This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2026.3725587

IEEE Open Journal of the Communications Society

Received XX Month, XXXX; revised XX Month, XXXX; accepted XX Month, XXXX; Date of publication XX Month, XXXX; date of current version 11 January, 2024.

Digital Object Identifier 10.1109/OJCOMS.2024.011100

Largely Constrained Noise-guessing Decoding of Short LDPC codes

SAHAR ALLAHKARAM[1,2] (Student Member, IEEE), FRANCISCO A. MONTEIRO[1,2] (Member, IEEE), JOÃO RIBEIRO[1,3], AND CARLOS SALEMA[1,3] (Life Member, IEEE)

[1]Instituto de Telecomunicações, Portugal
[2]ISCTE – Instituto Universitário de Lisboa, Portugal
[3]Instituto Superior Técnico, Universidade de Lisboa, Portugal

Corresponding author: Francisco A. Monteiro (e-mail: francisco.monteiro@lx.it.pt).

This work is funded by national funds through FCT – Fundação para a Ciência e a Tecnologia, I.P., and, when eligible, co-funded by EU funds under project/support UID/50008/2025 – Instituto de Telecomunicações, with DOI identifier https://doi.org/10.54499/UID/50008/2025. João Ribeiro is also supported by FCT – Fundação para a Ciência e a Tecnologia, I.P., under project 2024.12595.CMU with DOI https://doi.org/10.54499/2024.12595.CMU, and by the European Union (LESYNCH, 101218842). Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the European Union or the European Research Council Executive Agency. Neither the European Union nor the granting authority can be held responsible for them. Sahar Allahkaram is funded by a merit scholarship from ISCTE – Instituto Universitário de Lisboa.

ABSTRACT When using guessing random additive noise decoding (GRAND) to decode linear block codes (LBCs), the list of error patterns to be tested is blind to any structure in the parity-check matrix. This paper starts by proposing simple and general set partitioning (SP) constraints that reduce the number of possible patterns to less than half when decoding random linear codes (RLCs). Sparser parity-check matrices induce faster decoding when using SP, but at the expense of a severe performance loss. This loss can be mostly recovered by considering regular or quasi-regular low-density parity-check (LDPC) codes. Given their structure, one can apply the proposed Sieving-GRAND that, for a given support of the syndrome, "filters" possible error positions. Subsequently, a guessing approach is applied to those few positions. The filtering threshold can be successively relaxed to extend the set of candidate positions. We further propose a largely constrained GRAND mechanism (LC-GRAND), which takes advantage of zeroed syndrome positions to efficiently eliminate incompatible error patterns, while not requiring quasi-regular LDPC codes. The two main proposed techniques lead to similar sharp pruning of the search space. For example, LC-GRAND, in combination with SP, reduces the number of membership tests to $\simeq 1\%$. All the proposed hard-decoding techniques retain the same decoding performance of unconstrained GRAND.

INDEX TERMS Constrained GRAND, LDPCs, RLCs, set partitioning, short codes, sieving error positions, URLLC.

I. INTRODUCTION

FOR error protection, binary messages with a length of k bits should be mapped to codewords n bits long, taken from a finite codebook $\mathcal{C}$, with a code rate $R = \frac{k}{n}$. Shannon showed that it is possible to communicate with an error probability that tends to zero when using a code with sufficiently long codewords ($n \to \infty$), as long as the code rate is below capacity [1]. When employing an error correction code (ECC), the asymptotic bit error probability (BER) decays as BER $\propto e^{-n\epsilon(R)}$ [2]. The capacity corresponds to the highest code rate for which $\epsilon(R) > 0$. The statistics of the error exponent of random codes have been very recently analyzed in [3] when $n \to \infty$, proving that it converges to its expected value for both high and low code rates. Shannon's non-constructive proof showed that *random* codes could reach capacity [1], [4], [5], however, a simple maximum-likelihood (ML) decoder does not exist for such codes [6]. Decades-long research looked for codes capable of achieving capacity with some embedded *structure*, for which a feasible decoder could exist [7]. The focus was on reaching capacity, and the length of the codewords was not a major constraint. Low-density parity-check (LDPC) codes, also known as Gallager codes [8], later rediscovered by MacKay and Neal (MN) [9], [10], reach capacity for extremely large codewords.

Short codes became relevant for ultra-reliable low-latency communications (URLLC) [11], [12] in 6G [13], [14], even

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2026.3725587

more than in 5G [15]. The capacity for the *finite* blocklength regime was found by Polyanskiy et al. [16], recovering Shannon's capacity when $n \to \infty$. The authors pointed out that very few prior works had looked at that problem, as considered by Salema [17, sec. 6.12-6.13], and by Dolinar et al. [18], who revisited Shannon's work considering finite codewords [2].

Ordered statistics decoding (OSD), which has similarities with the concept of noise-guessing decoding, can be used to decode any linear block code (LBC) with quasi-ML performance [19], [20], has been proposed for short codes [21], [22]. Although OSD involves costly gaussian elimination for general LBCs, a low-latency OSD has recently been proposed for the particular case of Bose-Chaudhuri-Hocquenghem (BCH) codes, which eliminated the need of gaussian elimination [23]. The concepts of noise-guessing decoding can be traced back to [24]. Using a constrained serial list Viterbi algorithm, a very low number of codewords needs to be tested [25], however, this comes at the expense of a Gaussian elimination preprocessing for each received codeword, with a complexity that scales with $\approx \mathcal{O}(n^3)$.

High-speed decoding of LDPC codes (with $n = 1027$) has been presented in [26], which includes a list of other fast decoders implemented for n between 672 and 1994 bits. With a focus on fast decoding, short protographs LDPC codes have been proposed for URLLC, unfortunately enforcing rather resource-intensive very low code rates (from $\frac{1}{3}$ to $\frac{1}{12}$) [12].

Guessing random additive noise decoding (GRAND) is a *universal* ML decoder for codes with moderate redundancy, i.e., with high rate. This includes short or high-rate random linear codes (RLCs), which attain the maximum possible rates of the finite blocklength regime [27]. It should be noted that, while OSD appeared as a solution to decode short block codes (including LDPCs) with ML decoding performance, OSD has limited decoding throughput and is not adequate to meet the low-latency decoding aimed for 6G [28]. In addition to the Gaussian elimination required for each received codeword, OSD operates by flipping the k most reliable bits of the n-bits received words, its complexity exponentially increases due to the binomial coefficient $\binom{n}{k}$. For that reason, OSD is only a low-complexity viable alternative when low-rate codes are considered. For the high-rate codes considered in this work (for higher spectral efficiency), OSD is not competitive with noise-guessing approaches regarding decoding latency.

GRAND-type decoders [29] were originally proposed by Duffy et al. as a hard ML decoder [11], [27], [30]. The concept shifted the decoding paradigm from finding the transmitted codeword to finding the error pattern that corrupted that codeword. This is done by successively testing error patterns in decreasing order of likelihood. In the case of RLCs, as for any LBC, the membership test depends entirely on the observed syndrome [4], [29]. GRAND has been extended to soft-decoding [31], [32] and GRAND with joint detection has been analyzed in [33]. GRAND has been applied to fading channels in single-antenna [34] and multi-antenna systems [35] [36]. GRAND has been implemented in hardware [37], [38], showing record energy-efficient decoding [39]. The noise-guessing concept has also been applied to network coding [40], to quantum ECCs [41] [42], [43], and even to the purification of quantum links [44]. With GRAND, error-*detecting* codes, such as CRCs [45], or even the AES cryptosystem [46], can be turned into an ECC. GRAND can also compromise physical-layer security [47].

Although the statistics of the error patterns are central in determining the *order* in which they should be tested [11], [31], one should also restrict the *number* of possible error patterns that are worth testing. This was done in [48] by taking into account the modulation symbols. One can also take advantage of the parity-check matrix, $\mathbf{H}$. Given a syndrome, not all error patterns are compatible with the observed syndrome, which can greatly reduce the list of possible error patterns $\mathcal{L}$. Other recent proposals have appeared to reduce the decoding complexity of GRAND-type algorithms while preserving their near-ML decoding performance. In [49], the authors make use of a single parity-check constraint to eliminate guesswork in GRAND-based decoding, demonstrating that decoding complexity can be halved without compromising performance, particularly at lower signal-to-noise ratios (SNRs). Taking into account $\mathbf{H}$ and the observed syndrome, some constrained versions of GRAND have been proposed, achieving ML performance and requiring many fewer membership tests. In [50], constraints associated with the rows of $\mathbf{H}$ are used: for each considered row, the average number of tests is halved. However, processing those constraints becomes an extra burden, and the authors recommend using a single row. The work in [51] proposes splitting $\mathbf{H}$ in two matrices: one is used to generate low-weight error patterns via a (partial) trellis search, and the other is used to check whether the candidate error patterns can generate the remaining partial syndrome. In [52] the idea is to start with a list of partial error patterns ordered in decreasing order of likelihood, as defined by the soft metrics of the bits, and complete the remaining error position by testing which position, associated with a column of $\mathbf{H}$, is compatible with the observed syndrome. A unified analytical framework was introduced in [53] to distinguish between guessing noise decoding (GND) and guessing codeword decoding (GCD) algorithms for linear block codes. In this work, GND, representing the family of GRAND-type decoders, was contrasted with GCD approaches, such as OSD, which explicitly exploit the algebraic structure of the code. Both decoding paradigms were shown to achieve ML decoding performance; however, GCD demonstrated greater efficiency in terms of the number of required guesses, particularly for low-rate or highly redundant codes. In [54], [55], a segmented ORBGRAND method was proposed, in which the noise-guessing search space is partitioned into syndrome-consistent sub-patterns and recombined using logistic-weighted compositions. This segmentation reduces redundancy, cutting the average number of decoding queries

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2026.3725587

by over 66% across all SNR while maintaining near-ML accuracy. In [56], a soft-output GRAND (SOGRAND) decoder was introduced to extend the GRAND framework to soft-information processing. This method generates accurate a posteriori reliability estimates for both single and list decodings, supporting iterative architectures that outperform standardized LDPC codes from 5G New Radio in reliability and latency.

Collectively, these works mark an important evolution of the GRAND family from purely combinatorial noise-guessing toward adaptive, probabilistic, and structure-aware decoding methods. A recent overview and comparison between GRAND, OSD, and GCD is provided in [57], where the authors point out in which regions of coding rate and block length each decoding approach is preferable. Building upon these foundations, the present work proposes a largely constrained GRAND framework that makes use of two newly proposed techniques specifically designed for LDPCs, and later combined with one general purpose complexity-reduction technique. Our proposals introduce parity-check–driven structural constraints into the noise-guessing process. One of our main proposals leverages the inherent sparsity and quasi-regularity of LDPC parity-check matrices to perform syndrome-guided sieving of candidate error positions. The other main proposal looks at the zero-syndrome "filtering", thereby eliminating incompatible error patterns. In doing that, one achieves ML decoding performance for *any* short LDPC code with significantly reduced computational overhead and latency, extending the practicality of GRAND-type decoders for URLLC scenarios.

Note that the proposed decoders are distance-bounded decoders, which do not try to decode beyond the number of errors t dictated by the minimum Hamming distance, $d_{\min}$. Distance-bounded decoders do not reach capacity, with $n \to \infty$, as to reach capacity one has to be able to decode errors beyond $t = \frac{d_{\min}-1}{2}$ [10] [4, Ch.13]. However, in the finite blocklength regime imposed in URLLC, this distinction has no practical meaning.

A. Contributions:

Belief-propagation (BP) decoders are known to offer a low-complexity solution suitable for long block lengths, with a complexity that grows linearly with n. However, in the case of short LDPCs, BP falls short from the ML decoding performance [13], [14], and near-ML decoding of short LDPCs has been classified as an open problem in [58]. Noise-guessing decoders are seen as an efficient decoding approach to that goal [13], however none of them takes advantage of the sparsity of the LDPCs. The decoder in [50], which introduced the idea of "pre-filtering" error patterns by applying one or more constraints associated with one or more rows (or combinations of rows) of the parity-check matrix is a state-of-the-art decoding technique that fits well the hard-decoding of sparse codes and will be compared with our proposals. This paper focuses on low-complexity noise-guessing hard-decoders fit for quasi-regular LDPC codes, while some of the proposed techniques are not restricted to LDPCs. The contributions of this work are the following:

1) Short RLCs with GRAND have shown faster decoding than CRC-aided polar (CA-Polar) codes of similar length or than standalone polar codes using the Tal-Vardy list decoder [31], [35], [59]. This paper starts by proposing in Section III simple constraints based on a set partitioning (SP) of the positions of the errors, which can be applied to *any* LBC, and is tested with RLCs. The existence of a very unbalanced number of ones and zeros in some row of $\mathbf{H}$ determines the reduction of complexity. Sparse RLCs would lead to faster decoding, however, at the expense of the codes' performance. By considering short LDPC codes, one can retain a good performance.

2) To mitigate the performance degradation observed with sparse random codes, we use short quasi-regular low-density parity-check (LDPC) codes that maintain high reliability while enabling efficient constrained decoding. By relaxing the strict overlap constraints of Gallager and MacKay–Neal constructions from one to two overlaps allowed between any two columns [8]–[10], the proposed LDPC matrices retain uniformly weighted columns suitable for rapid error-pattern filtering in GRAND decoders that is crucial for the Sieving-GRAND technique proposed in Section IV. This is a syndrome-guided "sieving mechanism" that uses the structure of the LDPC matrix to iteratively identify and prioritize likely error positions. This filtering process dynamically enlarges the candidate set of positions in a controlled manner, allowing the decoder to find the ML codeword solution with minimal computational effort.

3) Next, in Section VI we develop the largely constrained GRAND (LC-GRAND) algorithm, which introduces a novel constraint based on zero-valued syndrome components. This additional layer of structural restriction enables the early elimination of incompatible error patterns, providing sharper search-space pruning and faster convergence toward the correct error pattern. Likewise SP, this technique is applicable to both regular and irregular LDPCs. Without loss of generality, for a fair performance comparison, LC-GRAND is assessed with the quasi-regular LDPC code used to assess Sieving-GRAND.

4) Finally, by applying SP to the reduced candidate positions obtained from the Sieving-GRAND and LC-GRAND stages one obtains the most efficient decoding strategy, as shown in the results in Section VII. The integration achieves maximum-likelihood performance while reducing the number of membership tests to approximately 2% of those required by unconstrained GRAND, enabling decoding speeds up to $50\times$ faster.

To improve the manuscript's readability, Table 1 summarizes the proposed GRAND-based decoding variants and clarifies their relationships. Each variant introduces a distinct constraint or filtering mechanism, and the combined schemes apply multiple mechanisms jointly to further reduce

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2026.3725587

TABLE 1. Overview of the GRAND-based decoding variants.

Decoder	Mechanism
GRAND	Unconstrained noise-guessing decoding.
SP-GRAND	Classifies the n potential error positions in disjoint subsets where the errors can lie.
Sieving-GRAND	Syndrome-guided filtering of candidate error positions using column-weight thresholding of a reduced parity-check matrix.
LC-GRAND	From the zero-valued syndrome components only tests feasible combinations of error positions based on the weight of the columns of a reduced parity-check matrix.
SPS-GRAND	Applies set partitioning over the reduced candidate-position set obtained from the Sieving technique.
SPLC-GRAND	Applies set partitioning and additionally enforces the LC constraints for sharper pruning.

the search space while always preserving the same ML performance obtained by GRAND [27].

Notation: we use MATLAB's notation to denote rows and columns of a matrix: the i-th row is $\mathbf{H}(i,:)$ and the j-th column is $\mathbf{H}(:,j)$. The Hamming weight of a vector is denoted by its 1-norm, $||\cdot||_1$. A set of positions is denoted as $\mathcal{P} \subseteq \{p_1, p_2, \ldots, p_n\}$, with size $|\mathcal{P}| \leq n$. This set of positions can be written as $\mathcal{P} = \{p_{j_1}, p_{j_2}, \ldots, p_{j_{|\mathcal{P}|}}\}$, where $j_1, j_2, \ldots, j_{|\mathcal{P}|}$ are all different. A set can be a set of positions (integers) or a set of binary vectors (error patterns). The addition in GF(2) is denoted by $\oplus$. The *overlap* between two binary column vectors is the number of common positions holding 1s. $\lceil x \rceil$ denotes the smallest integer larger than x. $\mathbb{1}_n$ is the row-vector of ones of size n, $\mathbf{0}_n$ is a column-vector of zeros of size n, $\mathbf{a}^\mathsf{T}$ denotes the transpose of vector $\mathbf{a}$, and $\times$ denotes the Cartesian product of sets.

II. Encoding schemes

A. Random Linear Codes: dense and sparse

A column vector $\mathbf{a}$ with k independent and identically distributed (i.i.d.) information bits is encoded by an LBC onto n-bits long codewords, $\mathbf{x}_\mathrm{b}$, and then mapped onto binary phase-shift keying (BPSK) or quadrature phase-shift keying (QPSK), subjected to additive white Gaussian noise (AWGN).

We start by considering short (systematic) RLCs, given their performance guarantees, mentioned in Section I. The elements of the generator matrix $\mathbf{G} \in \mathbb{F}_2^{n\times k}$ can be chosen uniformly at random from a Galois field $\mathbb{F}_q = \{0, 1, \ldots, q-1\}$, where in this work, $q = 2$. The $\mathbf{G}$ matrix defines a codebook $\mathcal{C} = \left\{\mathbf{x}_\mathrm{b} = \mathbf{G}\mathbf{a} : \mathbf{a} \in \mathbb{F}_2^k\right\}$ with $2^k = 2^{nR}$ codewords of length n, which is a linear subspace of the discrete vector space $\mathbb{F}_2^n$. For systematic codes, $\mathbf{G}$ is of the form $\mathbf{G} = \left[\frac{\mathbf{P}}{\mathbf{I}_k}\right]$, where $\mathbf{P} \in \mathbb{F}_2^{(n-k)\times k}$ is a random binary matrix, whose elements have probabilities: $P(1) = p$ and $P(0) = 1 - p$. When $p >= 0.5$, we say that the RLC is dense, and for $p < 0.5$ we say that the RLC is p-sparse. $\mathbf{I}_k$ is the $k \times k$ identity matrix associated with the systematic part of the encoding.

