Modified belief propagation decoders for quantum low-density parity-check codes

Alex Rigby,[*] J. C. Olivier, and Peter Jarvis

College of Sciences and Engineering, University of Tasmania, Hobart, Tasmania 7005, Australia

Quantum low-density parity-check codes can be decoded using a syndrome based GF(4) belief propagation decoder (where GF denotes Galois field). However, the performance of this decoder is limited both by unavoidable 4-cycles in the code's factor graph and the degenerate nature of quantum errors. For the subclass of CSS codes, the number of 4-cycles can be reduced by breaking an error into an X and Z component and decoding each with an individual GF(2) based decoder. However, this comes at the expense of ignoring potential correlations between these two error components. We present a number of modified belief propagation decoders that address these issues. We propose a GF(2) based decoder for CSS codes that reintroduces error correlations by reattempting decoding with adjusted error probabilities. We also propose the use of an augmented decoder, which has previously been suggested for classical binary low-density parity-check codes. This decoder iteratively reattempts decoding on factor graphs that have a subset of their check nodes duplicated. The augmented decoder can be based on a GF(4) decoder for any code, a GF(2) decoder for CSS code, or even a supernode decoder for a dual-containing CSS code. For CSS codes, we further propose a GF(2) based decoder that combines the augmented decoder with error probability adjustment. We demonstrate the performance of these new decoders on a range of different codes, showing that they perform favorably compared to other decoders presented in literature.

I. INTRODUCTION

In the classical setting, low-density parity-check (LDPC) codes are effective at protecting information against noise. LDPC codes are particularly useful as their sparse structure permits the use of an iterative belief propagation decoder that is of relatively low complexity [1,2]. Belief propagation is a message passing algorithm that takes place on a code's factor graph. This is a bipartite graph defined by a parity-check matrix for the code, with each row corresponding to a check node and each column to an error node. Quantum LDPC (QLDPC) codes, which are stabilizer codes with sparse generators, can be used to protect against the effects of a noisy quantum channel. The generators of an n-qubit stabilizer code can be represented as elements of $GF(4)^n$ [3,4]. This representation can be used to define a GF(4) parity-check matrix, which allows for slightly altered GF(4) belief propagation decoding of QLDPC codes [5]. The requirement that all stabilizer generators must commute results in unavoidable 4-cycles in the factor graph associated with the GF(4) parity-check matrix [6], which can be detrimental to decoding performance [7]. Belief propagation performance is also limited by the fact that it attempts to converge to the single most likely error (in a symbol-wise fashion), rather than accounting for the

degenerate nature of quantum errors [6]. For the subclass of Calderbank-Shor-Steane (CSS) codes, the number of 4-cycles can be reduced by instead representing generators as elements of $GF(2)^{2n}$ [4,8]. This allows an error to be broken into an X and Z component, which can then be decoded individually using two GF(2) belief propagation decoders [9]. However, for many channels, including the depolarizing channel, this has the effect of ignoring correlations between the two components [5].

Modified belief propagation based decoders have been proposed that aim to improve QLDPC decoding performance. Several decoders are presented in Ref. [6] that aim to alleviate so-called symmetric degeneracy errors, which occur as a result of symbol-wise decoding in the face of error degeneracy. The best performing of these is the random perturbation decoder, which attempts to break decoding symmetries by iteratively reattempting decoding with randomly modified channel error probabilities. The enhanced feedback (EFB) decoder of Ref. [10] behaves similarly in that it also iteratively reattempts decoding with modified error probabilities. However, unlike the random perturbation decoder, this modification is informed by the decoder's output. The supernode decoder of Ref. [5] is a modification to the standard GF(4) decoder for the subclass of dual-containing CSS codes. For this decoder, pairs of check nodes in the factor graph are combined to form supernodes. This both reduces decoding complexity and lowers the number of 4-cycles in the factor graph, which can lead to improved decoding performance.

The augmented decoder that we investigate has previously been proposed for classical binary LDPC codes in Ref. [11]. Like the random perturbation and EFB decoders, it also iteratively reattempts decoding. Each of these attempts employs a

---

*alex.rigby@utas.edu.au

version of the standard factor graph with a randomly selected subset of check nodes duplicated. In the classical case, this simple approach gives performance that compares favorably with other, typically more complicated, decoders presented in literature. In this paper we show that augmented decoders can be applied to QLDPC codes whether the underlying decoder is GF(2), GF(4), or supernode based. For CSS codes we propose the GF(2) based adjusted decoder, which attempts to reintroduce correlations between the X and Z components of an error that are lost when using a standard GF(2) decoder. If one of the two constituent GF(2) decoders fail, then the adjusted decoder reattempts decoding of this component using error probabilities that are modified according to the output of the other constituent decoder (this is a slight generalization of the decoder presented in Ref. [12]). We also present a GF(2) based decoder for CSS codes that combines the augmented and adjusted decoders. We simulate the performance of our decoders on six different codes: two dual-containing CSS codes, two nondual-containing CSS codes, and two non-CSS codes. We show that for dual-containing CSS codes our augmented GF(4), augmented supernode, and combined decoders all outperform random perturbation and EFB decoders. For the four other codes, we demonstrate that augmented GF(4) and supernode decoders perform similarly to the random perturbation and EFB decoders.

The paper is organized as follows. Section II gives an overview of belief propagation decoding for classical LDPC codes and extends this to the quantum case. Section III details the operation of existing modified decoders (random perturbation, EFB, and supernode) and describes the adjusted, augmented, and combined decoders that we propose. Section IV presents simulation results for our decoders on six different codes, comparing them to existing decoders. The paper is concluded in Sec. V.

II. BACKGROUND

A. Classical codes

A classical channel is the map $\Phi : \mathcal{A}_x \rightarrow \mathcal{A}_y$, where $\mathcal{A}_x$ is the set of possible inputs and $\mathcal{A}_y$ is the set of possible outputs. We are concerned with channels where the input and output sets are finite fields with q elements; that is, $\mathcal{A}_x = \mathcal{A}_y = \text{GF}(q)$. In this case the action of the channel can be expressed as

$$\Phi(x) = x + e = y, \tag{1}$$

where $x \in \text{GF}(q)$ is the channel input, $y \in \text{GF}(q)$ is the channel output, and $e \in \text{GF}(q)$ is an error (or noise) symbol that occurs with probability $P(e)$. A channel Φ is called symmetric if $P(0) = 1 - p$ and $P(e_i) = p/(q-1)$ for $e_i \neq 0$. A code $\mathcal{C} \subseteq \text{GF}(q)^n$ can be used to protect against the noise introduced by the channel. Elements $\boldsymbol{x} \in \mathcal{C}$, called codewords, are transmitted as n sequential uses of Φ or, equivalently, as a single use of the combined channel Φ^n, which is comprised of n copies of Φ. The action of Φ^n on some input $\boldsymbol{x} \in \mathcal{C}$ is

$$\Phi^n(\boldsymbol{x}) = \boldsymbol{x} + \boldsymbol{e} = \boldsymbol{y}, \tag{2}$$

where $\boldsymbol{y} \in \text{GF}(q)^n$ is the channel output and $\boldsymbol{e} \in \text{GF}(q)^n$ is an error "vector." Assuming the error components are

independent, the probability of an error $\boldsymbol{e} = (e_1, \ldots, e_n)$ occurring is

$$P(\boldsymbol{e}) = \prod_{i=1}^{n} P(e_i), \tag{3}$$

where $P(e_i)$ is the probability of the error symbol e_i occurring on Φ. The weight of a codeword $\boldsymbol{x} \in \mathcal{C}$ or an error $\boldsymbol{e} \in \text{GF}(q)^n$ is the number of nonzero components it contains. It follows from Eq. (3) that if Φ is symmetric, then the probability of $\boldsymbol{e} \in \text{GF}(q)^n$ occurring depends only on its weight. The distance between two codewords $\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{C}$, denoted $\Delta(\boldsymbol{x}_i, \boldsymbol{x}_j)$, is the number of components in which they differ. The distance of $\mathcal{C}$ is

$$d = \min_{\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{C}} \Delta(\boldsymbol{x}_i, \boldsymbol{x}_j). \tag{4}$$

Equivalently, the distance of $\mathcal{C}$ is equal to the weight of the lowest weight error that maps one codeword to another.

If a code $\mathcal{C} \subseteq \text{GF}(q)^n$ forms an (additive) group, then it is called additive; if it forms a vector space, then it is called linear (note that there is no distinction between additive and linear codes in the binary case). Suppose a linear code $\mathcal{C}$ has a basis $\mathcal{B} = \{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_k\}$. This defines a generator matrix

$$G^T = (\boldsymbol{b}_1 \quad \cdots \quad \boldsymbol{b}_k), \tag{5}$$

where the basis elements are considered as column vectors. A generator matrix can be defined in the same way for an additive code; however, in this case $\mathcal{B}$ is a generating set. For a linear code, the generator matrix defines a bijective encoding operation that maps some $\boldsymbol{d} \in \text{GF}(q)^k$ to a codeword $\boldsymbol{x} = G^T \boldsymbol{d} \in \mathcal{C}$ ($\boldsymbol{d}$ is also considered as a column vector). A linear code can also be defined as the kernel of a GF(q) parity-check matrix H; that is,

$$\mathcal{C} = \{\boldsymbol{x} \in GF(q)^n : H\boldsymbol{x} = \boldsymbol{0}\}. \tag{6}$$

Note that for a given code, neither the generator or parity-check matrix is unique. If H has m rows, then $\dim(\mathcal{C}) = k \geqslant n - m$, with equality when H is full rank. If $\mathcal{C}$ is linear with dimension k and distance d, then it is called an $[n, k]_q$ or $[n, k, d]_q$ code (the q is typically omitted for binary codes, where $q = 2$). For a linear code, this distance is equal to weight of the minimum weight nonzero codeword (as the errors that map one codeword to another are the nontrivial $\boldsymbol{e} \in \mathcal{C}$). The rate of a code is given by $R = k/n$.

The dual code of some code $\mathcal{C} \subseteq \text{GF}(q)^n$ with respect to the inner product $\langle \cdot, \cdot \rangle : \text{GF}(q)^n \times \text{GF}(q)^n \rightarrow \text{GF}(q)$ is

$$\mathcal{C}^{\perp} = \{\boldsymbol{c} \in \text{GF}(q)^n : \langle \boldsymbol{c}, \boldsymbol{x} \rangle = 0 \; \forall \; \boldsymbol{x} \in \mathcal{C}\}. \tag{7}$$

$\mathcal{C}^{\perp}$ is the annihilator of $\mathcal{C}$ and is therefore a linear code. If $\mathcal{C}^{\perp} \subseteq \mathcal{C}$, then $\mathcal{C}$ is called dual containing; if $\mathcal{C} \subseteq \mathcal{C}^{\perp}$, then $\mathcal{C}$ is called self-orthogonal; and if $\mathcal{C}^{\perp} = \mathcal{C}$, then $\mathcal{C}$ is called self-dual. Unless otherwise specified, the dual code is with respect to the Euclidean inner product

$$\langle \boldsymbol{c}, \boldsymbol{x} \rangle = \boldsymbol{c} \cdot \boldsymbol{x} = \sum_{i=1}^{n} c_i x_i. \tag{8}$$

In this case, if $\mathcal{C}$ is linear with generator matrix G, then a necessary and sufficient condition for $\boldsymbol{c} \in \mathcal{C}^{\perp}$ is $G\boldsymbol{c} = \boldsymbol{0}$; that is, a generator matrix for $\mathcal{C}$ is a parity-check matrix for $\mathcal{C}^{\perp}$.

Conversely, if H is a parity-check matrix for $\mathcal{C}$, then it is a generator matrix for $\mathcal{C}^{\perp}$.

The aim of a decoder is to determine the channel's input given its output. For a linear code $\mathcal{C}$, this decoder can make use of the error syndrome. If $\mathcal{C}$ has an $m \times n$ parity-check matrix H and the channel output is $\mathbf{y}$, then the syndrome is

$$z = Hy = H(x + e) = He \in \mathrm{GF}(q)^m. \tag{9}$$

An optimal decoder returns the most probable error given the syndrome measurement

$$\hat{e} = \underset{e \in \mathrm{GF}(q)^n}{\mathrm{argmax}} P(e|z) = \underset{e \in \mathrm{GF}(q)^n}{\mathrm{argmax}} P(e)\delta(He = z), \tag{10}$$

where $\delta(He = z) = 1$ if $He = z$ and 0 otherwise. The channel input can then be estimated as $\hat{x} = y - \hat{e}$. If $\hat{e} = e$ (and hence $\hat{x} = x$), then decoding is successful; otherwise, a decoding error has occurred. Unfortunately, even in the simple case of a binary code operating on the binary symmetric channel (a symmetric channel with $q = 2$), this decoding problem can be shown to be NP complete [13].

It follows from Eq. (9) that the syndrome resulting from some error $e \in \mathrm{GF}(q)^n$ depends only on which coset of $\mathrm{GF}(q)^n/\mathcal{C}$ it belongs to. If $\hat{e}$ is the most probable error in the coset $e + \mathcal{C}$, then the probability of a decoding failure given the syndrome $z = He$ is

$$P(e \neq \hat{e}|z) = \frac{P(e + \mathcal{C}) - P(\hat{e})}{P(e + \mathcal{C})}, \tag{11}$$

where $P(e + \mathcal{C})$ is the probability of any error in $e + \mathcal{C}$ occurring. Therefore, the probability of a decoding error is high if the error probability distribution over $e + \mathcal{C}$ is not sharply peaked [that is, if $P(\hat{e})$ is small]. If the channel is symmetric, then this corresponds to $e + \mathcal{C}$ containing errors with similar weight to $\hat{e}$, which will be the case if $\mathcal{C}$ contains low weight codewords. It therefore follows that the distance of $\mathcal{C}$ gives some indication of the fraction of transmissions that will not be decoded correctly, which is called call the frame error rate (FER).

B. Factor graphs and belief propagation

The factor graph of a linear code is a bipartite graph $G = (V, C, E)$. The error nodes $V = \{v_1, \ldots, v_n\}$ correspond to the n error components, and the check nodes $C = \{c_1, \ldots, c_m\}$ correspond to the m constraints imposed by the rows of a parity-check matrix H. An edge $\{c_i, v_j\} \in E$ connects check node c_i to error node v_j if $H_{ij} \neq 0$. For example, the [7,4,3] Hamming code of Ref. [14] can be defined by the parity-check matrix

$$H = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}, \tag{12}$$

which gives the factor graph shown in Fig. 1. In general a given code does not have a unique factor graph as the parity-check matrix from which it is defined is not unique. Furthermore, except in the case of a binary code, the mapping from a parity-check matrix to its corresponding factor graph is not one-to-one as an edge only indicates that $H_{ij} \neq 0$, it does not give the value of H_{ij} (although this information can be included by decorating the edges). A walk is a sequence whose
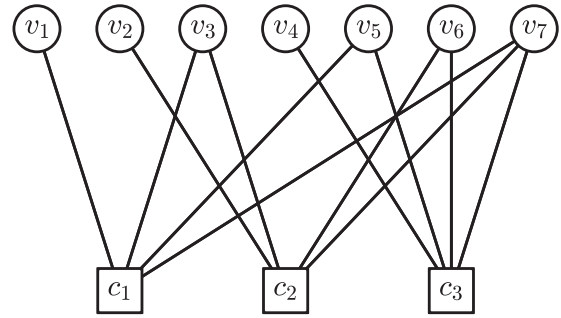

FIG. 1. The factor graph of the [7,4,3] Hamming code corresponding to the parity-check matrix given in Eq. (12). Error nodes are represented as circles and check nodes as squares.

elements alternate between connected nodes and the edges that connect them. The length of a walk is the number of edges it contains. A path is a walk containing no repeated nodes or edges with the exception that the first and last node can be the same, in which case the path is called a cycle. The bipartite nature of a code's factor graph ensures that the size of all cycles is even and greater than or equal to four. As an example, the walk $c_1, \{c_1, v_5\}, v_5, \{c_3, v_5\}, c_3, \{c_3, v_7\}, v_7, \{c_1, v_7\}, c_1$ in the graph of Fig. 1 is a 4-cycle (that is, a cycle of length four). Typically a code's factor graph will not be cycle free (that is, it will not be a tree) as if a code has such a representation, then its distance is bounded by [15]

$$d \leqslant \left\lfloor \frac{n}{k+1} \right\rfloor + \left\lfloor \frac{n+1}{k+1} \right\rfloor. \tag{13}$$

For $R \geqslant 1/2$ this reduces to $d \leqslant 2$, and for $R > 1/2$ it reduces to $d \lesssim 2\lfloor 1/R \rfloor$.

