IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 22 January 2025, accepted 2 February 2025, date of publication 5 February 2025, date of current version 10 February 2025.

Digital Object Identifier 10.1109/ACCESS.2025.3539475

RESEARCH ARTICLE

Quaternary Neural Belief Propagation Decoding of Quantum LDPC Codes With Overcomplete Check Matrices

SISI MIAO[1], (Student Member, IEEE), ALEXANDER SCHNERRING[2], HAIZHENG LI[1], (Graduate Student Member, IEEE), AND LAURENT SCHMALEN[1], (Fellow, IEEE)

[1]Communications Engineering Laboratory (CEL), Karlsruhe Institute of Technology (KIT), 76187 Karlsruhe, Germany
[2]Institute of Solar Research, German Aerospace Center (DLR), 04005 Almería, Spain

Corresponding author: Sisi Miao (sisi.miao@kit.edu)

This work was supported by the European Research Council (ERC) through the European Union's Horizon 2020 Research and Innovation Program under Grant 101001899.

ABSTRACT Quantum low-density parity-check (QLDPC) codes are promising candidates for error correction in quantum computers. One of the major challenges in implementing QLDPC codes in quantum computers is the lack of a universal decoder. In this work, we first propose to decode QLDPC codes with a belief propagation (BP) decoder operating on overcomplete check matrices. Then, we extend the neural BP (NBP) decoder, which was originally studied for suboptimal binary BP decoding of QLPDC codes, to quaternary BP decoders. Numerical simulation results demonstrate that both approaches as well as their combination yield a low-latency, high-performance decoder for several short to moderate length QLDPC codes.

INDEX TERMS Quantum error correction, quantum low-density parity-check codes, neural networks, channel coding, decoding algorithms.

I. INTRODUCTION

Quantum error correction (QEC) is an essential part of fault-tolerant quantum computing. Thanks to the potential of providing fault-tolerance with constant overhead [2] and the recent breakthroughs in designing asymptotically good quantum low-density parity-check (QLDPC) codes [3], [4], [5], [6], QLDPC codes are among the most promising candidates for future QEC schemes.

One of the major challenges in implementing QLDPC codes for practical quantum computers is the lack of a universal decoder that provides good decoding performance across a wide range of QLDPC codes. Two main families of potential decoders have been explored in the literature. The first family is the class of graph-theory-based decoders, such as the minimum weight perfect matching (MWPM) decoder [7], [8], which is commonly used for topological codes. However, a significant drawback of this decoder is its decoding latency. The second family is the class of message-passing decoders such as the belief propagation (BP) decoder or its low-complexity variant, the min-sum (MS) decoder. The decoding latency of such decoders is linear w.r.t. the block length and the number of decoding iterations. However, their excellent decoding performance for classical low-density parity-check (LDPC) codes is based on the condition that the Tanner graph has no short cycles (of length 4). This condition cannot be fulfilled in the case of QLDPC codes, where 4-cycles are unavoidable by construction. Therefore, for QEC, it is crucial to modify the BP decoder such that short cycles can be tolerated.

A variety of methods have been proposed to address this issue, which can be categorized into two categories. The first category contains approaches that modify the BP decoder itself, for example, message normalization and offsets [9],

The associate editor coordinating the review of this manuscript and approving it for publication was Faissal El Bouanani.

[10], layered scheduling [11], [12], and matrix augmentation [13]. The second category focuses on post-processing the output of the BP decoder, e.g., ordered statistics decoding (OSD) [11], [14], random perturbation [15], enhanced feedback [16], and stabilizer inactivation [17]. However, most of these post-processing methods introduce additional decoding latency, which makes them less appealing for QEC where decoding has to be performed with ultra-low latency.

In [1], we have proposed and studied two approaches to enhance the quaternary belief propagation (BP4) decoder for QLDPC codes [18], [19]. The first approach is to perform BP4 decoding on an overcomplete check matrix constructed by adding redundant low-weight checks to the original full-rank check matrix. The second approach is based on neural networks. We extend the neural belief propagation (NBP) decoder, which was originally proposed in [20] for the binary belief propagation (BP2) decoder, to the BP4 decoder. In this work, we extend the results of [1] by presenting an improved decoding strategy and including a detailed methodology for parameter optimization. A more comprehensive derivation and explanation of the proposed method is provided. The proposed decoder is also evaluated on various exemplary codes. Numerical simulation results demonstrate that both approaches as well as their combination yield a low-latency high-performance decoder for a wide range of short to moderate-length QLDPC codes. For example, for the evaluated toric codes, the proposed decoder achieves up to an order of magnitude reduction in post-decoding error rates compared to the MWPM decoder.

The paper's primary contribution is the novel NBP decoder for QLDPC codes, specifically addressing the degeneracy of quantum codes. Additionally, the tailored loss function for BP4 decoders enables efficient training of the decoder weights.

The remainder of this paper is structured as follows: In Sec. II, we briefly review the notions of QEC with quantum stabilizer codes (QSCs). Section III reviews the refined BP4 decoder and investigates the initialization of the BP decoder. In Sec. IV, we introduce decoding with overcomplete check matrices. The construction method of the redundant checks is given, as well as a heuristic explanation for the performance improvement. In Sec. V, the NBP decoder is introduced based on the refined BP4 decoder. In Sec. VI, the proposed decoder is evaluated with numerical simulations and compared with the state-of-the-art decoding algorithms in the literature. Section VII concludes this paper.

Notation: We use boldface letters to denote vectors and matrices, e.g., $\boldsymbol{a}$ and $\boldsymbol{A}$. The i-th component of vector $\boldsymbol{a}$ is denoted by a_i, and the element in the i-th row and j-th column of $\boldsymbol{A}$ is denoted by $A_{i,j}$. Let $\boldsymbol{A}_i$ be the i-th row of a matrix $\boldsymbol{A}$. $\boldsymbol{A}^{\mathsf{T}}$ denotes the matrix transpose. Pauli operators corresponding to quaternary vectors are denoted with the same letter in calligraphic form, such as $\boldsymbol{e}$ and $\mathcal{E}$. For brevity, the maximum number of decoding iterations L of a decoder is given as a superscript, e.g., BP4^{32} for $L = 32$.

II. PRELIMINARIES

A. STABILIZER CODES

QSCs [21], [22] are the quantum analogons of classical linear codes. To define a QSC, we first need to define the Pauli operators. For simplicity, we ignore the global phase and consider the n-qubit Pauli group $\mathcal{G}_n$ consisting of Pauli operators on n qubits $\boldsymbol{\mathcal{P}} = \mathcal{P}_1 \otimes \mathcal{P}_2 \otimes \cdots \otimes \mathcal{P}_n$ where $\mathcal{P}_i \in \{\boldsymbol{I}, \boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}\}$ are Pauli operators on the i-th qubit. Without risk of confusion, the tensor product $\otimes$ and the identity operator $\boldsymbol{I}$ can be omitted. The weight w of a Pauli operator $\boldsymbol{\mathcal{P}}$ is the number of non-identity components in the tensor product. To find a valid stabilizer code, it is crucial to find the stabilizer group $\mathcal{S}$, which is an Abelian subgroup of $\mathcal{G}_n$. This problem can be transferred into a classical coding problem using the mapping of Pauli errors to binary strings $\boldsymbol{\mathcal{P}} \mapsto (x_1 \cdots x_n \mid z_1 \cdots z_n) =: (\boldsymbol{p}_X \mid \boldsymbol{p}_Z)$ with $\mathcal{P}_i = \boldsymbol{X}^{x_i}\boldsymbol{Z}^{z_i}$. We can check that Pauli errors $\mathcal{A}$ and $\mathcal{B}$ in $\mathcal{G}_n$ commute if the symplectic product of their corresponding binary strings $(\boldsymbol{a}_X \mid \boldsymbol{a}_Z)$ and $(\boldsymbol{b}_X \mid \boldsymbol{b}_Z)$ is 0, i.e.,

$$\sum_{i=1}^{n} a_{X,i} \cdot b_{Z,i} + \sum_{i=1}^{n} a_{Z,i} \cdot b_{X,i} = 0, \tag{1}$$

where the addition is performed in the binary field (modulo 2). Therefore, a stabilizer code can be constructed from a $[2n, k]$ classical binary code given by its full rank parity check matrix (PCM) $\boldsymbol{H} = (\boldsymbol{H}_X \mid \boldsymbol{H}_Z)$ of size $m = 2n - k$ by $2n$ fulfilling the symplectic criterion:

$$\boldsymbol{H}_X\boldsymbol{H}_Z^{\mathsf{T}} + \boldsymbol{H}_Z\boldsymbol{H}_X^{\mathsf{T}} = \boldsymbol{0}. \tag{2}$$

Calderbank-Shor-Steane (CSS) codes are a special kind of stabilizer codes where

$$\boldsymbol{H} = \left(\begin{array}{c|c} \boldsymbol{H}'_X & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{H}'_Z \end{array}\right).$$

In this case, (2) holds if $\boldsymbol{H}'_X\boldsymbol{H}'^{\mathsf{T}}_Z = \boldsymbol{0}$ and the code construction can be simplified. QLDPC codes are defined as a family of stabilizer codes whose row and column weights are upper bounded by a relatively small constant independent of the block length, i.e., $\boldsymbol{H}$ is sparse. Most of the promising QLDPC codes constructed so far are CSS codes and all the QLDPC codes considered in this paper are CSS codes.

