700, 11.03, 3]$ and $[m_2, t_2, x_2, y_2] = [12262, 9507, 6.9, 3]$ that uses the concatenation $P[(8, 8); (5, 100)]$ to its equivalent when the block length is doubled ($N = 2\times 19014 = 38028$), i.e $[m_1, t_1, x_1, y_1] = [4000, 1400, 11.03, 3]$ and $[m_2, t_2, x_2, y_2] = [24524, 19014, 6.9, 3]$. Once again, the considered channel is the Pauli channel with $\alpha = 10^2$. The results are shown in Fig. 10, where the betterment of the decoding threshold associated to a larger block size can be clearly observed.

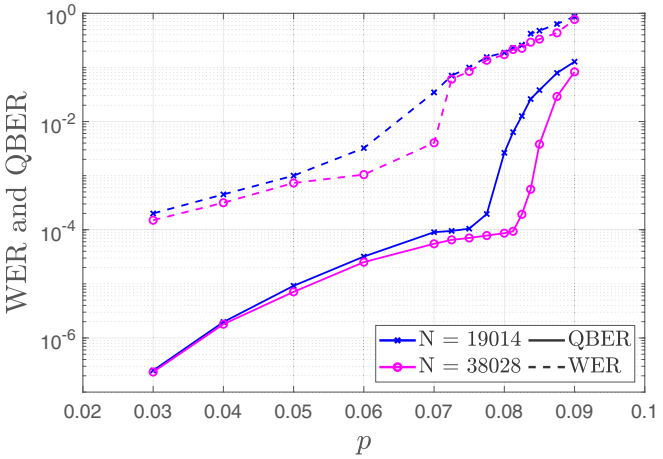

FIG. 10. Simulated QBER and WER for two asymmetric CSS codes with block lengths $N = 19014$ and 39028. p represents the gross flip probability of the Pauli channel with asymmetry coefficient $\alpha = 10^2$.

C. Simulations and adaptation to other asymmetric parameter values

A matter that has yet to be discussed is the behavior of our scheme over Pauli channels with different degrees of asymmetry. in Refs. [12,34,40], the values of the asymmetry coefficient $\alpha = [1, 10^2, 10^4, 10^6]$ are said to provide a realistic representation of practical quantum devices. Earlier in this work, we predicted that for larger degrees of asymmetry our proposed schemes would benefit from allowing more syndrome information to be used to decode Z errors (increasing the value of m_2). In a similar manner, this implies that for smaller degrees of asymmetry, the asymmetric CSS codes should provide more syndrome information to the X decoder. Essentially, asymmetric CSS QLDGM schemes should have larger Z operator decoding subgraphs and smaller X operator decoding subgraphs as the parameter α grows. To verify this hypothesis, we simulate different asymmetric CSS codes for the asymmetry coefficients $\alpha = [10, 10^4, 10^6]$. The particular configurations of the considered CSS codes are shown in Table V, while the performance of these codes over the Pauli channels with asymmetry coefficients $\alpha = [10, 10^4, 10^6]$ is shown in Fig. 11.

These results prove that our hypothesis is correct. For $\alpha =$ 10, code C1 outperforms C2 and C3, but as alpha grows, the

TABLE V. Parameter values and configurations of the CSS codes simulated over Pauli channels with $\alpha = [10, 10^4, 10^6]$. The results of these simulations are shown in Fig. 11.

Code No.	$[m_1, t_1, x_1, y_1]$	$[m_2, t_2, x_2, y_2]$
C1	[2000, 700, 11.03, 3]	[12262, 9507, 6.9, 3]
C2	[1262, 300, 29.3, 3]	[12676, 9507, 6, 3]
C3	[1000, 100, 31.5, 3]	[13262, 9507, 5, 3]