The factor graph representation of a linear code serves as the foundation for a belief propagation decoder. Instead of determining the most likely error as given in Eq. (10) a belief propagation decoder approximates it in a symbol-wise fashion. This gives an estimate $\hat{e} = (\hat{e}_1, \ldots, \hat{e}_n)$ where

$$\hat{e}_j = \underset{e_j \in \mathrm{GF}(q)}{\mathrm{argmax}} P(e_j|z). \tag{14}$$

An expression for $P(e_j|z)$ can be obtained by marginalizing $P(e|z)$. Assuming that the error components are independent

$$P(e|z) \propto \prod_{l=1}^{n} P(e_l)\delta(He = z)$$

$$= \prod_{l=1}^{n} P(e_l) \prod_{i=1}^{m} \delta\left(\sum_{j=1}^{n} H_{ij}e_j = z_i\right). \tag{15}$$

Fixing $e_j = a$ and summing over all other components gives

$$P(e_j = a|z) \propto \sum_{e:e_j = a} \prod_{l=1}^{n} P(e_l) \prod_{i=1}^{m} \delta\left(\sum_{j=1}^{n} H_{ij}e_j = z_i\right). \tag{16}$$

Belief propagation efficiently approximates these marginals by passing messages on the code's factor graph. For a code over $\mathrm{GF}(q)$, these messages will be vectors of length q. Initially, a message is sent from every error node v_j to the check nodes in the neighborhood $\mathcal{N}(v_j) = \{c_i \in C : \{c_i, v_j\} \in E\}$.

In particular, the message sent to check node $c_i \in \mathcal{N}(v_j)$ is $\mu_{j \to i}$ where the element corresponding to $a \in \mathrm{GF}(q)$ is

$$\mu_{j \to i}^a = P(e_j = a). \tag{17}$$

Note that this message simply gives the channel error probabilities. Every check node c_i then sends a message back to the error nodes in the neighborhood $\mathcal{M}(c_i) = \{v_j \in V : \{c_i, v_j\} \in E\}$. In particular, the message sent to error node $v_j \in \mathcal{M}(c_i)$ is $\lambda_{i \to j}$ with

$$\lambda_{i \to j}^a = K \sum_{\mathbf{e}:e_j=a} \delta \left(\sum_{j' \in \mathcal{M}(i)} H_{ij'} e_{j'} = z_i \right) \prod_{j' \in \mathcal{M}(i) \backslash j} \mu_{j' \to i}^{e_{j'}}, \tag{18}$$

where, through slight abuse of notation, $\mathcal{M}(i) = \{j \in \{1, \ldots, n\} : v_j \in \mathcal{M}(c_i)\}$ and K is a normalization factor chosen such that $\sum_a \lambda_{i \to j}^a = 1$. An estimate of the marginal probability $P(e_j | \mathbf{z})$ can then be made with

$$\hat{P}(e_j = a | \mathbf{z}) = KP(e_j = a) \prod_{i \in \mathcal{N}(j)} \lambda_{i \to j}^a, \tag{19}$$

where $\mathcal{N}(j) = \{i \in \{1, \ldots, m\} : c_i \in \mathcal{N}(v_j)\}$ and K is a normalization factor. From this, $\hat{\mathbf{e}}$ can be estimated in a symbol-wise fashion as in Eq. (14). If $\hat{\mathbf{z}} = H\hat{\mathbf{e}} = \mathbf{z}$, then decoding is complete; otherwise, another message is sent from each error node to its connected check nodes. The elements of this message are

$$\mu_{j \to i}^a = KP(e_j = a) \prod_{i' \in \mathcal{N}(j) \backslash i} \lambda_{i' \to j}^a, \tag{20}$$

where K is again a normalization factor. There is then another round of check to error node messages as in Eq. (18), followed by an approximation of marginals as in Eq. (19). This process of sending error to check messages followed by check to error messages and a computation of marginals proceeds iteratively until either $\hat{\mathbf{z}} = \mathbf{z}$ or a maximum number of iterations $I_{\max}$ is reached. The most computationally complex component of belief propagation is the check to error node message calculation of Eq. (18). However, it can be performed efficiently using a Fourier transform as outlined in Appendix A 1.

There are two types of decoding error exhibited by a belief propagation decoder. The first type is the detected error where decoding ends with $\hat{\mathbf{z}} \neq \mathbf{z}$ (and hence $\hat{\mathbf{e}} \neq \mathbf{e}$). Such errors do not occur when using an optimal decoder and as such are fundamentally a failing of the belief propagation decoder itself. The second type of error is the undetected error where decoding ends with $\hat{\mathbf{z}} = \mathbf{z}$ but $\hat{\mathbf{e}} \neq \mathbf{e}$. These are the same type of error exhibited by the optimal decoder and as such can be attributed to a failing of the code. It therefore follows that for a symmetric channel, using a code with a lower distance will tend to result in a higher rate of undetected errors.

Belief propagation decoding is an approximation on two levels. Firstly, it assumes that the most likely error is equal to the symbol-wise most likely error. Secondly, the estimate of the symbol-wise most likely error is based on the approximate marginal probabilities $\hat{P}(e_j | \mathbf{z})$ that are only exact when the code's factor graph is a tree [16], which as previously outlined is unlikely. However, good decoding performance can still be achieved when the factor graph is sparsely connected [16]. Linear codes with such a representation are called low-density

parity-check (LDPC) codes (most codes do not have such a representation [16,17]). Decoding performance is further improved when the factor graph contains few short cycles [7].

C. Stabilizer codes

The action of a quantum channel Φ on a quantum state described by the density operator ρ is

$$\Phi(\rho) = \sum_k A_k \rho A_k^\dagger, \tag{21}$$

where the A_k, called Kraus operators, satisfy $\sum_k A_k^\dagger A_k = I$ (the identity operator) [18]. In this paper we are interested in qubit systems; that is, systems where states $|\phi\rangle$ belong to a two-dimensional Hilbert space $\mathcal{H} \cong \mathbb{C}^2$. Furthermore, we are concerned with Pauli channels. These are channels of the form

$$\Phi(\rho) = p_I \rho + p_X X \rho X + p_Y Y \rho Y + p_Z Z \rho Z, \tag{22}$$

where in the computational $\{|0\rangle, |1\rangle\}$ basis

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \tag{23}$$

The action of this channel can be interpreted as mapping a pure state $|\phi\rangle$ to $E|\phi\rangle$ where the error E is I with probability p_I, X with probability p_X, Y with probability p_Y, or Z with probability p_Z [19]. X can be viewed as a bit flip operator as $X|0\rangle = |1\rangle$ and $X|1\rangle = |0\rangle$. Z can be viewed as a phase flip as $Z|0\rangle = |0\rangle$ and $Z|1\rangle = -|1\rangle$. $Y = iXZ$ can be viewed as a combined bit and phase flip. Of particular interest is the depolarizing channel where $p_I = 1 - p$ and $p_X = p_Y = p_Z = p/3$. We are also interested in the XZ channel for which the X and Z components of an error $E \propto X^u Z^v$, where $u, v \in \mathrm{GF}(2)$, occur independently with equal probability q. It follows from the independence of the error components that $p_X = p_Z = q(1 - q)$ and $p_Y = q^2$. These values can be expressed in terms of the total error probability $p = p_X + p_Y + p_Z$ as $q = 1 - \sqrt{1 - p}$, $p_X = p_Z = \sqrt{1 - p}(1 - \sqrt{1 - p})$, and $p_Y = (1 - \sqrt{1 - p})^2$.

The Pauli matrices are Hermitian, unitary, and anticommute with each other. Furthermore, they form a group called the Pauli group

$$\mathcal{P}_1 = \{\pm I, \pm iI, \pm X, \pm iX, \pm Y, \pm iY, \pm Z, \pm iZ\} = \langle X, Y, Z \rangle. \tag{24}$$

The n-qubit Pauli group $\mathcal{P}_n$ is defined as all n-fold tensor product combinations of elements of $\mathcal{P}_1$. For example, $\mathcal{P}_8$ contains the element $I \otimes I \otimes X \otimes I \otimes Y \otimes Z \otimes I \otimes I$, which is often written more compactly as $IIXIYZII$ or $X_3 Y_5 Z_6$. The weight of some $g \in \mathcal{P}_n$ is the number of elements in the tensor product that are not equal to the identity up to phase. The commutation relations of the Pauli matrices mean that elements of $\mathcal{P}_n$ must either commute or anticommute, with two elements anticommuting if their nonidentity components differ in an odd number of places.

Similar to the classical case, the noise introduced by a quantum channel can be protected against by employing a code. A quantum (qubit) code is a subspace $\mathcal{Q} \subseteq (\mathbb{C}^2)^{\otimes n}$. Codewords $|\phi\rangle \in \mathcal{Q}$ are transmitted across the combined n-qubit channel $\Phi^{\otimes n}$. If Φ is a Pauli channel, then $\Phi^{\otimes n}$ maps codewords $|\phi\rangle$ to $E|\phi\rangle$ where $E \in \mathcal{P}_n$. Assuming the channel

acts on each qubit independently, the probability of an error E occurring (up to phase) is

$$P(E) = \prod_{i=1}^{n} P(E_i), \tag{25}$$

where $P(E_i)$ is the probability of the error E_i occurring (up to phase) on the single qubit channel Φ. Note that errors are considered up to phase as the resulting state is equivalent up to such a phase factor. A convenient way of handling this is to group errors in $\mathcal{P}_n$ up to phase with $\tilde{E} = \{E, -E, iE, -iE\} \in \mathcal{P}_n/\{\pm I, \pm iI\} = \tilde{\mathcal{P}}_n$.

Stabilizer codes are defined by an Abelian subgroup $\mathcal{S} < \mathcal{P}_n$, called the stabilizer, that does not contain $-I$ [4]. The code $\mathcal{Q}$ is the space of states that are fixed by every element $s_i \in \mathcal{S}$; that is,

$$\mathcal{Q} = \{|\phi\rangle \in (\mathbb{C}^2)^{\otimes n} : s_i|\phi\rangle = |\phi\rangle \, \forall \, s_i \in \mathcal{S}\}. \tag{26}$$

The requirement that $-I \notin \mathcal{S}$ both means that no $s \in \mathcal{S}$ can have a phase factor of $\pm i$, and that if $s \in \mathcal{S}$, then $-s \notin \mathcal{S}$. If $\mathcal{S}$ is generated by $M = \{M_1, \ldots, M_m\} \subset \mathcal{P}_n$, then it is sufficient (and obviously necessary) for $\mathcal{Q}$ to be stabilized by every M_i. Assuming that the set of generators is minimal, it can be shown that $\dim(\mathcal{Q}) = 2^{n-m} = 2^k$ [19]; that is, $\mathcal{Q}$ encodes the state of a k-qubit system. If the generators of $\mathcal{S}$ are sparse, then $\mathcal{Q}$ is called a quantum LDPC (QLDPC) code.

Suppose an error E occurs mapping some codeword $|\phi\rangle \in \mathcal{Q}$ to $E|\phi\rangle$. A projective measurement of a generator M_i will give the result $+1$ if $[E, M_i] = EM_i - M_iE = 0$ or -1 if $\{E, M_i\} = EM_i + M_iE = 0$. These measurement values define the syndrome $z \in \mathrm{GF}(2)^m$ with

$$z_i = \begin{cases} 0 & \text{if } [E, M_i] = 0, \\ 1 & \text{if } \{E, M_i\} = 0. \end{cases} \tag{27}$$

There are three classes of error that can occur. The first class are those errors $\tilde{E} = \{E, -E, iE, -iE\} \in \tilde{\mathcal{S}}$ where $\tilde{\mathcal{S}}$ is the group

$$\tilde{\mathcal{S}} = \{\tilde{s} = \{s, -s, is, -is\} : s \in \mathcal{S}\}. \tag{28}$$

Such errors have no effect on the code and result in the trivial syndrome $z = 0$ (as the stabilizer is Abelian). The second class of errors are those $\tilde{E} \in C(\tilde{\mathcal{S}}) \backslash \tilde{\mathcal{S}}$ where $C(\tilde{\mathcal{S}})$ is the centralizer of $\tilde{\mathcal{S}}$ in $\tilde{\mathcal{P}}_n$, which in this case is actually equal to $N(\tilde{\mathcal{S}})$ (the normalizer of $\tilde{\mathcal{S}}$ in $\tilde{\mathcal{P}}_n$) [4]. These are errors that commute with every stabilizer and therefore also yield $z = 0$; however, the effect of such errors on the code is nontrivial. The final class of errors are those $\tilde{E} \in \tilde{\mathcal{P}}_n \backslash N(\tilde{\mathcal{S}})$, which yield nontrivial syndromes $z \neq 0$ and also act nontrivially on the code. In general, the syndrome resulting from some error $\tilde{E} \in \tilde{\mathcal{P}}_n$ depends only on which coset of $\tilde{\mathcal{P}}_n/N(\tilde{\mathcal{S}})$ it belongs to, while its effect on the code depends only on which coset of $\tilde{\mathcal{P}}_n/\tilde{\mathcal{S}}$ it belongs to [note that $\tilde{\mathcal{S}} \lhd N(\tilde{\mathcal{S}}) \lhd \tilde{\mathcal{P}}_n$ as $\tilde{\mathcal{P}}_n$ is Abelian]. This phenomena of distinct errors having an identical effect on a code is called degeneracy and has no classical analog. In the classical case, the distance d of a linear code is equal to the weight of the lowest weight error yielding a trivial syndrome while having a nontrivial effect on the code. This extends to the quantum case, with the distance d of a stabilizer code being the weight of the lowest weight element in $N(\tilde{\mathcal{S}}) \backslash \tilde{\mathcal{S}}$ [4]. An n-qubit code of dimension 2^k with distance d is called an

$[[n, k]]$ or $[[n, k, d]]$ code (the double brackets differentiate it from a classical code).

From a decoding point of view, the syndrome measurement determines which coset of $\tilde{\mathcal{P}}_n/N(\tilde{\mathcal{S}})$ an error $\tilde{E}$ belongs to. If this coset has the representative $\tilde{g} \in \tilde{\mathcal{P}}_n$, then an ideal decoder determines the coset $\hat{A}$ in $(\tilde{g}N(\tilde{\mathcal{S}}))/\tilde{\mathcal{S}}$ that $\tilde{E}$ is most likely to belong to. Importantly, $\hat{A}$ does not necessarily contain the individually most likely error in $\tilde{g}N(\tilde{\mathcal{S}})$. If $\hat{A}$ has the representative $\hat{\tilde{E}} = \{\hat{E}, -\hat{E}, i\hat{E}, -i\hat{E}\}$, then the decoder attempts to correct the channel error by applying $\hat{E}$ to the channel output. If $\tilde{E} \in \hat{A}$, then $\hat{\tilde{E}}\tilde{E} \in \tilde{\mathcal{S}}$ and as such this process corrects the error; otherwise, if $\tilde{E} \notin \hat{A}$, then a decoding error has occurred. Similar to the classical case, the probability of a decoding failure given some syndrome measurement z is

$$P(\tilde{E} \notin \hat{A}|z) = \frac{P(\tilde{g}N(\tilde{\mathcal{S}})) - P(\hat{A})}{P(\tilde{g}N(\tilde{\mathcal{S}}))}, \tag{29}$$

where $P(\tilde{g}N(\tilde{\mathcal{S}}))$ and $P(\hat{A})$ are the probabilities of an error being in $\tilde{g}N(\tilde{\mathcal{S}})$ or $\hat{A}$, respectively. From this it follows that the probability of a decoding error is high if the probability distribution over $(\tilde{g}N(\tilde{\mathcal{S}}))/\tilde{\mathcal{S}}$ is not sharply peaked, which will occur if $N(\tilde{\mathcal{S}}) \backslash \tilde{\mathcal{S}}$ contains high probability errors. For the depolarizing channel this corresponds to $N(\tilde{\mathcal{S}}) \backslash \tilde{\mathcal{S}}$ containing low weight elements, meaning that the distance d gives some indication of decoder performance.

D. Stabilizer code representations

It is possible to represent elements of $\tilde{\mathcal{P}}_1$ as elements of $\mathrm{GF}(2)^2$ according to the isomorphism [4,8]

$$I \leftrightarrow (0, 0), \quad X \leftrightarrow (1, 0), \quad Y \leftrightarrow (1, 1), \quad Z \leftrightarrow (0, 1). \tag{30}$$

This can be extended to elements of $\tilde{\mathcal{P}}_n$ according to

$$X^{u_1}Z^{v_1} \otimes \cdots \otimes X^{u_n}Z^{v_n} \leftrightarrow (u_1, \ldots, u_n|v_1, \ldots, v_n). \tag{31}$$

This can be written more compactly as $X^{\boldsymbol{u}}Z^{\boldsymbol{v}} \leftrightarrow (\boldsymbol{u}|\boldsymbol{v}) \in \mathrm{GF}(2)^{2n}$ where $\boldsymbol{u} = (u_1, \ldots, u_n)$, $\boldsymbol{v} = (v_1, \ldots, v_n) \in \mathrm{GF}(2)^n$. The product of elements in $\tilde{\mathcal{P}}_n$ corresponds to addition in $\mathrm{GF}(2)^{2n}$. Representatives of elements in $\tilde{\mathcal{P}}_n$ commute if the symplectic inner product of the binary representations is zero; otherwise, they anticommute. Note that the symplectic inner product of $\boldsymbol{a} = (\boldsymbol{u}|\boldsymbol{v}) \in \mathrm{GF}(2)^{2n}$ and $\boldsymbol{b} = (\boldsymbol{u}'|\boldsymbol{v}') \in \mathrm{GF}(2)^{2n}$ is

$$\boldsymbol{a} \circ \boldsymbol{b} = \boldsymbol{u} \cdot \boldsymbol{v}' + \boldsymbol{u}' \cdot \boldsymbol{v} = \sum_{i=1}^{n}(u_iv_i' + u_i'v_i). \tag{32}$$

Considering $\boldsymbol{a}$ and $\boldsymbol{b}$ as row vectors, this simplifies to $\boldsymbol{a} \circ \boldsymbol{b} = \boldsymbol{a}P\boldsymbol{b}^T$ where P is the $2n \times 2n$ matrix

$$P = \begin{pmatrix} 0 & I \\ I & 0 \end{pmatrix}. \tag{33}$$

The binary representations of the m generators of some stabilizer $\mathcal{S}$ define the rows of an $m \times 2n$ binary matrix H. This matrix has the form

$$H = (H_X|H_Z), \tag{34}$$

where H_X and H_Z are each $m \times n$ matrices. Note that while H only defines a stabilizer up to phase $\tilde{\mathcal{S}}$, the codes defined by

TABLE I. GF(4) addition.