Any LBC can be also defined by the $(n-k) \times n$ parity-check matrix, $\mathbf{H} = [\mathbf{I}_{n-k} \,|\, \mathbf{P}_{(n-k)\times k}] \in \mathbb{F}_2^{(n-k)\times n}$, which spans the null space of $\mathbf{G}$, so that $\mathbf{H}\mathbf{G} = \mathbf{0}_{n-k}$, and thus $\mathbf{H}\mathbf{x}_\mathrm{b} = \mathbf{0}_{n-k}, \forall \mathbf{x}_\mathrm{b} \in \mathcal{C}$. A received block, after hard-detection, can be written as $\mathbf{y}_\mathrm{b} = \mathbf{x}_\mathrm{b} \oplus \mathbf{e}$, where $\mathbf{e}$ is the binary error vector after detection. The syndrome, $\mathbf{s} = [s_1, s_2, ..., s_{(n-k)}]^T \in \mathbb{F}_2^{n-k}$, is computed as $\mathbf{s} = \mathbf{H}\mathbf{y}_\mathrm{b} = \mathbf{H}(\mathbf{x}_\mathrm{b} \oplus \mathbf{e}) = \mathbf{H}\mathbf{e}$. GRAND uses the syndrome for the membership test, given that $\mathbf{s} = \mathbf{0}_{n-k}$ if and only if $\mathbf{y}_\mathrm{b} \oplus \hat{\mathbf{e}} \in \mathcal{C}$, for some candidate error pattern $\hat{\mathbf{e}}$.

B. Quasi-regular LDPC codes with relaxed overlapping

Although short RLCs exist for any length and rate, the rules to design the parity-check matrices of MN codes and LDPC codes may not be feasible to fulfill for some desired lengths or rates. In its simplest form, a quasi-regular MN or LDPC code is constructed via its parity-check matrix with two simple rules [9], [10]:

1) randomly generate different sparse columns for $\mathbf{H}$, all with equal weight w_c;
2) any new column, $\mathbf{h}_\mathrm{new}$, can only be allowed to have an overlap ≤ 1 with all previously accepted columns.

The resulting code is a non-systematic one. To construct short LDPC codes we relax the second rule and tolerate overlaps ≤ 2. The overlaps can be easily checked by the inner product between columns of the extended $\mathbf{H}_\mathrm{ext} = [\mathbf{H} \,|\, \mathbf{h}_\mathrm{new}]$. The acceptance of a new column must check the off-diagonal elements of the Gram matrix: a new column $\mathbf{h}_\mathrm{new}$ is rejected if any element in of the matrix constructed as $\mathbf{H}_\mathrm{ext}^T\mathbf{H}_\mathrm{ext} - w_\mathrm{c}\mathbf{I}$ is > 2.

As noted earlier, short RLC codes outperform short LDPC codes. However, for reasons that will become apparent in Section IV.B, having an $\mathbf{H}$ with equal column weights (quasi-regular LDPC codes), will allow to strongly reduce the number of candidate error positions. Regular LDPC codes would additionally have an equal weight, w_r, in all their rows and could be used equally. Given its simpler construction, and similar performance when the codes are short (operating far from capacity), quasi-regular codes will be considered.

III. Set partitioning for Linear Block Codes

We start by proposing the partition of the n positions of a received block $\mathbf{y}_\mathrm{b}$ into sets. This allows us to generate a shorter list of admissible error patterns, $\mathcal{L}$.

Definitions: the positions of the syndrome containing a 1 are called the *flagged positions* of the syndrome, and their number is $F = ||\mathbf{s}||_1$. In all of them, $\mathbf{s}(f) = 1$, for $f \in \{1, 2, \ldots, n-k\}$. The rows of $\mathbf{H}$ that are associated with these positions are called the *flagged rows*. The flagged

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2026.3725587

rows are stacked to form the matrix $\mathbf{H}$-reduced, denoted as $\mathbf{H}_{\text{red}}$. The *support* of a binary vector $\mathbf{a}$ of length n is defined as $\text{supp}(\mathbf{a}) \triangleq \{i \in \{1,\dots,n\} : a_i \neq 0\}$, as in [50]. Accordingly, the i-th row of $\mathbf{H}$ has a support $\text{supp}(\mathbf{H}(i,:))$, whose cardinality (i.e., number of 1s) is its Hamming weight: $||\mathbf{H}(i,:))||_1 = |\text{supp}(\mathbf{H}(i,:))|$. Because $\mathbf{s} = \mathbf{He}$, we say that an error pattern $\mathbf{e}$ *activates* the j-th column of $\mathbf{H}$ when $e(j) = 1$. Each row i of $\mathbf{H}_{\text{red}}$ has a *set of ones*, whose elements are the positions of the row that contain a 1, denoted by $\mathcal{S}_1^{(i)}$ (corresponding to the support of the row), and a *set of zeros*, which comprises the positions of the row holding a 0, denoted by $\mathcal{S}_0^{(i)}$. The size of these sets is given by $|\mathcal{S}_1^{(i)}|$ and $|\mathcal{S}_0^{(i)}|$, respectively.

A. Construction of the admissible error sets

For a clearer explanation, we start by independently describing how the error patterns of weights $t = ||\mathbf{e}||_1 = 1, 2, 3$ are generated, and later we generalize to any even or odd weight.

1) One error case

In the case of a single error corrupting the block of n bits, there is no need to run membership tests by successively flipping single bits. Given that $\mathbf{s} = \mathbf{He}$, this situation corresponds to the case where only one column of $\mathbf{H}$ is active. This amounts to identifying whether the observed $\mathbf{s}$ is a copy of some column of $\mathbf{H}$. If only one error exists in position $p_1 = j$, then $\mathbf{s} = \mathbf{H}(:, j)$.

2) Two errors case

With $t = 2$ errors, any flagged position, f, in $\mathbf{s}$ can only result from two active columns $\mathbf{H}(:, p_1)$ and $\mathbf{H}(:, p_2)$, where the flagged row $\mathbf{H}(f,:)$ has in the positions $\mathbf{H}(f, p_1)$ and $\mathbf{H}(f, p_2)$ different bits. This means that the position of one of the errors must be in the *set of ones*, $\mathcal{S}_1^{(i)}$, of the row $\mathbf{H}(f,:)$ while the other error must necessarily be contained in the *set of zeros*, $\mathcal{S}_0^{(i)}$, of that same row.

The flagged rows are used to build the smaller matrix $\mathbf{H}_{\text{red}}$. The row of $\mathbf{H}_{\text{red}}$ that discloses more information about the location of the two errors is the one with the highest unbalance between the number of 0s and the number of 1s, i.e., the one where the sizes of the sets $\mathcal{S}_1^{(i)}$ and $\mathcal{S}_0^{(i)}$ that are most different. While the location of two errors among the n positions, all drawn from the same set, entails $|\mathcal{L}_2| = \binom{n}{2}$ possible error patterns, the list of all possible error patterns when the erroneous positions come from two disjoint sets is defined by the Cartesian product $\mathcal{S}_1^{(i)} \times \mathcal{S}_0^{(i)}$, whose size is $|\mathcal{S}_1^{(i)} \times \mathcal{S}_0^{(i)}| = |\mathcal{S}_1^{(i)}| \cdot |\mathcal{S}_0^{(i)}|$. Note that $\mathcal{S}_1^{(i)} \times \mathcal{S}_0^{(i)}$ generates the list of candidate error patterns $\hat{\mathcal{E}}_1, \hat{\mathcal{E}}_2, \dots, \hat{\mathcal{E}}_{|\mathcal{S}_1^{(i)} \times \mathcal{S}_0^{(i)}|}$, where each error pattern is defined by a set of two positions, i.e., $\hat{\mathcal{E}}_i = \{p_{j_1}, p_{j_2}\}$. The associated error pattern $\hat{\mathbf{e}}_i$ is the binary vector with 1s in the two positions in $\hat{\mathcal{E}}_i$.

The number of candidate error patterns can be minimized by choosing the row of $\mathbf{H}_{\text{red}}$ that has the minimum *or* the maximum weight of the support. From an information-theoretic perspective, that is the most informative row about the locations of the errors. The remaining entropy is subsequently dealt with by applying a noise-guessing approach that only considers the reduced set of candidate error patterns. The two extreme cases correspond to: a) $\mathcal{S}_1^{(i)} = 1$, where the number of error patterns with two errors is $|\mathcal{L}_2| = 1 \cdot (n-1)$, which is the case where $\mathcal{S}_1^{(i)}$ fully specifies the location of one of the two errors, leaving the other error position to be guessed among the remaining $n-1$ positions; b) $|\mathcal{S}_1^{(i)}| = |\mathcal{S}_0^{(i)}| = \frac{n}{2}$, leading to $|\mathcal{L}_2| = \frac{n}{2} \cdot \frac{n}{2} = \frac{n^2}{4}$. For the benefit of the algorithmic complexity of the decoder, we opt for only looking at the minimum support row of $\mathbf{H}_{\text{red}}$, i.e., the row of $\mathbf{H}_{\text{red}}$ that has the fewest 1s, defined in the following manner:

$$\mathbf{h}_m^{\mathsf{T}} = \mathbf{H}(m,:), \qquad m = \arg\min_{i \in \mathcal{I}} |\text{supp}(\mathbf{H}(i,:))|, \tag{1}$$

where $\mathcal{I} \subseteq \{1,\dots,n-k\}$ denotes the set of row indices selected from $\mathbf{H}$ to form $\mathbf{H}_{\text{red}}$, so that $|\mathcal{I}| = F$.

This option becomes optimal when sparse codes are considered, as $\mathbf{h}_m^{\mathsf{T}}$ will always be the most unbalanced row [36]. In the case of $t = 2$, two other sets should also be formed: the *full set*: $\mathcal{F}$, which contains the positions p_j where the columns $\mathbf{H}_{\text{red}}(:, p_j)$, $p_j \in \mathcal{F}$, are column-vectors of 1s with size F:

$$\mathcal{F} = \left\{ p_j \ : \ \mathbf{H}_{\text{red}}(:, p_j) = \mathbb{1}_F^{\mathsf{T}}, \ j \in \{1,\dots,n\} \right\}, \tag{2}$$

and the *null set*, $\mathcal{N}$, which contains the positions p_j where the columns $\mathbf{H}_{\text{red}}(:, p_j)$, $p_j \in \mathcal{N}$, are a zero column-vector with $(n-k) - F$ elements:

$$\mathcal{N} = \left\{ p_j \ : \ \mathbf{H}_{\text{red}}(:, p_j) = \mathbf{0}_{(n-k)-F}, \ j \in \{1,\dots,n\} \right\}. \tag{3}$$

If one error $p_1 \in \mathcal{F}$, then it is easy to see that, necessarily, $p_2 \in \mathcal{N}$. Therefore, we have a new set of possible error patterns given by another Cartesian product: $\mathcal{F} \times \mathcal{N}$.

Finally, the complete list of admissible error patterns is $\mathcal{L}_2 = \left((\mathcal{S}_1^{(m)} \setminus \mathcal{F}) \times (\mathcal{S}_0^{(m)} \setminus \mathcal{N}) \right) \cup (\mathcal{F} \times \mathcal{N})$, with size $|\mathcal{L}_2| = |\mathcal{S}_1^{(m)} \setminus \mathcal{F}| \cdot |\mathcal{S}_0^{(m)} \setminus \mathcal{N}| + |\mathcal{F}| \cdot |\mathcal{N}|$. This further splitting of the sets in the particular case of $||\mathbf{e}||_1 = 2$ allows for an additional reduction of the number of error patterns to be tested, which is more accentuated when the codes are very sparse and more zero-columns appear in $\mathbf{H}_{\text{red}}$.

3) Three errors case

In the case of $t = 3$ errors, the flagged rows, and in particular the minimum support row, $\mathbf{h}_m^{\mathsf{T}}$, can only trigger the flag if one of the two situations happens:

a) all three positions in error $p_1, p_2, p_3 \in |\mathcal{S}_1^{(m)}|$ and thus one has to search through all the possible error patterns with three positions in $\mathcal{S}_1^{(m)}$;
b) one error activated a 1 in the row and the other two lie "hidden" in the 0s, i.e., $p_1 \in \mathcal{S}_1^{(m)}$ and $p_2, p_3 \in \mathcal{S}_0^{(m)}$. Thus, one has to search through error patterns resulting

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2026.3725587

from the Cartesian product of $\mathcal{S}_1^{(m)} \times \mathcal{H}$, where $\mathcal{H}$ is the set of error patterns having all the combinations of two error positions "hiding" in $\mathcal{S}_0^{(m)}$.

With these constraints, the number of admissible error patterns with weight three is $|\mathcal{L}_3| = \binom{|\mathcal{S}_1^{(m)}|}{3} + |\mathcal{S}_1^{(m)}| \cdot \binom{|\mathcal{S}_0^{(m)}|}{2} \ll \binom{n}{3}$.

4) Any even or odd number of errors

Given that only an odd number of error positions in $\mathcal{S}_1^{(m)}$ can cause the syndrome flag associated with row $\mathbf{h}_m^{\mathsf{T}}$, when considering t errors, the set split is performed considering $l = 2a + 1 \leq t$ positions in $\mathcal{S}_1^{(m)}$, for $a \in \mathbb{Z}_0^+$, and the remaining $t - l$ positions are errors "hidden" in $\mathcal{S}_0^{(m)}$. The cases detailed for $t = 1, 2, 3$ are particular examples of this simple rule. For simplicity, the full set and the null set are recommended to be used only in the $t = 2$ case. Without those two sets, when applying set-partitioning, the maximum number of error patterns of weight up to t having to be tested is:

$$|\mathcal{L}| = \sum_{l=0}^{t} |\mathcal{L}_l| = 1 + \sum_{l=1}^{t} \sum_{\substack{i=1 \\ i \text{ odd}}}^{l} \binom{|\mathcal{S}_1^{(m)}|}{i} \cdot \binom{|\mathcal{S}_0^{(m)}|}{l-i}. \quad (4)$$

The following section will show that, for the parity-check matrices of RLCs or LDPCs, $|\mathcal{L}|$ is much smaller than the number of all error patterns of weight $||\mathbf{e}||_1 < t$, which amounts to $\sum_{i=0}^{t} \binom{n}{i}$.

B. Probability mass function of the minimum weight of the rows of $\mathbf{H}_{red}$

The impact of splitting the n positions into the set of ones, $\mathcal{S}_1^{(m)}$, and the set of zeros, $\mathcal{S}_0^{(m)}$, depends on how low the weight of the minimum weight of the support row of $\mathbf{H}_{\text{red}}$ is. Therefore, it is important to look at the probability distribution of this weight $||\mathbf{h}_m^{\mathsf{T}}||_1$, which corresponds to the size $|\mathcal{S}_1^{(m)}|$ in the minimum support row. The probability mass function (PMF) of $|\mathcal{S}_1^{(m)}|$ is plotted in Fig. 1 for RLCs (128,103) with SP-GRAND. Given that $\mathbf{H} = [\mathbf{I}_{n-k} \,|\, \mathbf{P}_{(n-k)\times k}] \in \mathbb{F}_2^{(n-k)\times n}$, dense RLCs (128,103) have on average 65 1s' in each row of $\mathbf{H}$: $\frac{n}{2}$ coming from the elements in the $\mathbf{P} \in \mathbb{F}_2^{(n-k)\times k}$ submatrix, plus one coming from the identity. The average weight of the minimum support row, $\mathbb{E}\{||\mathbf{h}_m^{\mathsf{T}}||_1\} = \mathbb{E}\{|\mathcal{S}_1^{(m)}|\}$, must be lower than $\frac{n}{2} + 1$. As observed in Fig. 1, the mean value $\mathbb{E}\{|\mathcal{S}_1^{(m)}|\} = 45$, and therefore $\mathbb{E}\{|\mathcal{S}_0^{(m)}|\} = n - \mathbb{E}\{|\mathcal{S}_1^{(m)}| = 83$. In this case, and considering $t = 3$, the number of possible error patterns is $|\mathcal{L}_3| = \binom{45}{3} + 45 \cdot \binom{83}{2} \ll \binom{128}{3}$.

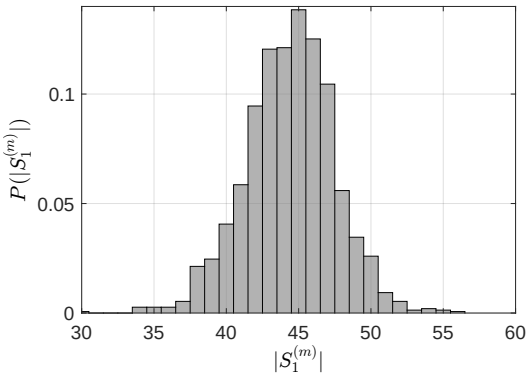


FIGURE 1. Simulated PMF for the number of 1s in the most unbalanced row of $\mathbf{H}_{\text{red}}$, using RLC (128,103).