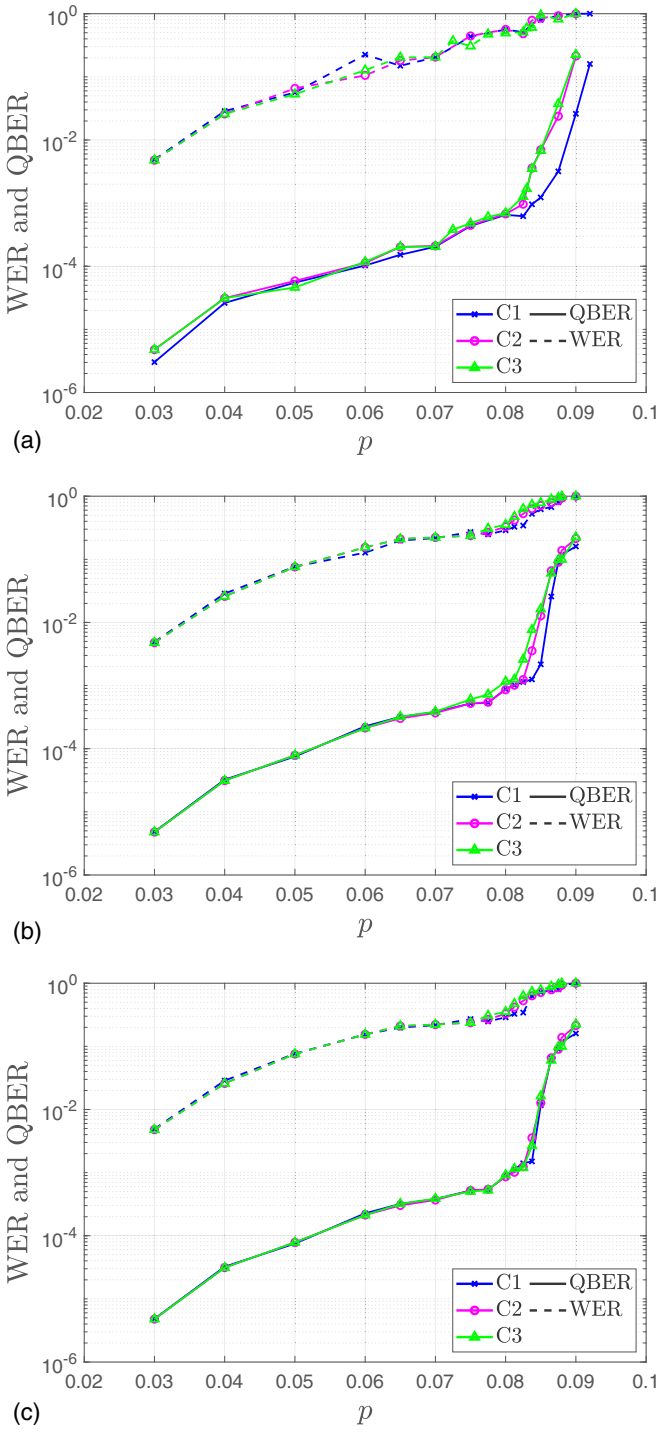

FIG. 11. Simulated QBER and WER for the asymmetric CSS schemes of Table V: (a) p represents the gross flip probability of the Pauli channel with an asymmetry coefficient $\alpha = 10$. (b) p represents the gross flip probability of the Pauli channel with asymmetry coefficient $\alpha = 10^4$. (c) p represents the gross flip probability of the Pauli channel with asymmetry coefficient $\alpha = 10^6$.

performance of C1 becomes increasingly degraded while that of C2 and C3 improves. This outcome is consistent with our initial hypothesis because for the smallest value of α that we have simulated, $\alpha = 10$, the code with the best performance is C1 which has the smallest value of m_2 (number of syndrome nodes used by the Z decoder), whereas for $\alpha = 10^6$ codes C2 and C3, which have larger values of m_2, overtake code C1. This behavior is also congruous with the fact that the hashing bound of a Pauli channel increases as this channel becomes more asymmetric [40].

However, the results of Fig. 11 are somewhat surprising in the sense that, even for the most asymmetric instance that we have simulated $\alpha = 10^6$, the performance improvement provided by the more asymmetric codes, C2 and C3, is relatively small. In fact, for $\alpha = 10^4$ codes C2 and C3, which have larger values of m_2, do not outperform the optimal scheme (code C1) that was derived for a Pauli channel with $\alpha = 10^2$. Let us discuss why this happens.