+	0	1	ω	$\bar{\omega}$
0	0	1	ω	$\bar{\omega}$
1	1	0	$\bar{\omega}$	ω
ω	ω	$\bar{\omega}$	0	1
$\bar{\omega}$	$\bar{\omega}$	ω	1	0

different stabilizers corresponding to $\tilde{\mathcal{S}}$ will all have the same error correction properties. Considering H as the parity-check matrix of a classical binary code $\mathcal{C}$, the stabilizer elements correspond to elements of the dual code $\mathcal{C}^{\perp}$. The requirement that all stabilizer generators commute becomes

$$H_X H_Z^T + H_Z H_X^T = 0. \tag{35}$$

Any classical linear code with a parity-check matrix H that satisfies this constraint can be used to define a stabilizer code. Furthermore, if H is sparse, then this stabilizer code is a QLDPC code. Errors can also be considered within the binary framework. Suppose that some error $E \propto X^{e_X} Z^{e_Z}$ occurs. This error has the binary representation $\boldsymbol{e} = (\boldsymbol{e}_X^T | \boldsymbol{e}_Z^T)^T$, and the corresponding syndrome is simply $\boldsymbol{z} = HP\boldsymbol{e}$ (where $\boldsymbol{e}_X$, $\boldsymbol{e}_Z$, and $\boldsymbol{e}$ are column vectors for consistency with the classical case).

A subclass of stabilizer codes are the Calderbank-Shor-Steane (CSS) codes [20,21], which have a binary representation of the form

$$H = \begin{pmatrix} \tilde{H}_X & 0 \\ 0 & \tilde{H}_Z \end{pmatrix}. \tag{36}$$

The commutation condition of Eq. (35) becomes $\tilde{H}_Z \tilde{H}_X^T = 0$ (or equivalently $\tilde{H}_X \tilde{H}_Z^T = 0$). Considering $\tilde{H}_X$ and $\tilde{H}_Z$ as parity-check matrices for classical codes $\mathcal{C}_X$ and $\mathcal{C}_Z$ respectively this commutation condition requires that $\mathcal{C}_X^{\perp} \subseteq \mathcal{C}_Z$ (or equivalently $\mathcal{C}_Z^{\perp} \subseteq \mathcal{C}_X$). If $\tilde{H}_Z = \tilde{H}_X$, then $\mathcal{C}_Z = \mathcal{C}_X$, which gives $\mathcal{C}_X^{\perp} \subseteq \mathcal{C}_X$. Such codes are called dual-containing CSS codes.

Elements of $\tilde{\mathcal{P}}_1$ can also be represented as elements of $GF(4) = \{0, 1, \omega, \omega^2 = \bar{\omega}\}$ according to the isomorphism [3,4]

$$I \leftrightarrow 0, \quad X \leftrightarrow 1, \quad Y \leftrightarrow \bar{\omega}, \quad Z \leftrightarrow \omega. \tag{37}$$

Elements of $\tilde{\mathcal{P}}_n$ then map to elements of $GF(4)^n$, with the product of elements in $\tilde{\mathcal{P}}_n$ corresponding to addition in $GF(4)^n$ [GF(4) addition and multiplication are defined in Tables I and II, respectively]. Representatives of elements in $\tilde{\mathcal{P}}_n$ commute if the trace inner product of the corresponding elements

TABLE II. GF(4) multiplication.

$\times$	0	1	ω	$\bar{\omega}$
0	0	0	0	0
1	0	1	ω	$\bar{\omega}$
ω	0	ω	$\bar{\omega}$	1
$\bar{\omega}$	0	$\bar{\omega}$	1	ω

of $GF(4)^n$ is zero. Note that the trace inner product of $\boldsymbol{a}, \boldsymbol{b} \in GF(4)^n$ is

$$\boldsymbol{a} * \boldsymbol{b} = \mathrm{tr}(\boldsymbol{a} \cdot \bar{\boldsymbol{b}}) = \mathrm{tr}\left(\sum_{i=1}^{n} a_i \bar{b}_i\right), \tag{38}$$

where $\bar{0} = 0$, $\bar{1} = 1$, $\bar{\omega} = \omega^2$, and $\bar{\omega^2} = \omega$; and $\mathrm{tr}(x) = x + \bar{x}$ [that is, $\mathrm{tr}(0) = \mathrm{tr}(1) = 0$ and $\mathrm{tr}(\omega) = \mathrm{tr}(\bar{\omega}) = 1$].

The $GF(4)^n$ representations of the m generators of some stabilizer $\mathcal{S}$ define an $m \times n$ GF(4) matrix H in much the same way as the binary case. A stabilizer with the GF(2) representation of Eq. (34) has the GF(4) representation

$$H = H_X + \omega H_Z. \tag{39}$$

For a CSS code this becomes

$$H = \begin{pmatrix} \tilde{H}_X \\ \omega \tilde{H}_Z \end{pmatrix}, \tag{40}$$

with $\tilde{H}_X$ and $\tilde{H}_Z$ as defined in Eq. (36). The stabilizer corresponds to the additive group generated by the rows of H. This group can be considered as an additive classical code $\mathcal{C}$ over GF(4). The rows of H must be orthogonal with respect to the trace inner product. Therefore, if $\mathcal{C}^{\perp}$ is the dual code of $\mathcal{C}$ with respect to the trace inner product, then $\mathcal{C} \subseteq \mathcal{C}^{\perp}$. Any such self-orthogonal additive GF(4) code can be used to define a stabilizer code. Errors can also be considered in the GF(4) framework. An error E with GF(4) representation $\boldsymbol{e}$ (again, taken to be a column vector) will yield a syndrome $\boldsymbol{z} = \mathrm{tr}(H\boldsymbol{e})$. Note that while H is a generator matrix for $\mathcal{C}$, we essentially consider it as a parity-check matrix because of the role it plays in syndrome calculation and hence in belief propagation decoding.

E. Belief propagation decoding for stabilizer codes

Belief propagation decoding can be applied to stabilizer codes using the GF(2) and GF(4) representations of the previous section. Such a belief propagation decoder aims to estimate the symbol-wise most likely error (up to phase) $\hat{E} = \hat{E}_1 \otimes \cdots \otimes \hat{E}_n$ where

$$\hat{E}_j = \underset{E_j}{\mathrm{argmax}}\, P(E_j|\boldsymbol{z}). \tag{41}$$

A GF(4) based belief propagation decoder can be used for any QLDPC code. This decoder attempts to make a symbol-wise estimate $\hat{\boldsymbol{e}} \in GF(4)^n$ that maps to $\hat{E}$ according to the isomorphism outlined in Sec. II D. The GF(4) decoder behaves very similarly to the belief propagation decoder presented for classical linear codes in Sec. II B. The only change is to account for the difference in syndrome calculation. In particular, the check to error node message is modified to

$$\lambda_{i \to j}^{a} = K \sum_{\boldsymbol{e}:e_j=a} \delta\left[\mathrm{tr}\left(\sum_{j' \in \mathcal{M}(i)} H_{ij'} \bar{e}_{j'}\right) = z_i\right]$$
$$\times \prod_{j' \in \mathcal{M}(i) \backslash j} \mu_{j' \to i}^{e_{j'}}. \tag{42}$$

This calculation can also be performed efficiently using a Fourier transform as outlined in Appendix A 2. The channel error probabilities used in error to check node messages

[Eqs. (17) and (20)] and in marginal calculation [Eq. (19)] are $P(e_j = 0) = 1 - p$, $P(e_j = 1) = p_X$, $P(e_j = \bar{\omega}) = p_Y$, and $P(e_j = \omega) = p_Z$.

For the subclass of CSS codes, it is also possible to use two separate GF(2) based belief propagation decoders. For some error $E \propto X^{e_X} Z^{e_Z}$, the corresponding binary error is $e = (e_X^T | e_Z^T)^T$, which yields the syndrome

$$z = HPe = \begin{pmatrix} \tilde{H}_X e_Z \\ \tilde{H}_Z e_X \end{pmatrix} = \begin{pmatrix} z_Z \\ z_X \end{pmatrix}. \qquad (43)$$

Using z_Z and $\tilde{H}_X$ an estimate $\hat{e}_Z$ of e_Z can be made using a classical binary belief propagation decoder. The same can be done with z_X and $\tilde{H}_Z$ to make an estimate $\hat{e}_X$ of e_X. The jth component of e_X, denoted $e_X^{(j)}$, is equal to one if $E_j \propto X$ or $E_j \propto Y$. Therefore, $P(e_X^{(j)} = 1) = p_X + p_Y$ and similarly $P(e_Z^{(j)} = 1) = p_Y + p_Z$. These values are used as the channel error probabilities for the two decoders, which amounts to considering the quantum channel as two binary symmetric channels. Note that for depolarizing channel $P(e_X^{(j)} = 1) = P(e_Z^{(j)} = 1) = 2p/3$, while for the XZ channel $P(e_X^{(j)} = 1) = P(e_Z^{(j)} = 1) = 1 - \sqrt{1 - p}$.

As in the classical case, belief propagation decoding can result in both detected and undetected errors. If $\hat{z} \neq z$, where $\hat{z}$ is the syndrome associated with the error estimate $\hat{E}$, then a detected error has occurred. Again, these detected errors are a failing of the decoder. If $\hat{z} = z$ but $\hat{\tilde{E}}\tilde{E} \notin \tilde{S}$, then an undetected error has occurred, which is fundamentally a failing of the code itself. It therefore follows that for the depolarizing channel, using a code with a lower distance will tend to result in a higher rate of undetected errors.

Using belief propagation in the quantum case is an even greater approximation than in the classical case. As outlined in Sec. II C, an optimal decoder for a stabilizer code will determine the most likely coset of errors rather than the single most likely error. By definition, QLDPC codes have many low weight stabilizers, which means there will be a large number of elements of the most likely coset with similar weight and hence similar probability. This spreading of probability increases the chance that the single most likely error will not belong to the most likely coset of errors. Approximating the ideal decoder with one that determines the single most likely error will therefore lead to an increased error rate. Belief propagation goes one step further away from the optimal decoder by estimating the single most likely error in a symbol-wise fashion, which can lead to so-called symmetric degeneracy errors. Such errors are well explained by the example of Ref. [6], which is as follows. Consider a two-qubit stabilizer code with generators $M_1 = XX$ and $M_2 = ZZ$, and assume that the error $E = IX$ occurs leading to a syndrome $z = (0, 1)^T$. The coset of errors that give this syndrome is $\{XI, IX, YZ, ZY\}$ (grouping errors up to phase). As a result, the error probabilities on both qubits are $P(E_i = I|z) = P(E_i = X|z) = Kp_I p_X$ and $P(E_i = Y|z) = P(E_i = Z|z) = Kp_Y p_Z$ where $K = 1/(2p_I p_X + 2p_Y p_Z)$. This symmetry of error probabilities results in the decoder estimating the same error on each qubit. This is not a symmetry exhibited by any of the errors that yield z and as such even an ideal symbol-wise decoder will yield a detected error.

The requirement that all stabilizer generators must commute also degrades belief propagation performance as it results in 4-cycles. Consider some qubit j, there must be (at least) two stabilizer generators, say M_i and $M_{i'}$, that act nontrivially on j with different Pauli matrices. If this is not the case, then there will be a weight one element of $N(\tilde{S})\backslash\tilde{S}$, meaning that the code will have distance $d = 1$ (making it of little to no interest). As M_i and $M_{i'}$ contain different Pauli matrices in position j, they must also contain different Pauli matrices at some other position j' to ensure that they commute with each other. This results in a 4-cycle in the GF(4) factor graph as check nodes c_i and $c_{i'}$ both connect to error nodes v_j and $v_{j'}$. In the case of a CSS code, any 4-cycles resulting from an overlap between one row from $\tilde{H}_X$ and one row from $\tilde{H}_Z$ can be removed by decoding with a pair of GF(2) decoders rather than a GF(4) decoder. If it is a dual-containing CSS code, then there must still be 4-cycles in the GF(2) factor graph as the rows of $\tilde{H} = \tilde{H}_X = \tilde{H}_Z$ must overlap in an even number of positions to ensure that $\tilde{H}\tilde{H}^T = 0$. If the code is not dual containing, then it is possible for $\tilde{H}_X$ and $\tilde{H}_Z$ to have corresponding GF(2) factor graphs with no 4-cycles.

The reduction in 4-cycles, along with the reduced inherent complexity, makes GF(2) decoding attractive for CSS codes. However, treating a Pauli channel as a pair of binary symmetric channels ignores potential correlations between the X and Z components of an error $E \propto X^{e_X} Z^{e_Z}$. These correlations are described by the conditional probabilities

$$P(e_Z^{(j)} = 1|e_X^{(j)} = 1) = \frac{p_Y}{p_X + p_Y}, \qquad (44)$$

$$P(e_Z^{(j)} = 1|e_X^{(j)} = 0) = \frac{p_Z}{1 - (p_X + p_Y)}, \qquad (45)$$

$$P(e_X^{(j)} = 1|e_Z^{(j)} = 1) = \frac{p_Y}{p_Y + p_Z}, \qquad (46)$$

$$P(e_X^{(j)} = 1|e_Z^{(j)} = 0) = \frac{p_X}{1 - (p_Y + p_Z)}. \qquad (47)$$

The X and Z components are uncorrelated if they occur independently, which requires $P(e_Z^{(j)} = 1|e_X^{(j)}) = P(e_Z^{(j)} = 1) = p_Y + p_Z$ and $P(e_X^{(j)} = 1|e_Z^{(j)}) = P(e_X^{(j)} = 1) = p_X + p_Y$. This is equivalent to the requirement that $p_Y = (p_X + p_Y)(p_Y + p_Z)$, which is satisfied by the XZ channel but not by the depolarizing channel.

III. MODIFIED DECODERS

A. Existing decoders

1. Random perturbation

A number of modified decoders have been presented in Ref. [6] to address symmetric degeneracy errors. The best performing of these is the random perturbation decoder, which attempts to break decoding symmetries by randomizing the channel error probabilities. Initially, decoding is attempted using a standard GF(4) decoder. If this results in $\hat{z} = z$, then decoding is complete. Otherwise, if $\hat{z} \neq z$, then decoding is iteratively reattempted with modified error probabilities until either decoding is successful or a maximum number of

attempts N is reached. In each decoding attempt a frustrated check is selected. This is a check node c_i such that $\hat{z}_i \neq z_i$. The channel probabilities of all qubits $j \in \mathcal{M}(i)$ involved in this check are then perturbed (up to normalization) as follows:

$$P(E_j = I) \rightarrow P(E_j = I), \tag{48}$$

$$P(E_j = X) \rightarrow (1 + \delta_X)P(E_j = X), \tag{49}$$

$$P(E_j = Y) \rightarrow (1 + \delta_Y)P(E_j = Y), \tag{50}$$

$$P(E_j = Z) \rightarrow (1 + \delta_Z)P(E_j = Z). \tag{51}$$

Here δ_X, δ_Y, and δ_Z are realizations of a random variable that is uniformly distributed over $[0, \delta]$, where δ is called the perturbation strength. The increasing of nonidentity error probabilities is motivated by the empirical observation that the decoder is naturally too biased towards the trivial error [6].

2. Enhanced feedback

The enhanced feedback (EFB) decoder of Ref. [10], which is specifically tailored for the depolarizing channel, behaves somewhat similarly to the random perturbation decoder in that it also iteratively reattempts decoding with modified channel probabilities. Again, decoding is first attempted using a standard GF(4) decoder. If this results in $\hat{z} = z$, then decoding is complete. If instead $\hat{z} \neq z$, then a frustrated check c_i is selected along with an involved qubit $j \in \mathcal{M}(i)$. If $z_i = 1$ but $\hat{z}_i = 0$, then the estimated error $\hat{E}$ commutes with the stabilizer generator M_i while the error E anticommutes with M_i. To address this, the channel probabilities for E_j are adjusted such that an anticommuting error is more likely than the commuting trivial error that the decoder is naturally too biased towards. This adjustment is

$$P(E_j = \sigma) \rightarrow \begin{cases} \frac{p}{2} & \text{if } \sigma = I \text{ or } M_i^{(j)}, \\ \frac{1-p}{2} & \text{otherwise,} \end{cases} \tag{52}$$

where $M_i^{(j)}$ is the jth component of the generator M_i. Conversely, if $z_i = 0$ but $\hat{z}_i = 1$, then the adjustment is

$$P(E_j = \sigma) \rightarrow \begin{cases} \frac{1-p}{2} & \text{if } \sigma = I \text{ or } M_i^{(j)}, \\ \frac{p}{2} & \text{otherwise.} \end{cases} \tag{53}$$

Decoding is then reattempted with these adjusted probabilities. If this fails, then a different qubit $j \in \mathcal{M}(i)$ is selected and the process is repeated. If all qubits involved in check c_i have been exhausted and decoding is still unsuccessful, then a different check is selected and the process continues. Again, decoding is halted if a maximum number of attempts N is reached.