Fig. 2 compares the number of error patterns required by unconstrained GRAND with the number of error patterns generated by the Cartesian products when set partitioning is applied, considering different weights of the minimum support row, $|\mathcal{S}_1^{(m)}|$. The number of error patterns is plotted as a function of the codewords' length $2 \leq n \leq 128$, and in the case when one is trying to guess the locations of $t = 2$ errors. In the case of unconstrained GRAND (blue curve), the number of error patterns is $|\mathcal{L}_2| = \binom{n}{2}$. When applying set partitioning GRAND, the worst-case scenario (black curve) corresponds to the case where $|\mathcal{S}_1^{(m)}| = |\mathcal{S}_0^{(m)}| = \frac{n}{2}$, in which case $|\mathcal{L}_2| = \frac{n}{2} \cdot \frac{n}{2} = \frac{n^2}{4}$. This upper bound for SP-GRAND is always much lower than $\binom{n}{2}$. The figure also depicts several curves for a varying number of $|\mathcal{S}_1^{(m)}|$ and $|\mathcal{S}_0^{(m)}| = n - |\mathcal{S}_1^{(m)}|$ pairs. In all cases, $|\mathcal{L}_2| = |\mathcal{S}_1^{(m)} \times \mathcal{S}_0^{(m)}| = |\mathcal{S}_1^{(m)}| \cdot |\mathcal{S}_0^{(m)}| \ll \binom{n}{2}$.

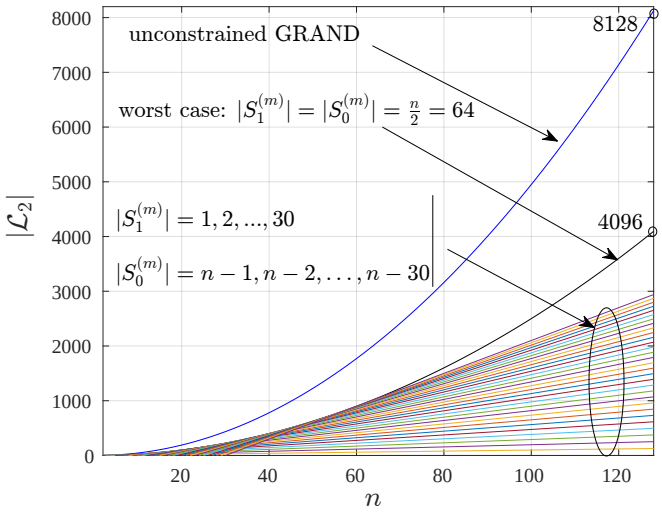


FIGURE 2. Number of error patterns considered in unconstrained GRAND and in set partitioning GRAND (SP-GRAND) for different weights of the minimum support row, $|\mathcal{S}_1^{(m)}|$, when looking for t=2 errors in codewords of length n.

IV. Sieving-GRAND for quasi-regular LDPC codes

When the parity-check matrix is both sparse and all columns have equal Hamming weight, such as the ones of quasi-

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2026.3725587

regular LDPC codes, the number of candidate error positions that need to be tested can be greatly reduced using a simple constraint (or "filter"), capable of gradually sieving candidate positions into a set, $\mathcal{P}$, with more positions being added to the set at different iterations. (See sieve algorithms for the closest vector problem in lattices [60], [61]). At each iteration, a GRAND approach only considers error patterns formed by error positions that belong to that reduced set of positions. We will next define that set and then explain the two proposed filtering mechanisms.

After the l-th iteration, the set of candidate error positions is $\mathcal{P}^{(l)} = \{p_{j_1}, p_{j_2}, \ldots, p_{j_{|\mathcal{P}^{(l)}|}}\}$. The size of the set increases monotonically, with more positions added to it in each iteration. The number of iterations is upper bounded by $w_c - 1$. When the algorithm reaches iteration $l = w_c - 1$ without finding the ML error pattern in any of the previous iterations, all the n positions will be in the set in that last iteration. After the first iteration, the set of candidate positions $\mathcal{P}^{(1)}$ is usually very small and, although $|\mathcal{P}^{(l)}|$ grows rapidly with each iteration, the number of error patterns to be tested for t errors in the l-th iteration is $\binom{|\mathcal{P}^{(l)}|}{t} \ll \binom{n}{t}$, for $k = 1, 2, \cdots, w_c - 1$. Note that the maximum number of iterations depends on the column weights of the LDPC code; nevertheless, the decoding can successfully terminate at a much earlier iteration if the ML $\hat{\mathbf{e}}$ has been found.

The proposed Sieving-GRAND algorithm implements two techniques involving a) the weight of the syndrome, and b) a Sieving mechanism of candidate positions where the errors may be. They are described in the following.

A. Discarding error patterns by weight

In the case of regular or quasi-regular-LDPC codes, by looking at the weight of the error syndrome, $||\mathbf{e}||_1$ it is possible to immediately discard some error patterns. For an $\mathbf{H}$ where all columns have constant weight w_c, the maximum weight of the syndrome can bear when t errors affect a codeword and is $\max\{||\mathbf{s}||_1\} = t \cdot w_c$. This happens when all the 1s in the t activated columns of $\mathbf{H}$ are in disjoint rows, i.e., with none of them overlapping. When overlaps of columns $\mathbf{H}_{\text{red}}(:, j)$ take place, $||\mathbf{s}||_1 \leq t \cdot w_c$. Therefore, when $||\mathbf{s}||_1 > m \cdot w_c$, for some $m \in \mathbb{Z}^+$, the decoder should only test error pattern with weights $||\hat{\mathbf{e}}||_1 > m$. Therefore, the first t to be tested should be $t = \lceil \frac{||\mathbf{s}||_1}{w_c} \rceil$.

B. Filtering candidate error positions

We start by constructing $\mathbf{H}_{\text{red}} \in \mathbb{F}_2^{F \times n}$, consisting of the F flagged rows of $\mathbf{H}$, as defined in Section III. The Sieving (or filtering) mechanism depends on the Hamming weights of the columns of $\mathbf{H}_{\text{red}}$:

$$\begin{aligned} \Sigma(j) &= ||\mathbf{H}_{\text{red}}(:, j)||_1 \\ &= \sum_{i=1}^{F} \mathbf{H}_{\text{red}}(i, j), \quad \text{for } j = 1, \ldots n, \end{aligned} \tag{5}$$

corresponding to the number of 1s in each column of $\mathbf{H}_{\text{red}}$. These sums form a row vector that can be directly obtained via the inner product of the syndrome vector with each column of the parity-check matrix:

$$\boldsymbol{\Sigma} = \mathbf{s}^{\mathsf{T}} \cdot \mathbf{H}. \tag{6}$$

The iterations begin with an empty set of candidate positions $\mathcal{P}^{(0)} = \emptyset$. Next, more candidate positions are added such that $\mathcal{P}^{(1)} = \emptyset \cup \{p_i\}_{\text{new}}$. The mechanism for accepting positions into the candidate set at the l-th iteration is the following:

$$\mathcal{P}^{(l)} = \{p_j \ : \ \Sigma(j) \geq w_{\text{th}}(l), \ j \in \{1, \ldots, n\}\}, \tag{7}$$

where j indexes the columns of $\mathbf{H}_{\text{red}}$ and $w_{\text{th}}(l)$ is the value of a weight threshold during the l-th iteration, starting at $w_{\text{th}}(1) = w_c$. While the number of possible positions is $|\mathcal{P}^{(l)}| < t$, w_{th} is lowered by one unit: $w_{\text{th}}(l+1) = w_{\text{th}}(l) - 1$ until it reaches a pre-defined minimum value $w_{\text{th}_{\min}}$. By choosing $w_{\text{th}_{\min}} = 1$ in Sieving-GRAND, the decoding performance must coincide with the one of GRAND (i.e., ML); a choice of $w_{\text{th}_{\min}} > 1$ will lead to faster but sub-optimal decoding, leaving some errors uncorrected. At this point, the error patterns are tested by running GRAND on the reduced set of positions $\mathcal{P}^{(l)}$, involving only $|\mathcal{L}| = \binom{|\mathcal{P}^{(l)}|}{t} \ll \binom{n}{t}$ candidate error patterns. The threshold is successively reduced in unit steps until the set of candidate positions $\mathcal{P}^{(l)}$ includes *all* the t positions where the ML errors lie, in which case the decoding of the ML error pattern $\hat{\mathbf{e}}$ will take place during the guessing stage. The need to have equal-weight columns in $\mathbf{H}$ assures an unbiased selection of positions to be added to $\mathcal{P}^{(l)}$ at iteration l, meaning that they all hold an equal *a priori* probability. If the columns of $\mathbf{H}$ were to have different weights, the columns with larger weights would have more matches with the syndrome flags and would be selected for inclusion in $\mathcal{P}^{(l)}$ at early iterations, even when that column has not been activated by an error position.

For a visualization of these calculations, consider the example in Fig. 3 with $n = 30$. An overview of the procedure is provided through the flowchart in Fig. 4, and detailed in Algorithm 1.

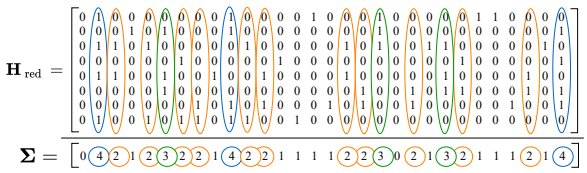


FIGURE 3. Hamming weights of the columns of $\mathbf{H}_{\text{red}}$ and formation of the candidate set $\mathcal{P}^{(l)}$, for iterations l= 1, 2, 3, which respectively correspond to the column-weight thresholds $w_{\text{th}} = 4, 3, 2$.

In this example, the different colors indicate the set of new positions that are added to the pool at each iteration. In the first iteration, the selected candidate positions are $\mathcal{P}^{(1)} = \{p_1, p_2, p_3\} = \{2, 10, 30\}$ (colored in blue) for the first threshold $w_{\text{th}}(1) = 4$. We denote such

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2026.3725587

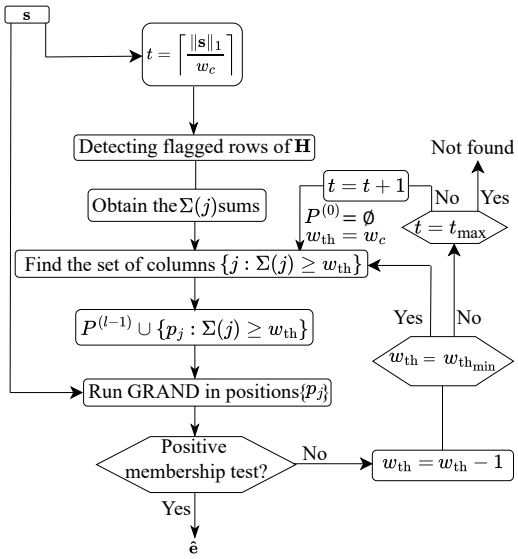


FIGURE 4. **Sieving (or filtering) candidate positions using i) a column weight threshold, and ii) the weight of the syndrome to discard error weights.**

Algorithm 1 Sieving (filtering) candidate positions using (i) a column-weight threshold and (ii) the syndrome weight to discard error weights

Input: syndrome $\mathbf{s}$
Construct $\mathbf{H}_{\text{red}}$ from the flagged rows of $\mathbf{H}$
Compute $\Sigma(j) = \|\mathbf{H}_{\text{red}}(:, j)\|_1$, $j = 1, \ldots, n$
$w_{\text{th}_{\min}} \leftarrow 1$
for $t = \lceil \frac{||\mathbf{s}||_1}{w_c} \rceil$ to $t_{\max}$ **do**
    $w_{\text{th}} \leftarrow w_c$
    $P^{(0)} \leftarrow \emptyset$
    $l \leftarrow 0$
    while $w_{\text{th}} \geq w_{\text{th}_{\min}}$ **do**
        $l \leftarrow l + 1$
        $P^{(l)} \leftarrow P^{(l-1)} \cup \{p_j : \Sigma(j) \geq w_{\text{th}}(l)\}$
        Run GRAND restricted to positions in $P^{(l)}$
        if positive membership test **then**
            Output: $\hat{\mathbf{e}}$
            return
        end if
        $w_{\text{th}} \leftarrow w_{\text{th}} - 1$
    end while
end for
Output: Not found

columns as columns of type $\mathbb{S}_4$, defined by their weight. The set of candidate positions is subsequently extended to $\mathcal{P}^{(2)} = \{p_1, p_2, p_3, p_4, p_5\} = \{2, 6, 10, 19, 23, 30\}$ (colored in green), for $w_{\text{th}}(2) = 3$; similarly, we denote such columns as columns of type $\mathbb{S}_3$. Finally, the candidate positions are extended (if needed) to include the remaining positions where $||\mathbf{H}_{\text{red}}(:, j)||_1 \geq 2$ (colored orange), which are denoted as columns of type $\mathbb{S}_2$.

Error patterns that have been tested in a previous iteration do not need to be retested after adding new positions to $\mathcal{P}$, as only error patterns that involve *new* positions added during each iteration need to be considered.

One should note that the proposed Sieving mechanism is rather different from majority logic decoding (MLD) for LDPC codes, first proposed by Gallager as one-shot hard-decoder [8], and which later evolved to the iterative (hard or soft) bit-flipping algorithm [62, Ch. 8], [63, Ch. 8]. In MLD, parity-check equations cast votes to iteratively determine and flip symbol values, directly modifying the received word during decoding. The Sieving mechanism does not perform bit decisions nor iterative correction. Instead, it leverages syndrome–parity-check consistency to rank candidate error positions that are successively added to the pool of candidate locations for the errors. This preserves maximum-likelihood decoding (the ML error pattern is never discarded without testing) while greatly reducing the search space in comparison to GRAND.

C. Probability mass function of the set of candidate positions $|\mathcal{P}^{(l)}|$ containing the t errors

The Sieving approach adds at each iteration l a new set of positions to the set $\mathcal{P}^{(l-1)}$, leading to $\mathcal{P}^{(l)}$. Here we analyze the probability distribution of the increasing set of candidate positions, containing the t errors after l iterations. A positive membership test takes place at the l-th iteration if and only if the correct t error positions $\{p_1, p_2, \ldots, p_t\} \in \mathcal{P}^{(l)}$. Fig. 5a) shows the continuous-variable approximated PMF of the size $|\mathcal{P}^{(l)}|$ that contains *all* the $t = 3$ errors for an LDPC code (128,104) with the Sieving-GRAND decoder. We get that the set sizes $|\mathcal{P}^{(l)}| \leq 10$ concentrate $\approx 74\%$ of the probability mass function (as seen in the cumulative distribution function (CDF) plotted in Fig. 5b). This very small pool of candidate positions translates into an extreme reduction of the average number of error patterns to be tested.

V. Mixing set partitioning and Sieving-GRAND

By combining set partitioning and the Sieving technique, one obtains set partitioning Sieving-GRAND (SPS-GRAND). To shed light on the concept, an example with a small parity-check matrix, with $n = 30$ and $n-k = 20$, will be discussed in the following (and illustrated in Fig. 6).

- Let us consider a number of flags in the syndrome $F = ||\mathbf{s}||_1 = 8$.
- both the *flags* and the *flagged rows* are highlighted by red arrows (rows $\{1, 3, 4, 5, 7, 9, 14, 16\}$).
- the flagged rows are stacked to form $\mathbf{H}_{\text{red}}$.
- the columns' Hamming weights are obtained: $\Sigma(j) = ||\mathbf{H}_{\text{red}}(:, j)||_1$, and the positions $\mathcal{P}^{(1)} = \{2, 10, 30\}$ are first selected, for a threshold $w_{\text{th}} = 4$, $\mathcal{P}^{(2)} = \{2, 6, 10, 19, 23, 30\}$, for $w_{\text{th}} = 3$, and finally extended (if needed) to include the remaining positions where $||\mathbf{H}_{\text{red}}(:, j)||_1 \geq 2$.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2026.3725587

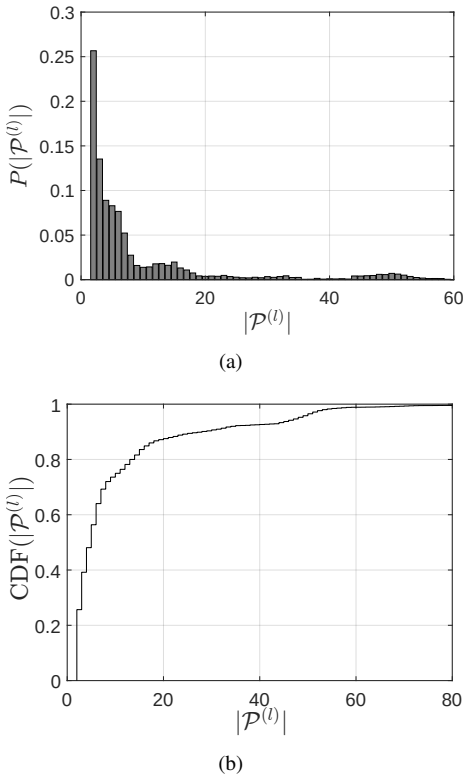


FIGURE 5. Simulated PMF and CDF for the number of positions in the set $\mathcal{P}^{(l)}$ that contain all the t errors. The results are for the LDPC code (128,104), with the decoder searching for up to t=3 errors.

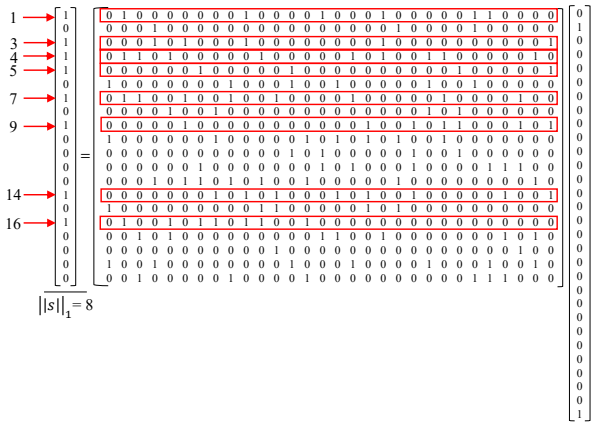


FIGURE 6. Construction of $\mathbf{H}_{\text{red}}$ for the Sieving procedure. The non-zero syndrome components (their number is $\|s\|_1 = 8$ in this example) select the corresponding flagged rows of $\mathbf{H}$, which are highlighted and stacked to form $\mathbf{H}_{\text{red}}$.