It is clear from our initial simulation results (Fig. 4) that asymmetric CSS schemes in which more syndrome information is used to decode Z operators perform better over a general Pauli channel than symmetric CSS codes. The extent to which m_2 must be increased has been thoroughly discussed throughout this section and has been shown to also be dependant on other characteristics of the asymmetric CSS construction. The entirety of this analysis has been performed considering a Pauli channel with $\alpha = 10^2$. We can now use the knowledge we have obtained for channels with $\alpha = 10^2$ to try and understand what happens over Pauli channels with $\alpha = 10^4$ and 10^6. In reality, the only difference between these channels lies in the error probabilities of the X and Z operators, which, assuming a channel gross flip probability of $p = 0.075$ go from $p_z = 0.0735$ and $p_x = 7\times10^{-4}$ when $\alpha = 10^2$, to $p_z \approx 0.075$ and $p_x \approx 7\times10^{-6}$ when $\alpha = 10^4$, to $p_z \approx 0.075$ and $p_x \approx 7\times10^{-8}$ when $\alpha = 10^6$. These changes in the values of p_z and p_x as α grows, show why in Fig. 11, the optimum code for $\alpha = 10^2$ performs as well as C2 and C3 when $\alpha = 10^4$ and $\alpha = 10^6$. The change in p_z when going from $\alpha = 10^2$ to 10^4 or 10^6 is too subtle to appreciate improvements[15] when increasing the value of m_2 from 12 262 to $m_2 = 12\,676$ and 13 262. In stark contrast, when we go from $\alpha = 1$ to 10^2, i.e., we compare our schemes to symmetric CSS codes, the improvements in performance when increasing m_2 are evident throughout the results provided in this paper. This happens because the change in p_z and p_x is substantial enough when going from $\alpha = 1$ to 10^2, that changing $m_2 = 7131$ to 12 262 results in a palpable boost in performance. This also coincides with what is shown in Ref. [40], where the capacity of a Pauli channel grows sharply when the asymmetry coefficient goes from $\alpha = 1$ to 10^2. The authors of this work also mention that only a marginal capacity improvement is exhibited when α increases further beyond 10^2. Thus the main reason for the almost negligible improvements that C2 and C3 provide with regard to C1 when $\alpha > 10^2$ lies behind the fact that the change in the nature of the asymmetric channel when increasing α beyond 10^2 is too small to allow us to appreciate improvements in performance when increasing m_2 for the selected block length. An interesting future research problem will be to study whether more drastic performance improvements can be obtained by increasing the block length

[15]It may even slightly impinge on performance due to imperfect configuration of the other parameters of the scheme.

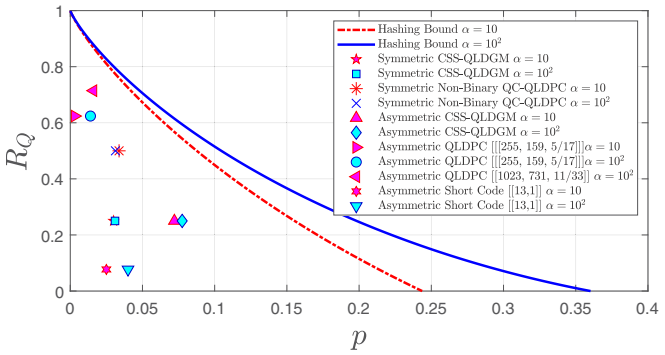

FIG. 12. Achievable coding rate at a WER of 10^{-3} for various QLDPC codes over Pauli channels with $\alpha = 10$ and 10^2.

of the code, as it may be that for a sufficiently large value of N (for the same quantum rate, increasing the block length implies an increase in the total number of syndrome nodes), for channels with values of $\alpha > 10^2$, the improvements provided by codes with larger values of m_2 when compared to the optimum scheme for the channel with $\alpha = 10^2$ will be more noticeable.

D. Distance to the Hashing bound of the Pauli channel model for asymmetry

We close this section by benchmarking the performance of our proposed schemes against the theoretical limit for the Pauli channel. As is shown in Ref. [12], the Hashing bound for a Pauli channel with asymmetry coefficient α can be computed as

$$C_Q(p,\alpha) = 1 + (1-p)\log_2(1-p) + \left(\frac{2p}{\alpha+2}\right)\log_2\left(\frac{p}{\alpha+2}\right) + \left(\frac{\alpha p}{\alpha+2}\right)\log_2\left(\frac{\alpha p}{\alpha+2}\right).$$

Based on this expression, we can assess the distance to the Hashing bound for a specific rate and asymmetry coefficient. This is reflected in Fig. 12, where the Hashing bounds for a Pauli channel with asymmetry coefficients $\alpha = 10$ and 10^2 are shown alongside the points at which the best $R_Q = \frac{1}{4}$ asymmetric schemes[16] designed in the previous sections for each of these channels can function with WER $= 10^{-3}$. As was done earlier in Sec. IV, in Fig. 12 we also show the highest possible coding rate at which the asymmetric codes that have been proposed in the literature can function with WER $= 10^{-3}$ over each of these asymmetric channels. These codes are the following.