3. Supernodes

The supernode decoder of Ref. [5] is a modification of the GF(4) decoder for dual-containing CSS codes. Decoding is performed on the factor graph corresponding to $\tilde{H} = \tilde{H}_X = \tilde{H}_Z$ with checks c_i and $c_{i+m/2}$ grouped to form a single

supernode. The check node calculation is modified to

$$\lambda_{i \rightarrow j}^a = K \sum_{\mathbf{e}:e_j = a} \delta\left[\text{tr}\left(\sum_{j' \in \mathcal{M}(i)} \bar{e}_{j'}\right) = z_Z^{(i)}\right]$$
$$\times \delta\left[\text{tr}\left(\sum_{j' \in \mathcal{M}(i)} \omega\bar{e}_{j'}\right) = z_X^{(i)}\right] \prod_{j' \in \mathcal{M}(i)\setminus j} \mu_{j' \rightarrow i}^{e_{j'}}. \tag{54}$$

Here $\mathbf{z}_Z$ contains the first $m/2$ values of $\mathbf{z}$ and $\mathbf{z}_X$ contains the last $m/2$ values; $z_Z^{(i)}$ and $z_X^{(i)}$ are the ith values of $\mathbf{z}_Z$ and $\mathbf{z}_X$, respectively. Defining $\tilde{z}_i = \omega z_Z^{(i)} + z_X^{(i)} \in \text{GF}(4)$, the two constraints of Eq. (54) can be combined to give

$$\lambda_{i \rightarrow j}^a = K \sum_{\mathbf{e}:e_j = a} \delta\left(\sum_{j' \in \mathcal{M}(i)} e_{j'} = \tilde{z}_i\right) \prod_{j' \in \mathcal{M}(i)\setminus j} \mu_{j' \rightarrow i}^{e_{j'}}. \tag{55}$$

Note that this is of the same form as the classical check to error message given in Eq. (18), and it can therefore be computed using the same Fourier transform approach. The effect of combining nodes into supernodes is twofold. Firstly, it reduces decoding complexity by halving the number of check node calculations. Secondly, it can improve decoder performance as it reduces the number of 4-cycles present in the factor graph. Note that random perturbation and EFB can also be implemented using an underlying supernode decoder rather than a standard GF(4) decoder.

B. New decoders

1. Adjusted

The first decoder we propose is the adjusted decoder for CSS codes. This is a GF(2) based decoder that aims to reintroduce the correlations between X and Z errors that are lost when using a standard GF(2) decoder. Initially, decoding is attempted using a standard GF(2) decoder. If this is successful, then decoding is complete. If both $H_Z\hat{e}_X = \hat{z}_X \neq z_X$ and $H_X\hat{e}_Z = \hat{z}_Z \neq z_Z$, then the adjusted decoder also halts. However, if one of $\hat{z}_X = z_X$ or $\hat{z}_Z = z_Z$, then we reattempt decoding for the incorrect component using channel probabilities that are adjusted according to Eqs. (44) to (47). In particular, if $\hat{z}_X = z_X$ but $\hat{z}_Z \neq z_Z$, then the adjustment is

$$P\left(e_Z^{(j)} = 1\right) \rightarrow \begin{cases} \frac{p_Y}{p_X + p_Y} & \text{if } \hat{e}_X^{(j)} = 1, \\ \frac{p_Z}{1 - (p_X + p_Y)} & \text{if } \hat{e}_X^{(j)} = 0. \end{cases} \tag{56}$$

Alternatively, if $\hat{z}_Z = z_Z$ but $\hat{z}_X \neq z_X$, then the adjustment is

$$P\left(e_X^{(j)} = 1\right) \rightarrow \begin{cases} \frac{p_Y}{p_Y + p_Z} & \text{if } \hat{e}_Z^{(j)} = 1, \\ \frac{p_X}{1 - (p_Y + p_Z)} & \text{if } \hat{e}_Z^{(j)} = 0. \end{cases} \tag{57}$$

We note that the adjusted decoder presented here is similar to the decoder presented for the depolarizing channel in Ref. [12]. The decoder of Ref. [12] first attempts decoding of the X component using standard channel probabilities. If this is successful, then decoding is attempted for the Z components using the modified probabilities of Eq. (56).

2. Augmented

The second decoder we propose is the augmented decoder, which was first presented in Ref. [11] for classical binary codes. An augmented decoder for QLDPC codes can be based on a GF(4) decoder for any code, a GF(2) decoder for a CSS code, or a supernode decoder for a dual-containing CSS code. The simplest of these cases is when the underlying decoder is a GF(4) decoder. In this case, decoding is initially attempted using a standard GF(4) decoder with a standard GF(4) parity-check matrix H. If this is unsuccessful, then decoding is reattempted using a randomly generated augmented parity-check matrix

$$H_A = \begin{pmatrix} H \\ H_\delta \end{pmatrix}. \tag{58}$$

H_δ is comprised of a subset of rows selected at random from H. The fraction of rows selected is dictated by the augmentation density δ. The syndrome used for decoding is

$$z_A = \begin{pmatrix} z \\ z_\delta \end{pmatrix}, \tag{59}$$

where z is the measured syndrome and z_δ contains the syndrome values corresponding to the rows selected to form H_δ. Decoding is iteratively reattempted using different augmented matrices until either decoding is successful or a maximum number of attempts N is reached. Note that duplicating rows results in a duplication of the corresponding check nodes in the factor graph.

The behavior of a supernode based augmented decoder is very similar. In this case the augmented parity-check matrices are of the form

$$H_A = \begin{pmatrix} \tilde{H} \\ \tilde{H}_\delta \end{pmatrix}, \tag{60}$$

where $\tilde{H}_\delta$ consists of the rows selected from $\tilde{H} = \tilde{H}_X = \tilde{H}_Z$. The augmented syndrome is

$$z_A = \begin{pmatrix} z_Z \\ z_{Z\delta} \\ z_X \\ z_{X\delta} \end{pmatrix}, \tag{61}$$

where the values of $z_{Z\delta}$ and $z_{X\delta}$ are taken from z_Z and z_X respectively according to the rows selected for repetition.

In the GF(2) case two augmented decoders are used, one for the X component and one for the Z component. The augmented parity-check matrices used by the X decoder are of the form

$$H_A = \begin{pmatrix} \tilde{H}_Z \\ \tilde{H}_{Z\delta} \end{pmatrix}, \tag{62}$$

and the augmented syndrome is

$$z_A = \begin{pmatrix} z_X \\ z_{X\delta} \end{pmatrix}. \tag{63}$$

The syndrome and augmented parity-check matrices used by the Z decoder are of the same form.

In all three cases [GF(2), GF(4), and supernode], decoding with an augmented parity-check matrix H_A is equivalent to running a slightly altered belief propagation algorithm using the standard parity-check matrix H. We define the function r such that

$$r(i) = \begin{cases} 1 & \text{if } c_i \text{ duplicated in } H_A, \\ 0 & \text{otherwise.} \end{cases} \tag{64}$$

Decoding with H_A is then equivalent to decoding using H with the marginal marginal probability approximation of Eq. (19) changed to

$$\hat{P}(e_j = a|z) = KP(e_j = a) \prod_{i \in \mathcal{N}(j)} \left(\lambda_{i \to j}^a \right)^{1+r(i)}, \tag{65}$$

and the error to check message of Eq. (20) changed to

$$\mu_{j \to i}^a = KP(e_j = a) \left(\lambda_{i \to j}^a \right)^{r(i)} \prod_{i' \in \mathcal{N}(j) \setminus i} \left(\lambda_{i' \to j}^a \right)^{1+r(i')}. \tag{66}$$

As a result of this equivalence, we can consider one iteration of an augmented decoder to be of the same complexity as one iteration of the underlying decoder. This formulation also gives some insight into the effect of decoding with an augmented parity-check matrix. It can be seen that duplicating a check has the effect of increasing its influence in estimating the error. Furthermore, the message $\mu_{j \to i}$ is now no longer independent of the message $\lambda_{i \to j}$ if c_i is duplicated. This amplification and feedback will alter the convergence of the marginal probability estimates. This altered convergence can help the decoder to give a different (and hopefully correct) error estimate.

3. Combined

The third decoder we propose combines the augmented GF(2) and adjusted decoders for CSS codes. Initially, standard GF(2) decoding is attempted. If this is successful, then decoding is complete. If both $\hat{z}_X \neq z_X$ and $\hat{z}_Z \neq z_Z$, then we reattempt decoding for the X component using augmented parity-check matrices up to N times. If this is unsuccessful, then we repeat this procedure for the Z component. If we still have $\hat{z}_X \neq z_X$ and $\hat{z}_Z \neq z_Z$, then decoding halts. However, if one of $\hat{z}_X = z_X$ or $\hat{z}_Z = z_Z$ (either from the initial decoding or after attempting decoding with augmented parity-check matrices if required), then we reattempt decoding for the unsatisfied component with adjusted channel error probabilities as outlined in Sec. III B 1. If this is unsuccessful, then decoding for this component will be reattempted with augmented parity-check matrices up to N times using the same adjusted probabilities.

IV. SIMULATION RESULTS

A. Bicycle

The first code we have considered is a [[400,200]] bicycle code of Ref. [9]. Bicycle codes are dual-containing CSS codes that are constructed by first generating an $n/2 \times n/2$ binary circulant matrix A with row weight $w/2$. A is used to define the $n/2 \times n$ matrix $H_0 = [A \quad A^T]$ from which $(n - m)/2$ rows are removed to give $\tilde{H}$ (following the heuristic that column weight should be kept as uniform as possible). Taking $\tilde{H}_X = \tilde{H}_Z = \tilde{H}$ defines the GF(2) and GF(4) parity-check matrices according to Eqs. (36) and (39) respectively. The associated stabilizer code will have $k \geqslant n - m$, with equality when the
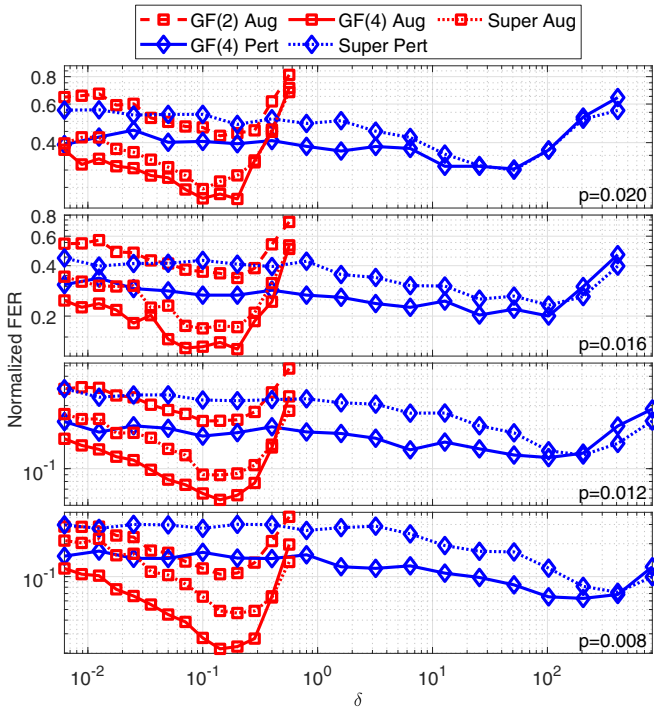
FIG. 2. The effect of augmentation density and random perturbation strength on decoder performance for the [[400,200]] bicycle code on the depolarizing channel. Each decoder uses $N = 10$ maximum attempts.
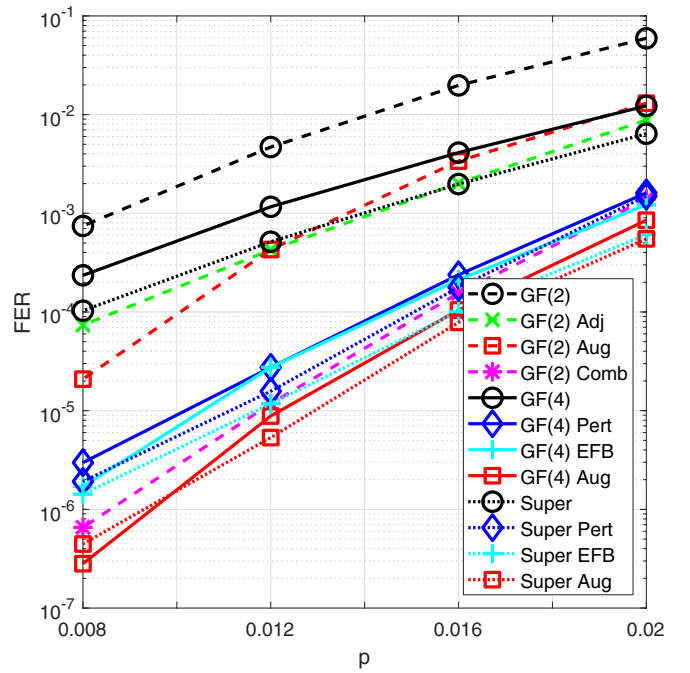

FIG. 3. FER performance of decoders with $N = 100$ attempts (where applicable) for the [[400,200]] bicycle code on the depolarizing channel.

parity-check matrix is full rank (this is the case for our code). Removing rows from H_0 corresponds to removing stabilizer generators of weight w. Unless a removed row belongs to the span of the remaining rows, which is unlikely, the removed generator will be in $N(\tilde{\mathcal{S}}) \backslash \tilde{\mathcal{S}}$. A bicycle code's distance is therefore upper bounded by w (we have chosen $w = 20$ for our code).

1. Depolarizing channel

We first consider the depolarizing channel. Both the augmented and random perturbation decoders have a tunable parameter δ, which controls the augmentation density and perturbation strength, respectively. As shown for classical codes in Ref. [11], this δ value can have a significant impact on the performance of an augmented decoder. We observe the same behavior for both augmented and random perturbation decoders in the quantum case as shown in Fig. 2. Here decoders with $N = 10$ maximum decoding attempts and varying δ have been tested at four different depolarizing probabilities (we use a maximum of $I_{\max} = 100$ iterations per attempt for every decoder in this paper). The vertical axis gives normalized FER, which is the modified decoder's FER divided by the underlying (standard) decoder's FER. Note that each data point in these figures, as well as all other figures presented in this paper, corresponds to at least 100 decoding errors. Based on these results, we have selected values of $\delta = 0.1$ for the augmented GF(2) decoder, $\delta = 0.15$ for the augmented GF(4) and supernode decoders, $\delta = 100$ for the random perturbation GF(4) decoder, and $\delta = 200$ for the random perturbation supernode decoder. Note that the δ

value we use for the combined decoder is always the same as the value used for the augmented GF(2) decoder.