To jointly decode the four types of Pauli errors, it is convenient to consider the quaternary form of $\boldsymbol{H}$, denoted as $\boldsymbol{S} \in \text{GF}(4)^{m\times n}$, where GF(4) consists of the elements $\{0, 1, \omega, \bar{\omega}\}$. $\boldsymbol{H}$ can be converted to $\boldsymbol{S}$ using the mapping of binary vectors to vectors over GF(4) $(x_1 \cdots x_n \mid z_1 \cdots z_n) \mapsto (p_1 \cdots p_n) =: \boldsymbol{p}$ with $p_i = x_i\omega + z_i\bar{\omega}$. Then, we can check that for $\mathcal{A}$ and $\mathcal{B}$, mapped to vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ over GF(4), (1) is equivalent to $\langle \boldsymbol{a}, \boldsymbol{b}\rangle = \sum_{i=1}^{n}\langle a_i, b_i\rangle = 0$, where $\langle\cdot,\cdot\rangle$ denotes the *trace Hermitian inner product* over GF(4), which is defined as $\text{tr}(a \cdot \bar{b})$ where $\bar{b}$ is the conjugate of b and $a \cdot \bar{b}$ is the Hermitian inner product of a and b. The trace operator is computed as $\text{tr}(\omega) = \text{tr}(\bar{\omega}) = 1$ and $\text{tr}(0) = \text{tr}(1) = 0$.

The matrix $\boldsymbol{S}$ is called the *check matrix* and every row of $\boldsymbol{S}$ is called a *check*. The checks correspond to the m stabilizers

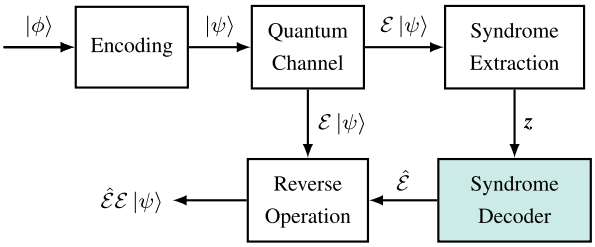


FIGURE 1. Block diagramm of QEC using QSCs.

which generate the stabilizer group $\mathcal{S}$. An $[[n, k, d]]$ stabilizer code is a 2^k-dimensional subspace of the n-qubit Hilbert space $(\mathbb{C}^2)^{\otimes n}$ defined as the common +1 eigenspace of $\mathcal{S}$. Additionally, we define $\mathcal{N}(\mathcal{S})$ to be the normalizer of $\mathcal{S}$ and $\boldsymbol{S}^{\perp}$ to be the matrix containing the vectors corresponding to the $2n-m$ generators of $\mathcal{N}(\mathcal{S})$. The minimum distance d of a QSC is defined as the lowest error weight in $\mathcal{N}(\mathcal{S})\backslash\mathcal{S}$. A code is called *degenerate* if $\mathcal{S}$ contains a stabilizer of weight less than d. Moreover, a code is highly degenerate if d is much larger than the minimum weight of the stabilizers.

B. CODES USED IN THIS STUDY

Two families of codes are considered in this work to evaluate the proposed decoder, namely generalized bicycle (GB) codes [11] and toric codes [23].

GB codes are constructed by choosing $\boldsymbol{H}'_X = \begin{pmatrix}\boldsymbol{A} & \boldsymbol{B}\end{pmatrix}$ and $\boldsymbol{H}'_Z = \begin{pmatrix}\boldsymbol{B}^{\mathsf{T}} & \boldsymbol{A}^{\mathsf{T}}\end{pmatrix}$ where $\boldsymbol{A}$ and $\boldsymbol{B}$ are $\frac{n}{2} \times \frac{n}{2}$ square circulant matrices of rank $m/2$. For constructing the check matrix, $m/2$ linearly independent rows are chosen from both $\boldsymbol{A}$ or $\boldsymbol{B}$. Therefore, GB codes naturally possess an overcomplete set of checks.

An $[[n = 2d^2, k = 2, d]]$ toric code is a topological code constructed from a $d \times d$ lattice embedded on the surface of a torus. The qubits are placed on the edges of the lattice. The vertex and plaquette operators define the $\boldsymbol{X}$ and $\boldsymbol{Z}$ stabilizers, respectively. A toric code has d^2 $\boldsymbol{X}$ stabilizers of weight 4 from the vertex operators and d^2 $\boldsymbol{Z}$ stabilizers of weight 4 from the plaquette operators. Among them, one $\boldsymbol{X}$ stabilizer and one $\boldsymbol{Z}$ stabilizer are redundant.

The two code families are connected by the lifted product construction [4] while having some contrasting properties. The GB codes considered in this work usually have a larger minimal distance than the considered toric codes. However, the GB codes are not degenerate while the toric codes contain many low-weight stabilizers which lead to degenerate errors.

C. ERROR CORRECTION WITH STABILIZER CODES

The workflow of error correction with stabilizer codes is depicted in Fig. 1. To protect a logical quantum state $|\phi\rangle$ with k qubits, it is encoded to a state $|\psi\rangle$ of n qubits using a stabilizer code. The noise in the quantum memory can be modeled as a depolarizing channel with depolarizing probability ϵ. In this channel, the errors $\boldsymbol{X}, \boldsymbol{Z}$ and $\boldsymbol{Y}$ occur equally likely with probability $\frac{\epsilon}{3}$. For a potentially corrupted state, the syndrome vector $\boldsymbol{z}$ is extracted and sent to a decoder which aims to find the most probable error given the syndrome $\boldsymbol{z}$. The focus of this work is to improve the syndrome decoder, which is a BP4 decoder. The decoding result is used as a reverse operator and applied to the corrupted quantum state. This will hopefully recover the state $|\psi\rangle$. To be more precise: Let $\hat{\boldsymbol{e}}$ be the estimate of $\boldsymbol{e}$ by the decoder and $\hat{\boldsymbol{\mathcal{E}}}$ be its corresponding Pauli error. The decoder aims to find an $\hat{\boldsymbol{e}}$ yielding the same syndrome $\boldsymbol{z}$ and fulfilling $\hat{\boldsymbol{\mathcal{E}}}\boldsymbol{\mathcal{E}} \in \mathcal{S}$. The latter can be checked by

$$\langle(\boldsymbol{e} + \hat{\boldsymbol{e}}), \boldsymbol{S}_i^{\perp}\rangle = 0 \tag{3}$$

for every row $i \in \{1, 2, \ldots, 2n - m\}$ of $\boldsymbol{S}^{\perp}$.

From (3), we can see that when correcting errors using a QSC, four outcomes may happen:

- Type I success: the estimated error is exactly the error that occurred, i.e., $\boldsymbol{e} + \hat{\boldsymbol{e}} = \boldsymbol{0}$.
- Type II success (with degeneracy): the estimated error is not exactly the same as $\boldsymbol{e}$ but (3) holds.
- Type I failure (flagged): the BP decoder is not able to find an $\hat{\boldsymbol{e}}$ that matches the syndrome $\boldsymbol{z}$.
- Type II failure (unflagged): the syndrome matches but (3) does not hold. This will lead to an undetectable erroneous quantum state.

One way to improve the decoding of QSCs is by designing a decoder that maximizes the probability of type I success, following classical decoding principles. However, this approach often falls short in achieving satisfactory decoding performance for QSCs. To obtain a near-optimal decoder for quantum codes, it is crucial to exploit degeneracy and improve type I and II success rates jointly. This motivates the utilization of neural network (NN) decoders since there is no good decoding algorithm that systematically exploits degeneracy so far.

III. BELIEF PROPAGATION DECODER

BP decoding of QLDPC codes is performed on the *Tanner graph* associated with the code. The Tanner graph is a bipartite graph with two classes of vertices. The first class of vertices are the variable nodes (VNs), each corresponding to a code bit (or qubit) and thus to a column of the check matrix $\boldsymbol{S}$. The second class of vertices are the check nodes (CNs), each corresponding to a check and thus to a row of the check matrix. A VN v_i is connected to a CN c_j if the corresponding entry $S_{j,i} \neq 0$, where $S_{j,i} \in \mathrm{GF}(4)\backslash\{0\}$ is called the coefficient of the edge. For example, consider the $[[7, 1, 3]]$ quantum Bose–Chaudhuri–Hocquenghem (BCH) code with both $\boldsymbol{H}'_X$ and $\boldsymbol{H}'_Z$ being the PCM of a classical $[7, 4, 3]$ BCH code:

$$\boldsymbol{H}_{\mathrm{BCH}} = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}.$$

The check matrix $\boldsymbol{S} \in \mathrm{GF}(4)^{6\times 7}$ is obtained as

$$\boldsymbol{S} = \begin{pmatrix} \omega\boldsymbol{H}_{\mathrm{BCH}} \\ \bar{\omega}\boldsymbol{H}_{\mathrm{BCH}} \end{pmatrix}. \tag{4}$$

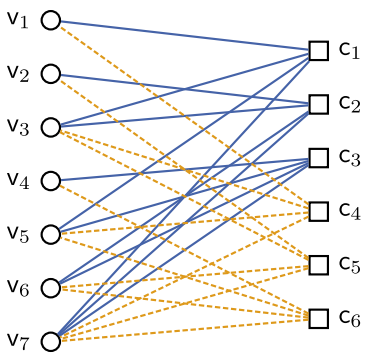


FIGURE 2. Tanner graph of the [[7, 1, 3]] quantum BCH code, with VNs represented by circles and CNs by squares. Blue solid edges represent coefficients ω (X type), while yellow dashed edges represent coefficients $\bar{\omega}$ (Z type).

The Tanner graph of this code is depicted in Fig. 2.

In this work, the syndrome decoder in Fig. 1 is the quaternary BP4 decoder [10], [18], [19], as binary decoding ignores the correlation between $\boldsymbol{X}$ and $\boldsymbol{Z}$ type of errors and is thus sub-optimal and has a higher error floor compared with a BP4 decoder, which takes the correlation between $\boldsymbol{X}$ and $\boldsymbol{Z}$ errors into account [10], [11]. The main disadvantage of the conventional BP4 decoder is the high complexity due to the passing of vector messages instead of scalar messages as in BP2. The problem is solved by the recently proposed refined BP4 decoder with scalar messages [9], [10].

A. REFINED BP4 DECODER

In this work, we use the log-domain refined BP4 decoder proposed in [10]. It exploits the fact that the syndrome of a stabilizer code is binary, indicating whether the error commutes with the stabilizer or not, enabling scalar message passing. Next, we briefly review the algorithm.