- At iteration $l = 2$, the candidate error patterns are tested by running GRAND exclusively on a smaller subset of positions $|\mathcal{P}^{(l)}| = 6$ including only $|\mathcal{L}_t| = \binom{|\mathcal{P}^{(l)}|}{t} \ll \binom{n}{t}$ candidate error patterns. When looking for $t = 3$ errors, $|\mathcal{L}_3| = \binom{6}{3} = 20 \ll \binom{30}{3} = 4060$.

The simultaneous application of set partitioning and Sieving techniques is exemplified in Fig. 7. It begins with $\mathcal{S}_1^{(m)}$ and $\mathcal{S}_0^{(m)}$, as defined by the minimum support row,

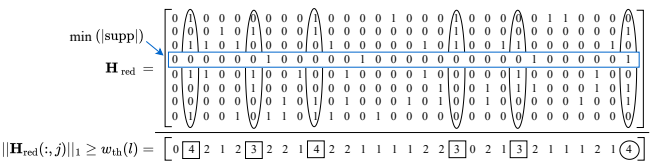


FIGURE 7. Identification of the minimum support row of $\mathbf{H}_{\text{red}}$, and the combined application of set partitioning and Sieving-GRAND (SPS-GRAND). The squares highlight the positions in the set of zeros, $\mathcal{S}_0^{(m)}$, and the circle highlights the position in the set of ones, $\mathcal{S}_1^{(m)}$.

and next, the set $\mathcal{P}^{(l)}$ intersects both $\mathcal{S}_1^{(m)}$ and $\mathcal{S}_0^{(m)}$. The admissible error patterns are generated by applying the sets partitioning technique while considering only the remaining positions in $\mathcal{P}^{(l)} \cap \mathcal{S}_1^{(m)}$ and $\mathcal{P}^{(l)} \cap \mathcal{S}_0^{(m)}$. In this example, $|\mathcal{P}^{(l)} \cap \mathcal{S}_1^{(m)}| = 1$ (indicated by a circle), and $|\mathcal{P}^{(l)} \cap \mathcal{S}_0^{(m)}| = 5$ (indicated by squares). The number of candidate error patterns is $|\mathcal{L}_3| = 1 \times 5 \ll 4060$.

A. Probability mass function of $|\mathcal{S}_1^{(m)}|$ with SPS-GRAND

After applying the filtering mechanism described in Section IV, the positions in the set $\mathcal{P}^{(l)}$ can be intersected with the ones in the $\mathcal{S}_1^{(m)}$ and $\mathcal{S}_0^{(m)}$ sets, resulting in the elimination of many of the positions in both $\mathcal{S}_1^{(m)}$ and $\mathcal{S}_0^{(m)}$. Fig. 8 depicts the simulated histogram of the number of positions in the intersection $|\mathcal{S}_1^{(m)}| \cap \mathcal{P}^{(l)}$, using the LDPC code (128,104), for $t = 3$. Compared with Fig. 1, where $\mathbb{E}\{|\mathcal{S}_1^{(m)}|\} = 45$, we now see in Fig. 8 that, when set partitioning and the Sieving technique are applied simultaneously, $|\mathcal{S}_1^{(m)}|$ is always below 20 positions with probability 1.

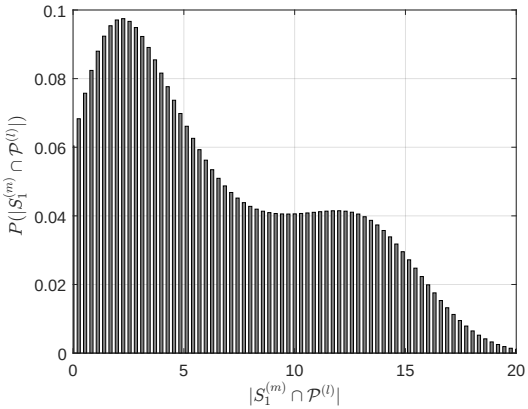


FIGURE 8. Simulated PMF for the remaining number of 1s in $|\mathcal{S}_1^{(m)}| \cap \mathcal{P}^{(l)}$ after applying the intersection of SPS-GRAND, using the LDPC code (128,104), with t =3.

VI. Largely constrained GRAND

While the Sieving mechanism introduced earlier significantly reduces the number of candidate error patterns in GRAND-based decoding of LDPCs, further computational savings can be obtained by exploiting zero-valued syndrome positions. We refer to this enhanced approach as largely constrained GRAND (LC-GRAND). The key idea is to utilize the

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2026.3725587

zero-valued syndrome positions and column-wise Hamming weight properties of the parity-check matrix.

For a binary syndrome vector $\mathbf{s}$, define the set F_0 as the positions in the syndrome vector where the value is zero, which will be called *zero-flagged positions*. The number of such positions is $|F_0| = (n-k) - F$, and the corresponding rows are called *the zero-flagged rows* of $\mathbf{H}$, which satisfy $\mathbf{H}(f_0, :) \cdot \mathbf{e} = 0$, for $f_0 \in \{1, 2, \ldots, n-k\}$. These positions are stacked to form matrix $\mathbf{H}_{\overline{\text{red}}} \in \mathbb{F}_2^{|F_0| \times n}$. The decoding strategy depends on analyzing the column-wise weights of $\mathbf{H}_{\overline{\text{red}}}$. Similarly to vector $\mathbf{\Sigma}$, defined in Subsection IV.B, we define the row vector $\overline{\mathbf{\Sigma}}$, whose elements are the sums of the columns of matrix $\mathbf{H}_{\overline{\text{red}}}$:

$$\begin{aligned} \overline{\Sigma}(j) &= ||\mathbf{H}_{\overline{\text{red}}}(:, j)||_1 \\ &= \sum_{i=1}^{|F_0|} \mathbf{H}_{\overline{\text{red}}}(i, j), \quad \text{for } j = 1, \ldots n. \end{aligned} \tag{8}$$

This "sums vector" can be obtained via the product of the syndrome vector with the parity-check matrix, noting the following relation between $\mathbf{H}_{\overline{\text{red}}}$ and $\mathbf{H}_{\text{red}}$:

$$\begin{aligned} \overline{\mathbf{\Sigma}} &= w_{\text{c}} \mathbb{1}_n - \mathbf{\Sigma} \\ &= w_{\text{c}} \mathbb{1}_n - \mathbf{s}^{\mathsf{T}} \cdot \mathbf{H}, \end{aligned} \tag{9}$$

where $\mathbb{1}$ is the row-vector of ones of size n. The relation captured in (9) comes from the construction of the considered quasi-regular LDPC codes. Considering that all columns of $\mathbf{H}$ have weight w_{c}, when the j-th column of $\mathbf{H}_{\text{red}}$ has weight $\Sigma(j)$, then the corresponding j-th column of $\mathbf{H}_{\overline{\text{red}}}$ necessarily holds the remaining $\overline{\Sigma}(j) = w_{\text{c}} - \Sigma(j)$ 1s.

The columns of $\mathbf{H}_{\overline{\text{red}}}$ are classified according to their weights. Specifically, $\mathbb{C}_i$ denotes the set of columns having $\left\|\mathbf{H}_{\overline{\text{red}}}(:, j)\right\|_1 = i$, for $i = 0, 1, \ldots, w_{\text{th}}^{(\text{LC})}$. For a fair comparison between Sieving-GRAND and LC-GRAND, we use the same quasi-regular LDPC and therefore the maximum weight a column of $\mathbf{H}_{\overline{\text{red}}}$ can take is

$$w_{\text{th}}^{(\text{LC})} = w_{\text{c}} - w_{\text{th}_{\min}}, \tag{10}$$

where $w_{\text{th}_{\min}}$ has been defined in Section IV.B. An error pattern is valid only if the activated columns from these sets yield even parity across all flagged zero rows. This principle allows many patterns to be eliminated in advance. To illustrate how the proposed LC-GRAND method narrows down viable error patterns, we examine cases of error patterns with weights $t = 2, 3, 4$. In each example, we analyze whether the given Cartesian product of sets $\mathbb{C}_i$ can generate a valid error pattern. Each set $\mathbb{C}_i$ contributes exactly i ones in the corresponding column, i.e., $\mathbb{C}_0$ contains no 1s, $\mathbb{C}_1$ contains one 1, $\mathbb{C}_2$ contains two 1s, and so on, up to $\mathbb{C}_{w_{\text{th}}^{(\text{LC})}}$. We aim to cancel 1s across rows so that each row reaches even parity. If all 1s can be paired or grouped in such a way that every row ends up with an even number of 1s (or dashes after cancellation), the result is a valid error pattern (even). Otherwise, if any row has an unpaired 1 (odd parity), it cannot lead to a valid error pattern. This rule enables us to identify feasible error patterns in combinations of sets by examining how the 1s align and whether full cancellation is achievable. The error sets of weight t have a structure $\mathbb{C}_{i_1} \times \mathbb{C}_{i_2} \cdots \times \mathbb{C}_{i_t} = \prod_{a=1}^{t} \mathbb{C}_{i_a}$, where $i_a \in \{1, 2, \ldots, w_{\text{th}}^{(\text{LC})}\}$. Determining whether one of such structures is feasible, is an algorithmic problem that can be generally framed and solved by making use of the following simple rules:

1) Place the sets $\mathbb{C}_i$ with increasing cardinality from left to right;
2) group horizontal even numbers of 1s and eliminate them;
3) if there is a remaining 1 in a row, it descends to a row below where it can turn that row even; if no such row is found, that 1 drops to an extra row at the bottom.

Algorithm 2 formally describes how these rules are applied, and to make this procedure clearer some examples will be explored in detail in the following.

Algorithm 2 Checking feasibility of an error pattern with LC-GRAND constraints

Input: A collection of t sets $\{\mathbb{C}_i\}$
Sort the sets $\mathbb{C}_i$ in increasing order of their cardinality
Place the sets from left to right according to this order
for each row r **do**
    while exists a horizontal pair of 1s in row r **do**
        Group two 1s and eliminate them
    end while
    if exactly one 1 remains in row r **then**
        Search for a row $r' > r$ that becomes even by adding this 1
        if such a row r' exists **then**
            Move remaining 1 to row r'
            Eliminate the pair
        else
            Move remaining 1 to extra row at the bottom
        end if
    end if
end for
if all rows have no 1 left unpaired **then**
    Output: Feasible error pattern type
else
    Output: Infeasible error pattern type
end if

Feasible example for $t = 2$:

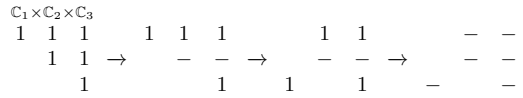
$$\begin{array}{c} \mathbb{C}_1 \times \mathbb{C}_1 \\ 1 \quad 1 \end{array} \rightarrow \; -$$

- each $\mathbb{C}_1$ contributes a single 1. These two 1s cancel each other out, leaving no unmatched 1s, so the resulting parity is even, meaning this set is a valid error pattern.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

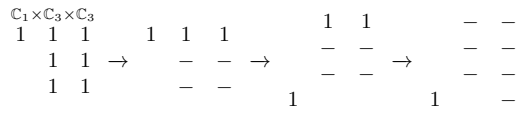
This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2026.3725587

Feasible example for $t = 3$:



- The two 1s in the second row cancel each other,
- one 1 remains in the first and third rows. We regroup to pair the remaining 1s in the first and third rows,
- all 1s have been canceled, each row has even parity. Consequently, this composition is a valid error pattern

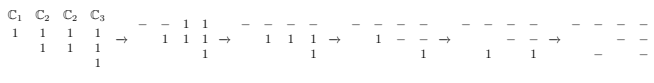
Infeasible example for $t = 3$:



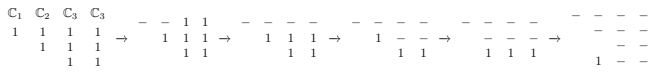
- The 1s in the second and third rows cancel in their overlapping positions, leaving three 1s in the first row,
- we bring down the remaining 1 in the first row to try canceling with others. But now we have an odd number of 1s distributed,
- one unpaired 1 remains (bottom row), resulting in odd parity. Thus, a Cartesian product of the form $\mathbb{C}_1 \times \mathbb{C}_3 \times \mathbb{C}_3$ is not a valid error pattern.

The process for $t = 4$ follows the same logic as in the earlier examples with $t = 2$ and $t = 3$: we progressively cancel or pair 1s in overlapping rows, and then assess whether all rows achieve even parity.

Feasible example for $t = 4$:



Infeasible example for $t = 4$:



We now present in Fig. 9 an example to visualize these calculations using a small parity-check matrix with $n = 30$ and $n - k = 20$:

- Let us consider a number of zero-flagged positions in the syndrome $|F_0| = 12$;
- the *flagged zero positions* (highlighted by red arrows) are rows $\{2, 6, 8, 10, 11, 12, 13, 15, 17, 18, 19, 20\}$);
- the flagged zero rows are stacked to form $\mathbf{H}_{\overline{\text{red}}}$;
- vector $\overline{\mathbf{\Sigma}}$ is obtained according to (9);

TABLE 2. Feasible error patterns for two errors (t=2) using LC-GRAND filtering.

$\mathbf{H}_{\overline{\text{red}}}$	Possible patterns
$\mathbb{C}_0 \times \mathbb{C}_0$	Yes
$\mathbb{C}_1 \times \mathbb{C}_1$	Yes
$\mathbb{C}_2 \times \mathbb{C}_2$	Yes

- at this stage, GRAND only tests the candidate error patterns from the sets $\mathbb{C}_i$, where $i \in \{0, 1, 2\}$, as shown in Table 2.

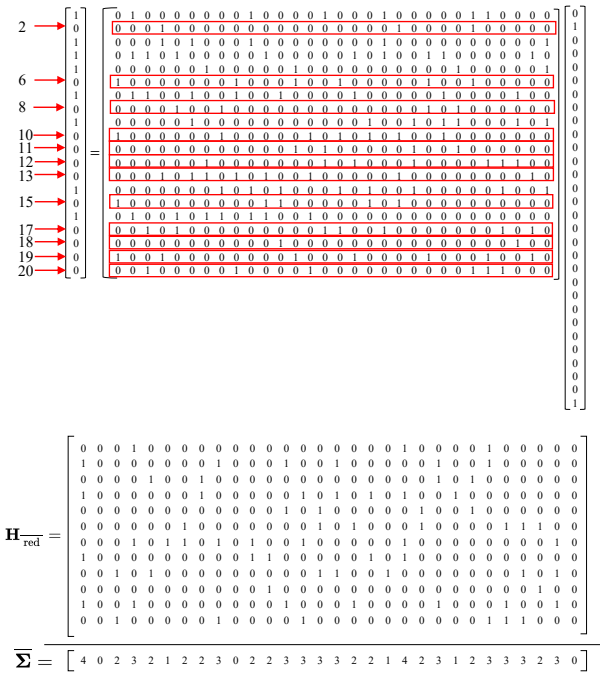


FIGURE 9. Example of the LC-GRAND: (top) construction of $\mathbf{H}_{\overline{\text{red}}}$ from the zero-flagged rows; (bottom) obtaining the weights of the columns of $\mathbf{H}_{\overline{\text{red}}}$.

By applying these rules, all valid configurations for error weights $t = 2, 3, 4$ have been obtained, which are summarized in Tables 3, 4, and 5 respectively. Each row in the tables represents a candidate error pattern formed by selecting columns from the constraint sets $\mathbb{C}_i$, intending to preserve the zero-valued syndrome condition. The Cartesian products that form errors of weight t in positions $\hat{\mathcal{E}}_i = \{p_{j_1}, p_{j_2} \dots p_{j_t}\}$, as defined in Subsection III.A.2), are specified in these tables in two different but entirely related notations: in the first column, related with $\mathbf{H}_{\overline{\text{red}}}$, they appear as in the notation defined for the Sieving-GRAND presented in Section IV, where the indexes of $\mathbb{S}_i$ are $i = w_c, w_c - 1, \dots, 1$, considering that $w_{\text{th}_{\min}} = 1$; and in the second column, related with $\mathbf{H}_{\overline{\text{red}}}$, the error sets are denoted using the notation of LC-GRAND, where the indexes of $\mathbb{C}_i$ are $i = 0, 1, \dots, w_c - 1$, according to (10), also considering $w_{\text{th}_{\min}} = 1$. The third column indicates whether the Cartesian product of these sets can produce an error pattern compatible with the zero-flagged positions of the syndrome.

VII. Results

The system model described in Section II was numerically evaluated with RLCs (128,103), and an LDPC code (128,104) with $w_c = 4$ and $d_{\min} = 7$ (i.e., capable of correcting at least $t = 3$ errors), constructed as described in Subsection B, so that both systems have $R \approx 0.8$ (the

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2026.3725587

TABLE 3. Valid and invalid error patterns for two errors (t =2) under LC-GRAND constraints.

$\mathbf{H}_{\text{red}}$	$\mathbf{H}_{\overline{\text{red}}}$	Possible patterns
$\mathbb{S}_4 \times \mathbb{S}_4$	$\mathbb{C}_0 \times \mathbb{C}_0$	Yes
$\mathbb{S}_4 \times \mathbb{S}_3$	$\mathbb{C}_0 \times \mathbb{C}_1$	No
$\mathbb{S}_4 \times \mathbb{S}_2$	$\mathbb{C}_0 \times \mathbb{C}_2$	No
$\mathbb{S}_4 \times \mathbb{S}_1$	$\mathbb{C}_0 \times \mathbb{C}_3$	No
$\mathbb{S}_3 \times \mathbb{S}_3$	$\mathbb{C}_1 \times \mathbb{C}_1$	Yes
$\mathbb{S}_3 \times \mathbb{S}_2$	$\mathbb{C}_1 \times \mathbb{C}_2$	No
$\mathbb{S}_3 \times \mathbb{S}_1$	$\mathbb{C}_1 \times \mathbb{C}_3$	No
$\mathbb{S}_2 \times \mathbb{S}_2$	$\mathbb{C}_2 \times \mathbb{C}_2$	Yes
$\mathbb{S}_2 \times \mathbb{S}_1$	$\mathbb{C}_2 \times \mathbb{C}_3$	No
$\mathbb{S}_1 \times \mathbb{S}_1$	$\mathbb{C}_3 \times \mathbb{C}_3$	No, Overlap > 2

TABLE 4. Feasible and infeasible error patterns for three errors (t =3) using LC-GRAND constraints.