(1) The [[255, 159, $\frac{5}{17}$]] asymmetric QLDPC code of rate $R_Q \approx 0.624$ introduced in Ref. [54].

(2) The [[1023, 731, $\frac{11}{33}$]] asymmetric QLDPC code of rate $R_Q \approx 0.714$ introduced in Ref. [47].

(3) The [[13,1]] asymmetric short code of rate $R_Q \approx 0.077$ introduced in Ref. [41].

To provide further context, we also include the coding rates that can be achieved while maintaining WER $= 10^{-3}$ by the symmetric CSS QLDGM codes of Refs. [25,26] and the nonbinary QC-QLDPC codes of [27,28] over the Pauli channels with asymmetry coefficients $\alpha = 10$ and 10^2. The results for the symmetric CSS QLDGM codes have been obtained via Monte Carlo simulations. The results for the codes of Refs. [27,28] have been derived as follows.

As is mentioned in Sec. III B, the most commonly employed noise model in the literature of CSS codes [13,25–28] approximates the action of a depolarizing channel by means of two independent BSCs with marginal bit-flip probabilities $f_m = \frac{2p}{3}$. Given the fact that each constituent code of a CSS scheme is decoded separately, the use of this model simplifies the simulation process because it only requires one of the constituent codes to be executed in most cases. This can be done because the error rate of the CSS code over the complete channel is computed as the sum of the error rates of each constituent code over each separate BSC, and considering the fact that each BSC will have the same bit-flip probability, it will be possible to compute the performance of the scheme over the overall channel by simply obtaining the error rate of one of the constituent codes and summing it to itself. We can use this framework to estimate the performance over the general Pauli channel model by adjusting the flip probabilities of the separate BSCs as $f_m^x = \frac{2p}{\alpha+2}$ and $f_m^z = \frac{p(\alpha+1)}{\alpha+2}$ [47,54]. This means that each BSC serves as an X and Z error channels, respectively. Against this backdrop, we can compute the individual flip probabilities that each constituent code of the symmetric $R_Q = \frac{1}{2}$ code of Refs. [27,28] would have to function at by substituting the value of p into the expressions that have been given for f_m^x and f_m^z. For instance, over a Pauli channel with $\alpha = 10$, for the same value of f_m at which the code performs with WER $= 10^{-3}$ over the depolarizing channel, its X error decoder will now have to operate at $f_m^x \approx 0.0077$ while the Z operator decoder has to function at $f_m^z \approx 0.0426$. These flip probabilities can then be used to obtain the WER of each constituent code of the nonbinary QC-QLDPC CSS code from the results given in Refs. [27,28]. This yields $\text{WER}_x \ll 10^{-5}$ and $\text{WER}_z \approx 10^{-1}$. Finally, we can add these error probabilities to obtain the overall WER over the complete asymmetric quantum channel,[17] which in this particular case would be $\text{WER}_{\text{non-bin-QC-QLDPC}} = \text{WER}_z + \text{WER}_x \approx 10^{-1}$. Analogously, since $\alpha \geqslant 10$, p_z will be much larger than p_x and we will be able to compute the depolarizing probability at which the code will have WER $= 10^{-3}$ over the general Pauli channel by solving for p in $f_m^{\text{dep}} \approx f_m^z = \frac{p(\alpha+1)}{\alpha+2}$, where f_m^{dep} is the flip probability at which the code performs with WER $= 10^{-3}$ over the iid X/Z channel model.

This discussion proves that a CSS code designed for a symmetric channel over which the probability distribution for X and Z errors is identical will be unable to yield the same

[16]These schemes are defined by the parameters $[m_1, t_1, x_1, y_1] = [4000, 1400, 11.03, 3]$ and $[m_2, t_2, x_2, y_2] = [24524, 19014, 6.9, 3]$, $N = 38\,028$, and $P[(8,8);(8,160)]$.