For every VN v_i, where $i \in \{1, 2, \ldots, n\}$, we initialize the log-likelihood ratio (LLR) vector $\Gamma_{i\to j}$ as $\Lambda_i = \left(\Lambda_i^{(1)}\ \Lambda_i^{(\omega)}\ \Lambda_i^{(\bar{\omega})}\right) \in \mathbb{R}^3$ with

$$\Lambda_i^{(\zeta)} = \ln\left(\frac{P(e_i = 0)}{P(e_i = \zeta)}\right) = \ln\left(\frac{1-\epsilon_0}{\frac{\epsilon_0}{3}}\right),$$

where $\zeta \in \mathrm{GF}(4)\backslash\{0\}$ and ϵ_0 is the estimated physical error probability of the channel. To exchange scalar messages, a *belief-quantization operator* $\lambda_\eta : \mathbb{R}^3 \to \mathbb{R}$ is defined as

$$\begin{aligned}\lambda_\eta(\Lambda_i) &= \ln\left(\frac{P(\langle e_i, \eta\rangle = 0)}{P(\langle e_i, \eta\rangle = 1)}\right)\\ &= \ln\left(\frac{1 + \mathrm{e}^{-\Lambda_i^{(\eta)}}}{\sum_{\zeta\neq 0,\zeta\neq\eta} \mathrm{e}^{-\Lambda_i^{(\zeta)}}}\right).\end{aligned}$$

The operator λ_η maps the LLR vector onto a scalar LLR of the binary random variable $\langle e_i, \eta\rangle$ where η runs over the nonzero entries of $\boldsymbol{S}$. The initial scalar VN messages are calculated as

$$\lambda_{i\to j} := \lambda_{S_{j,i}}(\Gamma_{i\to j}) \tag{5}$$

and are passed to the neighboring CNs where $i \to j$ denotes the message from VN v_i to CN c_j.

The outgoing messages of CN c_j, $j \in \{1, 2, \ldots, m\}$, are calculated using

$$\Delta_{i\leftarrow j} = (-1)^{z_j} \cdot \boxplus_{i'\in\mathcal{N}(j)\backslash\{i\}} \lambda_{i'\to j}, \tag{6}$$

where $\mathcal{N}(j)$ denotes the indices of the neighboring VNs of c_j and the $\boxplus$ operation is defined as

$$\boxplus_{i=1}^{I} x_i := 2\tanh^{-1}\left(\prod_{i=1}^{I}\tanh\frac{x_i}{2}\right).$$

At the VN update, we first calculate the LLR vector $\Gamma_{i\to j} = \left(\Gamma_{i\to j}^{(1)}\ \Gamma_{i\to j}^{(\omega)}\ \Gamma_{i\to j}^{(\bar{\omega})}\right)$ with

$$\Gamma_{i\to j}^{(\zeta)} = \Lambda_i^{(\zeta)} + \sum_{\substack{j'\in\mathcal{M}(i)\backslash\{j\},\\ \langle\zeta, S_{j',i}\rangle = 1}} \Delta_{i\leftarrow j'}, \tag{7}$$

for all $\zeta \in \mathrm{GF}(4)\backslash\{0\}$ with $\mathcal{M}(i)$ denoting the indices of the neighboring CNs of VN v_i. Then the outgoing messages $\lambda_{i\to j} = \lambda_{S_{j,i}}(\Gamma_{i\to j})$ are calculated and passed to the neighboring CNs.

To estimate the error, a hard decision is performed at the VNs by calculating Γ_i for $i \in \{1, 2, \ldots, n\}$ with

$$\Gamma_i^{(\zeta)} = \Lambda_i^{(\zeta)} + \sum_{\substack{j\in\mathcal{M}(i)\\ \langle\zeta, S_{j,i}\rangle = 1}} \Delta_{i\leftarrow j}, \tag{8}$$

for all $\zeta \in \mathrm{GF}(4)\backslash\{0\}$. If all $\Gamma_i^{(\zeta)} > 0$, then $\hat{e}_i = 0$, otherwise $\hat{e}_i = \mathrm{argmin}_\zeta \Gamma_i^{(\zeta)}$.

The iterative process is performed until the maximum number of iterations L is reached or the syndrome is matched.

B. INITIALIZATION OF THE BP DECODER

During our experiments, we noticed that often a certain fixed initialization of ϵ_0 tends to provide the best decoding performance for a wide range of physical error probabilities ϵ. This is overlooked in many previous works, where ϵ_0 is simply taken as the physical channel probability ϵ.

To explain this phenomenon, we look at the concept of a correctable error set proposed in [24] in the study of classical binary symmetric channels (BSCs). For a given QLDPC code with a fixed Tanner graph, the set of correctable errors of a BP decoder depends solely on the initialization value ϵ_0. Depending on the weight distribution of the correctable error set, the optimal choice of ϵ_0 is related to ϵ but it is often a distinct value [24].

To illustrate this point, we present in Fig. 3 the BP4 decoding results for the toric code with $d = 8$ and the GB codes A1 and A2 for different values of ϵ_0. The code parameters are given in Tab. 1 and the construction matrices for the GB codes are given in [11, Appendix B]. For the toric code (leftmost figure), no difference is seen for different ϵ_0. This is due to the very low success rate of plain BP decoding for toric codes, regardless of the chosen ϵ_0.

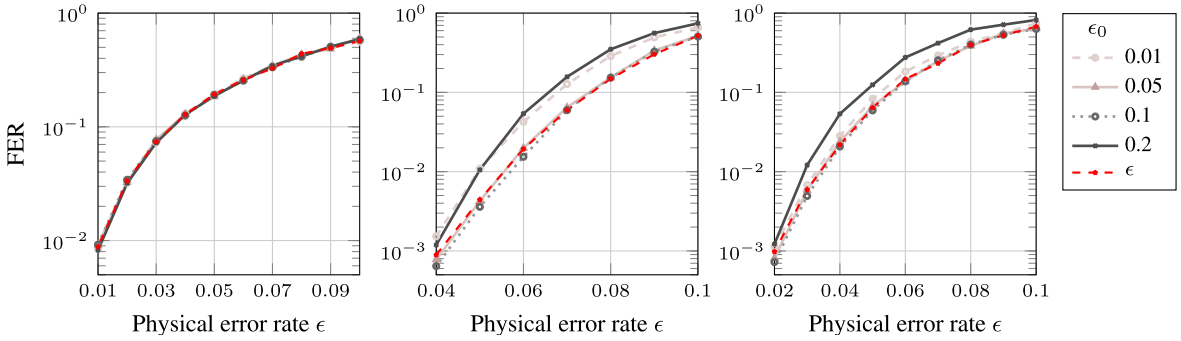


FIGURE 3. FER under different initial ϵ_0 values: Left figure: toric code ($d = 8$), middle figure: GB-A1 code, right figure: GB-A2 code.

TABLE 1. Parameters of the codes used to evaluate the proposed decoding algorithms. Parameter w_c is the CN degree of the original check matrix and m_{oc} is the number of rows of the overcomplete check matrix. The optimal values of ϵ_0 for OBP4 and (ϵ_0, w_r) for OBP4-opt used to obtain the results in Fig. 8 are listed on the right side.

Code		n	k	d	w_c	m_{oc}	ϵ_0 (OBP4)	(ϵ_0, w_r) (OBP4-opt)
GB	A1	254	28	14-20	10	254	0.1	-
	A2	126	28	8	10	126	0.1	-
	A3	48	6	8	8	2000	0.3	-
	A4	46	2	9	8	800	0.3	-
toric	$d = 4$	32	2	4	4	96	0.45	-
	$d = 6$	72	2	6	4	216	0.48	(0.35, 0.1)
	$d = 8$	128	2	8	4	384	0.49	(0.37, 0.1)
	$d = 10$	200	2	10	4	600	0.48	(0.45, 0.15)

However, when we examine the decoding results for the GB A1 code (middle figure) and A2 code (rightmost figure), it becomes evident that different ϵ_0 values yield distinct decoding performances and choosing $\epsilon_0 = \epsilon$ leads to sub-optimal decoding performance. Similar results are observed for codes with other parameters as well. Therefore, to determine a good decoder configuration, a line search or another optimization method should be conducted to identify the best ϵ_0 value.

Furthermore, in what follows, the significance of using different initial ϵ_0 values becomes more pronounced when redundant checks are introduced.

IV. OVERCOMPLETE CHECK MATRICES

Overcomplete check matrices refer to check matrices with redundant rows in addition to the original full-rank check matrix. BP4 can be performed on the Tanner graph associated with the overcomplete check matrices. We call this decoder *BP4 using overcomplete check matrix (OBP4)*. In this section, we give two possible approaches to construct overcomplete check matrices and then investigate the reason for performance improvement from OBP4.

A. CONSTRUCTING REDUNDANT CHECKS

To construct redundant checks, we treat $\boldsymbol{H}_X$ and $\boldsymbol{H}_Z$ as two separate binary matrices. Then, constructing overcomplete check matrices is identical to finding a large number of low-weight stabilizers. Two approaches are used in this work. The first one is based on an exhaustive search, which is generally intractable. However, it is possible to use probabilistic approaches [25] to increase the chance of finding low-weight stabilizers in the exhaustive search. This approach works well for the short-length codes considered in this work.

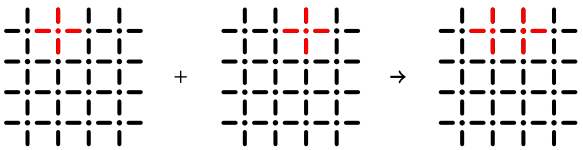


FIGURE 4. Example of the construction of a weight-6 stabilizer for the toric code with $d = 4$. The red edges are the qubits which support an X stabilizer.

The second approach relies on the topological structure of topological codes. For example, as depicted in Fig. 4, combining two neighboring stabilizers in a toric code yields a new weight-6 stabilizer. This results from each toric code stabilizer being supported by four qubits, with adjacent stabilizers overlapping by one qubit. Consequently, we can construct $2n$ redundant weight-6 $\boldsymbol{X}$ stabilizers and $2n$ redundant weight-6 $\boldsymbol{Z}$ stabilizers for every toric code. Combined with the n weight-4 stabilizers, we obtain an overcomplete check matrix of size $3n \times n$ for toric codes, which is used in this work.