$\mathbf{H}_{\text{red}}$	$\mathbf{H}_{\overline{\text{red}}}$	Possible patterns
$\mathbb{S}_4 \times \mathbb{S}_4 \times \mathbb{S}_4$	$\mathbb{C}_0 \times \mathbb{C}_0 \times \mathbb{C}_0$	Yes
$\mathbb{S}_4 \times \mathbb{S}_4 \times \mathbb{S}_3$	$\mathbb{C}_0 \times \mathbb{C}_0 \times \mathbb{C}_1$	No
$\mathbb{S}_4 \times \mathbb{S}_3 \times \mathbb{S}_3$	$\mathbb{C}_0 \times \mathbb{C}_1 \times \mathbb{C}_1$	Yes
$\mathbb{S}_4 \times \mathbb{S}_4 \times \mathbb{S}_2$	$\mathbb{C}_0 \times \mathbb{C}_0 \times \mathbb{C}_2$	No
$\mathbb{S}_4 \times \mathbb{S}_3 \times \mathbb{S}_2$	$\mathbb{C}_0 \times \mathbb{C}_1 \times \mathbb{C}_2$	No
$\mathbb{S}_4 \times \mathbb{S}_2 \times \mathbb{S}_2$	$\mathbb{C}_0 \times \mathbb{C}_2 \times \mathbb{C}_2$	Yes
$\mathbb{S}_4 \times \mathbb{S}_4 \times \mathbb{S}_1$	$\mathbb{C}_0 \times \mathbb{C}_0 \times \mathbb{C}_3$	No
$\mathbb{S}_4 \times \mathbb{S}_3 \times \mathbb{S}_1$	$\mathbb{C}_0 \times \mathbb{C}_1 \times \mathbb{C}_3$	No
$\mathbb{S}_4 \times \mathbb{S}_2 \times \mathbb{S}_1$	$\mathbb{C}_0 \times \mathbb{C}_2 \times \mathbb{C}_3$	No
$\mathbb{S}_4 \times \mathbb{S}_1 \times \mathbb{S}_1$	$\mathbb{C}_0 \times \mathbb{C}_3 \times \mathbb{C}_3$	Yes
$\mathbb{S}_3 \times \mathbb{S}_3 \times \mathbb{S}_3$	$\mathbb{C}_1 \times \mathbb{C}_1 \times \mathbb{C}_1$	No
$\mathbb{S}_3 \times \mathbb{S}_3 \times \mathbb{S}_2$	$\mathbb{C}_1 \times \mathbb{C}_1 \times \mathbb{C}_2$	Yes
$\mathbb{S}_3 \times \mathbb{S}_2 \times \mathbb{S}_2$	$\mathbb{C}_1 \times \mathbb{C}_2 \times \mathbb{C}_2$	No
$\mathbb{S}_3 \times \mathbb{S}_3 \times \mathbb{S}_1$	$\mathbb{C}_1 \times \mathbb{C}_1 \times \mathbb{C}_3$	No
$\mathbb{S}_3 \times \mathbb{S}_2 \times \mathbb{S}_1$	$\mathbb{C}_1 \times \mathbb{C}_2 \times \mathbb{C}_3$	Yes
$\mathbb{S}_3 \times \mathbb{S}_1 \times \mathbb{S}_1$	$\mathbb{C}_1 \times \mathbb{C}_3 \times \mathbb{C}_3$	No
$\mathbb{S}_2 \times \mathbb{S}_2 \times \mathbb{S}_2$	$\mathbb{C}_2 \times \mathbb{C}_2 \times \mathbb{C}_2$	No
$\mathbb{S}_2 \times \mathbb{S}_2 \times \mathbb{S}_1$	$\mathbb{C}_2 \times \mathbb{C}_2 \times \mathbb{C}_3$	No
$\mathbb{S}_2 \times \mathbb{S}_1 \times \mathbb{S}_1$	$\mathbb{C}_2 \times \mathbb{C}_3 \times \mathbb{C}_3$	No
$\mathbb{S}_1 \times \mathbb{S}_1 \times \mathbb{S}_1$	$\mathbb{C}_3 \times \mathbb{C}_3 \times \mathbb{C}_3$	No

TABLE 5. Feasible and infeasible error patterns for four errors (t =4) using LC-GRAND constraints.

$\mathbf{H}_{\text{red}}$	$\mathbf{H}_{\overline{\text{red}}}$	Possible patterns
$\mathbb{S}_4 \times \mathbb{S}_4 \times \mathbb{S}_4 \times \mathbb{S}_4$	$\mathbb{C}_0 \times \mathbb{C}_0 \times \mathbb{C}_0 \times \mathbb{C}_0$	Yes
$\mathbb{S}_4 \times \mathbb{S}_4 \times \mathbb{S}_4 \times \mathbb{S}_3$	$\mathbb{C}_0 \times \mathbb{C}_0 \times \mathbb{C}_0 \times \mathbb{C}_1$	No
$\mathbb{S}_4 \times \mathbb{S}_4 \times \mathbb{S}_3 \times \mathbb{S}_3$	$\mathbb{C}_0 \times \mathbb{C}_0 \times \mathbb{C}_1 \times \mathbb{C}_1$	Yes
$\mathbb{S}_4 \times \mathbb{S}_3 \times \mathbb{S}_3 \times \mathbb{S}_3$	$\mathbb{C}_0 \times \mathbb{C}_1 \times \mathbb{C}_1 \times \mathbb{C}_1$	No
$\mathbb{S}_4 \times \mathbb{S}_4 \times \mathbb{S}_4 \times \mathbb{S}_2$	$\mathbb{C}_0 \times \mathbb{C}_0 \times \mathbb{C}_0 \times \mathbb{C}_2$	Yes
$\mathbb{S}_4 \times \mathbb{S}_4 \times \mathbb{S}_3 \times \mathbb{S}_2$	$\mathbb{C}_0 \times \mathbb{C}_0 \times \mathbb{C}_1 \times \mathbb{C}_2$	No
$\mathbb{S}_4 \times \mathbb{S}_3 \times \mathbb{S}_3 \times \mathbb{S}_2$	$\mathbb{C}_0 \times \mathbb{C}_1 \times \mathbb{C}_1 \times \mathbb{C}_2$	Yes
$\mathbb{S}_4 \times \mathbb{S}_4 \times \mathbb{S}_2 \times \mathbb{S}_2$	$\mathbb{C}_0 \times \mathbb{C}_0 \times \mathbb{C}_2 \times \mathbb{C}_2$	Yes
$\mathbb{S}_4 \times \mathbb{S}_3 \times \mathbb{S}_2 \times \mathbb{S}_2$	$\mathbb{C}_0 \times \mathbb{C}_1 \times \mathbb{C}_2 \times \mathbb{C}_2$	No
$\mathbb{S}_4 \times \mathbb{S}_2 \times \mathbb{S}_2 \times \mathbb{S}_2$	$\mathbb{C}_0 \times \mathbb{C}_2 \times \mathbb{C}_2 \times \mathbb{C}_2$	Yes
$\mathbb{S}_4 \times \mathbb{S}_4 \times \mathbb{S}_4 \times \mathbb{S}_1$	$\mathbb{C}_0 \times \mathbb{C}_0 \times \mathbb{C}_0 \times \mathbb{C}_3$	No
$\mathbb{S}_4 \times \mathbb{S}_4 \times \mathbb{S}_3 \times \mathbb{S}_1$	$\mathbb{C}_0 \times \mathbb{C}_0 \times \mathbb{C}_1 \times \mathbb{C}_3$	Yes
$\mathbb{S}_4 \times \mathbb{S}_3 \times \mathbb{S}_3 \times \mathbb{S}_1$	$\mathbb{C}_0 \times \mathbb{C}_1 \times \mathbb{C}_1 \times \mathbb{C}_3$	No
$\mathbb{S}_4 \times \mathbb{S}_4 \times \mathbb{S}_2 \times \mathbb{S}_1$	$\mathbb{C}_0 \times \mathbb{C}_0 \times \mathbb{C}_2 \times \mathbb{C}_3$	No
$\mathbb{S}_4 \times \mathbb{S}_3 \times \mathbb{S}_2 \times \mathbb{S}_1$	$\mathbb{C}_0 \times \mathbb{C}_1 \times \mathbb{C}_2 \times \mathbb{C}_3$	Yes
$\mathbb{S}_4 \times \mathbb{S}_2 \times \mathbb{S}_2 \times \mathbb{S}_1$	$\mathbb{C}_0 \times \mathbb{C}_2 \times \mathbb{C}_2 \times \mathbb{C}_3$	No
$\mathbb{S}_4 \times \mathbb{S}_4 \times \mathbb{S}_1 \times \mathbb{S}_1$	$\mathbb{C}_0 \times \mathbb{C}_0 \times \mathbb{C}_3 \times \mathbb{C}_3$	Yes
$\mathbb{S}_4 \times \mathbb{S}_3 \times \mathbb{S}_1 \times \mathbb{S}_1$	$\mathbb{C}_0 \times \mathbb{C}_1 \times \mathbb{C}_3 \times \mathbb{C}_3$	No
$\mathbb{S}_4 \times \mathbb{S}_2 \times \mathbb{S}_1 \times \mathbb{S}_1$	$\mathbb{C}_0 \times \mathbb{C}_2 \times \mathbb{C}_3 \times \mathbb{C}_3$	Yes
$\mathbb{S}_4 \times \mathbb{S}_1 \times \mathbb{S}_1 \times \mathbb{S}_1$	$\mathbb{C}_0 \times \mathbb{C}_3 \times \mathbb{C}_3 \times \mathbb{C}_3$	No
$\mathbb{S}_3 \times \mathbb{S}_3 \times \mathbb{S}_3 \times \mathbb{S}_3$	$\mathbb{C}_1 \times \mathbb{C}_1 \times \mathbb{C}_1 \times \mathbb{C}_1$	Yes
$\mathbb{S}_3 \times \mathbb{S}_3 \times \mathbb{S}_3 \times \mathbb{S}_2$	$\mathbb{C}_1 \times \mathbb{C}_1 \times \mathbb{C}_1 \times \mathbb{C}_2$	No
$\mathbb{S}_3 \times \mathbb{S}_3 \times \mathbb{S}_2 \times \mathbb{S}_2$	$\mathbb{C}_1 \times \mathbb{C}_1 \times \mathbb{C}_2 \times \mathbb{C}_2$	Yes
$\mathbb{S}_3 \times \mathbb{S}_2 \times \mathbb{S}_2 \times \mathbb{S}_2$	$\mathbb{C}_1 \times \mathbb{C}_2 \times \mathbb{C}_2 \times \mathbb{C}_2$	No
$\mathbb{S}_3 \times \mathbb{S}_3 \times \mathbb{S}_3 \times \mathbb{S}_1$	$\mathbb{C}_1 \times \mathbb{C}_1 \times \mathbb{C}_1 \times \mathbb{C}_3$	Yes
$\mathbb{S}_3 \times \mathbb{S}_3 \times \mathbb{S}_2 \times \mathbb{S}_1$	$\mathbb{C}_1 \times \mathbb{C}_1 \times \mathbb{C}_2 \times \mathbb{C}_3$	No
$\mathbb{S}_3 \times \mathbb{S}_2 \times \mathbb{S}_2 \times \mathbb{S}_1$	$\mathbb{C}_1 \times \mathbb{C}_2 \times \mathbb{C}_2 \times \mathbb{C}_3$	Yes
$\mathbb{S}_3 \times \mathbb{S}_3 \times \mathbb{S}_1 \times \mathbb{S}_1$	$\mathbb{C}_1 \times \mathbb{C}_1 \times \mathbb{C}_3 \times \mathbb{C}_3$	Yes
$\mathbb{S}_3 \times \mathbb{S}_2 \times \mathbb{S}_1 \times \mathbb{S}_1$	$\mathbb{C}_1 \times \mathbb{C}_2 \times \mathbb{C}_3 \times \mathbb{C}_3$	No
$\mathbb{S}_3 \times \mathbb{S}_1 \times \mathbb{S}_1 \times \mathbb{S}_1$	$\mathbb{C}_1 \times \mathbb{C}_3 \times \mathbb{C}_3 \times \mathbb{C}_3$	Yes
$\mathbb{S}_2 \times \mathbb{S}_2 \times \mathbb{S}_2 \times \mathbb{S}_2$	$\mathbb{C}_2 \times \mathbb{C}_2 \times \mathbb{C}_2 \times \mathbb{C}_2$	Yes
$\mathbb{S}_2 \times \mathbb{S}_2 \times \mathbb{S}_2 \times \mathbb{S}_1$	$\mathbb{C}_2 \times \mathbb{C}_2 \times \mathbb{C}_2 \times \mathbb{C}_3$	No
$\mathbb{S}_2 \times \mathbb{S}_2 \times \mathbb{S}_1 \times \mathbb{S}_1$	$\mathbb{C}_2 \times \mathbb{C}_2 \times \mathbb{C}_3 \times \mathbb{C}_3$	Yes
$\mathbb{S}_2 \times \mathbb{S}_1 \times \mathbb{S}_1 \times \mathbb{S}_1$	$\mathbb{C}_2 \times \mathbb{C}_3 \times \mathbb{C}_3 \times \mathbb{C}_3$	No
$\mathbb{S}_1 \times \mathbb{S}_1 \times \mathbb{S}_1 \times \mathbb{S}_1$	$\mathbb{C}_3 \times \mathbb{C}_3 \times \mathbb{C}_3 \times \mathbb{C}_3$	Yes

construction of the LDPC requires $k = 104$). For bit energy E_b and noise power spectral density N_0, we use the block error rate (BLER) as the performance metric, and the average number of error patterns tested as the complexity metric, both assessed as a function of E_b/N_0. In the case of RLCs, each transmitted message was assigned a randomly generated code so that the average performance of RLCs is measured. Let $t = 1, 2, 3$ be the maximum number of errors that a distance-bounded GRAND-based decoder aims to correct.

Fig. 10 depicts the results using RLCs with set partitioning GRAND (SP-GRAND) compared to unconstrained GRAND. All BLERs are identical (all curves overlap), and the complexity curves show that SP-GRAND is substantially faster than unconstrained GRAND. For $t = 1$, only one membership check is needed, as seen in Section 1, as it is well known. For the more interesting $t = 2, 3$, SP-GRAND achieves 68% and 55% reductions if the average number of membership tests, respectively. We note that the results for $t = 2$ considered the sets $\mathcal{F}$ and $\mathcal{N}$, however it was observed that the extra reduction attained by considering the $\mathcal{F} \times \mathcal{N}$ is negligible. Given the extra pre-processing complexity they induce, we recommend these extra partitions to be implemented only for very sparse codes, where a larger $\mathcal{N}$ can arise in $\mathbf{H}_{\text{red}}$.

Fig. 11 compares Sieving-GRAND and unconstrained GRAND for the LDPC code (128,104). The overlapping BLER curves confirm that Sieving-GRAND attains the same performance of unconstrained GRAND, given that all possible error patterns can eventually be tested. It is important to

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2026.3725587

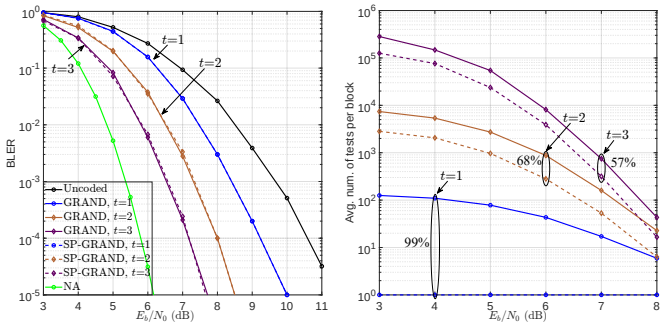


FIGURE 10. **BLER performance (left) and decoding complexity (right) for t =1,2,3, using RLC (128,103) with set partitioning GRAND (SP-GRAND) compared with unconstrained GRAND. The normal approximation bound for codewords $(128, 103)$ is also included.**

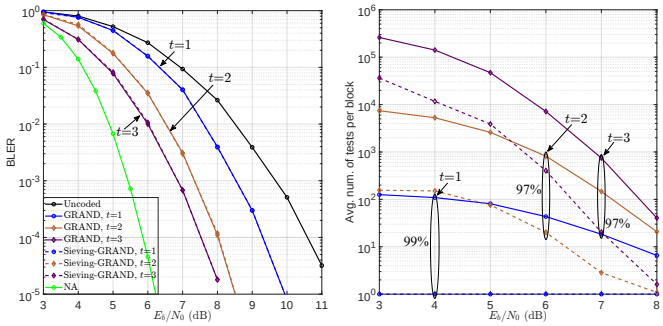


FIGURE 11. **BLER performance (left) and decoding complexity (right) for t =1,2,3, using an LDPC (128,104) with Sieving-GRAND compared with unconstrained GRAND. The normal approximation bound for codewords $(128, 104)$ is also included.**

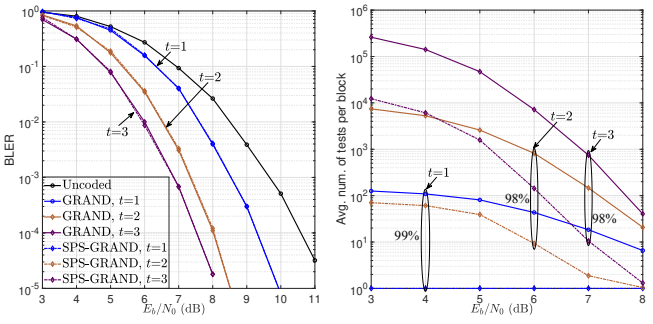


FIGURE 12. **BLER performance (left) and decoding complexity (right) for t =1,2,3, using an LDPC (128,104) with combined set partitioning and Sieving-GRAND (SPS-GRAND) compared with unconstrained GRAND.**

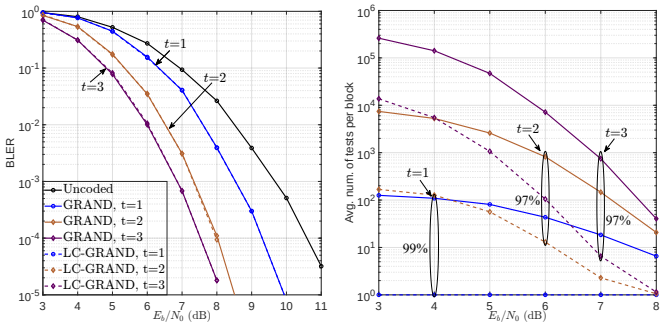


FIGURE 13. **BLER performance (left) and decoding complexity (right) for t =1,2,3, using an LDPC (128,104) with largely constrained GRAND (LC-GRAND) compared with unconstrained GRAND.**

note that the RLCs outperform the similar-size LDPC code (comparing the left figures of Fig. 10 and Fig. 11). The 0.5 dB loss is due to the LDPC code itself rather than to the filtering mechanism applied in the decoding. The code's structure allows for much faster decoding, which is traded off by its slightly less good performance.