We have tested all of the decoders outlined in Sec. III on this code. The random perturbation, EFB, augmented, and combined decoders all use $N = 100$ attempts. The FER performance of these decoders is shown in Fig. 3, and the average number of iterations required by each of them is shown in Fig. 4. It can be seen that the standard supernode decoder outperforms the standard GF(4) decoder, which in turn outperforms the standard GF(2) decoder. Furthermore, the supernode decoder requires fewer iterations on average than the standard GF(4) or GF(2) decoders [the number of iterations used by a GF(2) based decoder is taken to be the number used by one of the two constituent decoders]. However, note that comparing the number of iterations used by these different decoders, or indeed modified decoders based on different underlying decoders, is not particularly meaningful as their iterations are of differing complexity. The adjusted decoder can be seen to give a FER similar to the standard supernode decoder at the cost of a negligible increase in required iterations compared to the standard GF(2) decoder. This FER performance suggests that the adjusted decoder is successful in reintroducing the correlation between the X and Z error components. The random perturbation and EFB decoders based on either GF(4) or supernode decoders have similar FER performance and require a near-identical number of iterations on average. The augmented GF(4) and supernode decoders outperform both the random perturbation and EFB decoders while requiring a lower number of iterations on average. The augmented GF(2) decoder does give a reasonable FER reduction compared to the standard GF(2) decoder, but it is outperformed by all modified GF(4) and supernode
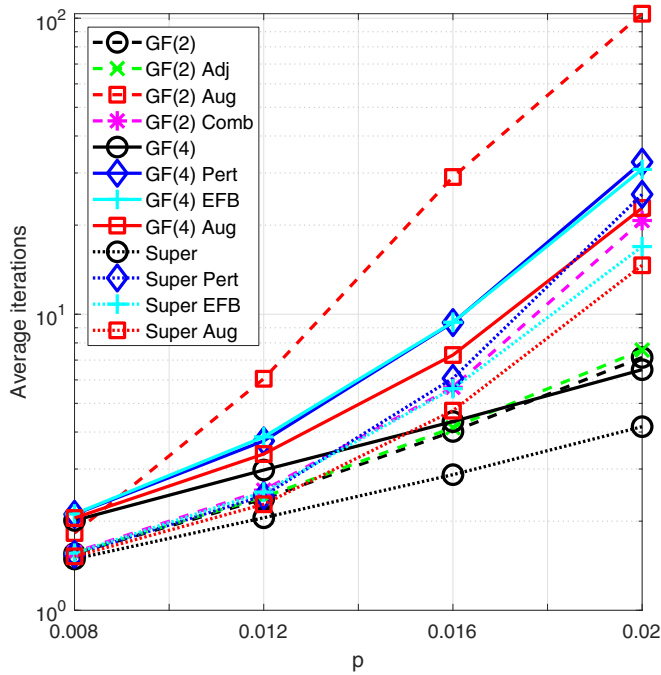
FIG. 4. Average number of iterations required by decoders with $N = 100$ attempts (where applicable) for the [[400,200]] bicycle code on the depolarizing channel.
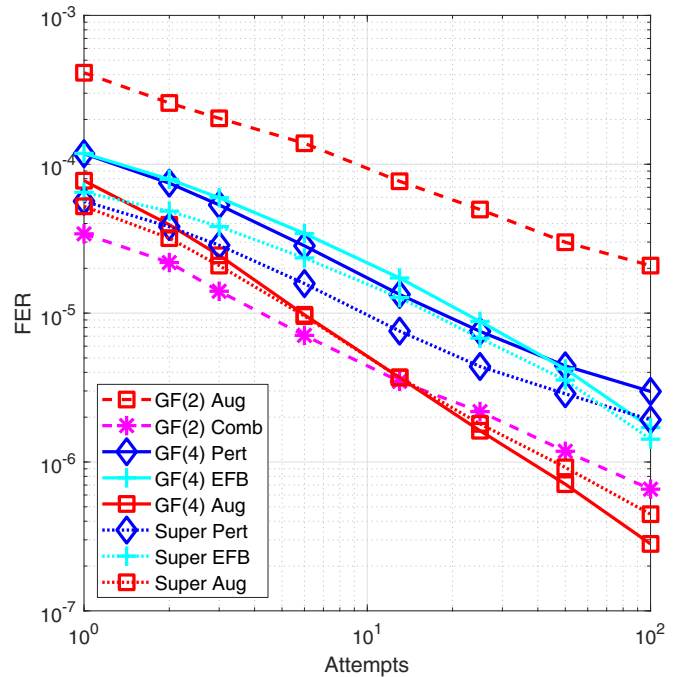

FIG. 5. FER performance of decoders at $p = 0.008$ with a varying number of decoding attempts for the [[400,200]] bicycle code on the depolarizing channel.

decoders. However, the combined decoder gives a FER lower than the random perturbation and EFB decoders. Furthermore, it also requires fewer iterations on average than the augmented GF(2) decoder. All of the decoding errors we have observed for this code are detected errors; that is, they are due to a failing of the decoder rather than the code's distance.

Figure 5 shows the effect of the maximum number of decoding attempts on the performance of the augmented, combined, random perturbation, and EFB decoders at a depolarizing probability of $p = 0.008$. For all decoders, the FER reduction with an increasing maximum number of attempts is approximately linear on a log-log plot. This suggests that we could continue to reduce the FER by increasing the maximum number of attempts beyond $N = 100$. It can be seen that the augmented and combined decoders only require approximately $N = 25$ maximum attempts to match the performance of random perturbation and EFB decoders with $N = 100$.

2. XZ channel

To isolate the effect of augmentation in the GF(2) case, we have repeated the analysis of the previous section for the XZ channel. As previously noted, the X and Z error components occur independently for this channel, therefore, there are no correlations to be ignored when using a GF(2) based decoder. As a result, the adjusted and combined decoders will give no performance increase over the standard GF(2) and augmented GF(2) decoders, respectively. While we have still employed the random perturbation decoder for comparison on this channel, we have not used the EFB decoder as it is specifically tailored to the depolarizing channel.

Again, we first tune the augmentation density and random perturbation strength using decoders with $N = 10$ as

shown in Fig. 6. It can be seen that the optimal value of δ is essentially independent of the underlying decoder. As such, we have selected a value of $\delta = 0.15$ for all augmented decoders and $\delta = 100$ for both of the random perturbation decoders. Note that these are the same values we have used for the GF(4) based decoders in the depolarizing case.

The FER performance and average required iterations for decoders with $N = 100$ maximum attempts are shown in Figs. 7 and 8, respectively. It can be seen that the standard GF(2), GF(4), and supernode decoders all exhibit near-identical performance on the XZ channel. That the GF(2) and supernode decoders yield the same FER is unsurprising and is consistent with the similar performance of the adjusted and supernode decoders on the depolarizing channel. The performance of the GF(4) decoder suggests that the 4-cycles involving one row from $\tilde{H}_X$ and one row from $\tilde{H}_Z$ have no effect on decoding performance when the error components are independent. The performance of the augmented and random perturbation decoders is also largely independent of the underlying decoder. Furthermore, the relative performance of the decoders is very similar to that observed for the GF(4) based decoders in the depolarizing case, with the augmented decoders outperforming the random perturbation decoders.

The effect of the maximum number of decoding attempts on decoder performance is shown for $p = 0.008$ in Fig. 9. Unsurprisingly, the performance of the augmented and random perturbation decoders remains largely independent of the underlying decoder over the range of N values tested. Furthermore, the relative performance is very similar to that exhibited by the GF(4) based decoders in the depolarizing case, with the augmented decoders only requiring approximately $N = 25$
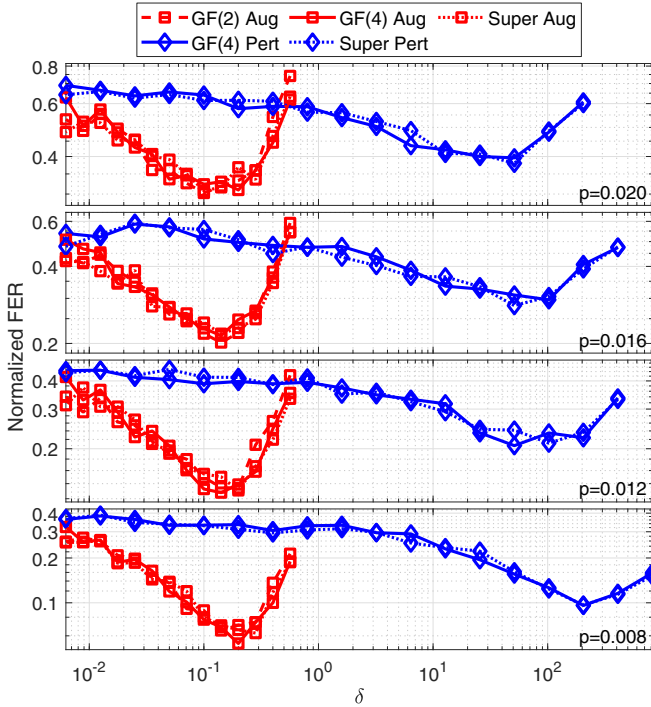
FIG. 6. The effect of augmentation density and random perturbation strength on decoder performance for the [[400,200]] bicycle code on the XZ channel. Each decoder uses $N = 10$ maximum attempts.

maximum attempts to match the performance of the random perturbation decoders with $N = 100$.

We have tested the performance of decoders on the XZ channel for all four CSS codes considered in this paper. However, we omit the results for the other three codes as they all follow the same trend outlined here. That is, the performance of decoders is essentially independent of the underlying decoder, and the relative performance of the augmented and random perturbation decoders is very similar to that exhibited by the GF(4) based decoders in the depolarizing case.

B. BIBD

The second code we have considered is a [[610,490]] balanced incomplete block design (BIBD) code from Ref. [22]. Like the bicycle code, this is also a dual-containing CSS code. A BIBD $(X, \mathcal{B})$, where $X = \{x_1, \ldots, x_v\}$ and $\mathcal{B} = \{B_1, \ldots, B_b\}$, is a collection of b subsets (blocks) of size k that are drawn from a set X containing v elements. Each pair of elements occurs in λ of the blocks, and every element occurs in r blocks. The $v \times b$ GF(2) incidence matrix A of $(X, \mathcal{B})$ has elements

$$A_{ij} = \begin{cases} 1 & x_i \in B_j, \\ 0 & x_i \notin B_j. \end{cases} \tag{67}$$

If λ is even, then A will satisfy $AA^T = 0$ as any two rows will overlap an even number of times. As such, taking $\tilde{H} = \tilde{H}_X = \tilde{H}_Z = A$ defines a dual-containing CSS code. The BIBD that we have selected follows the construction of Ref. [23]. If $6t + 1$ is a prime or prime power and α is a primitive element of
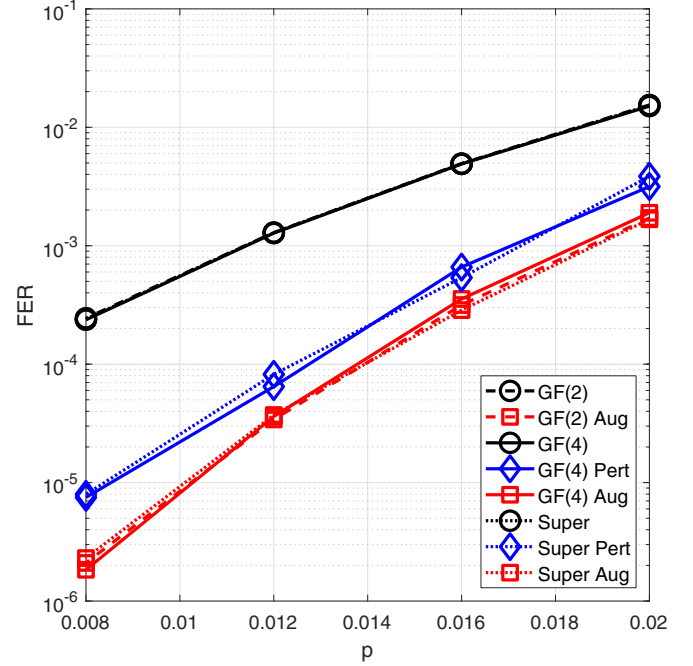
GF$(6t + 1)$, then a BIBD [GF$(6t + 1)$, $\mathcal{B}$] can be constructed with $v = 6t + 1$, $b = t(6t + 1)$, $r = 4t$, $k = 4$, and $\lambda = 2$. To do this, t base blocks $\tilde{B}_i$ are constructed for $0 \leqslant i \leqslant t - 1$ with

$$\tilde{B}_i = \{0, \alpha^i, \alpha^{2t+i}, \alpha^{4t+i}\}. \tag{68}$$

FIG. 7. FER performance of decoders with $N = 100$ attempts (where applicable) for the [[400,200]] bicycle code on the XZ channel.
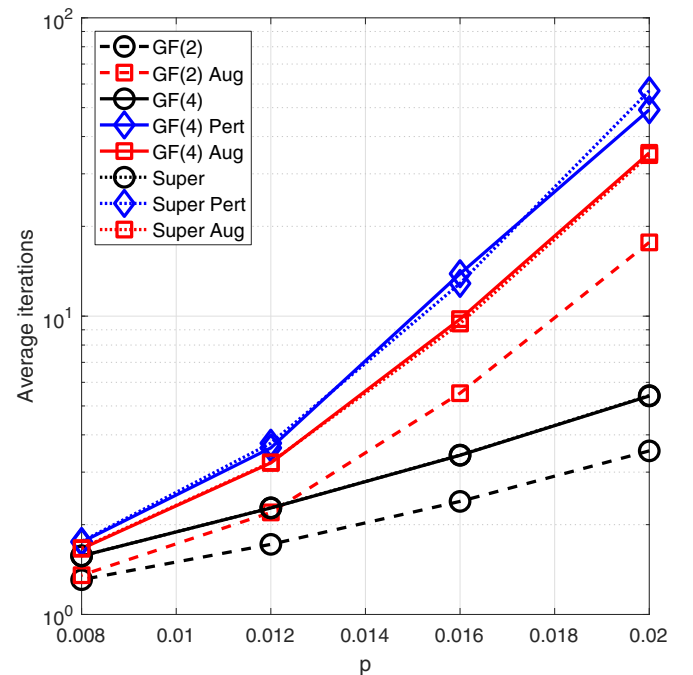
FIG. 8. Average number of iterations required by decoders with $N = 100$ attempts (where applicable) for the [[400,200]] bicycle code on the XZ channel.
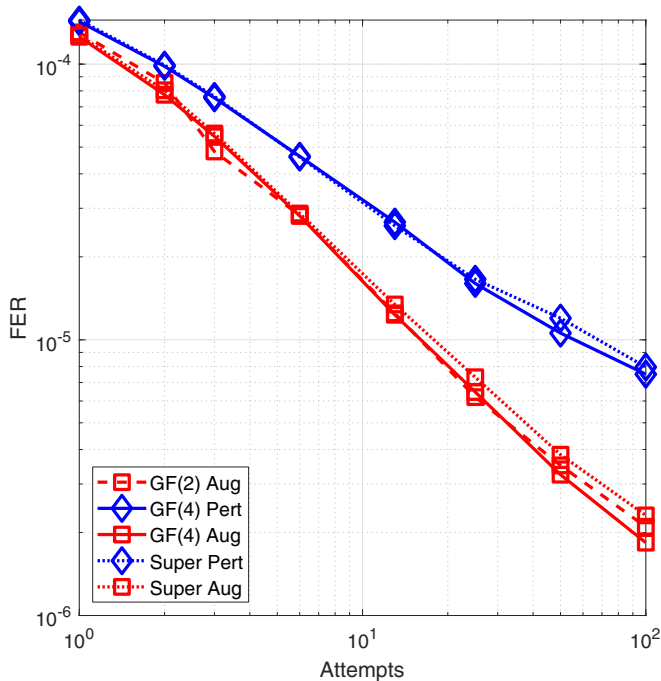
FIG. 9. FER performance of decoders at $p = 0.008$ with a varying number of decoding attempts for the [[400,200]] bicycle code on the XZ channel.
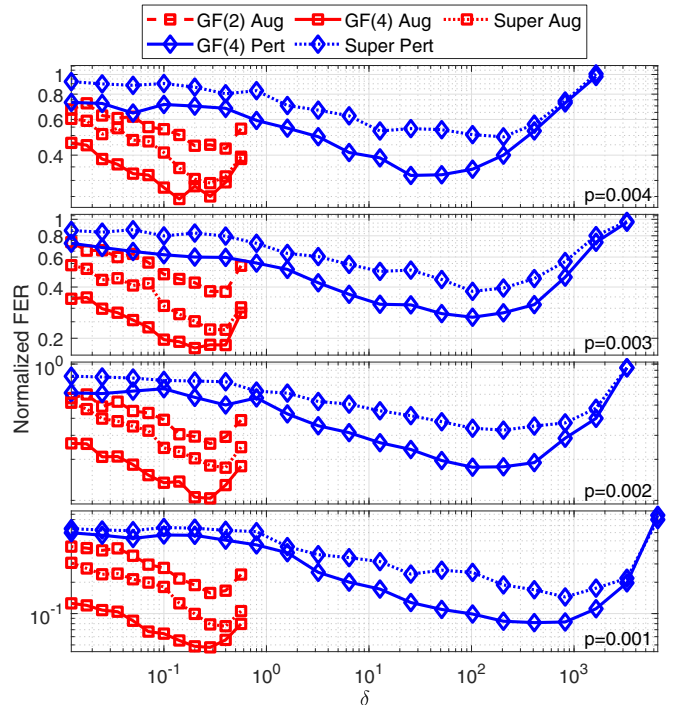

FIG. 10. The effect of augmentation density and random perturbation strength on decoder performance for the [[610,490]] BIBD code on the depolarizing channel. Each decoder uses $N = 10$ maximum attempts.

$6t + 1$ blocks of the form $\tilde{B}_i + \beta = \{\beta, \alpha^i + \beta, \alpha^{2t+i} + \beta, \alpha^{4t+i} + \beta\}$, where $\beta \in \mathrm{GF}(6t + 1)$, can then be constructed from each base block. This gives a total of $t(6t + 1)$ blocks and a corresponding incidence matrix of the form

$$\tilde{H} = A = (A_1 \quad A_2 \quad \cdots \quad A_t). \tag{69}$$

Here each A_i is a $(6t + 1) \times (6t + 1)$ circulant matrix of weight $k = 4$. We have selected $t = 10$ and $\alpha = 2$ for our code.

The results presented for this code and all codes that follow are on the depolarizing channel. The effect of augmentation density and random perturbation strength for decoders with $N = 10$ on this code is shown in Fig. 10. Based on these results, we have selected values of $\delta = 0.3$ for all augmented decoders, $\delta = 200$ for the random perturbation GF(4) decoder, and $\delta = 400$ for the random perturbation supernode decoder.

The FER performance and average required iterations for decoders with $N = 100$ maximum attempts are shown in Figs. 11 and 12, respectively. The results here are quite similar to those for the bicycle code. Again, the adjusted decoder gives performance similar to that of the supernode decoder. Furthermore, the random perturbation and EFB decoders perform similarly to one another. The augmented GF(2) decoder is outperformed by all modified GF(4) and supernode decoders. The combined, augmented GF(4), and augmented supernode decoders again outperform the random perturbation and EFB decoders. Overall there is less spread in the performance of the decoders on this BIBD code. This can be attributed to the fact that a large fraction of decoding errors are undetected. For example, approximately 65% of the errors exhibited by the augmented supernode decoder at $p = 0.001$

are undetected. This abundance of undetected errors suggests that decoding is being limited by the code's distance $d \leqslant 5$ [this value is based on the lowest weight element of $N(\tilde{S}) \backslash \tilde{S}$ that we have observed].