The value of the redundant syndromes can be obtained without additional syndrome extraction operations. Let $\boldsymbol{H}$ be either $\boldsymbol{H}_X$ or $\boldsymbol{H}_Z$. The PCM $\boldsymbol{H}_{oc}$ with redundant rows is obtained by $\boldsymbol{H}_{oc} = \boldsymbol{M}\boldsymbol{H}$ with $\boldsymbol{M}$ being a binary matrix of size

$m_{\text{oc}} \times m$. The original syndrome is calculated as $\boldsymbol{H}\boldsymbol{e}^{\mathsf{T}} = \boldsymbol{z}$. Then, the new syndrome associated with $\boldsymbol{H}_{\text{oc}}$ is given by

$$\boldsymbol{z}_{\text{oc}} = \boldsymbol{H}_{\text{oc}}\boldsymbol{e}^{\mathsf{T}} = \boldsymbol{M}\boldsymbol{H}\boldsymbol{e}^{\mathsf{T}} = \boldsymbol{M}\boldsymbol{z},$$

being a linear mapping of $\boldsymbol{z}$ represented by $\boldsymbol{M}$.

B. HEURISTIC ANALYSIS

In this section, we present the heuristics which inspired our use of overcomplete check matrices.

The idea of performing BP decoding over an overcomplete parity-check matrix has been investigated in [26], [27], [28], [29], [30], and [31] for classical linear codes. One of the initial motivations was to perform more node updates in parallel to reduce the effect of short cycles. It is interesting that this method is able to improve the decoding performance of QLDPC codes significantly, both in the case of a limited as well as an unlimited number of iterations. This differs from decoding classical linear codes where using overcomplete check matrices only improves convergence speed and does not show obvious gains when a large number of iterations is used.

For QLDPC codes, this approach resembles matrix augmentation [13], where a fraction of the rows of the check matrix are duplicated. The advantage is that the messages associated with the duplicated check node are magnified which helps in breaking the symmetry during decoding and leading to a (hopefully) correct error estimate [13].

We demonstrate an extra benefit of the proposed method with a toy example of the $[[7, 1, 3]]$ quantum BCH code described in Sec. III, whose check matrix is given in (4).

Consider the error $\boldsymbol{\mathcal{E}} = \boldsymbol{Y}_7$. The syndrome $\boldsymbol{z}$ of $\boldsymbol{e}$ is

$$\boldsymbol{z} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}.$$

Assume that we initialize the decoder with $\epsilon_0 = 0.1$. According to (5), all the initial VN messages are 2.64. Then, in the first CN update, all CN messages are -1.55 according to (6). Based on these messages, in the next hard decision step, the decoder estimates the error as $\hat{\boldsymbol{\mathcal{E}}} = \boldsymbol{Y}_3\boldsymbol{Y}_5\boldsymbol{Y}_6\boldsymbol{Y}_7$ which produces the same syndrome as $\boldsymbol{z}$. However, (3) does not hold and we end up with an unflagged error.

In this example, the decoder wrongly estimates the error because all CNs have a syndrome value of 1. This is a strong indication that the error has a high weight. Except for VNs v_1, v_2, and v_4 with degree 2, all VNs that are connected to more than 2 CNs are erroneously estimated in the first hard-decision step. Duplicating some rows of the check matrix does not help in this case.[1]

Now we use the overcomplete $\boldsymbol{H}_X$ and $\boldsymbol{H}_Z$ with 7 rows, i.e., we take all the linear combinations of the 3 rows of the original $\boldsymbol{H}_{\text{BCH}}$ except the trivial case (all-zero vector). The new syndrome is

$$\boldsymbol{z}_{\text{oc}} = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}.$$

[1]However, note that this problem could be solved by initializing the decoder with a very small ϵ_0 such as 0.001 at the cost of degrading the overall decoding performance (depicted in Fig. 5), as in most cases, the decoder has a tendency towards trivial errors.

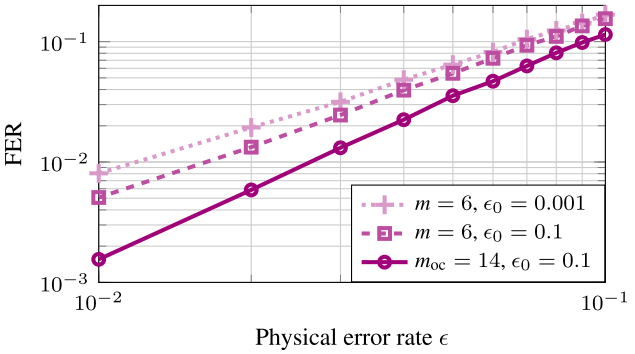


FIGURE 5. BP decoding results for the [[7,1,3]] CSS code with original and overcomplete check matrix ($L = 32$).

Although $\boldsymbol{z}_{\text{oc}}$ contains the same amount of information as $\boldsymbol{z}$ with respect to the coset where the error $\boldsymbol{\mathcal{E}}$ is located, the former is easier for the decoder to interpret. In the first hard-decision step, the error is correctly estimated. Figure 5 depicts the frame error rate (FER) curves corresponding to decoding using the overcomplete check matrix and the original check matrix for different ϵ_0.

V. NEURAL BELIEF PROPAGATION

NBP has been shown to be effective in improving the decoding performance of classical linear codes without increasing the decoding latency, see, e.g., [26], [32], and [33]. Using NBP to enhance the decoding of quantum stabilizer codes has been investigated in [20] and [34] for the BP2 decoder, which decodes $\boldsymbol{X}$ and $\boldsymbol{Z}$ errors separately. In this section, we propose to extend the NBP concept to the BP4 decoder with an adapted loss function tailored to exploit degeneracy. The proposed NBP model can be applied to both the original check matrix as well as the overcomplete check matrix, resulting in an *neural BP4 (NBP4)* decoder and an *NBP4 using an overcomplete check matrix (NOBP4)* decoder, respectively.

A. FRAMEWORK

An NBP decoder is an NN decoder obtained by unfolding a BP decoder over the decoding iterations [32]. The BP decoder can be based on the original Tanner graph of the QLDPC code or the Tanner graph associated with an overcomplete check matrix. The VNs and CNs are interpreted as neurons of an NN and the activation functions of the NN describe the node update equations. Trainable weights are added to the messages passed along the edges of the graph, i.e., the update rules (6) and (7) become

$$\Delta_{i\leftarrow j} = (-1)^{z_j} \cdot \boxplus_{i'\in\mathcal{N}(j)\setminus\{i\}} w^{(\ell)}_{\mathsf{v},i',j} \cdot \lambda_{i'\to j} \tag{9}$$

and

$$\Gamma^{(\zeta)}_{i\to j} = w^{(\ell)}_{\mathsf{v},i}\Lambda^{(\zeta)}_i + \sum_{\substack{j'\in\mathcal{M}(i)\setminus\{j\},\\ \langle\zeta,S_{j'i}\rangle=1}} w^{(\ell)}_{\mathsf{c},i,j'} \cdot \Delta_{i\leftarrow j'}, \tag{10}$$

where ℓ is the decoding iteration index and $w^{(\ell)}_{\mathsf{V},i',j}$, $w^{(\ell)}_{\mathsf{C},i,j'}$ and $w^{(\ell)}_{\mathsf{V},i}$ denote the real-valued trainable weights applied to the VN messages, CN messages, and channel LLR values, respectively.

B. LOSS FUNCTION

In [20], a loss function for BP2 decoding has been proposed which takes degeneracy into account. We extend this loss function to the BP4 case. Using Γ_i calculated by (8) in the hard-decision step, we compute

$$P(\langle \hat{e}_i, \eta\rangle = 1|\boldsymbol{z}) = \left(1 + \mathrm{e}^{-\lambda_\eta(\Gamma_i)}\right)^{-1} \tag{11}$$

for $\eta \in \mathrm{GF}(4)\backslash\{0\}$, denoting the estimated probability of the i-th error commuting with $\boldsymbol{X}$, $\boldsymbol{Z}$, and $\boldsymbol{Y}$, respectively. Then, the proposed per-error pattern loss function can be written as

$$\mathcal{L}(\Gamma; \boldsymbol{e}) = \sum_{j=1}^{2n-m} f\left(\sum_{i=1}^{n} P\left(\langle e_i + \hat{e}_i, S^{\perp}_{j,i}\rangle = 1|\boldsymbol{z}\right)\right) \tag{12}$$

where $f(x) = |\sin(\pi x/2)|$, which approaches 0 with x approaching any even number (i.e, it realizes a "soft modulo 2" function) [20]. The loss value is summed up over all rows $\boldsymbol{S}^{\perp}_j$ of $\boldsymbol{S}^{\perp}$. For each row j, we sum up the values $P\left(\langle e_i + \hat{e}_i, S^{\perp}_{j,i}\rangle = 1|\boldsymbol{z}\right)$ for all the elements $S^{\perp}_{j,i}$ in $\boldsymbol{S}^{\perp}_j$, representing the probability of $S^{\perp}_{j,i}$ being unsatisfied after estimating e_i as $\hat{e}_i$. It can be calculated as $P\left(\langle \hat{e}_i, S^{\perp}_{j,i}\rangle = 1 + \langle e_i, S^{\perp}_{j,i}\rangle|\boldsymbol{z}\right)$. If $\langle e_i, S^{\perp}_{j,i}\rangle = 0$, we directly calculate $P\left(\langle \hat{e}_i, S^{\perp}_{j,i}\rangle = 1|\boldsymbol{z}\right)$ using (11), with η being $S^{\perp}_{j,i}$. In contrast, when $\langle e_i, S^{\perp}_{j,i}\rangle = 1$, we calculate $P\left(\langle \hat{e}_i, S^{\perp}_{j,i}\rangle = 0|\boldsymbol{z}\right) = 1 - P\left(\langle \hat{e}_i, S^{\perp}_{j,i}\rangle = 1|\boldsymbol{z}\right)$. The loss function (12) is minimized if (3) holds.