The normal approximation (NA) is a metric that is commonly used in the finite blocklenght regime [64], [65], [66, Sec.1.5, App.C.1]. NA bounds have been included in both Fig. 10 and Fig. 11 for the respective code sizes. In Fig. 10 one can observe that the RLC (128,103) gets 1.5 dB close to the NA bound with GRAND or with SP-GRAND (both of which are hard-decoders). In [13], the performance results for RLC (128,106) with soft-GRAND gets 1 dB close to the respective NA. In Fig. 11, one can see that the LDPC (128,104) exhibits a gap of ≈ 2 dB to the respective NA.]

As seen, the Sieving-GRAND approach sharply reduces the number of membership tests needed by unconstrained GRAND, purging $\approx 97\%$ of the original $\sum_{i=0}^{t} \binom{n}{i}$ tests for $t = 2, 3$. This is achieved by considering a set of admissible error positions $\mathcal{P}^{(l)}$, of size $|\mathcal{P}^{(l)}| \ll n$, which includes the correct error positions with a monotonically increasing probability as the iterations unfold. Although in some cases, to find the ML $\hat{\mathbf{e}}$, the threshold has to be lowered down to $w_{\text{th}} = 2$, this happens with low probability, thus the average number of positions considered in the guessing stage remains consistently $\ll n$.

The SP and Sieving techniques can be easily combined by applying SP to the candidate positions filtered at each iteration, leading to the here proposed set partitioning and Sieving-GRAND (SPS-GRAND). The resulting decoding of the LDPC code (128,104) is shown in Fig. 12. As expected, SPS-GRAND has the same BLER as unconstrained GRAND. The complexity comparison reveals that SPS-GRAND outstandingly outperforms GRAND, eliminating on average over 98% of the membership tests, even when looking for $t = 3$ errors.

As shown in Fig. 13, the proposed LC-GRAND constrain for LDPC codes (128,104) significantly enhances the efficiency of error pattern selection by exploiting zero-valued positions in the syndrome. The results show that LC-GRAND also keeps the same decoding performance of GRAND and, even though the mechanism to attain a reduction of error patterns tested being quite different, the number of membership tests is virtually the same as the one attained by Sieving-GRAND. This constraint allows the decoder to eliminate a large portion of infeasible error patterns, thereby reducing the number of membership tests by 97% compared with the unconstrained GRAND approach, while maintaining near-maximum likelihood decoding accuracy.

For a fair comparison with Sieving-GRAND, the first error weight tested by LC-GRAND is also $t = \lceil \frac{||\mathbf{s}||_1}{w_c} \rceil$, given that

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2026.3725587

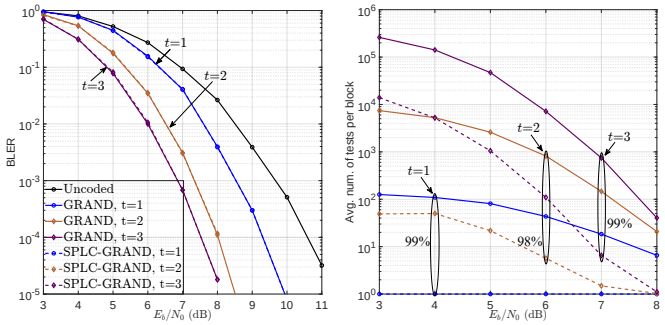


FIGURE 14. BLER performance (left) and decoding complexity (right) for t =1,2,3, using an LDPC (128,104) with combined set partitioning and largely constrained GRAND (SPLC-GRAND) compared with unconstrained GRAND.

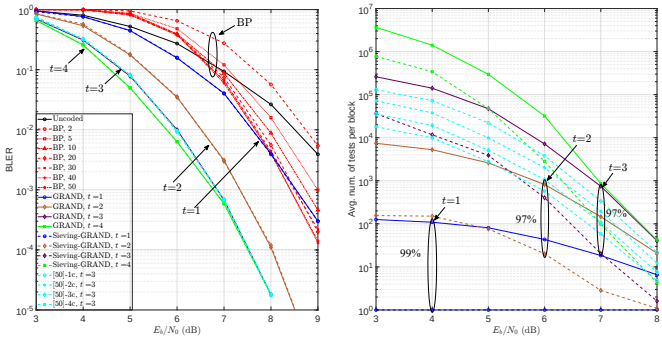


FIGURE 15. BLER performance (left) and decoding complexity (right) for t = 1, 2, 3, 4 using an LDPC (128,104) code, comparing BP decoding with different maximum iteration counts (2, 5, 10, 20, 30, 40, 50), with the proposed Sieving-GRAND (the results for LC-GRAND overlap those and are omitted) and with GRAND, for reference. The constrained decoder proposed in [50] is also included when decoding the same LDPC, and tested with a different number of row-constraints (1, 2, 3, 4).

this is a mechanism that can be applied to any quasi-regular (or regular) LDPC code, as presented in Section IV.A.

Furthermore, when LC-GRAND is combined with set partitioning constraints (referred to as SPLC-GRAND), additional complexity gains are observed. As illustrated in Fig. 14, SPLC-GRAND achieves a dramatic reduction in decoding complexity $\approx 99\%$, leading to a much larger decoding speed without compromising reliability.

As described in Sections III, IV and VI, SP-GRAND, Sieving-GRAND and LC-GRAND have a very lightweight pre-processing per incoming word when compared with a competitor such as OSD, which would involve computing Gaussian elimination for each received word, to obtain the positions of the k most reliable bits. In SP-GRAND one has to obtain $\mathbf{h}_m^{\mathsf{T}}$ from (1). In Sieving-GRAND and LC-GRAND, the pre-processing amounts to computing a vector-matrix product as in (6) or (9), and in the case of Sieving-GRAND, this is complemented by adding more positions to $\mathcal{P}^{(l)}$ at each iteration. In the main three proposed techniques (and also in the syndrome-weight technique applicable to quasi-regular LDPCs), there is no extra complexity when generating the error patterns, and no error pattern is ever generated to be discarded before any membership test is made. All these techniques use light pre-processing to reduce the number of positions where the errors may be and consequently they generate the error patterns by taking Cartesian products of t positions taken from a set of filtered candidate positions, $\hat{\mathcal{E}}_i = \{p_{j_1}, p_{j_2} \dots p_{j_t}\}$, while in unconstrained GRAND one would generate all Cartesian products of up to t sets, each set containing *all* the n positions (expunged from repetitions). Note that, as stated in Subsection III.A.2), each binary error pattern $\hat{\mathbf{e}}_i$ to be tested has 1s in the positions listed in $\hat{\mathcal{E}}_i$.

A. Noise-guessing vs belief-propagation decoding

The belief-propagation (BP) algorithm has been identified quite early as a possible decoder for LDPCs [10]. BP is an iterative message-passing algorithm operating on the Tanner graph of the code, where soft reliability information is exchanged between variable nodes and check nodes. Its decoding performance improves with the number of iterations, at the expense of increased decoding latency and computational effort, and it can reach the ML decoding performance when the LDPC codes are cycle-free. However, short LDPC codes tend to have many short cycles and many small trapping sets [62, Sec.2.7]. For that reason, the performance of short LDPC codes decoded via BP may lie far from the normal approximation (NA), as mentioned in [13], [14]

The performance of BP decoding has been assessed for different numbers of iterations (between 2 and 50) and compared with the one of GRAND and Sieving-GRAND in Fig. 15. Note that all the proposed noise-guessing receivers hold the same performance, and LC-GRAND and Sieving-GRAND lead to undistinguishable complexities; for that reason LC-GRAND is omitted in Fig. 15. In the same figure, we also show the results attained by the decoding technique proposed in [50] for an increasing number of constraints (1, 2, 3, and 4). While the number of *full* membership tests is halved each time an additional constraint is included, one should note that is due to a "pre-filtering" of error patterns by the considered constraints that entails a lighter complexity than a *full* membership test. When considering four constraints that approach can be competitive with Sieving-GRAND and LC-GRAND in at very low E_b/N_0 when many columns of $\mathbf{H}$ are activated by many errors. When looking at the average decoding time in the next Subsection, one can observe that the "pre-filtering" lighter operations add up in a non-negligible manner, contributing to a larger latency.

As expected, the BP algorithm converges extremely fast to a codeword (see next Subsection) when decoding the LDPC (128,104) code. However, it often converges to an incorrect codeword and consequently Fig. 15 exhibits a performance gap of ≈ 1.5 dB to the performance of any of the noise-guessing decoders when they look for error patterns of up to $t = 3$ errors. This performance loss happens even though the BP decoder makes use of soft detection metrics. This is a well-known phenomena when applying BP decoding to

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2026.3725587

short LDPC codes [28]. For LDPC codes of similar length to the tested LDPC, the performance distance in respect to the NA bound has been found to be between 1 dB and 1.75 dB, while OSD can perform 0.5 dB off from the NA curve [13].

A larger number of iterations improves the error-rate performance, but beyond 10 iterations the performance gain is low. Noticeably, even with a rather large number of iterations, BP does not fully achieve the performance attained by GRAND, which attains the ML performance [27], and which is seen to be kept by all the proposed decoders.

For the reference LDPC, with $d_{\min} = 7$, testing error patterns with weight up to $t = 3$ suffices to reach the asymptotic ML hard-decoding performance. Nevertheless, Fig. 15 also includes the performance results for GRAND and Sieving-GRAND testing error patterns up to weight $t = 4$. As expected, the asymptotic performance $t = 3$ and $t = 4$ coincide. However, interestingly, at low SNR, correcting up to $t = 4$ errors slightly reduces the BLER. This is an expected effect: while a minimum distance $d_{\min} = 2t+1$ ensures that a ball of Hamming distance t can exist around any two codewords containing only a codeword at its center and with the two balls not intercepting, in some error instances it is still possible to correct beyond $t = \frac{d_{\min}-1}{2}$ [4, Ch.13].

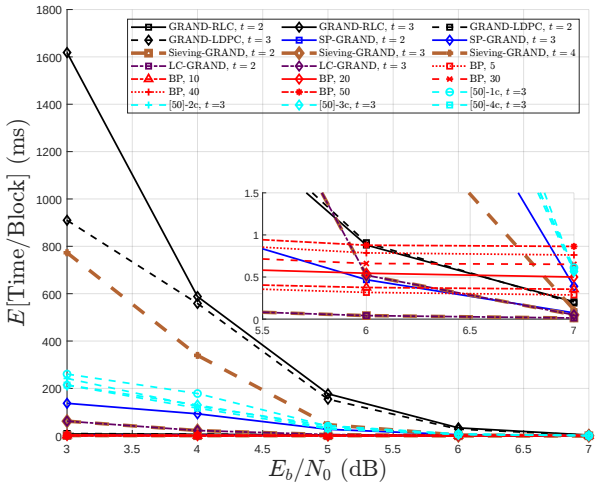


FIGURE 16. Average decoding time per block versus E_b/N_0 for LDPC(128,104) and RLC(128,104) codes for different decoders: the proposed SP-GRAND, Sieving-GRAND and LC-GRAND are compared with unconstrained GRAND and, in the case of the LDPC, also compared with BP decoding for different maximum iteration limits (2, 5, 10, 20, 30, 40, and 50). The noise-guessing decoders are plotted for $t = 2, 3$ (single errors amount to a column search in $\mathbf{H}$), while Sieving-GRAND is evaluated up to $t = 4$ (beyond its correction capability, as in Fig. 15). Note that for $t = 2$ and $t = 3$ the LC-GRAND curves fully overlap the corresponding curves with Sieving-GRAND. The constrained decoder proposed in [50] is also included and tested with a different number of row-constraints (1, 2, 3, 4).

B. Overall decoding time

The proposed decoders rely on some pre-processing based on the syndrome and the parity-check matrix, while the original concept of GRAND in [27] only requires the noise-guessing stage. It is therefore relevant to compare the average total decoding time per codeword taken by the proposed techniques, i.e., including the pre-processing stage, rather than only looking at the number of membership tests. We start by summarizing in Table 6 the complexities of both decoding stages of the most relevant proposed techniques by means of the big-O notation. Note that the types of feasible error patterns in LC-GRAND are entirely known in advance, and fixed for any LDPC code, according to the tables given in Section VI.

For a fair assessment, the total decoding time, including pre-processing and guessing stages, of the main proposed noise-guessing techniques have been measured, along with the decoding time of BP decoding (with a varying number of iterations). The decoding time of the decoder inspired by the idea of segmentation, as presented in the previous subsection, is also included. Such results are shown in Fig. 16. The tests where run using MATLAB in a 12-th gen Intel core i7 2.10 GHz computer (with no explicit parallelization in the code).

As mentioned in the previous Subsection, BP is known to converge very rapidly to a codeword and is therefore significantly faster than GRAND-based decoders at low E_b/N_0; however, it often does *not* converge to the ML codeword. Increasing the maximum number of iterations up to 20 in BP improves the error-rate performance, however, beyond that number no performance gain is noticeable, as seen in Fig. 15. Also as expected, at low E_b/N_0 more iterations are required before convergence or termination. Even so, for $E_b/N_0 > 6$ dB, Sieving-GRAND and LC-GRAND (considering error patterns up to $t = 3$) can even decode faster than the BP algorithm when it runs more than 20 iterations, and beyond 7 dB they even get faster than BP running 5 iterations. This is due to the very low number of error patterns tested when the noise entropy is low (and also because the proposed decoders have a minimal pre-processing per codeword). In Fig. 16 one can also observe that the Sieving- and LC-GRAND decoders hold the same decoding times. This happens due to i) their very similar pre-processing and ii) due to an indistinguishable number of membership tests, as observed in the complexity curves in the previous Subsection. Interestingly, the average decoding time of the proposed Sieving-GRAND and LC-GRAND is lower than the decoding times achieved by [50] for all the tested number of constraints. This happens even though, as seen in Fig. 15, when applying 4 constraints, the number of queries attained by the technique proposed in [50] being lower when the E_b/N_0 is very low. However, the computation of the "pre-filtering" of error patterns via the constraints do not translate into a lower decoding time. This is a clear example that complexity comparisons should not only rely on the number of membership tests.

C. Constrained GRAND techniques with coded 16-QAM

The proposed set partitioning and Sieving-GRAND (SPS-GRAND) operates at a bit level on the received LDPC codewords; nevertheless, the *M*-arity of the modulation mod-

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2026.3725587

TABLE 6. Worst-case complexity comparison (big-O per received block) for the main decoding techniques correcting up to t errors.

Decoder	Preprocessing	Membership tests (worst-case)
Unconstrained GRAND	None	$\sum_{i=0}^{t} \binom{n}{i} = O(n^t)$
SP-GRAND	$O\left(n(n-k)\right)$ (find min-weight row of $\mathbf{H}_{\text{red}}$)	$\sum_{\substack{i=1 \\ i \text{ odd}}}^{t} \binom{\lvert S_1^{(m)}\rvert}{i}\binom{\lvert S_0^{(m)}\rvert}{t-i}$ with $\lvert S_1^{(m)}\rvert + \lvert S_0^{(m)}\rvert = n$, $O(n^t)$ worst-case
Sieving-GRAND	$O(n(n-k))$ (compute $\mathbf{s}^T\mathbf{H}$, as in (6))	$\sum_{l} \binom{\lvert\mathcal{P}^{(l)}\rvert}{t}$, $\lvert\mathcal{P}^{(l)}\rvert \le n$ avg. $\lvert\mathcal{P}^{(l)}\rvert \ll n$
LC-GRAND	$O(n(n-k))$ (compute $w_c\mathbb{1}_n - \mathbf{s}^T\mathbf{H}$, as in (9))	Feasible t-fold products from $\mathbb{C}_i$ sets (see Tables 3–5), $O(n^t)$ bound

Note: $\lvert S_1^{(m)}\rvert$ and $\lvert S_0^{(m)}\rvert$: no. of ones/zeros in min-weight row of $\mathbf{H}_{\text{red}}$; $\lvert\mathcal{P}^{(l)}\rvert$: candidate position set size at iteration l; w_c: constant column weight.

ifies the performance of a coded system. Here, we present the BLER and complexity results when using the same coding schemes, now with 16-QAM (quadrature amplitude modulation; $M = 16$) with Gray mapping. The results are shown in Fig. 17, Fig. 18, Fig.19, Fig.20, and Fig. 21 in the same order as in the main text. Fig. 17 illustrates the performance and decoding complexity of RLC (128,103) using set partitioning GRAND (SP-GRAND), in comparison to unconstrained GRAND. The BLER remains identical for SP-GRAND and unconstrained GRAND across the analyzed range of E_b/N_0, with the curves overlapping. As with $M = 2$ or $M = 4$ (BPSK or QPSK), with 16-QAM SP-GRAND also significantly reduces the number of tests needed with unconstrained GRAND: for $t = 2, 3$ the savings are over 70%. (In the case with $t = 1$, as seen in Section 1, only a direct comparison of $\mathbf{s}$ with all columns $\mathbf{H}(:, j)$ is needed, and only one membership test is involved.)