The effect of these undetected errors can also be seen in Fig. 13. For the bicycle code, the reduction in FER with increasing maximum number of iterations was approximately linear on a log-log plot. However, the reduction in FER for the BIBD code can be seen to taper off; that is, increasing the maximum number of attempts has diminishing returns. Partially as a result of this, we only require approximately $N = 10$ maximum attempts for our augmented supernode decoder to match the performance of the random perturbation and EFB decoders with $N = 100$.

C. Quasicyclic

The third code we have considered is a [[506,240]] quasicyclic code from Ref. [24]. Unlike the first two codes, this is a nondual-containing CSS code. The parity-check submatrices $\tilde{H}_X$ and $\tilde{H}_Z$ can be defined in terms of base matrices $\mathcal{H}_X$ and $\mathcal{H}_Z$, respectively, whose elements belong to the set $\{0, 1, \ldots, P - 1\}$. $\tilde{H}_X$ ($\tilde{H}_Z$) is then constructed by replacing each element of $\mathcal{H}_X$ ($\mathcal{H}_Z$) with a $P \times P$ identity matrix shifted circularly to the right by an amount given by the replaced element. The base matrix construction of Ref. [24] gives a parity-check matrix that satisfies $\tilde{H}_Z \tilde{H}_X^T = 0$; it also ensures that the factor graphs associated with $\tilde{H}_X$ and $\tilde{H}_Z$ are free of 4-cycles. These base matrices are constructed from a so-called "perfume" (perfect fulfillment). Let $\mathbb{Z}_P$ be the set of integers $\{0, 1, \ldots, P - 1\}$ with addition, subtraction, and
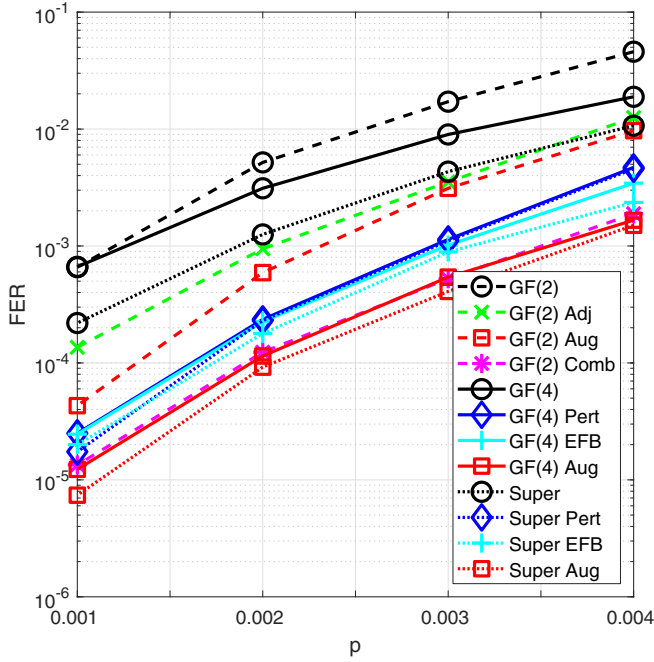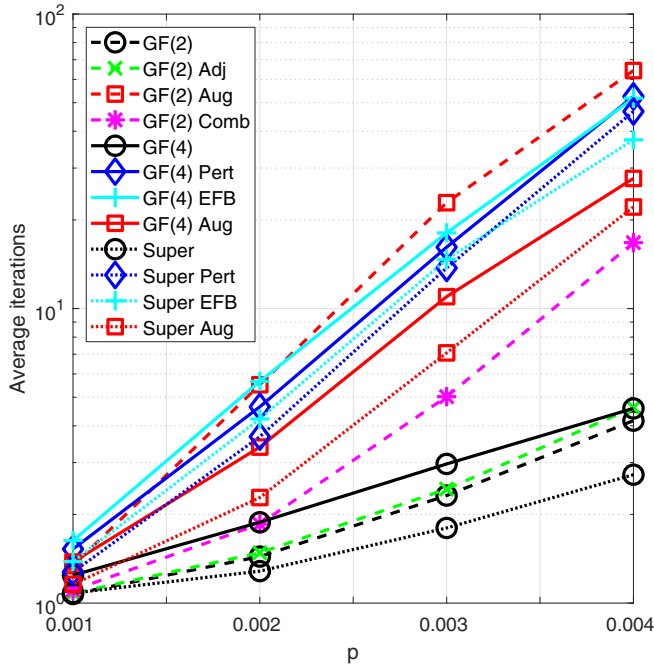
FIG. 11. FER performance of decoders with $N = 100$ attempts (where applicable) for the [[610,490]] BIBD code on the depolarizing channel.
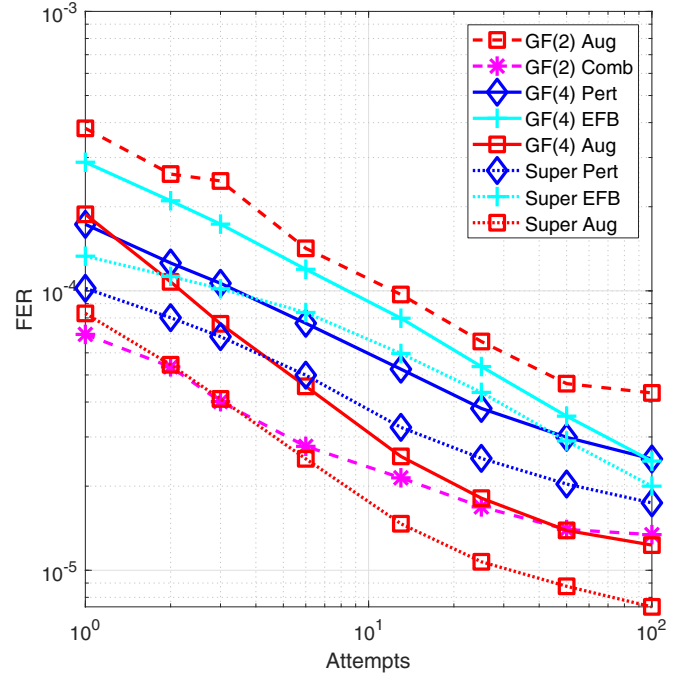

FIG. 13. FER performance of decoders at $p = 0.001$ with a varying number of decoding attempts for the [[610,490]] BIBD code on the depolarizing channel.

multiplication modulo P. $\mathbb{Z}_P^*$ is then the Abelian multiplicative group $\mathbb{Z}_P^* = \{z \in \mathbb{Z}_P : \gcd(z, P) = 1\}$. For positive integers P and σ, σ is a fulfillment of P if σ is coprime to P and $1 - \sigma^i$ is coprime to P for $1 \leqslant i < \text{ord}(\sigma)$. Here $\text{ord}(\sigma)$ is

FIG. 12. Average number of iterations required by decoders with $N = 100$ attempts (where applicable) for the [[610,490]] BIBD code on the depolarizing channel.

the order of σ in $\mathbb{Z}_P^*$. A triple of positive integers (P, σ, τ) is a perfume if σ is a fulfillment of P, τ is coprime to P, and $\tau \notin \{\sigma, \sigma^2, \dots, \sigma^{\text{ord}(\sigma)}\}$. Letting $L = 2\text{ord}(\sigma)$, we define

$$c_{jl} = \begin{cases} \sigma^{-j+l} & \text{if } 0 \leqslant l < \frac{L}{2}, \\ \tau\sigma^{-j+l} & \text{if } \frac{L}{2} \leqslant l \leqslant L, \end{cases} \tag{70}$$

$$d_{kl} = \begin{cases} -\tau\sigma^{k-l} & \text{if } 0 \leqslant l < \frac{L}{2}, \\ -\sigma^{k-l} & \text{if } \frac{L}{2} \leqslant l \leqslant L. \end{cases} \tag{71}$$

Indexing from zero, these are the elements of the $J \times L$ and $K \times L$ base matrices $\tilde{H}_X$ and $\tilde{H}_Z$, respectively, where $1 \leqslant J, K \leqslant L/2$. To construct our code, we have used the perfume $(23,8,20)$ (this gives $L = 22$) and have chosen $J = K = 6$.

The effect of augmentation density and random perturbation strength for decoders with $N = 10$ on this code is shown in Fig. 14. Note that for this code we can only use GF(2) and GF(4) based decoders as it is not dual containing. Based on these results, we have selected values of $\delta = 0.07$ for the augmented GF(2) decoder, $\delta = 0.05$ for the augmented GF(4) decoder, $\delta = 50$ for the random perturbation GF(4) decoder.

The FER performance and average required iterations for decoders with $N = 100$ maximum attempts are shown in Figs. 15 and 16, respectively. On the previous two codes, the augmented GF(2) decoder gave a similar or lower FER than the adjusted decoder. This is not the case here with the adjusted decoder giving a significantly lower FER. This suggests that the augmented decoder has some effect in alleviating the effect of 4-cycles in the code's factor graph [none of which
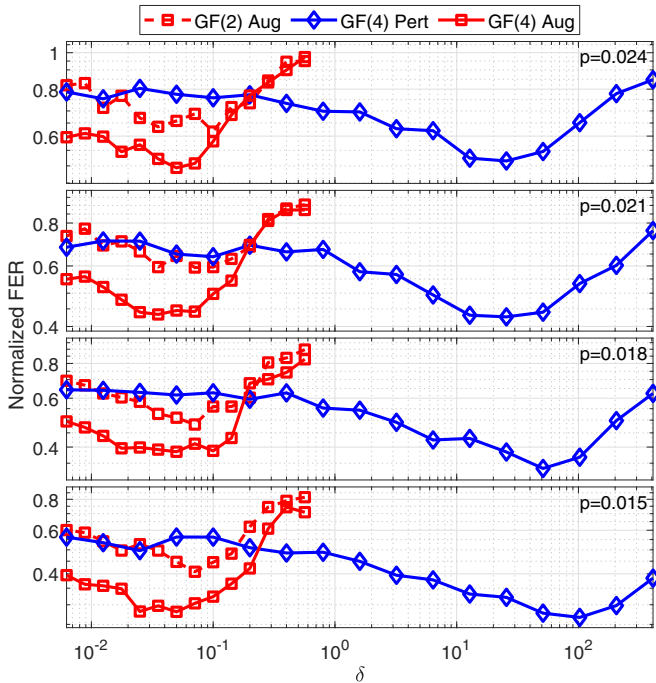
FIG. 14. The effect of augmentation density and random perturbation strength on decoder performance for the [[506,240]] quasicyclic code on the depolarizing channel. Each decoder uses $N = 10$ maximum attempts.
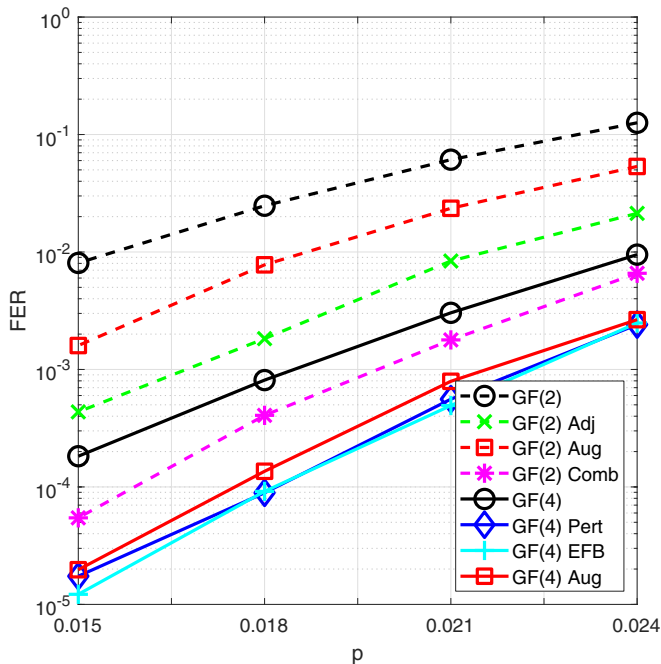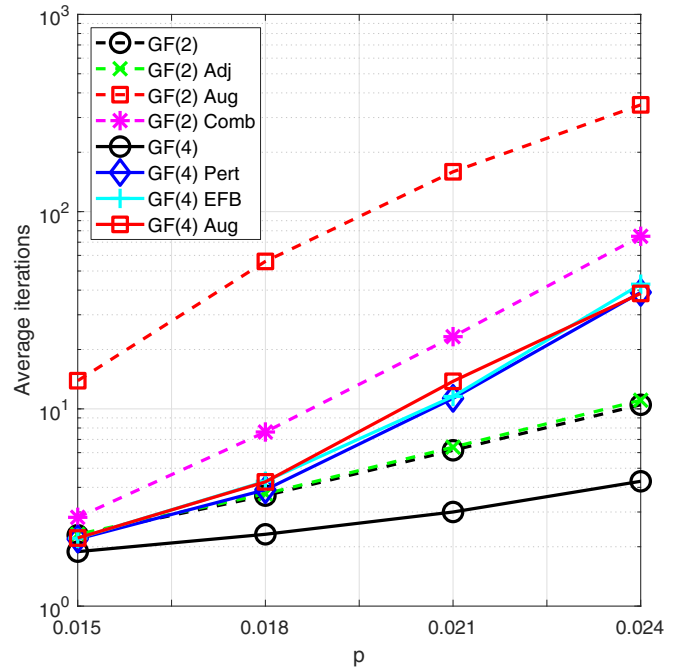

FIG. 16. Average number of iterations required by decoders with $N = 100$ attempts (where applicable) for the [[506,240]] quasicyclic code on the depolarizing channel.

are present when using a GF(2) decoder for this code]. The random perturbation, EFB, and augmented GF(4) decoders all perform similarly on this code. The combined decoder performs worse than the modified GF(4) decoders.

Like the bicycle code, all decoding errors observed for this code were detected errors. This is reflected in Fig. 17, which shows an approximately linear reduction in FER with an increasing number of maximum attempts on a log-log plot for all decoders considered.

FIG. 15. FER performance of decoders with $N = 100$ attempts (where applicable) for the [[506,240]] quasicyclic code on the depolarizing channel.
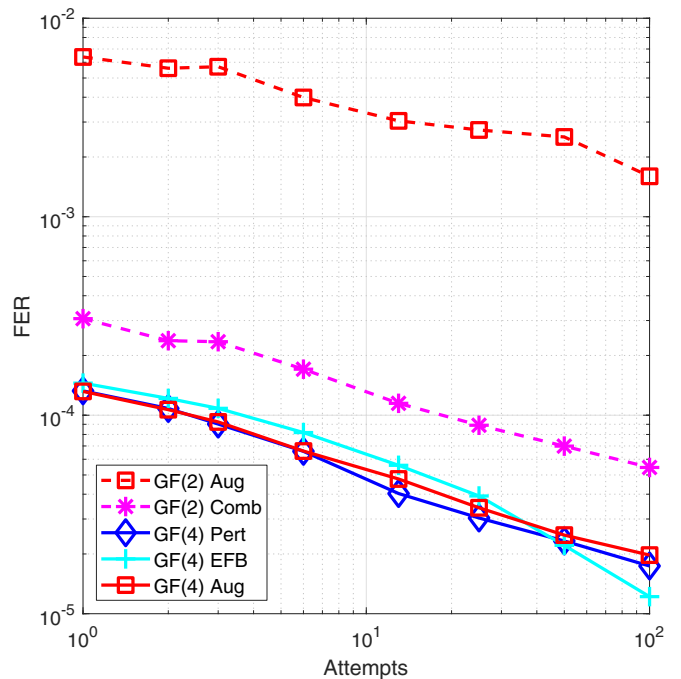
FIG. 17. FER performance of decoders at $p = 0.015$ with a varying number of decoding attempts for the [[506,240]] quasicyclic code on the depolarizing channel.
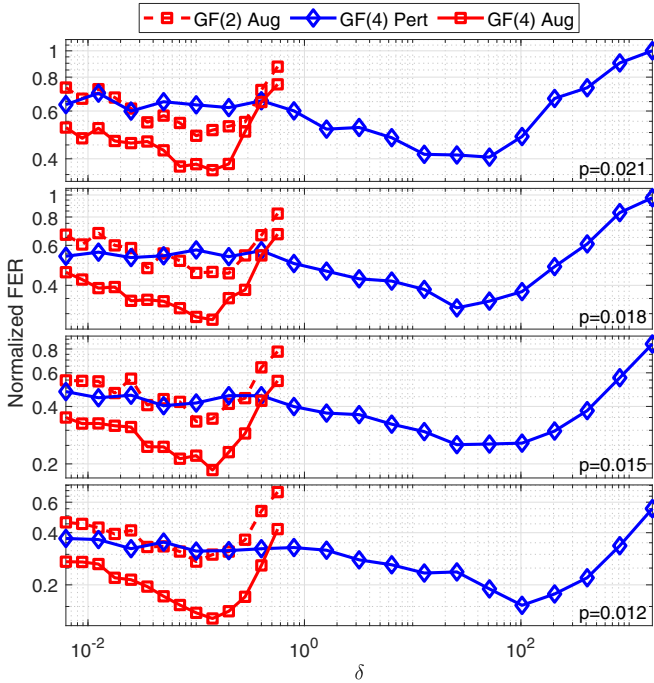
FIG. 18. The effect of augmentation density and random perturbation strength on decoder performance for the [[400,200]] bicyclelike code on the depolarizing channel. Each decoder uses $N = 10$ maximum attempts.