C. INITIAL WEIGHTS OF THE REDUNDANT CHECKS

When a large number of redundant CNs is used, it becomes crucial to properly normalize the messages as the dependency between the messages becomes severe during decoding. Therefore, sometimes using a relatively large ϵ_0 does not provide a sufficient degree of freedom for the normalization of the messages. Hence, we introduce another parameter w_{r}, which is used as a normalization factor for the check node messages, similarly to the $w^{(\ell)}_{\mathsf{C},i,j}$ in (10), but independent of the number of iterations ℓ and of the index of the connected CN j and VN i, i.e.,

$$\Gamma^{(\zeta)}_{i\to j} = \Lambda^{(\zeta)}_i + \sum_{\substack{j'\in\mathcal{M}(i)\backslash\{j\},\\ \langle\zeta, S_{j'i}\rangle = 1}} w_{\mathrm{r}} \cdot \Delta_{i\leftarrow j'}.$$

It can be used in the plain OBP4 decoder or as the initial weight of $w^{(\ell)}_{\mathsf{C},i,j'}$ for training an NOBP4 decoder.

When optimizing the decoder configuration, we simply set $w_{\mathrm{r}} = 1$ if optimizing ϵ_0 alone yields satisfactory decoding results, as it results in the best convergence speed. If this is not the case, a grid search is conducted to find the optimal pair $(\epsilon_0, w_{\mathrm{r}})$. This decoder is referred to as OBP4-opt decoder. In Appendix, details about the grid search for the optimal initial value pair $(\epsilon_0, w_{\mathrm{r}})$ are given. Table 1 summarizes the parameters of the exemplary codes and the corresponding decoder configurations used in this work.

TABLE 2. A summary of the trainable parameters.

parameter	meaning
$w^{(\ell)}_{\mathsf{V},i,j}$	weight applied on message from v_i to c_j in the ℓ-th iteration
$w^{(\ell)}_{\mathsf{C},j,i}$	weight applied on message from c_j to v_i in the ℓ-th iteration
$w^{(\ell)}_{\mathsf{V},i}$	weight applied on the channel LLR in the ℓ-th iteration

TABLE 3. A summary of the hyper-parameters.

model	batch size	batch number	sampling error rate ϵ	learning rate
NBP4	120	2000	$\{0.02, 0.03, \ldots, 0.07\}$	1 to 0.1 with linear scheduler
NOBP4	120	200	$\{0.06, 0.07, \ldots, 0.11\}$	1 to 0.1 with linear scheduler

D. TRAINING

To train an NBP decoder, we employ the following procedure. First, we perform a line search to determine the optimal initial value of ϵ_0 for the initial non-trained decoder. The initial values for $w^{(\ell)}_{\mathsf{V},i,j}$ and $w^{(\ell)}_{\mathsf{V},i}$ at the beginning of the training are set to 1. If an overcomplete check matrix is used, we also search for the optimal initial weight w_{r} for the redundant checks jointly with ϵ_0. In this case, the initial value of $w^{(\ell)}_{\mathsf{C},i,j}$ will be w_{r}. Otherwise, when no redundant checks are used, the initial value of $w^{(\ell)}_{\mathsf{C},i,j}$ is 1. Subsequently, the training of the NN decoder is carried out using the proposed loss function (12) through plain stochastic gradient descent [35].

During training, mini-batches of 120 samples are used. Each mini-batch comprises six sets of 20 randomly generated error patterns from depolarizing channels with six distinct physical error probabilities ϵ. For training the NBP4 decoder, the set of ϵ is $\{0.02, 0.03, \ldots, 0.07\}$, while for training the NOBP4 decoder, we use $\{0.06, 0.07, \ldots, 0.11\}$. The loss function (12) is evaluated after each decoding iteration ℓ, resulting in a loss $\mathcal{L}^{(\ell)}$. The per-error pattern loss is determined as the minimum value across all iterations, i.e., as $\min\{\mathcal{L}^{(\ell)} : \ell \in \{1, 2, \ldots, L\}\}$. In the case where multiple iterations yield the same value, we select the earliest iteration. The gradients are clipped to 10^{-3} to avoid exploding gradients in the first decoding iterations where the message magnitude is small. Additionally, we gradually decrease the learning rate from 1 to 0.1 using a linear scheduler. For the NBP4 decoder, we train for 2000 batches, whereas for the NOBP4 decoder, we only train for 200 batches.

The trainable model parameters are summarized in Tab. 2 and the hyper-parameters for training are summarized in Tab. 3.

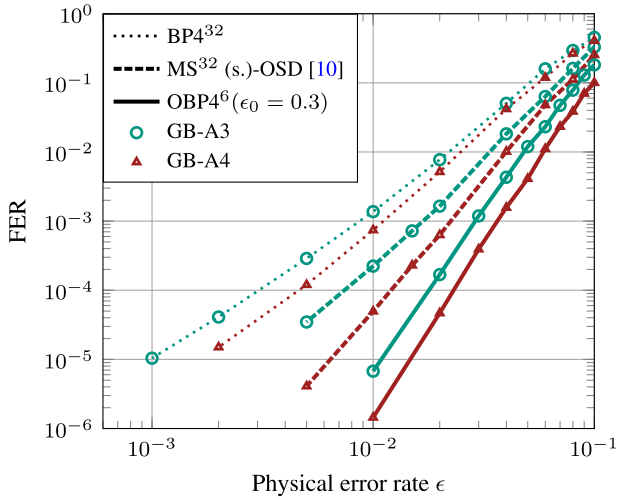


FIGURE 6. FER vs. depolarizing probability ϵ curves for the [[48, 6, 8]] GB-A3 code and [[46, 2, 9]] GB-A4 code. The reference curves are taken from [11], which use 32 iterations of serial (s.) normalized MS decoding concatenated with OSD-10 post-processing.

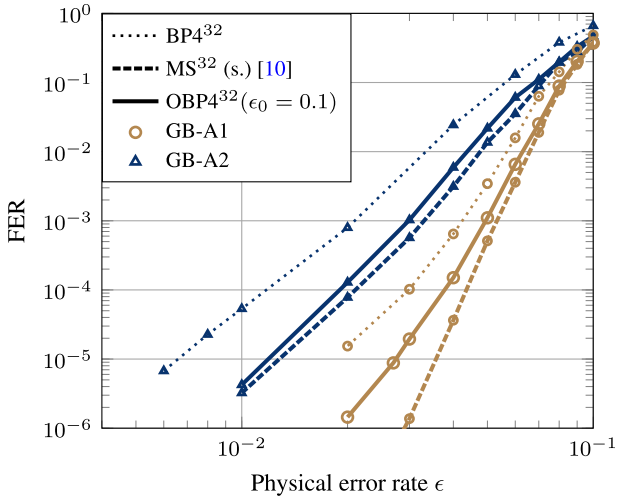


FIGURE 7. FER vs. depolarizing probability ϵ curves for the [[254, 28, d]] GB-A1 code and [[126, 28, 8]] GB-A2 code. The reference curves are taken from [11] where 32 iterations of serial normalized MS decoding is used.

E. COMPLEXITY

Firstly, the refined BP4 decoder has a computational complexity scaling linearly with the number of iterations and the number of edges in the Tanner graph, as one can see from (7) and (6). Compared to BP2 decoders, the source of complexity increase is the belief-quantization step which can be implemented with complexity $\mathcal{O}(1)$ per edge [10]. Therefore, the complexity of the BP4 decoders is only slightly higher than the complexity of the BP2 decoders.

The proposed NBP4 decoders are based on the refined BP4 decoder. As shown in (9) and (10), the proposed neural decoders require one additional multiplication with weights per edge compared to the refined BP4 decoder. Therefore, compared to conventional binary decoders, the complexity increase of the proposed NBP4 decoders is moderate.

Using an overcomplete check matrix increases the number of CNs by a factor of 1.5 to 10 compared to a full-rank matrix, based on the example codes presented. Additionally, while the computational complexity per CN remains unchanged, VNs require more additions. However, the number of required iterations is significantly reduced, often to half or even one-tenth, when using overcomplete matrices. Generally, we can decrease the maximum number of iterations as the number of redundant CNs increases, preserving similar overall complexity in both cases. Note that for QEC, the decrease in the number of iterations is advantageous, as low-latency decoding is crucial due to the limited coherent time of the qubits.

VI. NUMERICAL RESULTS

We assess the performance of our proposed novel decoder on the codes described in Sec. II using Monte Carlo simulations. Specifically, we use the GB A1-A4 codes and toric codes with $d \in \{4, 6, 8, 10\}$. The code parameters used are provided in Tab. 1. To ensure sufficiently accurate results, 300 frame errors are collected for each data point. Throughout this work, we use a flooding schedule where all VNs/CNs are updated in parallel in each decoding iteration. The initial ϵ_0 is set to 0.1 unless explicitly stated.

First, we investigate the effects of overcomplete check matrices in Sec. VI-A. Later, in Sec. VI-B, we show performance improvements due to NBP.

A. RESULTS WITH OVERCOMPLETE CHECK MATRICES

1) GB CODES

Figure 6 shows the decoding results using the original BP4 and the proposed OBP4 for the GB-A3 and GB-A4 codes. As described in Sec. IV-A, the overcomplete check matrices are constructed using the algorithm from [25]. For the GB-A3 code, we construct the overcomplete check matrix using 48 rows of weight 8 and 1952 rows of weight 12. No checks of weight 10 were found. For the GB-A4 code, we use 46 rows of weight 8 and 754 rows of weight 10. An improvement in post-decoding FER by more than an order of magnitude is observed using the OBP4 decoder with only 6 decoding iterations, especially when the physical error rate is small. The proposed OBP4 decoder also outperforms the reference decoder from [11], which uses 32 iterations of serial normalized MS decoding concatenated with OSD-10 post-processing.