Fig. 18 compares performances and decoding complexities of Sieving-GRAND with the original unconstrained GRAND, when decoding the LDPC code (128,104). The BLER curves for both Sieving-GRAND and GRAND perfectly overlap, indicating that in this setup Sieving-GRAND also achieves the ML decoding performance. As observed, Sieving-GRAND significantly reduces the number of membership tests required by unconstrained GRAND, eliminating over 95% of the original $\sum_{i=0}^{t} \binom{n}{i}$ membership tests.

The decoded output of mixing set partitioning GRAND and Sieving-GRAND for an LDPC code (128,104) is shown in Fig. 19. As anticipated, within the specified E_b/N_0 range, SPS-GRAND holds the same BLER as unconstrained GRAND. Regarding complexity savings, the results also show the superiority of SPS-GRAND, ditching over 98% of the membership tests needed on average.

Fig. 20 compares the performances and decoding complexity of the LC-GRAND and the original unconstrained GRAND for $t = 1, 2, 3$, using an LDPC code (128, 104). Without sacrificing performance, LC-GRAND significantly reduces the number of error patterns that are tested.

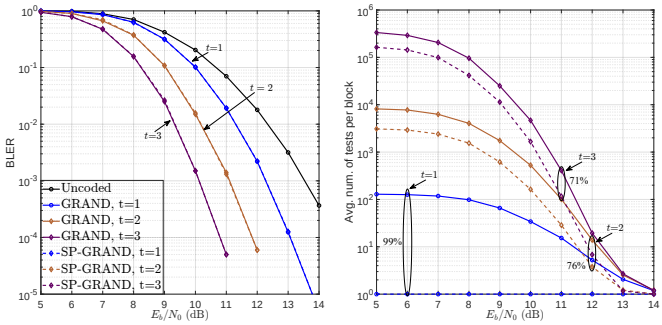


FIGURE 17. BLER performance (left) and decoding complexity (right) for t =1,2,3, using RLC (128,103) and 16-QAM, with set partitioning GRAND (SP-GRAND) compared with unconstrained GRAND.

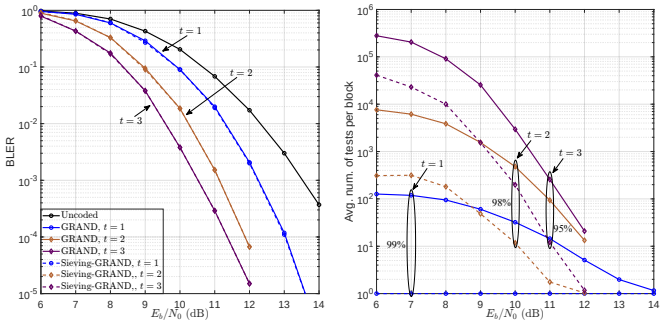


FIGURE 18. BLER performance (left) and decoding complexity (right) for t =1,2,3, using an LDPC (128,104) and 16-QAM, with Sieving-GRAND compared with unconstrained GRAND.

The performance and complexity results for $t = 1, 2, 3$, using an LDPC code (128, 104) with combined set partitioning and LC-GRAND (i.e., SPLC-GRAND), are compared with unconstrained GRAND in Fig. 21. The BLER performance remains identical, while the decoding complexity is reduced by approximately 99% compared with unconstrained GRAND.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2026.3725587

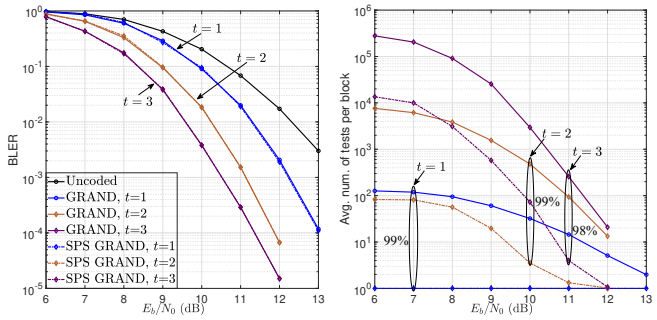


FIGURE 19. BLER performance (left) and decoding complexity (right) for t =1,2,3, using an LDPC (128,104) and 16-QAM, with combined set partitioning and Sieving GRAND (SPS-GRAND) compared with unconstrained GRAND.

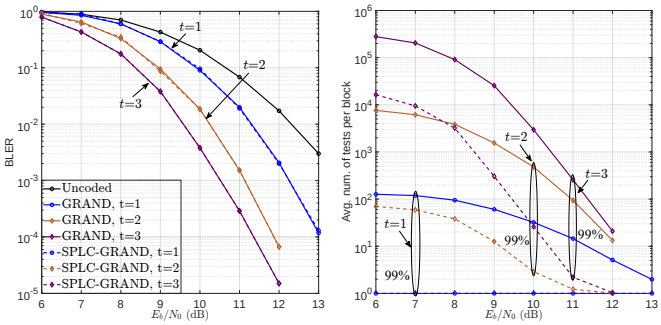


FIGURE 21. BLER performance (left) and decoding complexity (right) for t =1,2,3, using an LDPC (128,104) and 16-QAM with combined set partitioning and largely constrained GRAND compared with unconstrained GRAND.

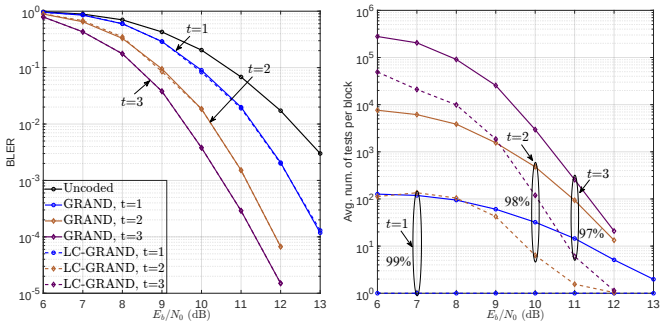


FIGURE 20. BLER performance (left) and decoding complexity (right) for t =1,2,3, using an LDPC (128,104) and 16-QAM with largely constrained GRAND compared with unconstrained GRAND.

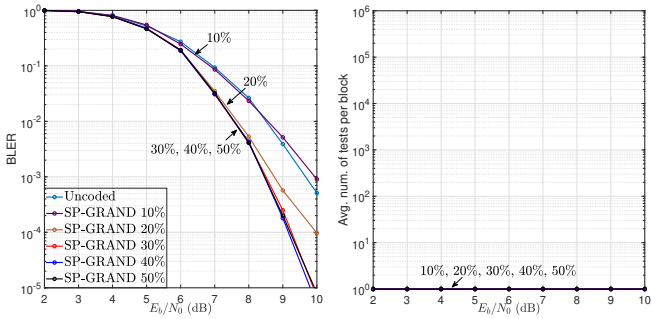


FIGURE 22. BLER performance (left) and decoding complexity (right) for the number of error t=1, using RLC (128,103) and BPSK, with p-sparse (indicated as a percentage) parity-check matrices, using set partitioning GRAND.

VIII. CONCLUSION

This paper starts by proposing a simple constraint for the decoding of any LBC. The constraint splits the positions of the errors into sets, allowing one to construct the error patterns by means of the Cartesian product between sets. With this set partitioning the complexity reduction is more accentuated with sparser codes. By using short quasi-regular LDPC codes, one can retain the benefits of sparsity without incurring a severe performance loss. For such codes, a simple filtering technique is used to greatly reduce the candidate positions where the error positions may lie. The technique is based on the "fingerprints" that the active columns of the parity-check matrix leave on the observed syndrome. This technique starts by selecting a small pool of positions where the errors most likely lie, and then GRAND runs on those few positions only. This is done iteratively, allowing more positions to be added to that pool until the ML error pattern is found. The number of these iterations is bounded by the weight of the columns of the parity-check matrix.

The third decoding approach proposed, the largely constrained mechanism, applies a more straightforward constraint to any short LDPC code (not necessarily quasi-regular), and can be extended to any short linear block code – which is suggested as a natural extension of the present work. In this case, the decoder avoids testing infeasible error patterns. It only tests error patterns derived from valid Cartesian products that are computed *a priori*, ditching the need of an iterative process as the one in the Sieving approach. This enhanced decoding strategy reduces computational complexity also without compromising performance, particularly when combined with set partitioning constraints. It reduced the membership tests to $\approx 1\%$ of those required by GRAND. We conclude by highlighting the pressing need for ML-performance decoders for short LDPC codes in the context of quantum error correction, where LDPC-based quantum codes are emerging as a practical solution [67]–[69], and where noise-guessing decoders may play a relevant role [41], [43]. The presented techniques can be of interest to tackle the present challenge of decoding quantum LDPCs.

Appendix: Set partitioning GRAND with sparse RLCs

The number of candidate error patterns constructed via Cartesian products of two position sets gets smaller when the entropy of one of the sets is lower; fewer elements in a set provide more information about the errors' locations. Sparse RLCs necessarily verify the objective of having a low minimum support row $\mathbf{h}_m^{\mathsf{T}}$. We assessed the performance and the complexity of SP-GRAND for RLCs (128, 103) with different sparsity values in their parity-check matrix, as shown in Fig. 22, Fig. 23, and Fig. 24 for bounded-decoders with $t = 1, 2, 3$, respectively. All figures include the performances of uncoded transmissions as well as the BLER and complexity results with SP-GRAND. With a sparser

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2026.3725587

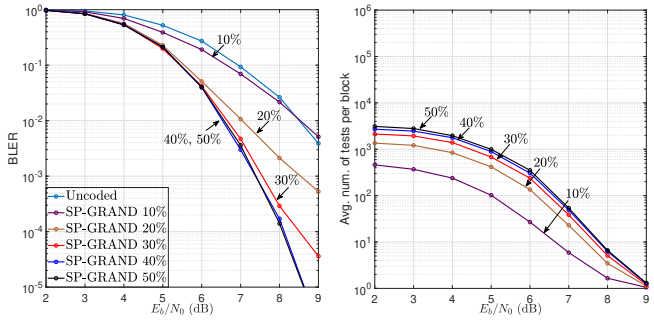


FIGURE 23. BLER performance (left) and decoding complexity (right) for the number of errors t =2, using RLC (128,103) and BPSK, with p-sparse (indicated as a percentage) parity-check matrices, using set partitioning GRAND.

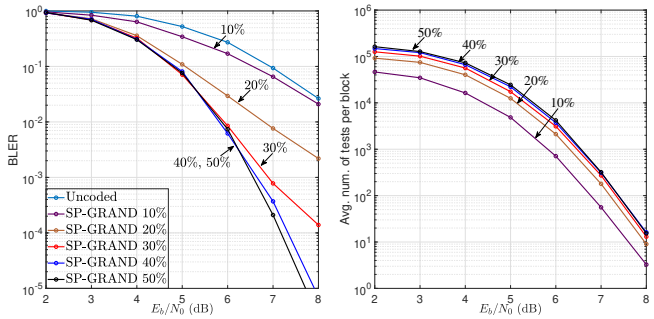


FIGURE 24. BLER performance (left) and decoding complexity (right) for the number of errors t =3, using RLC (128,103) and BPSK, with p-sparse (indicated as a percentage) parity-check matrices, using set partitioning GRAND.

parity-check matrix, the weight of the minimum support row is lower and faster decoding is possible, however, at the expense of a severe deterioration of the codes' performance, as one would expect (as they lack the proper construction of LDPC codes). It should be noted that the very sparse RLC codes can lead to a BLER even higher than the one of the uncoded transmission. This is because the added redundant bits contribute to more errors and the energy spent in their transmission is assigned to E_b, while this extra redundancy does not contribute to a codeword correction that compensates for that extra spent energy.

Acknowledgment

Francisco Monteiro thanks Dr. Bruno Coutinho (German Aerospace Center – DLR), and Prof. Marco Gomes and Prof. Vitor Silva (University of Coimbra) for discussions on set partitioning and sieving techniques. Carlos Salema thanks Prof. Paulo Mateus (IST, University of Lisbon) for discussions on low-complexity decoding techniques of block codes.

REFERENCES

[1] C. E. Shannon, "A mathematical theory of communication," *The Bell Syst. Tech. J.*, vol. 27, no. 3, pp. 379–423, Jul. 1948. doi: 10.1002/j.1538-7305.1948.tb01338.x

[2] ——, "Probability of error for optimal codes in a gaussian channel," *The Bell Syst. Tech. J.*, vol. 38, no. 3, pp. 611–656, May. 1959. doi: 10.1002/j.1538-7305.1959.tb03905.x

[3] L. V. Truong, G. Cocco, J. Font-Segura, and A. Guillén i Fàbregas, "Concentration properties of random codes," *IEEE Trans. Inf. Theory*, vol. 69, no. 12, pp. 7499–7537, Dec. 2023. doi: 10.1109/TIT.2023.3312326

[4] D. MacKay, *Inf. Theory, Inference and Learning Algorithms*. Cambridge, UK: Cambridge Univ. Press, 2003.

[5] R. Gallager, "The random coding bound is tight for the average code (corresp.)," *IEEE Trans. Inf. Theory*, vol. 19, no. 2, p. 244–246, Mar. 1973. doi: 10.1109/TIT.1973.1054971

[6] J. Coffey and R. Goodman, "Any code of which we cannot think is good," *IEEE Trans. Inf. Theory*, vol. 36, no. 6, pp. 1453–1461, Nov. 1990. doi: 10.1109/18.59944

[7] D. J. Costello and G. D. Forney, "Channel coding: The road to channel capacity," *Proc. of the IEEE*, vol. 95, no. 6, pp. 1150–1177, Jun. 2007. doi: 10.1109/JPROC.2007.895188

[8] R. Gallager, "Low-density parity-check codes," *IEEE Trans. Inf. Theory*, vol. 8, no. 1, pp. 21–28, Jan. 1962. doi: 10.1109/TIT.1962.1057683

[9] D. MacKay and R. Neal, "Good codes based on very sparse matrices," in *Cryptography and Coding 5th IMA Conf.*, ser. Lecture Notes in Computer Science, C. Boyd, Ed., no. 1025, 1995, pp. 100–111. doi: 10.1007/3-540-60693-9_13

[10] D. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Trans. Inf. Theory*, vol. 45, no. 2, pp. 399–431, Mar. 1999. doi: 10.1109/18.748992

[11] W. An, M. Médard, and K. R. Duffy, "Keep the bursts and ditch the interleavers," *IEEE Trans. Commun.*, vol. 70, no. 6, pp. 3655–3667, Jun. 2022. doi: 10.1109/TCOMM.2022.3171798

[12] X. Wu, M. Jiang, C. Zhao, L. Ma, and Y. Wei, "Low-rate PBRL-LDPC codes for URLLC in 5G," *IEEE Wireless Commun. Lett.*, vol. 7, no. 5, pp. 800–803, Oct. 2018. doi: 10.1109/LWC.2018.2825988

[13] C. Yue, V. Miloslavskaya, M. Shirvanimoghaddam, B. Vucetic, and Y. Li, "Efficient decoders for short block length codes in 6G URLLC," *IEEE Commun. Mag.*, vol. 61, no. 4, pp. 84–90, Apr. 2023. doi: 10.1109/MCOM.001.2200275

[14] M. Shirvanimoghaddam, M. S. Mohammadi, R. Abbas, A. Minja, C. Yue, B. Matuz, G. Han, Z. Lin, W. Liu, Y. Li, S. Johnson, and B. Vucetic, "Short block-length codes for ultra-reliable low latency communications," *IEEE Commun. Mag.*, vol. 57, no. 2, pp. 130–137, Feb. 2019. doi: 10.1109/MCOM.2018.1800181

[15] K. Arora, J. Singh, and Y. S. Randhawa, "A survey on channel coding techniques for 5G wireless networks," *Telecommun. Syst.*, vol. 73, no. 4, p. 637–663, Apr. 2020. doi: 10.1007/s11235-019-00630-3

[16] Y. Polyanskiy, H. V. Poor, and S. Verdu, "Channel coding rate in the finite blocklength regime," *IEEE Trans. Inf. Theory*, vol. 56, no. 5, pp. 2307–2359, May. 2010. doi: 10.1109/TIT.2010.2043769

[17] C. Salema, *Microwave Radio Links: From Theory to Design*. London, UK: Wiley, 2008.

[18] S. Dolinar, D. Divsalar, and F. Pollara, "Code performance as a function of block size," TMO Progress Report, JPL-NASA, techreport 42-133, May. 1998.