D. Bicyclelike

The fourth code we have considered is a [[400,200]] nondual-containing CSS based on the bicyclelike construction of Ref. [25]. The codes of Ref. [25] are constructed using a BIBD in a similar way to the code of Sec. IV B. $\tilde{H}_X$ is constructed by taking the first a (where a is even) submatrices of the BIBD's adjacency matrix as given in Eq. (69); that is,

$$\tilde{H}_X = \begin{pmatrix} A_1 & A_2 & \cdots & A_a \end{pmatrix}. \tag{72}$$

$\tilde{H}_Z$ is then a cyclically shifted version of $\tilde{H}_X$ with

$$\tilde{H}_Z = \begin{pmatrix} A_{\frac{a}{2}+1} & A_{\frac{a}{2}+2} & \cdots & A_a & A_1 & A_2 & \cdots & A_{\frac{a}{2}} \end{pmatrix}. \tag{73}$$

The use of a BIBD with $\lambda = 1$ ensures that $\tilde{H}_X$ and $\tilde{H}_Z$ are both free of 4-cycles. However, we have observed that codes constructed in this way have low distances and are therefore not appropriate for comparing decoders. We have found that this distance can be increased by generalizing the construction to allow the circulant matrices $A_1, \ldots, A_a$ to be randomly generated. Note that this comes at the expense of introducing 4-cycles. For our code we have constructed $\tilde{H}_X$ from four 100×100 circulant matrices of weight five. Each of $\tilde{H}_X$ and $\tilde{H}_Z$ yield factor graphs with 1700 4-cycles, compared to the 2737 4-cycles of the bicycle code considered in Sec. IV A.

The effect of augmentation density and random perturbation strength for decoders with $N = 10$ on this code is shown in Fig. 18. Based on these results, we have selected values of $\delta = 0.1$ for the augmented GF(2) decoder, $\delta = 0.15$ for the augmented GF(4) decoder, and $\delta = 100$ for the random perturbation GF(4) decoder.
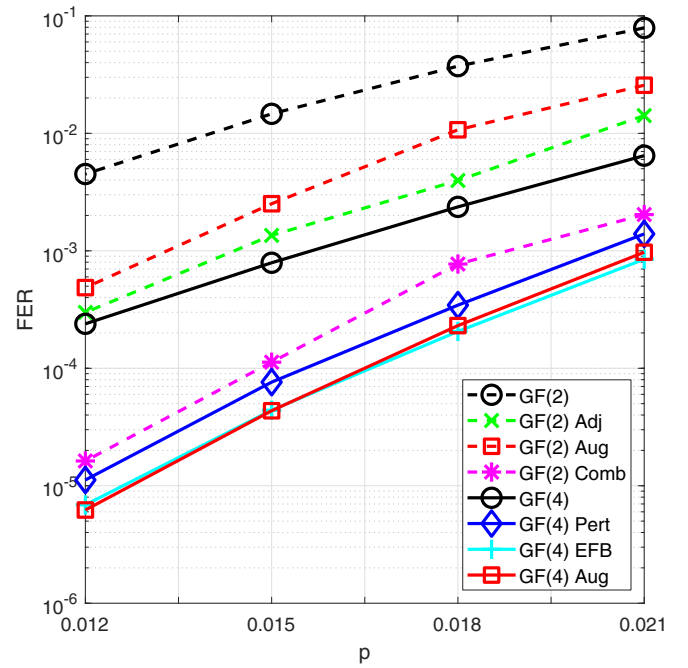

FIG. 19. FER performance of decoders with $N = 100$ attempts (where applicable) for the [[400,200]] bicyclelike code on the depolarizing channel.

The FER performance and average required iterations for decoders with $N = 100$ maximum attempts are shown in Figs. 19 and 20, respectively. Again, the adjusted decoder outperforms the augmented GF(2) decoder; however, the gap in their performance is smaller than for the quasicyclic code of Sec. IV C. The EFB and augmented GF(4) decoders perform
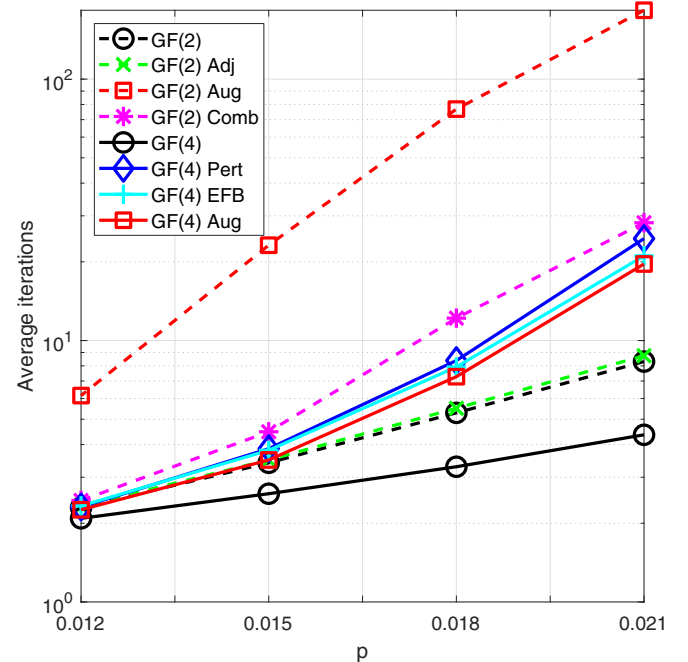

FIG. 20. Average number of iterations required by decoders with $N = 100$ attempts (where applicable) for the [[400,200]] bicyclelike code on the depolarizing channel.
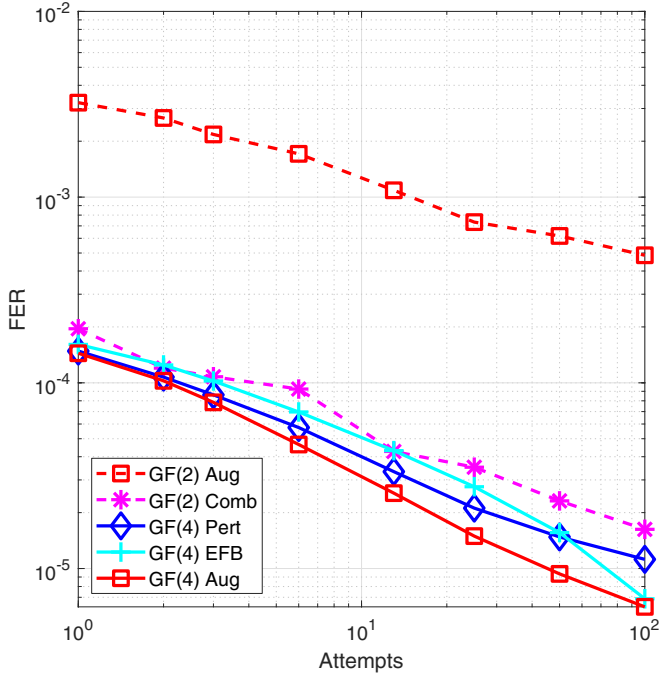
FIG. 21. FER performance of decoders at $p = 0.012$ with a varying number of decoding attempts for the [[400,200]] bicyclelike code on the depolarizing channel.

FIG. 22. The effect of augmentation density and random perturbation strength on decoder performance for the [[400,202]] non-CSS code A on the depolarizing channel. Each decoder uses $N = 10$ maximum attempts.

similarly on this code, both outperforming the random perturbation decoder. The combined decoder is again outperformed by all modified GF(4) decoders, although the performance gap is smaller than in the quasicyclic case.

While our modified construction gives a higher distance than the codes presented in Ref. [25], we still observed a moderate number of undetected errors, which can be attributed to the codes moderately low distance of $d \leqslant 10$. For example, at $p = 0.015$ approximately 15% of errors are undetected for both the EFB and augmented GF(4) decoders. However, this is not significant enough fraction of errors to prevent the FER reducing near linearly on a log-log plot with an increasing maximum number of attempts as shown in Fig. 21.

E. Non-CSS A

The fifth code we have considered is a [[400,202]] non-CSS code based on construction three of Ref. [26]. The GF(2) and GF(4) parity-check matrices for this code are defined by the matrices H_X and H_Z as outlined in Eqs. (34) and (39). For this code these matrices are of the form

$$H_X = \begin{pmatrix} A_X^{(1)} & A_X^{(2)} & \cdots & A_X^{(a)} \end{pmatrix}, \tag{74}$$

$$H_Z = \begin{pmatrix} A_Z^{(1)} & A_Z^{(2)} & \cdots & A_Z^{(a)} \end{pmatrix}. \tag{75}$$

The submatrices $A_X^{(i)}$ and $A_Z^{(i)}$ are given by

$$A_X^{(i)} = \begin{pmatrix} B_X^{(i)} & B_X^{(i)T} P_i^T \\ P_i B_X^{(i)T} & P_i B_X^{(i)} P_i^T \end{pmatrix}, \tag{76}$$

$$A_Z^{(i)} = \begin{pmatrix} B_Z^{(i)} & B_Z^{(i)T} P_i^T \\ P_i B_Z^{(i)T} & P_i B_Z^{(i)} P_i^T \end{pmatrix}. \tag{77}$$
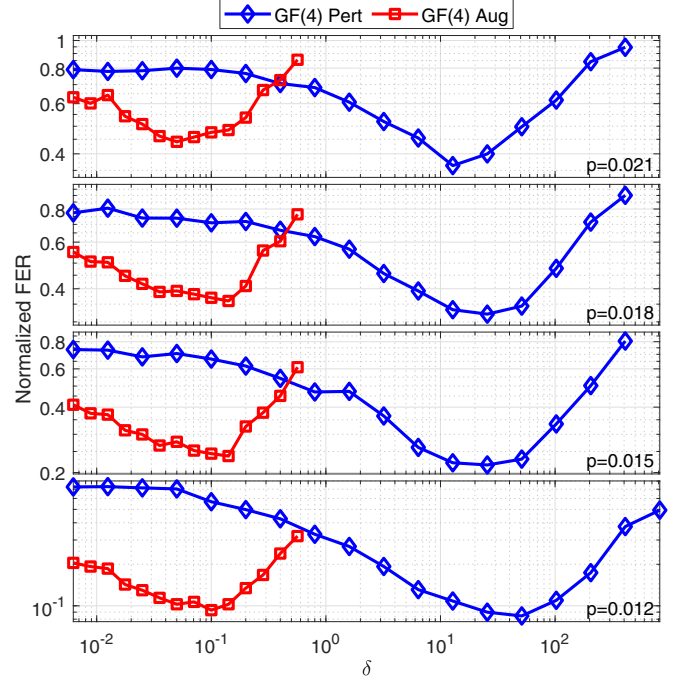
Here $B_X^{(i)}$ and $B_Z^{(i)}$ are square matrices of the same size that are either both symmetric or both circulant; P_i is a square matrix satisfying $P_i^T = P_i^{-1}$. For our code, we have taken $a = 2$, each $B_X^{(i)}$ and $B_Z^{(i)}$ to be a 100×100 circulant matrix of weight three, and each P_i to be a 100×100 permutation matrix.

The effect of augmentation density and random perturbation strength for decoders with $N = 10$ on this code is shown in Fig. 22. Note that for non-CSS codes we can only use GF(4) based decoders. Based on these results, we have chosen $\delta = 0.1$ for the augmented decoder and $\delta = 25$ for the random perturbation decoder.

The FER performance and average required iterations for decoders with $N = 100$ maximum attempts are shown in Figs. 23 and 24, respectively. It can be seen that the random perturbation, EFB, and augmented decoders all perform similarly on this code.

The majority of decoding errors are detected errors for this code, which also has distance $d \leqslant 10$. At $p = 0.012$ only 1%–2% of errors are undetected for the random perturbation, EFB, and augmented decoders. This is again reflected in the near-linear reduction in FER with increasing maximum number of attempts on the log-log plot given in Fig. 25.

F. Non-CSS B

The final code we have considered is a [[400,201]] non-CSS code based on construction four of Ref. [26]. This construction is quite similar to that of the last section with H_X and H_Z defined as in Eqs. (74) and (75), respectively.
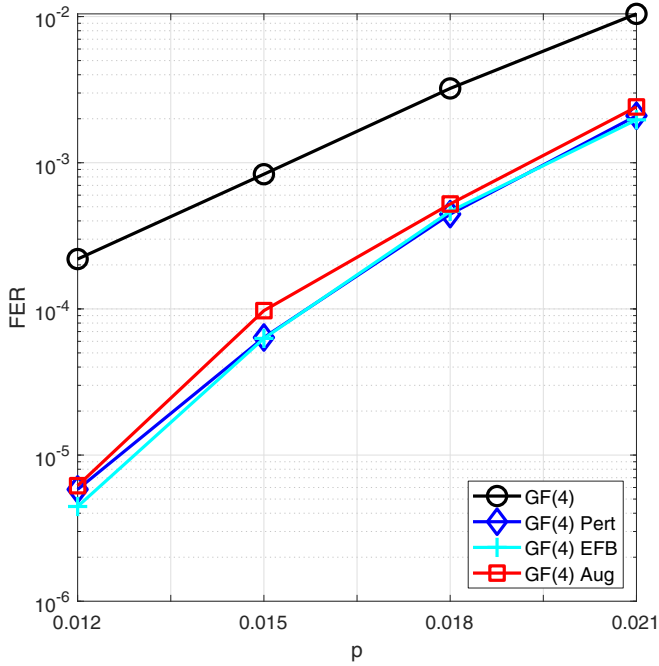
FIG. 23. FER performance of decoders with $N = 100$ attempts (where applicable) for the [[400,202]] non-CSS code A on the depolarizing channel.

However, the submatrices $A_X^{(i)}$ and $A_Z^{(i)}$ are now given by

$$A_X^{(i)} = \begin{pmatrix} B_X^{(i)} & B_X^{(i)T} P_i^T \end{pmatrix}, \quad (78)$$

$$A_Z^{(i)} = \begin{pmatrix} B_Z^{(i)} & B_Z^{(i)T} P_i^T \end{pmatrix}. \quad (79)$$
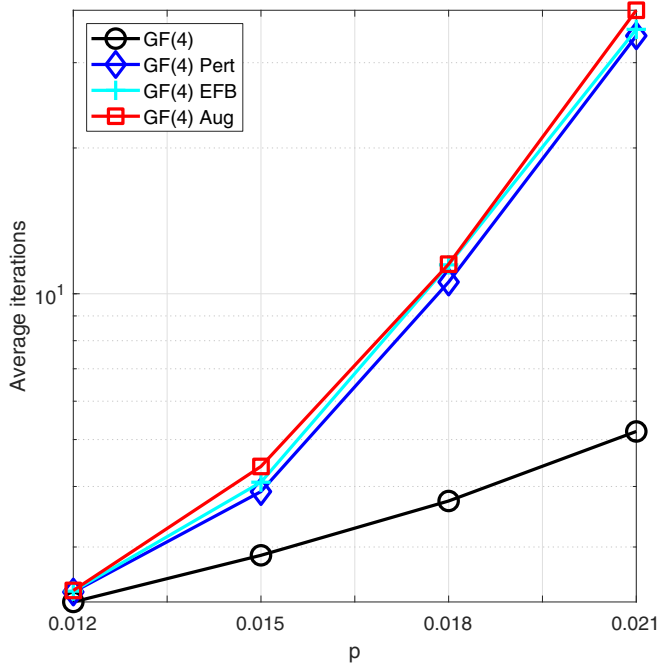

FIG. 24. Average number of iterations required by decoders with $N = 100$ attempts (where applicable) for the [[400,202]] non-CSS code A on the depolarizing channel.
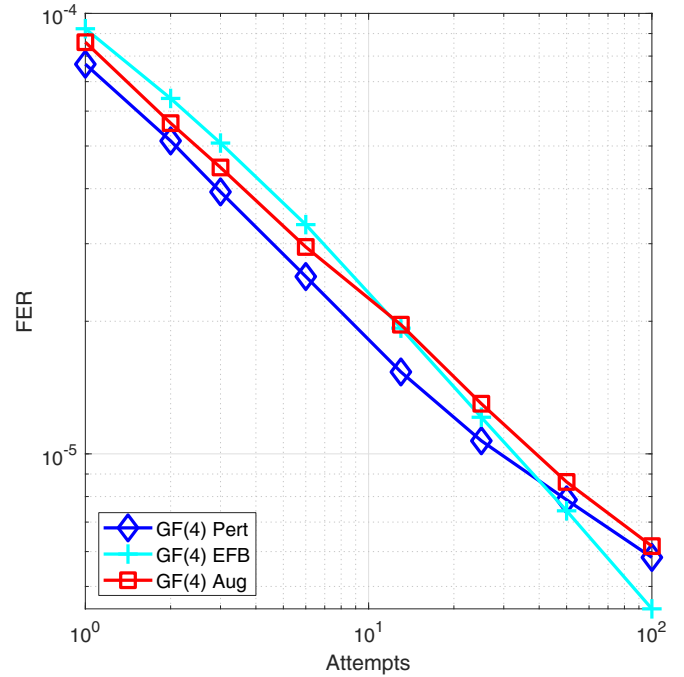

FIG. 25. FER performance of decoders at $p = 0.012$ with a varying number of decoding attempts for the [[400,202]] non-CSS code A on the depolarizing channel.