During our experiments, we observe that the OBP4 decoder performs especially well if the code possesses a large number of low-weight checks which can be used to construct overcomplete check matrices. For codes with only a small number of redundant low-weight stabilizers, a decoding performance gain is still observed but the gain is smaller when the ratio between the number of redundant rows and the block length is smaller. For example, in Fig. 7, we show the decoding results for the GB-A1 and GB-A2 codes. For both codes, the overcomplete check matrices consist of n

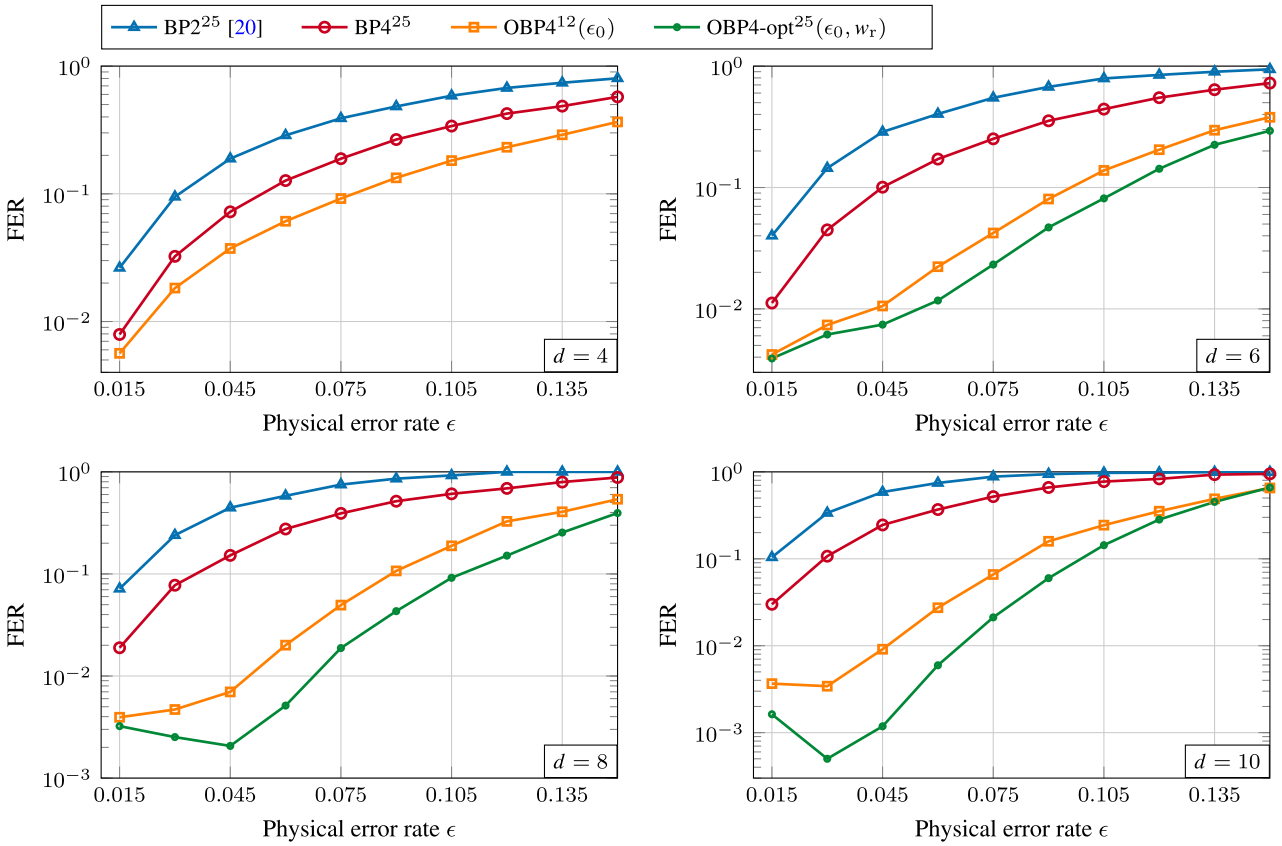


FIGURE 8. FER vs. depolarizing probability ϵ curves for the toric codes with $d \in \{4, 6, 8, 10\}$ using plain BP2 and BP4 decoders. The BP2 results are taken from [20]. The parameters used to produce the results for OBP4 and OBP4-opt are listed in Tab. 1.

rows of weight 10. No further redundant rows are added, as both codes do not possess any other low-weight check with a weight less than 16. For both codes, the OBP4 decoder outperforms the original BP4 decoding and has only a small gap to the reference results which uses normalized MS decoding with serial scheduling [11].

2) TORIC CODES

As depicted in Fig. 8, we also evaluate the decoding performance of BP4 and OBP4 decoding on toric codes with $d \in \{4, 6, 8, 10\}$. For comparison, we plot the BP2 decoding results taken from [20] which were evaluated over the XZ channel where the $\boldsymbol{X}$ and $\boldsymbol{Z}$ errors are assumed to happen independently with a probability ϵ_{b}. For fairness, we scale the results from [20] using $\epsilon_{\mathrm{b}} = \frac{2}{3}\epsilon$ as explained in [36].

We see that for plain BP decoding, BP4 performs consistently better than BP2 decoding. This highlights the advantage of using a quaternary decoder. The BP4 decoding result can be further improved by constructing an overcomplete check matrix of size $3n \times n$ using the topological structure of a toric code as described in Sec. IV-A. The performance improvement due to OBP4 is shown in Fig. 8 (orange curves). For toric codes with $d \geq 6$, we also use the OBP-opt decoder where a second initial parameter w_{r} is used (green curves). For $d = 4$, OBP-opt is not used as it does not show improvements compared to OBP. One can observe that the OBP4-opt decoder exhibits an error floor, especially for the codes with relatively large block lengths where the post-decoding error rate does not decrease with the decreasing channel error rate. This is caused by our choice of initial parameters to ensure a good performance after training. We observed during optimization that we need to focus on parameters that yield good performance for medium to high physical error rates. The decoder tends to converge to high-weight errors and thus performs worse for low-weight errors, which are dominant when the physical error rate is low. We optimize at this working point as the error floor can later be mitigated by NBP and we observe that this configuration yields the best overall decoding performance across different physical error rates.

B. RESULTS FOR NBP

We apply both NBP4 and NOBP4 decoders to toric codes and compare the decoding results with the neural BP2 decoder from [20] and the MWPM decoder. The numerical results of the MWPM decoder are obtained via the Pymatching library [37]. All results are plotted in Fig. 9.

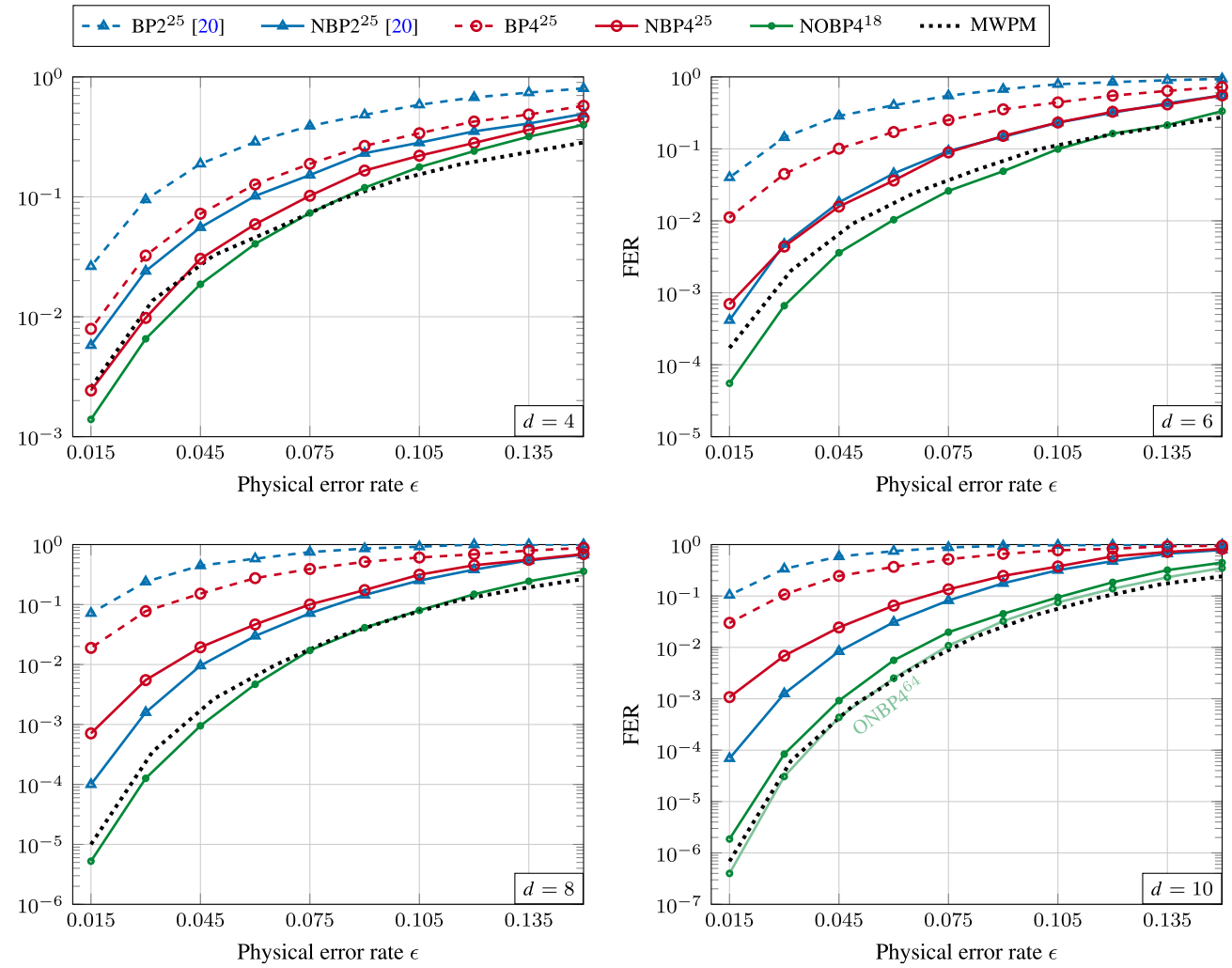


FIGURE 9. FER vs. depolarizing probability ϵ curves for the toric codes with $d \in \{4, 6, 8, 10\}$ using the trained NBP decoders.