[17]Note that this procedure would be valid to approximate the performance of any symmetric CSS code over a Pauli channel with asymmetry coefficient α.

performance over an asymmetric channel. This is reflected in Fig. 12, where a coding rate $R_Q = \frac{1}{2}$, which is achievable with the codes of Refs. [27,28] over the depolarizing channel for a value of $p \approx 0.0465$, can now only be achieved for $p \approx 0.0338$ and ≈ 0.0313 when the corresponding asymmetry coefficient of the general Pauli channel is $\alpha = 10$ and 100, respectively. The same phenomenon was exhibited by the symmetric CSS QLDGM codes in Sec. V A, where performance was shown to be substantially degraded over the general Pauli channel.

As was done previously for the comparison over the depolarizing channel, we can use the distance to the Hashing bound,[18] computed as shown in (11), to analyze the quality of the strategies shown in Fig. 12. For instance, for the Pauli channel with asymmetry coefficient $\alpha = 10$, the proposed asymmetric CSS QLDGM scheme exhibits a distance to the Hashing bound of $\delta^{\alpha}_{\text{asym-CSS}} = 3.1$ dB. In the case of the nonbinary QC-QLDPC codes of [27,28] applied to this same channel, the distance to the Hashing bound is $\delta^{\alpha}_{\text{non-bin-QC-QLDPC}} = 4.2$ dB. This distance is $\delta^{\alpha}_{\text{sym-CSS}} = 6.66$ dB for the symmetric CSS QLDGM codes. Clearly, these outcomes showcase the improvements provided by building CSS designs specifically for the Pauli channel model for asymmetry. However, it must be noted that existing asymmetric quantum codes have a short block length (which most likely has a negative impact on their performance) and that, in light of their excellent performance over the depolarizing channel, asymmetric adaptations of other types of QLDPC codes such as those of Refs. [27,28,30] may exhibit better performance over asymmetric channels than the asymmetric CSS QLDGM codes derived in this paper. Nonetheless, given the increased complexity of these error correction strategies, optimizing them for asymmetric quantum channels will be more complex than the schemes proposed in this work.

All in all, the discussion provided in this section along with the results that are portrayed in Fig. 12 prove that the best symmetric CSS codes of Refs. [27,28] and asymmetric codes that can be found in the literature are outperformed over asymmetric Pauli channels by the codes proposed in this paper.

[18]We denote the distance to the Hashing bound of a Pauli channel with asymmetry coefficient α by δ^{α}. The superscript α is introduced to distinguish this distance measure from δ, the measure used for the depolarizing channel.

VI. CONCLUSION

We have introduced a technique to design CSS quantum codes based on the use of the generator and parity check matrices of LDGM codes specifically for the general Pauli channel. The proposed methods are based on simple modifications to the upper layer of the decoding graph of a symmetric CSS QLDGM code designed for the depolarizing channel. For the block length used in this paper, an asymmetric CSS code has been found for practical Pauli channels with different values of the asymmetry coefficient. Additionally, we have shown how for larger block lengths, the proposed asymmetric CSS codes can be further optimized based on the asymmetry coefficient α by increasing the block length of the code and the value of m_2 according to the guidelines provided in the paper. Over Pauli channels with $\alpha = 10$ and 10^2, the schemes proposed in this paper are closer to the theoretical limit than other existing asymmetric codes and the best codes designed for the depolarizing channel.

ACKNOWLEDGMENTS

This work was supported by the Spanish Ministry of Science and Innovation through the ADELE (PID2019-104958RB-C44) project. This work has been funded in part by NSF Award No. CCF-2007689. J.E. is funded by a Basque Government predoctoral research grant (Hezkuntza, Hizkuntza Politika Eta Kultura Saila, Eusko Jaurlaritza).