[19] M. Fossorier and S. Lin, "Soft-decision decoding of linear block codes based on ordered statistics," *IEEE Trans. Inf. Theory*, vol. 41, no. 5, pp. 1379–1396, Sep. 1995. doi: 10.1109/18.412683

[20] S. Alnawayseh and P. Loskot, "Ordered statistics-based list decoding techniques for linear binary block codes," *EURASIP J. Wireless Commun. Netw.*, vol. 2012, no. 314, pp. 1–12, Oct. 2012. doi: 10.1186/1687-1499-2012-314

[21] C. Kim, D. Kam, S. Kim, G. Park, and Y. Lee, "Simplified ordered statistic decoding for short-length linear block codes," *IEEE Commun. Lett.*, vol. 26, no. 8, pp. 1720–1724, Aug. 2022. doi: 10.1109/LCOMM.2022.3176646

[22] J. Liang, Y. Wang, S. Cai, and X. Ma, "A low-complexity ordered statistic decoding of short block codes," *IEEE Commun. Lett.*, vol. 27, no. 2, pp. 400–403, Feb. 2023. doi: 10.1109/LCOMM.2022.3222819

[23] L. Yang, J. Zhao, X. Li, L. Chen, H. Zhang, and J. Tong, "Efficient ordered statistics decoding of BCH codes without Gaussian elimination," *IEEE Trans. Inf. Theory*, vol. 71, no. 11, pp. 8294–8311, Nov. 2025. doi: 10.1109/TIT.2025.3613748

[24] B. Dorsch, "A decoding algorithm for binary block codes and J-ary output channels (corresp.)," *IEEE Trans. Inf. Theory*, vol. 20, no. 3, pp. 391–394, May 1974. doi: 10.1109/TIT.1974.1055217

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2026.3725587

[25] Y. Wang, J. Liang, and X. Ma, "Local constraint-based ordered statistics decoding for short block codes," in *2022 IEEE Inf. Theory Workshop (ITW)*, Mumbai, India, Nov. 2022, pp. 107–112. doi: 10.1109/ITW54588.2022.9965916

[26] M. Li, V. Derudder, K. Bertrand, C. Desset, and A. Bourdoux, "High-speed LDPC decoders towards 1 Tb/s," *IEEE Trans. Circuits Syst. I: Regul. Pap.*, vol. 68, no. 5, pp. 2224–2233, May. 2021. doi: 10.1109/TCSI.2021.3060880

[27] K. R. Duffy, J. Li, and M. Médard, "Capacity-achieving guessing random additive noise decoding," *IEEE Trans. Inf. Theory*, vol. 65, no. 7, pp. 4023–4040, Jul. 2019. doi: 10.1109/TIT.2019.2896110

[28] G. Li and X. Yu, "Boosting ordered statistics decoding of short LDPC codes with simple neural network models," *IEEE Commun. Lett.*, vol. 28, no. 12, pp. 2714–2718, Dec. 2024. doi: 10.1109/LCOMM.2024.3475874

[29] S. M. Abbas, M. Jalaleddine, and W. J. Gross, *Guessing Random Additive Noise Decoding: A Hardware Perspective*. Switzerland: Springer, 2023.

[30] K. R. Duffy, J. Li, and M. Médard, "Guessing noise, not code-words," in *Proc. of IEEE Int. Symp on Inf. Theory (ISIT)*, Vail, CO, USA, Jun. 2018, pp. 671–675. doi: 10.1109/ISIT.2018.8437648

[31] K. R. Duffy, W. An, and M. Médard, "Ordered reliability bits guessing random additive noise decoding," *IEEE Trans. Signal Process.*, vol. 70, pp. 4528–4542, Aug. 2022. doi: 10.1109/TSP.2022.3203251

[32] S. M. Abbas, M. Jalaleddine, and W. J. Gross, "List-GRAND: A practical way to achieve maximum likelihood decoding," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 31, no. 1, pp. 43–54, Jan. 2023. doi: 10.1109/TVLSI.2022.3223692

[33] H. Sarieddeen, P. Yuan, M. Médard, and K. R. Duffy, "Soft-input, soft-output joint data detection and GRAND: A performance and complexity analysis," in *Proc. of IEEE Int. Symp on Inf. Theory (ISIT)*, Taipei, Taiwan, Jun. 2023, pp. 1090–1095. doi: 10.1109/ISIT54713.2023.10206823

[34] S. M. Abbas, M. Jalaleddine, and W. J. Gross, "GRAND for Rayleigh fading channels," in *Proc. of IEEE Globecom Workshops (GC Wkshps)*, Rio de Janeiro, Brazil, Dec. 2022, pp. 504–509. doi: 10.1109/GCWkshps56602.2022.10008698

[35] S. Allahkaram, F. A. Monteiro, and I. Chatzigeorgiou, "URLLC with coded massive MIMO via random linear codes and GRAND," in *Proc. of IEEE 96th Vehic. Tech. Conf. (VTC2022-Fall)*, London, UK, Sep. 2022, pp. 1–5. doi: 10.1109/VTC2022-Fall57202.2022.10012803

[36] ——, "Constrained symbol-level noise-guessing decoding with antenna sorting for massive MIMO," *IEEE Open J. Commun. Soc.*, vol. 6, pp. 9486–9503, Nov. 2025. doi: 10.1109/OJCOMS.2025.3630551

[37] A. Riaz, V. Bansal, A. Solomon, W. An, Q. Liu, K. Galligan, K. R. Duffy, M. Médard, and R. T. Yazicigil, "Multi-code multi-rate universal maximum likelihood decoder using GRAND," in *In Proc. of Eur. Solid State Circuits. Conf. (ESSCIRC)*, Grenoble, France, Sep. 2021, pp. 239–246. doi: 10.1109/ESSCIRC53450.2021.9567867

[38] C. Condo, "A fixed latency ORBGRAND decoder architecture with LUT-aided error-pattern scheduling," *IEEE Trans. Circuits Syst. I: Regul. Pap.*, vol. 69, no. 5, pp. 2203–2211, May. 2022. doi: 10.1109/TCSI.2022.3150583

[39] A. Riaz, A. Yasar, F. Ercan, W. An, J. Ngo, K. Galligan, M. Médard, K. R. Duffy, and R. T. Yazicigil, "A sub-0.8pj/b 16.3gbps/mm2 universal soft-detection decoder using ORBGRAND in 40nm CMOS," in *2023 IEEE Int. Solid-State Circuits Conf. (ISSCC)*, San Francisco, CA, USA, Feb. 2023, pp. 432–434. doi: 10.1109/ISSCC42615.2023.10067519

[40] I. Chatzigeorgiou, "Transversal GRAND for network coded data," in *Proc. of IEEE Int. Symp on Inf. Theory (ISIT)*, Espoo, Finland, Jun. 2022, pp. 1773–1778. doi: 10.1109/ISIT50566.2022.9834692

[41] D. Cruz, F. A. Monteiro, and B. C. Coutinho, "Quantum error correction via noise guessing decoding," *IEEE Access*, vol. 11, pp. 119 446–119 461, Oct. 2023. doi: 10.1109/ACCESS.2023.3327214

[42] D. Chandra, Z. B. Kaykac Egilmez, Y. Xiong, S. X. Ng, R. G. Maunder, and L. Hanzo, "Universal decoding of quantum stabilizer codes via classical guesswork," *IEEE Access*, vol. 11, pp. 19 059–19 072, Feb. 2023. doi: 10.1109/ACCESS.2023.3247966

[43] D. Cruz, F. A. Monteiro, A. Roque, and B. C. Coutinho, "Fault-tolerant noise guessing decoding of quantum random codes," *IEEE Trans. Quantum Eng.*, vol. 6, pp. 1–26, Aug. 2025. doi: 10.1109/TQE.2025.3595778

[44] A. Roque, D. Cruz, F. A. Monteiro, and B. C. Coutinho, "Efficient entanglement purification based on noise guessing decoding," *Quantum*, vol. 8, p. 1476, Sep. 2024. doi: 10.22331/q-2024-09-19-1476

[45] W. An, M. Médard, and K. R. Duffy, "CRC codes as error correction codes," in *IEEE Int. Conf. on Comm. (ICC)*, Montreal, QC, Canada, Jul. 2021, pp. 1–6. doi: 10.1109/ICC42927.2021.9500279

[46] A. Cohen, R. G. L. D'Oliveira, K. R. Duffy, J. Woo, and M. Medard, "AES as error correction: Cryptosystems for reliable communication," *IEEE Commun. Lett.*, vol. 27, no. 8, pp. 1964–1968, Aug. 2023. doi: 10.1109/LCOMM.2023.3285404

[47] M. Médard and K. R. Duffy, "Physical layer insecurity," in *Proc. 57th Annu. Conf. Inf. Sci. Syst. (CISS)*, Baltimore, MD, USA, Mar. 2023. doi: 10.1109/CISS56502.2023.10089749

[48] I. Chatzigeorgiou and F. A. Monteiro, "Symbol-level GRAND for high-order modulation over block fading channels," *IEEE Commun. Lett.*, vol. 27, no. 2, pp. 447–451, Feb. 2023. doi: 10.1109/LCOMM.2022.3227593

[49] J. Griffin, P. Yuan, K. R. Duffy, and M. Médard, "Using a single parity-check to reduce the guesswork of guessing codeword decoding," in *Proc. of 59th Annual Conf. on Information Sciences and Systems (CISS)*, Baltimore, MD, USA, Mar. 2025, pp. 1–6. doi: 10.1109/CISS64860.2025.10944685

[50] M. Rowshan and J. Yuan, "Constrained error pattern generation for GRAND," in *Proc. of IEEE Int. Symp on Inf. Theory (ISIT)*, Espoo, Finland, Jun. 2022, pp. 1767–1772. doi: 10.1109/ISIT50566.2022.9834343

[51] Y. Wang, J. Liang, and X. Ma, "Partially constrained GRAND of linear block codes," in *Proc. of 9th International Conference on Computer and Communication Systems (ICCCS)*, Apr. 2024, pp. 617–622. doi: 10.1109/ICCCS61882.2024.10602848

[52] R. Hadavian, D. Truhachev, K. El-Sankary, H. Ebrahimzad, and H. Najafi, "Ordered reliability direct error pattern testing ORDEPT algorithm," in *Proc. of IEEE Global Commun. Conf.*, Dec. 2023, pp. 6983–6988. doi: 10.1109/GLOBECOM54140.2023.10437894

[53] X. Ma, "Guessing what, noise or codeword?" in *Proc. of IEEE Inf. Theory Workshop (ITW)*, Shenzhen, China, Nov. 2024, pp. 460–465. doi: 10.1109/ITW61385.2024.10806987

[54] M. Rowshan and J. Yuan, "Low-complexity GRAND by segmentation," in *Proc. of 2023 IEEE Global Communications Conference (GLOBECOM)*, Kuala Lumpur, Malaysia, Dec. 2023, pp. 6145–6151. doi: 10.1109/GLOBECOM54140.2023.10436895

[55] ——, "Segmented GRAND: Complexity reduction through sub-pattern combination," *IEEE Trans. Commun.*, vol. 73, no. 8, pp. 5607–5620, Aug. 2025. doi: 10.1109/TCOMM.2025.3541094

[56] P. Yuan, M. Médard, K. Galligan, and K. R. Duffy, "Soft-output (SO) GRAND and iterative decoding to outperform LDPC codes," *IEEE Trans. Wireless Commun.*, vol. 24, no. 4, pp. 3386–3399, Jan. 2025. doi: 10.1109/TWC.2025.3530880

[57] Q. Wang, J. Liang, P. Yuan, K. R. Duffy, M. Médard, and X. Ma, "Guessing decoding of short blocklength codes," *arXiv:2511.12108 [cs.IT]*, Nov. 2025. [Online]. Available: https://arxiv.org/abs/2511.12108

[58] M. Geiselhart, F. Krieg, J. Clausius, D. Tandler, and S. ten Brink, "6G: A welcome chance to unify channel coding?" *IEEE BITS the Information Theory Magazine*, vol. 3, no. 1, pp. 67–80, Mar. 2023. doi: 10.1109/MBITS.2023.3322974

[59] I. Tal and A. Vardy, "List decoding of polar codes," *IEEE Trans. Inf. Theory*, vol. 61, no. 5, pp. 2213–2226, May. 2015. doi: 10.1109/TIT.2015.2410251

[60] F. Monteiro, "Lattices in MIMO spatial multiplexing: Detection and geometry," Ph.D. dissertation, Univ. of Cambridge, UK, 2012.

[61] Z. Sun, C. Gu, and Y. Zheng, "A review of sieve algorithms in solving the shortest lattice vector problem," *IEEE Access*, vol. 8, pp. 190 475–190 486, Oct. 2020. doi: 10.1109/ACCESS.2020.3031276

[62] T. Richardson and R. Urbanke, *Modern Coding Theory*. Cambridge, UK: Cambridge Univ. Press, 2012.

[63] K. D. Rao, *Channel Coding Techniques for Wireless Communications*. London, UK: Springer, 2016.

[64] T. Erseghe, "Coding in the finite-blocklength regime: Bounds based on laplace integrals and their asymptotic approximations," *IEEE Transactions on Information Theory*, vol. 62, no. 12, pp. 6854–6883, Dec. 2016. doi: 10.1109/TIT.2016.2616900

[65] M. Rowshan, M. Qiu, Y. Xie, X. Gu, and J. Yuan, "Channel coding toward 6G: Technical overview and outlook," *IEEE Open Journal of*

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

This article has been accepted for publication in IEEE Open Journal of the Communications Society. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/OJCOMS.2026.3725587

the Communications Society, vol. 5, pp. 2585–2685, Apr. 2024. doi: 10.1109/OJCOMS.2024.3390000

[66] M. Rowshan and E. Viterbo, *Polar Codes: From Theory to Practice*. New Jersey, USA: Wiley, 2026.

[67] Google Quantum AI and Collaborators, "Quantum error correction below the surface code threshold," *Nature*, vol. 638, no. 8052, pp. 920–926, Feb. 2025. doi: 10.1038/s41586-024-08449-y

[68] S. Bravyi, A. W. Cross, J. M. Gambetta, D. Maslov, P. Rall, and T. J. Yoder, "High-threshold and low-overhead fault-tolerant quantum memory," *Nature*, vol. 627, no. 8005, pp. 778–782, Mar. 2024. doi: 10.1038/s41586-024-07107-7

[69] O. Ferraz, B. Coutinho, G. Falcão, M. Gomes, F. A. Monteiro, and V. Silva, "GPU-accelerated syndrome decoding for quantum LDPC codes below the 63 µs latency threshold"," in *Proc. of the 59th Asilomar Conf. on Signals, Systems, and Computers*, Pacific Grove, CA, USA, Oct. 2025, pp. 1878–1882. doi: 10.1109/IEEECONF67917.2025.11443687



Sahar Allahkaram is a PhD student in the Dep. of Information Science and Technology at Iscte - University Institute of Lisbon, Portugal. She is working on Signal Processing and Coding Techniques for 6G Ultra-reliable Low-latency Wireless Machine-type Communications. She obtained her MSc in Aerospace Engineering from Sapienza University of Rome, Italy, in 2020 and her BSc in Electronic Engineering from Azad University of Tehran, Iran, in 2015.



Francisco A. Monteiro (M'07) is an Associate Professor in the Dep. of Information Science and Technology at Iscte - University Institute of Lisbon, Portugal, and a researcher at Instituto de Telecomunicações, Lisbon, Portugal. He holds a PhD from the University of Cambridge, U.K., and the Licenciatura and MSc degrees in Electrical and Computer Engineering from Instituto Superior Técnico (IST), University of Lisbon, Portugal, where he also became a Teaching Assistant. He held visiting research positions at the Universities of Toronto (Canada), Lancaster (U.K.), Oulu (Finland), and Pompeu Fabra (Barcelona, Spain). He has won two best paper prizes awards at IEEE conferences (2004 and 2007), a Young Engineer Prize (3rd place) from the Portuguese Engineers Institution (Ordem dos Engenheiros) in 2002, and for two years in a row was a recipient of Exemplary Reviewer Awards from the IEEE Wireless Communications Letters (in 2014 and 2015). He co-edited the book "MIMO Processing for 4G and Beyond: Fundamentals and Evolution", published by CRC Press in 2014. In 2016 he was the Lead Guest Editor of a special issue on Network Coding of the EURASIP Journal on Advances in Signal Processing. He was a general chair of ISWCS 2018 - The 15th International Symposium on Wireless Communication Systems, an IEEE major conference in wireless communications.



João Ribeiro is currently an Assistant Professor with the Department of Mathematics, Instituto Superior Técnico (IST), University of Lisbon, Portugal, and a Researcher at the Instituto de Telecomunicações, Lisbon, Portugal. He received the B.Sc. degree in applied mathematics and computation from IST, Universidade de Lisboa in 2015, the M.Sc. degree in computer science from the Swiss Federal Institute of Technology (ETH), Zürich, Switzerland, in 2017, and the Ph.D. degree in computing from Imperial College London, U.K., in 2021. Previously, he was an Assistant Professor with the Department of Computer Science, NOVA School of Science and Technology, Caparica, Portugal; a Researcher at NOVA LINCS, Caparica; and a Post-Doctoral Fellow with the Computer Science Department, Carnegie Mellon University, Pittsburgh, PA, USA. His research interests include coding and information theory, cryptography, and pseudorandomness. In 2025 he was the recipient of a Starting Grant from the European Research Council (ERC) for research on coding against synchronization errors.



Carlos Salema (Life Member, IEEE) He is a Professor Emeritus at Instituto Superior Técnico (IST), University of Lisbon, Portugal. He graduated in electrical engineering from the IST, University of Lisbon, in 1965. He received the Ph.D. degree in electrical engineering from Queen Mary College, University of London, U.K., in 1972, and the Aggregate degree from IST in 1978, where he became a Full Professor in 1979. He acted as Chairperson of the National Foundation for Scientific Computing from 1986 to 1989, and from 1989 to 1992, he was the President of the Portuguese National Science and Technology Research Council (JNICT). From 1993 until 2024 he was the Chairperson of the Board of Directors of the Instituto de Telecomunicações, Portugal. He was twice distinguished by the students as the best professor in electrical engineering at IST. He is the author or a coauthor of several conference and journal articles, and of six textbooks. Carlos Salema is a Counselor Member of the Portuguese Institute of Engineers ("Ordem dos Engenheiros"), a fellow (and former President) of the Lisbon Academy of Sciences, and a fellow (and former President) of the Portuguese Academy of Engineering. He has served as a member of the Selection Committee of the Institute of Engineering and Technology (IET) A. F. Harvey Research Prize from 2015 to 2020. In 1974, he was awarded the Marconi Prize by the Institute of Electrical and Electronic Engineers (IEE).

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/