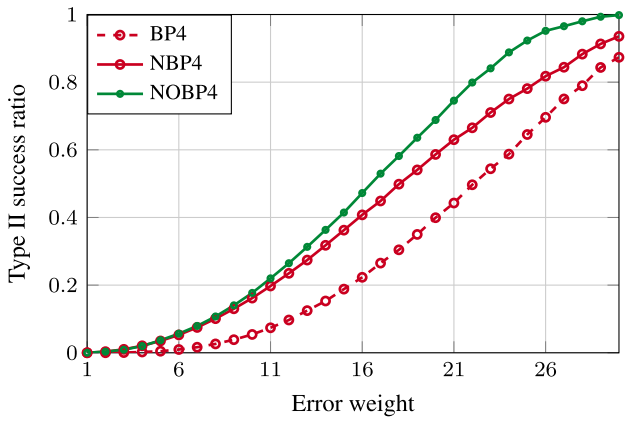


FIGURE 10. Fraction of type II success (success with degenerated errors) in relation to overall success plotted over the error weight for decoding the toric codes with $d = 10$.

1) FER RESULTS

We first compare the effect of NBP for both BP2 (red curves) and BP4 (blue curves) decoding without using any overcomplete check matrices. After training, both decoders improve the FER by orders of magnitude. This demonstrates the effectiveness of training. For toric codes with large d, the gain of NBP2 compared to BP2 is bigger than the gain of NBP4 compared to BP4, as NBP2 from [20] also implements other enhancements such as residual connections and soft weights. However, the performance of the NBP2 decoder is lower bounded by the MWPM decoder, which is considered to be near optimum for the XZ channel. Figure 9 shows that the decoding performance of the NOBP4^{18} decoder is better than the MWPM decoder except for $d = 10$. We observe that for codes with large block lengths, more BP iterations (e.g., 64 iterations) are needed to achieve a good decoding performance. This is shown with the light green curve.

Simulation results show that the NOBP4 decoder resolves both the error floor and convergence speed issues of the OBP4-opt decoder. Consequently, the NOBP4 decoder achieves better decoding performance and requires only 18 iterations, whereas the other decoders use 25 iterations.

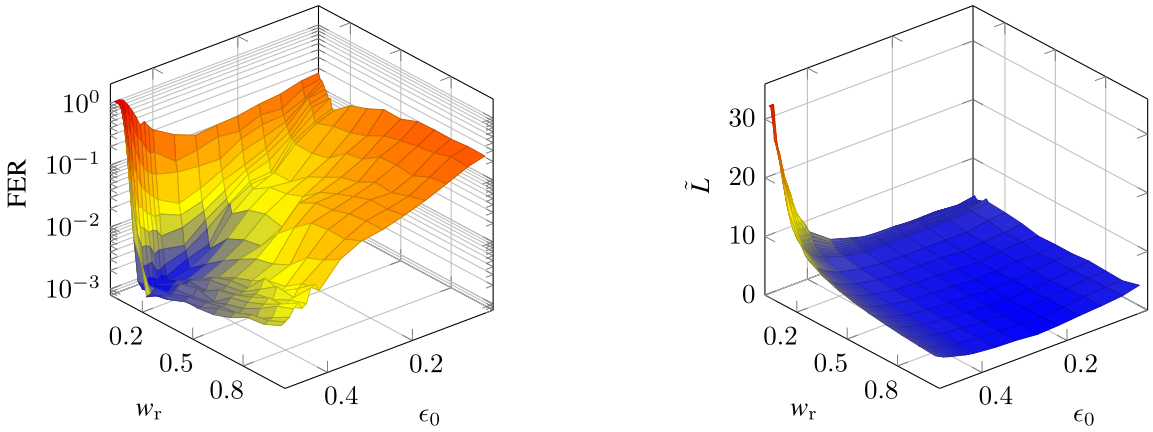


FIGURE 11. OBP4 decoding results for toric codes with $d = 10$ under different configurations of (ϵ_0, w_r) over a depolarizing channel with $\epsilon = 0.045$: Left figure shows the post decoding FER and the right figure shows the average number of BP decoding iterations until decoding success ($\tilde{L}$).

2) EXPLOITING DEGENERACY WITH NBP

In Fig. 10, we depict the fraction of type II decoding successes in relation to the total number of decoding successes when decoding error patterns of weights ranging from 1 to 30 occur. By employing both NBP4 and NOBP4 decoders, the fraction of successful decodings with degenerate errors (type II success) exhibits a significant increase compared to the original BP4 decoder. This highlights the effectiveness of training in exploiting degeneracy, which explains the substantial improvement when decoding toric codes with neural decoders. Toric codes, particularly those with large minimum distance d, possess many degenerate errors from their topological structure.

Furthermore, We observe that the NN is trained to exploit degeneracy from the weights $w^{(\ell)}_{\mathsf{c},i,j'}$. When trained solely with low-weight errors, we observe a clear trend where low-weight checks tend to have high message weights after training. This trend maximizes the type I success rate by increasing the weights on low-weight checks, which aligns with observations in decoding classical codes where low-degree check nodes are advantageous. However, when trained with mixed errors of both low and high weights, this tendency diminishes. After training, it becomes sometimes more beneficial to converge to an error that differs from the actual error that occurred (type II success). This indicates that the NN learns the degeneracy structure of the code.

For GB codes, the large minimum distance d ensures that they almost always achieve a valid error estimate as long as the syndrome is matched. The fraction of type II errors is extremely small (typically below 10^{-3}). Decoding failures mainly arise from the inability of the BP decoder to find an error estimate that matches the syndrome. Hence, enhancing the convergence of BP with overcomplete check matrices leads to improved performance. However, since GB codes are not highly degenerate, the type II success rate for GB codes is also quite low (typically below 10^{-3}). Training the NN decoder slightly improves the type II success rate but also significantly increases the type II failure rate. Consequently, training the neural decoder does not yield further performance improvement for GB codes.

C. SUMMARY OF THE NUMERICAL RESULTS

The proposed enhancements of the BP decoder significantly improve the decoding performance for both GB codes and toric codes. However, the reasons for improvement are different. We observe that plain BP4 decoding performs significantly better on GB codes than on toric codes. Furthermore, the decoding performance for both GB and toric codes is further enhanced by constructing a suitable overcomplete check matrix. However, the training of NBP decoding for GB code is not as successful as for toric codes, and thus, we omit results.

VII. CONCLUSION AND OUTLOOK

In this paper, we investigated two enhancements for the BP decoder of QLDPC codes, as well as their combination. The OBP4 decoder offers significant improvements in decoding performance and convergence speed. In contrast, the proposed NBP4 decoder effectively exploits degeneracy, but its improvement through training alone is limited without good initial parameters. By applying NBP to OBP4 decoding, we are able to address both of these disadvantages, resulting in superior decoding performance.

The proposed decoder enhancements have several natural extensions that are not discussed in the scope of this paper. Decoding based on overcomplete check matrices can also be applied to other message-passing decoders such as the MS decoder, which will yield similar decoding gains as for the BP decoder. The proposed loss function can also be used to train other model-based neural decoders such as a neural offset min-sum (NOMS) decoder [38] and other optimized min-sum decoders with efficient hardware implementation [39], [40]. As the proposed decoder reduces decoding latency, post-processing techniques could also

be applied after BP decoding. Additionally, considering circuit-level noise is also important for future work.

The source code for this work is available at https://github.com/kit-cel/Quantum-Neural-BP4-demo. The trained model parameters are also included in the repository. The provided Jupyter notebook can perform training and evaluation of the NBP model for arbitrary CSS with a given PCM.

APPENDIX
GRID SEARCH FOR THE INITIAL PARAMETERS

Section III emphasizes the significance of the initial value ϵ_0 in BP decoding, particularly when overcomplete check matrices are employed. To achieve the best decoding performance with a specific decoder, we conduct a line search to find the ϵ_0 that minimizes the post-decoding FER. In some cases, optimizing over ϵ_0 alone does not provide enough flexibility for optimization. Then we search for the optimal CN weight w_r for redundant checks jointly with ϵ_0 via a grid search.

Figure 11 illustrates the post-decoding FER (left figure) and the average number of decoding iterations $\tilde{L}$ until success (right figure) for the toric code with $d = 10$. The left figure reveals a specific region where the decoder exhibits the best performance, characterized by a relatively large ϵ_0 value and a small w_r value. However, the right figure shows that the number of iterations required for the BP decoder to converge increases with larger ϵ_0 and smaller w_r. Consequently, when the decoding performance is similar, we choose values that enable faster convergence. For other toric codes, the search results are similar to the ones shown in Fig. 11 and are summarized in Tab. 1.

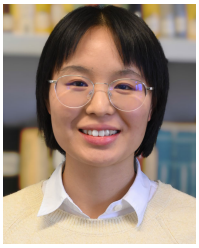
ACKNOWLEDGMENT

An earlier version of this paper was presented in part at the 2023 Information Theory Workshop (ITW) [DOI: 10.48550/arXiv.2308.08208].