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, in *Proceedings of the IEEE International Conference on Communications (ICC'93), Geneva, Switzerland* (IEEE, Piscataway, NJ, 1993), pp. 1064–1070.
[2] R. G. Gallagher, IRE Trans. Inf. Theory **8**, 21 (1962).
[3] D. J. C. MacKay, IEEE Trans. Inf. Theory **45**, 399 (1999).
[4] S.-Y. Chung, G. D. Forney, T. J. Richardson, and R. Urbanke, IEEE Commun. Lett. **5**, 58 (2001).
[5] D. Poulin, J. Tillich, and H. Ollivier, IEEE Trans. Inf. Theory **55**, 2776 (2009).
[6] M. Wilde, M. H. Hsieh, and Z. Babar, IEEE Trans. Inf. Theory **60**, 1203 (2014).
[7] Z. Babar, S. X. Ng, and L. Hanzo, IEEE Trans. Veh. Technol. **64**, 866 (2015).
[8] H. V. Nguyen, Z. Babar, D. Alanis, P. Botsinis, D. Chandra, S. X. Ng, and L. Hanzo, IEEE Access **4**, 10194 (2016).
[9] D. Chandra, Z. Babar, S. X. Ng, and L. Hanzo, IEEE Access **7**, 52712 (2019).
[10] J. E. Martinez, P. M. Crespo, and J. Garcia-Frias, Entropy **21**, 633 (2019).
[11] J. E. Martinez, P. M. Crespo, and J. Garcia-Frias, Entropy **21**, 1133 (2019).
[12] J. E. Martinez, P. Fuentes, P. M. Crespo, and J. Garcia-Frias, in *Proceedings of the IEEE International Conference on Quantum Computing and Engineering (QCE), Denver, CO* (IEEE, Piscataway, NJ, 2020), pp. 102–108.
[13] D. J. Mackay, G. Mitchison, and P. L. McFadden, IEEE Trans. Inf. Theory **50**, 2315 (2004).
[14] T. R. Oenning and J. Moon, IEEE Trans. Magn. **37**, 737 (2001).
[15] J. Garcia-Frias and W. Zhong, IEEE Commun. Lett. **7**, 266 (2003).
[16] W. Zhong, H. Lou, and J. Garcia-Frias, in *Proceedings of the International Conference on Image Processing, Barcelona, Spain* (IEEE, Piscataway, NJ, 2003), pp. 1–593.
[17] I. Granada, P. M. Crespo, and J. Garcia-Frías, EURASIP J. Wireless Commun. Network **2019**, 11 (2019).

[18] I. Granada, P. M. Crespo, and J. Garcia-Frias, Entropy **21**, 378 (2019).

[19] D. Gottesman, Phys. Rev. A **54**, 1862 (1996).

[20] Z. Babar, P. Botsinis, D. Alanis, S. X. Ng, and L. Hanzo, IEEE Access **3**, 2492 (2015).

[21] H. Lou and J. Garcia-Frias, in *Proceedings of the IEEE 6th Workshop on Signal Processing Advances in Wireless Communications, New York, NY* (IEEE, Piscataway, NJ, 2005), pp. 1043–1047.

[22] H. Lou and J. Garcia-Frias, in *Proceedings of the 4th International Symposium on Turbo Codes & Related Topics; 6th International ITG-Conference on Source and Channel Coding, Munich, Germany* (IEEE, Piscataway, NJ, 2006), pp. 1–6.

[23] A. R. Calderbank and P. W. Shor, Phys. Rev. A **54**, 1098 (1996).

[24] A. Steane, Proc. R. Soc. A **452**, 2551 (1996).

[25] J. Garcia-Frias and K. Liu, in *Proceedings of the 42nd Annual Conference on Information Sciences and Systems, Princeton, NJ* (IEEE, Piscataway, NJ, 2008), pp. 562–567.

[26] K. Liu and J. Garcia-Frias, in *Proceedings of the 48th Annual Allerton Conference on Communication, Control, and Computing (Allerton), Monticello, IL* (IEEE, Piscataway, NJ, 2010), pp. 881–886.

[27] K. Kasai, M. Hagiwara, H. Imai, and K. Sakaniwa, in *Proceedings of the IEEE International Symposium on Information Theory, St. Petersburg, Russia* (IEEE, Piscataway, NJ, 2011), pp. 653–657.

[28] K. Kasai, M. Hagiwara, H. Imai, and K. Sakaniwa, IEEE Trans. Inf. Theory **58**, 1223 (2012).

[29] I. Andriyanova, D. Maurice, and J. Tillich, in *Proceedings of the IEEE Information Theory Workshop, Lausanne, Switzerland* (IEEE, Piscataway, NJ, 2012), pp. 327–331.

[30] D. Maurice, J. Tillich, and I. Andriyanova, in *Proceedings of the IEEE International Symposium on Information Theory, Istanbul, Turkey* (IEEE, Piscataway, NJ, 2013), pp. 907–911.

[31] P. Fuentes, J. E. Martinez, P. M. Crespo, and J. Garcia-Frias, Phys. Rev. A **102**, 012423 (2020).