Here $B_X^{(i)}$ and $B_Z^{(i)}$ are either both symmetric, both circulant, or $B_X^{(i)} B_Z^{(i)T} + B_X^{(i)T} B_Z^{(i)}$ is symmetric; P_i is a permutation matrix. For our code we
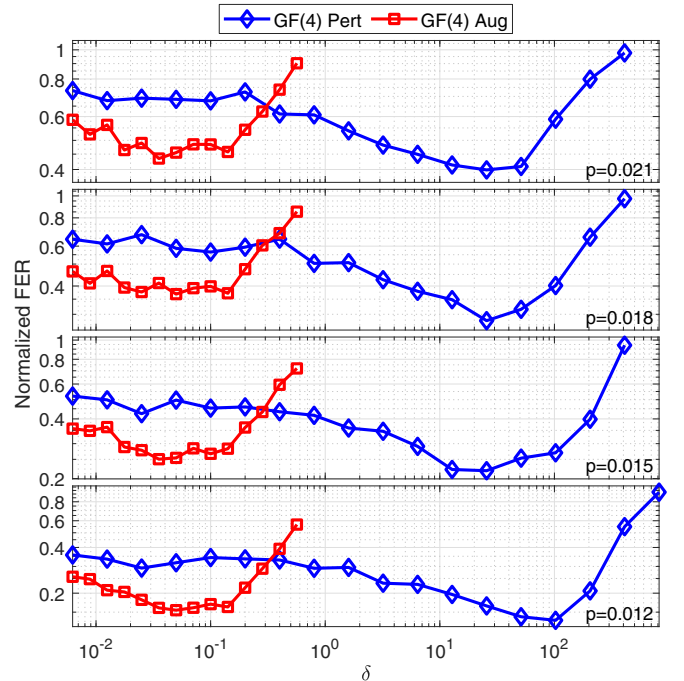

FIG. 26. The effect of augmentation density and random perturbation strength on decoder performance for the [[400,201]] non-CSS code B on the depolarizing channel. Each decoder uses $N = 10$ maximum attempts.
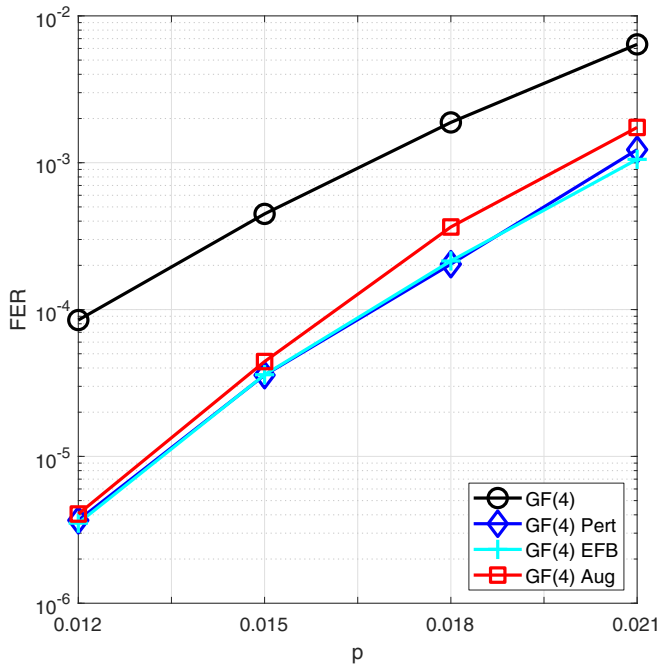
FIG. 27. FER performance of decoders with $N = 100$ attempts (where applicable) for the [[400,201]] non-CSS code B on the depolarizing channel.

have taken $a = 1$, $B_X^{(1)}$ and $B_Z^{(1)}$ to be 200×200 circulant matrices of weight six, and P_1 to be a 200×200 permutation matrix.

The effect of augmentation density and random perturbation strength for decoders with $N = 10$ on this code is
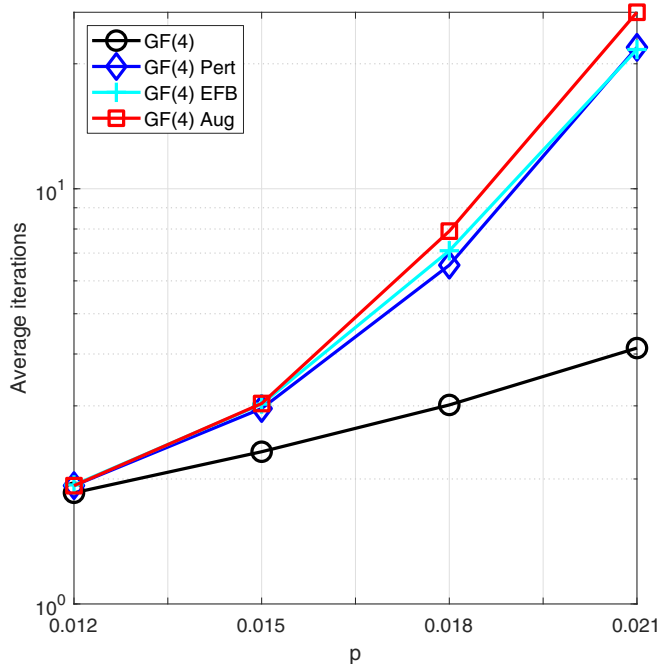

FIG. 28. Average number of iterations required by decoders with $N = 100$ attempts (where applicable) for the [[400,201]] non-CSS code B on the depolarizing channel.
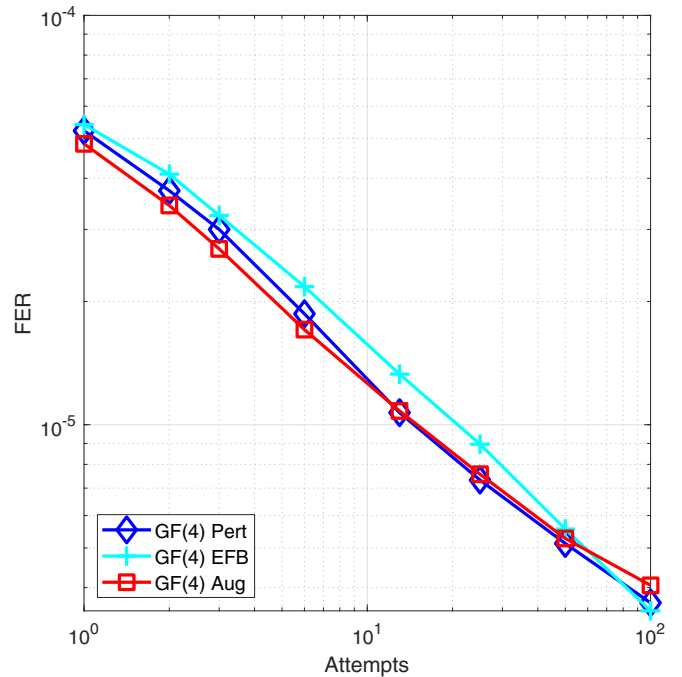

FIG. 29. FER performance of decoders at $p = 0.012$ with a varying number of decoding attempts for the [[400,201]] non-CSS code B on the depolarizing channel.

shown in Fig. 26. Based on these results, we have chosen $\delta = 0.05$ for the augmented decoder and $\delta = 25$ for the random perturbation decoder.

The FER performance and average required iterations for decoders with $N = 100$ maximum attempts are shown in Figs. 27 and 28, respectively. The results are consistent with those of Sec. IV E, with the random perturbation, EFB, and augmented decoders all performing fairly similarly on this code.

A moderate number of undetected errors were observed for this code, which also has distance $d \leqslant 10$. For example, at $p = 0.012$ approximately 10%–15% of errors are undetected for each decoder. It can also be seen that the reduction in FER with an increasing number of maximum attempts, while still near linear on the log-log plot of Fig. 29, tapers of slightly more than was observed for the code of the previous section.

V. CONCLUSION

We have presented modified belief propagation decoders for QLDPC codes that, depending on the code, either outperform or perform similarly to other decoders presented in literature. We have proposed the GF(2) based adjusted decoder, which uses modified error probabilities to reintroduce correlations between the X and Z components of an error that are lost when using a standard GF(2) decoder. Furthermore, we have demonstrated that the augmented decoder, which has previously been proposed for classical binary LDPC codes, can be applied in the quantum case and that it can be based on an underlying GF(2), GF(4), or supernode decoder. We have also proposed a combination of the augmented GF(2) and adjusted decoders. For the bicycle and BIBD based

dual-containing CSS codes tested, the augmented GF(4), augmented supernode, and combined decoders were shown to outperform random perturbation and EFB decoders. For the two nondual-containing CSS codes and the two non-CSS codes considered, augmented GF(4) and supernode decoders were shown to perform similarly to random perturbation and EFB decoders.

APPENDIX: CHECK NODE FOURIER TRANSFORM IMPLEMENTATIONS

1. Classical decoding

The check constraint of Eq. (18) can be written as

$$\sum_{j' \in \mathcal{M}(i) \setminus j} H_{ij'} e_{j'} = \sum_{j' \in \mathcal{M}(i) \setminus j} \tilde{e}_{j'} = z_i - H_{ij} a, \quad \text{(A1)}$$

where $\tilde{e}_{j'} = H_{ij'} e_{j'}$. $\tilde{e}_{j'}$ can be used to define

$$\tilde{\lambda}_{i \to j}^a = \sum_{\boldsymbol{e}: \sum_{j'} \tilde{e}_{j'} = a} \prod_{j'} \mu_{j' \to i}^{e_{j'}} = \sum_{\boldsymbol{e}: \sum_{j'} \tilde{e}_{j'} = a} \prod_{j'} \tilde{\mu}_{j' \to i}^{\tilde{e}_{j'}}, \quad \text{(A2)}$$

where $\tilde{\mu}_{j' \to i}^{\tilde{e}_{j'}} = \mu_{j' \to i}^{H_{ij'}^{-1} \tilde{e}_{j'}}$ (this corresponds to a permutation of elements) and $j' \in \mathcal{M}(i) \setminus j$. Equation (A2) is a convolution and as such can be efficiently computed using a Fourier transform $\mathcal{F}$ as

$$\tilde{\lambda}_{i \to j} = K \mathcal{F}^{-1} \left\{ \prod_{j'} \mathcal{F}\{\tilde{\mu}_{j' \to i}\} \right\}, \quad \text{(A3)}$$

where $\mathcal{F}^{-1}$ is the inverse Fourier transform and the product is element-wise (K is a normalization factor). A Hadamard transform can be used in the binary case; if $\tilde{\mu}_{j' \to i} = (\tilde{\mu}_{j' \to i}^0, \tilde{\mu}_{j' \to i}^1)$ is considered as a column vector, then

$$\mathcal{F}\{\tilde{\mu}_{j' \to i}\} = F \tilde{\mu}_{j' \to i}, \quad \text{(A4)}$$

where

$$F \propto \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}. \quad \text{(A5)}$$

The inverse transform is also achieved through multiplication by F (up to some unimportant scaling factor). $\lambda_{i \to j}$ is a permuted version of $\tilde{\lambda}_{i \to j}$ with

$$\lambda_{i \to j}^a = \tilde{\lambda}_{i \to j}^{z_i - H_{ij} a}. \quad \text{(A6)}$$

2. GF(4) stabilizer decoding

The check constraint of Eq. (42) can be written as

$$\mathrm{tr} \left(H_{ij} \bar{a} + \sum_{j' \in \mathcal{M}(i) \setminus j} H_{ij'} \bar{e}_{j'} \right) = \mathrm{tr} \left(H_{ij} \bar{a} + \sum_{j' \in \mathcal{M}(i) \setminus j} \tilde{e}_{j'} \right) = z_i, \quad \text{(A7)}$$

where $\tilde{e}_{j'} = H_{ij'} \bar{e}_{j'}$. $\tilde{\lambda}_{i \to j}^a$ is defined in the same was as Eq. (A2) with $\tilde{\mu}_{j' \to i}^{\tilde{e}_{j'}} = \mu_{j' \to i}^{(H_{ij'}^{-1} \tilde{e}_{j'})^{-1}} = \mu_{j' \to i}^{H_{ij'} \tilde{e}_{j'}^{-1}}$. Again, $\tilde{\lambda}_{i \to j}$ can be calculated using the Hadamard transform with

$$F \propto \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}^{\otimes 2} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}. \quad \text{(A8)}$$

$\tilde{\lambda}_{i \to j}^a$ corresponds to $\sum_{j'} \tilde{e}_{j'} = a$ and as such can be used to determine $\lambda_{i \to j}^a$, which corresponds to $e_j = a$. If $z_i = 0$, then $H_{ij} \bar{a} + \sum_{j'} \tilde{e}_{j'} = 0$ or 1; conversely, if $z_i = 1$, then $H_{ij} \bar{a} + \sum_{j'} \tilde{e}_{j'} = \omega$ or $\bar{\omega}$. Therefore, for $z_i = 0$ the elements of $\lambda_{i \to j}$ are

$$\lambda_{i \to j}^a = \tfrac{1}{2} \big[\tilde{\lambda}_{i \to j}^{-H_{ij} \bar{a}} + \tilde{\lambda}_{i \to j}^{1 - H_{ij} \bar{a}} \big], \quad \text{(A9)}$$

and for $z_i = 1$

$$\lambda_{i \to j}^a = \tfrac{1}{2} \big[\tilde{\lambda}_{i \to j}^{\omega - H_{ij} \bar{a}} + \tilde{\lambda}_{i \to j}^{\bar{\omega} - H_{ij} \bar{a}} \big]. \quad \text{(A10)}$$

These can be combined to give

$$\lambda_{i \to j}^a = \tfrac{1}{2} \big[\tilde{\lambda}_{i \to j}^{\omega z_i - H_{ij} \bar{a}} + \tilde{\lambda}_{i \to j}^{\omega z_i + 1 - H_{ij} \bar{a}} \big]. \quad \text{(A11)}$$

[1] R. Gallager, IRE Trans. Inf. Theory **8**, 21 (1962).

[2] D. J. MacKay, IEEE Trans. Inf. Theory **45**, 399 (1999).

[3] A. R. Calderbank, E. M. Rains, P. Shor, and N. J. Sloane, IEEE Trans. Inf. Theory **44**, 1369 (1998).

[4] D. E. Gottesman, Ph.D. thesis, California Institute of Technology, 1997.

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

[6] D. Poulin and Y. Chung, Quantum Inf. Comput. **8**, 987 (2008).

[7] J. A. McGowan and R. C. Williamson, in *Proceedings of IEEE Information Theory Workshop, Paris, France, 2003* (IEEE, Piscataway, NJ, 2003), pp. 230–233.

[8] A. R. Calderbank, E. M. Rains, P. W. Shor, and N. J. A. Sloane, Phys. Rev. Lett. **78**, 405 (1997).

[9] D. J. MacKay, G. Mitchison, and P. L. McFadden, IEEE Trans. Inf. Theory **50**, 2315 (2004).

[10] Y.-J. Wang, B. C. Sanders, B.-M. Bai, and X.-M. Wang, IEEE Trans. Inf. Theory **58**, 1231 (2012).

[11] A. Rigby, J. Olivier, H. C. Myburgh, C. Xiao, and B. P. Salmon, EURASIP J. Wireless Commun. Networking **2018**, 189 (2018).

[12] N. Delfosse and J.-P. Tillich, in *2014 IEEE International Symposium on Information Theory Honolulu, HI* (IEEE, Piscataway, NJ, 2014), pp. 1071–1075.

[13] E. Berlekamp, R. McEliece, and H. Van Tilborg, IEEE Trans. Inf. Theory **24**, 384 (1978).

[14] R. W. Hamming, Bell Syst. Tech. J. **29**, 147 (1950).

[15] T. Etzion, A. Trachtenberg, and A. Vardy, IEEE Trans. Inf. Theory **45**, 2173 (1999).

[16] T. Richardson and R. Urbanke, *Modern Coding Theory* (Cambridge University Press, Cambridge, 2008).

[17] D. J. MacKay, *Information Theory, Inference and Learning Algorithms* (Cambridge University Press, Cambridge, 2003).

[18] K. Kraus, *States, Effects and Operations: Fundamental Notions of Quantum Theory* (Springer, New York, 1983).

[19] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information: 10th Anniversary Edition* (Cambridge University Press, New York, NY, 2002).

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

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

[22] I. B. Djordjevic, IEEE Commun. Lett. **12**, 389 (2008).

[23] R. C. Bose, Ann. Eugenics **9**, 353 (1939).

[24] M. Hagiwara and H. Imai, in *IEEE International Symposium on Information Theory, Nice, France, 2007* (IEEE, Piscataway, NJ, 2007).

[25] Z. Babar, P. Botsinis, D. Alanis, S. X. Ng, and L. Hanzo, IEEE Commun. Lett. **20**, 9 (2016).

[26] P. Tan and J. Li, IEEE Trans. Inf. Theory **56**, 476 (2010).