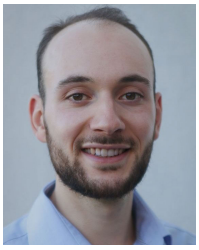
REFERENCES

[1] S. Miao, A. Schnerring, H. Li, and L. Schmalen, "Neural belief propagation decoding of quantum LDPC codes using overcomplete check matrices," in *Proc. IEEE Inf. Theory Workshop (ITW)*, Apr. 2023, pp. 215–220.
[2] D. Gottesman, "Fault-tolerant quantum computation with constant overhead," *Quantum Inf. Comput.*, vol. 14, pp. 1339–1371, Nov. 2014.
[3] J.-P. Tillich and G. Zémor, "Quantum LDPC codes with positive rate and minimum distance proportional to the square root of the blocklength," *IEEE Trans. Inf. Theory*, vol. 60, no. 2, pp. 1193–1202, Feb. 2014.
[4] P. Panteleev and G. Kalachev, "Quantum LDPC codes with almost linear minimum distance," *IEEE Trans. Inf. Theory*, vol. 68, no. 1, pp. 213–229, Jan. 2021.
[5] P. Panteleev and G. Kalachev, "Asymptotically good quantum and locally testable classical LDPC codes," in *Proc. 54th Annu. ACM SIGACT Symp. Theory Comput.*, Jun. 2022, pp. 375–388.
[6] N. P. Breuckmann and J. N. Eberhardt, "Balanced product quantum codes," *IEEE Trans. Inf. Theory*, vol. 67, no. 10, pp. 6653–6674, Oct. 2021.
[7] E. Dennis, A. Kitaev, A. Landahl, and J. Preskill, "Topological quantum memory," *J. Math. Phys.*, vol. 43, no. 9, pp. 4452–4505, Sep. 2002.
[8] J. Edmonds, "Paths, trees, and flowers," *Can. J. Math.*, vol. 17, pp. 449–467, Jan. 1965.
[9] K.-Y. Kuo and C.-Y. Lai, "Refined belief propagation decoding of sparse-graph quantum codes," *IEEE J. Sel. Areas Inf. Theory*, vol. 1, no. 2, pp. 487–498, Aug. 2020.
[10] C.-Y. Lai and K.-Y. Kuo, "Log-domain decoding of quantum LDPC codes over binary finite fields," *IEEE Trans. Quantum Eng.*, vol. 2, pp. 1–15, 2021.
[11] P. Panteleev and G. Kalachev, "Degenerate quantum LDPC codes with good finite length performance," *Quantum*, vol. 5, p. 585, Nov. 2021.
[12] N. Raveendran and B. Vasić, "Trapping sets of quantum LDPC codes," *Quantum*, vol. 5, p. 562, Oct. 2021.
[13] A. Rigby, J. C. Olivier, and P. Jarvis, "Modified belief propagation decoders for quantum low-density parity-check codes," *Phys. Rev. A, Gen. Phys.*, vol. 100, no. 1, Jul. 2019, Art. no. 012330.
[14] J. Roffe, D. R. White, S. Burton, and E. Campbell, "Decoding across the quantum low-density parity-check code landscape," *Phys. Rev. Res.*, vol. 2, no. 4, Dec. 2020, Art. no. 043423.
[15] D. Poulin and Y. Chung, "On the iterative decoding of sparse quantum codes," *Quantum Inf. Comput.*, vol. 8, no. 10, pp. 986–1000, Nov. 2008.
[16] Y.-J. Wang, B. C. Sanders, B.-M. Bai, and X.-M. Wang, "Enhanced feedback iterative decoding of sparse quantum codes," *IEEE Trans. Inf. Theory*, vol. 58, no. 2, pp. 1231–1241, Feb. 2012.
[17] J. D. Crest, M. Mhalla, and V. Savin, "Stabilizer inactivation for message-passing decoding of quantum LDPC codes," in *Proc. IEEE Inf. Theory Workshop (ITW)*, Nov. 2022, pp. 488–493.
[18] M. C. Davey and D. J. C. MacKay, "Low density parity check codes over GF(*q*)," in *Proc. Inf. Theory Workshop*, 1998, pp. 70–71.
[19] D. Declercq and M. Fossorier, "Decoding algorithms for nonbinary LDPC codes over GF(*q*)," *IEEE Trans. Commun.*, vol. 55, no. 4, pp. 633–643, Apr. 2007.
[20] Y.-H. Liu and D. Poulin, "Neural belief-propagation decoders for quantum error-correcting codes," *Phys. Rev. Lett.*, vol. 122, no. 20, May 2019, Art. no. 200501.
[21] D. Gottesman, "Stabilizer codes quantum error correction," Ph.D. thesis, Dept. Phys., Math. Astron., California Inst. Technol., Pasadena, CA, USA, 1997.
[22] A. R. Calderbank, E. M. Rains, P. W. Shor, and N. J. A. Sloane, "Quantum error correction via codes over GF(4)," *IEEE Trans. Inf. Theory*, vol. 44, no. 4, pp. 1369–1387, Jul. 1998.
[23] A. Kitaev, "Anyons in an exactly solved model and beyond," *Ann. Phys.*, vol. 321, no. 1, pp. 2–111, Jan. 2006.
[24] M. Hagiwara, M. P. C. Fossorier, and H. Imai, "Fixed initialization decoding of LDPC codes over a binary symmetric channel," *IEEE Trans. Inf. Theory*, vol. 58, no. 4, pp. 2321–2329, Apr. 2012.
[25] J. S. Leon, "A probabilistic algorithm for computing minimum weights of large error-correcting codes," *IEEE Trans. Inf. Theory*, vol. IT-34, no. 5, pp. 1354–1359, Sep. 1988.
[26] M. Lian, F. Carpi, C. Häger, and H. D. Pfister, "Learned belief-propagation decoding with simple scaling and SNR adaptation," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jul. 2019, pp. 161–165.
[27] T. Halford and K. Chugg, "Random redundant soft-in soft-out decoding of linear block codes," in *Proc. IEEE Int. Symp. Inf. Theory*, Jul. 2006, pp. 2230–2234.
[28] M. Bossert and F. Hergert, "Hard- and soft-decision decoding beyond the half minimum distance—An algorithm for linear codes (Corresp.)," *IEEE Trans. Inf. Theory*, vol. IT-32, no. 5, pp. 709–714, Sep. 1986.
[29] A. Kothiyal, O. Y. Takeshita, W. Jin, and M. Fossorier, "Iterative reliability-based decoding of linear block codes with adaptive belief propagation," *IEEE Commun. Lett.*, vol. 9, no. 12, pp. 1067–1069, Dec. 2005.
[30] J. Jiang and K. R. Narayanan, "Iterative soft-input soft-output decoding of Reed–Solomon codes by adapting the parity-check matrix," *IEEE Trans. Inf. Theory*, vol. 52, no. 8, pp. 3746–3756, Aug. 2006.
[31] A. Buchberger, C. Häger, H. D. Pfister, L. Schmalen, and A. Graell i Amat, "Pruning and quantizing neural belief propagation decoders," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 7, pp. 1957–1966, Jul. 2021.
[32] E. Nachmani, Y. Be'ery, and D. Burshtein, "Learning to decode linear codes using deep learning," in *Proc. 54th Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Sep. 2016, pp. 341–346.
[33] E. Nachmani, E. Marciano, L. Lugosch, W. J. Gross, D. Burshtein, and Y. Be'ery, "Deep learning methods for improved decoding of linear codes," *IEEE J. Sel. Topics Signal Process.*, vol. 12, no. 1, pp. 119–131, Feb. 2018.
[34] X. Xiao, "Neural-Net decoding of quantum LDPC codes with straight-through estimators," in *Proc. Inf. Theory Appl. Workshop (ITA)*, 2019, pp. 1–6.
[35] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[36] D. J. C. MacKay, G. Mitchison, and P. L. McFadden, "Sparse-graph codes for quantum error correction," *IEEE Trans. Inf. Theory*, vol. 50, no. 10, pp. 2315–2330, Oct. 2004.
[37] O. Higgott and C. Gidney, "Sparse blossom: Correcting a million errors per core second with minimum-weight matching," 2023, *arXiv:2303.15933*.
[38] L. Lugosch and W. J. Gross, "Neural offset min-sum decoding," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2017, pp. 1361–1365.
[39] Q. Lu, Z. Shen, C.-W. Sham, and F. C. M. Lau, "A parallel-routing network for reliability inferences of single-parity-check decoder," in *Proc. Int. Conf. Adv. Technol. Commun. (ATC)*, Oct. 2015, pp. 127–132.
[40] C.-W. Sham, X. Chen, W. M. Tam, Y. Zhao, and F. C. M. Lau, "A layered QC-LDPC decoder architecture for high speed communication system," in *Proc. IEEE Asia–Pacific Conf. Circuits Syst.*, Dec. 2012, pp. 475–478.



SISI MIAO (Student Member, IEEE) received the bachelor's degree in communication engineering from the Ocean University of China, in 2018, and the master's degree in infotech from the University of Stuttgart, in 2021. She is currently pursuing the Ph.D. degree with the Communications Engineering Laboratory, Karlsruhe Institute of Technology, focusing her research on classical and quantum error control coding.



ALEXANDER SCHNERRING was born in Heidelberg, Germany, in 1997. He received the B.Sc. and M.Sc. degrees in electrical engineering from Karlsruhe Institute of Technology, in 2019 and 2022, respectively. He is currently pursuing the Ph.D. degree with the Institute of Solar Research, German Aerospace Center (DLR), Almería, Spain, where he is investigating the calibration of CST plants using autonomous UAVs.

His research interests include signal processing, robotics, and computational geometry.

HAIZHENG LI (Graduate Student Member, IEEE) received the bachelor's degree in automation from Beijing Institute of Technology, in 2017, and the master's degree in ETIT from Karlsruhe Institute of Technology, in 2021, where he is currently pursuing the Ph.D. degree with the Communications Engineering Laboratory, focusing his research on spatially-coupled LDPC codes.

LAURENT SCHMALEN (Fellow, IEEE) received the Dipl.-Ing. degree in electrical engineering and information technology and the Dr.-Ing. degree from RWTH Aachen University, Germany.

From 2011 to 2019, he was with Alcatel-Lucent Bell Labs and Nokia Bell Labs. From 2014 to 2019, he was a Guest Lecturer with the University of Stuttgart, Stuttgart, Germany. Since 2019, he has been a Professor with Karlsruhe Institute of Technology, where he co-heads the Communications Engineering Laboratory. His research interests include channel coding, modulation formats, and optical communications.

Dr. Schmalen was a recipient and co-recipient of several awards, including the E-Plus Award for his Ph.D. thesis, the 2013 Best Student Paper Award from the IEEE Signal Processing Systems Workshop, and the 2016 and 2018 Journal of Lightwave Technology Best Paper Awards. He is an Associate Editor of IEEE TRANSACTIONS ON COMMUNICATIONS.