[32] P. Fuentes, J. E. Martinez, P. M. Crespo, and J. Garcia-Frias, in *Proceedings of the IEEE International Conference on Quantum Computing and Engineering (QCE), Denver, CO* (IEEE, Piscataway, NJ, 2020), pp. 93–101.

[33] L. Loffe and M. Mézard, Phys. Rev. A **75**, 032345 (2007).

[34] J. E. Martinez, P. Fuentes, P. M. Crespo, and J. Garcia-Frias, IEEE Access **8**, 172623 (2020).

[35] F. Schmidt-Kaler, S. Gulde, M. Riebe, T. Deuschle, A. Kreuter, G. Lancaster, C. Becher, J. Eschner, H. Häffner, and R. Blatt, J. Phys. B: At. Mol. Opt. Phys. **36**, 623 (2003).

[36] L. M. K. Vandersypen, M. Steffen, G. Breyta, C. S. Yannoni, M. H. Sherwood, and I. L. Chuang, Nature (London) **414**, 883 (2001).

[37] A. M. Tyryshkin, J. J. L. Morton, S. C. Benjamin, A. Ardavan, G. A. D. Briggs, J. W. Ager, and S. A. Lyon, J. Phys.: Condens. Matter **18**, S783 (2006).

[38] J. R. Petta, A. C. Johnson, J. M. Taylor, E. A. Laird, A. Yacoby, M. D. Lukin, C. M. Marcus, M. P. Hanson, and A. C. Gossard, Science **309**, 2180 (2005).

[39] P. Bertet, I. Chiorescu, G. Burkard, K. Semba, C. J. P. M. Harmans, D. P. DiVincenzo, and J. E. Mooij, Phys. Rev. Lett. **95**, 257002 (2005).

[40] H. V. Nguyen *et al.*, IEEE Access **4**, C1 (2016).

[41] M. Chiani and L. Valentini, IEEE J. Sel. Areas Inf. Theory **1**, 480 (2020).

[42] T. Brun, I. Devetak, and M. Hsieh, Science **314**, 436 (2006).

[43] W. Zhong, H. Chai, and Javier Garcia-Frias, in *Proceedings of the International Symposium on Information Theory (ISIT 2005), Adelaide, SA, Australia* (IEEE, Piscataway, NJ, 2005), pp. 1753–1757.

[44] J. Pearl, *Morgan Kaufmann Series in Representation and Reasoning* (Morgan Kauffman, Burlington, MA, 1989).

[45] F. R. Kschischang, B. J. Frey, and H. A. Loeliger, IEEE Trans. Inf. Theory **47**, 498 (2001).

[46] T. Camara, H. Ollivier, and J.-P. Tillich, in *Proceedings of the IEEE International Symposium on Information Theory, Nice, France* (IEEE, Piscataway, NJ, 2007), pp. 811–815.

[47] P. K. Sarvepalli, A. Klappenecker, and M. Rötteler, Proc. R. Soc. London A **465**, 1645 (2009).

[48] Z. W. E. Evans, A. M. Stephens, J. H. Cole, and L. C. L. Hollenberg, arXiv:0709.3875.

[49] C. P. Williams, *Explorations in Quantum Computing* (Springer-Verlag, Berlin, 2000).

[50] S. ten Brink, in *Proceedings of the IEEE International Symposium on Information Theory, Washington, DC* (IEEE, Piscataway, NJ, 2001), p. 235.

[51] M. Hagiwara, K. Kasai, H. Imai, and K. Sakaniwa, in *Proceedings of the IEEE International Symposium on Information Theory, St. Petersburg, Russia* (IEEE, Piscataway, NJ, 2011), pp. 638–642.

[52] F. J. Vázquez-Araújo, M. González-López, L. Castedo and J. Garcia-Frias, IEEE Trans. Commun. **59**, 352 (2011).

[53] K. Liu and J. Garcia-Frias, in *Proceedings of the 43rd Annual Conference on Information Sciences and Systems, Baltimore, MD* (IEEE, Piscataway, NJ, 2009), pp. 87–92.

[54] P. K. Sarvepalli, A. Klappenecker, and M. Rotteler, in *Proceedings of the IEEE International Symposium on Information Theory, Toronto, ON, Canada* (IEEE, Piscataway, NJ, 2008), pp. 305–309.