Partial Ordered Statistics Decoding with Enhanced Error Patterns

Marwan Jalaleddine, Huayi Zhou, Jiajie Li, Warren J. Gross
Department of Electrical and Computer Engineering
McGill University
Montreal, Quebec, Canada
Emails: {marwan.jalaleddine, huayi.zhou, jiajie.li}@mail.mcgill.ca, warren.gross@mcgill.ca

***Abstract*—Guessing Random Additive Noise Decoding (GRAND) excels at decoding high-rate codes but struggles to decode low-rate codes with reasonable complexity. Ordered Statistics Decoding (OSD) specifically excels in decoding short codes irrespective of rates; however, OSD necessitates the use of Gaussian elimination which introduces additional time, space and computational complexity. Partial Ordered Statistics Decoding (POSD) was proposed to reduce the time, space, and computational complexity of OSD; however, the current partition-based POSD has poor decoding performance since it does not generate test error patterns across partitions. In this paper, we propose to improve the decoding performance of POSD by incorporating test error patterns inspired by GRAND methods. This work offers a trade-off between performance and complexity compared to existing decoders such as GRAND and OSD. We enhance POSD by optimizing the scheduling of Test Error Patterns (TEPs) and show that our technique can be applied to any code in a standard form. At a target BER 10^{-4} with eBCH (128,64) the enhanced error patterns achieve more than 0.6 dB gain in performance compared to the POSD with partition-based error patterns. Moreover, at a target frame error rate of 10^{-5}, POSD uses $10\times$ less binary operations compared to GRAND when decoding eBCH (128,64) and RLC(128,64) codes. With BCH (127,29) and RLC(128,32), at a target frame error rate of 10^{-2}, POSD with enhanced error patterns with a maximum number of queries (MQ) of 10^4 achieves up to a 2 dB gain to its GRAND equivalent which is using 10^7 maximum number of queries.**

***Index Terms*—Ordered Statistics Decoding, Guessing Random Additive Noise Decoding, mMTC, Partial Ordered Statistics Decoding**

I. Introduction

Massive machine-type communication (mMTC) and ultra-reliable and low-latency communication (URLLC) are two use-cases of the 5G communication system [1]. With new applications such as wireless factory automation and wide-area disaster monitoring, critical mMTC [2], a new use case that is a hybrid of mMTC and URLLC, is set to emerge in the near future, posing new design challenges for wireless systems beyond 5G. In addition to necessitating strict limits on the power used by the decoder in the mMTC use case, critical mMTC requires high reliability which is usually associated with using short low-rate codes and a high-performing low-power code decoder for these codes.

Guessing Random Additive Noise Decoding (GRAND) [3] has been recently proposed as a versatile decoder that can decode any short and high-rate code with some variants approaching the decoding performance of Maximum Likelihood (ML) decoding [4]. One of the most notable soft decision variants of GRAND is ORBGRAND [5] which generates its test error patterns in increasing logistic weight order. Several improvements of the sequence of generation of the test error patterns were suggested in [6] where an improved logistic weight order penalizes queries based on their Hamming weight is discussed. Additionally [7] proposes the use of test error patterns based on an empirical analysis of the error patterns that appear on an AWGN channel. Despite GRAND's potential, the brute-force process of generating Test Error Patterns (TEPs) on the codeword bits and the use of a syndrome check to verify codeword membership results in unacceptably high complexity for low-rate codes.

An alternative to GRAND is Ordered Statistics Decoding (OSD) [8], [9] which is a code-agnostic near-ML decoder that leverages the soft channel information to improve on the decoding performance. OSD uses Gaussian elimination with any code of any code rate which results in a time, space, and computational complexity penalty for short-packet low-power applications.

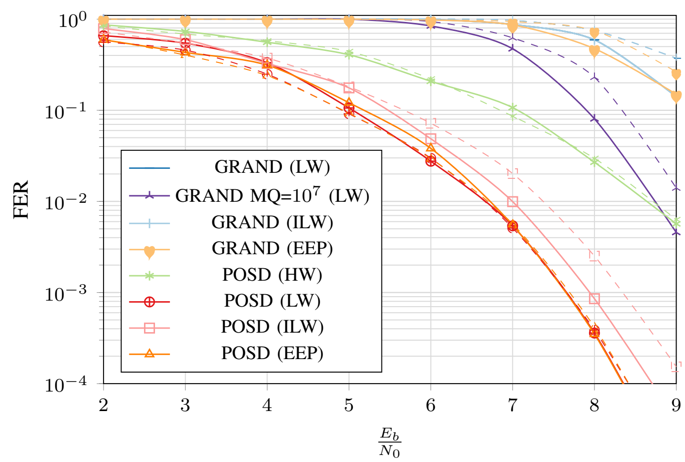

Fig. 1. FER performance of POSD and GRAND with various error patterns on RLC (128,32) (represented by solid lines) and BCH (127,29) (represented by dashed lines). If unspecified, the maximum number of queries used with these methods is 10^4.

Gaussian elimination in OSD can be avoided by using Partial OSD (POSD) which necessitates a generator matrix in standard form [10], [11]. The current implementation of POSD has several limitations. First, the length of the segments and the Hamming weight of the TEPs of the partition based POSD must be determined empirically through extensive computer simulations. Second, the partition based POSD does not use TEP candidates generated across partitions, resulting in poor decoding performance.

In this work, we propose to enhance the decoding performance of POSD by using a new scheduling of TEPs inspired by GRAND methods [5]–[7]. We demonstrate that in order to attain the same decoding performance as POSD with enhanced error patterns, GRAND uses $1698\times \rightarrow 7\times$ more binary operations per frame with eBCH (128,64) and RLC(128,64). Moreover, with BCH (127,29) and RLC(128,32), POSD with enhanced error patterns with a maximum number of queries (MQ) of 10^4 can outperform its GRAND equivalent which uses $1000\times$ the maximum number of queries, as depicted in Fig. 1. Our improved POSD with modified error patterns can operate with codes of varying rates and codebooks and for the same performance, can achieve lower computational complexity compared to GRAND on low-rate codes.

The rest of this paper is organized as follows: The preliminaries for this work are explained in Section II. The proposed POSD is presented in Section III, while the numerical simulation results are presented in Section IV. Finally, in Section V, concluding remarks are made.

II. Preliminaries

A. Notations and Definitions

Matrices are denoted by a bold upper-case letter ($\boldsymbol{M}$), while vectors are denoted with bold lower-case letters ($\boldsymbol{v}$). The number of k-combinations from a given set of n elements is noted by $\binom{n}{k}$. All the indices start at 1. The i'th element of vector $\mathbf{v}$ is represented as $\mathbf{v}_i$. The subvector composed of elements i to j from vector $\mathbf{v}$ is represented as $\mathbf{v}_{i:j}$. For this work, all operations are restricted to the Galois field with 2 elements, noted $\mathbb{F}_2$. Furthermore, we restrict ourselves to (n, k) linear block codes, where n is the code length and k is the code dimension. Throughout this analysis and simulations, we assume Binary Phase Shift Keying (BPSK) modulation over an Additive White Gaussian Noise (AWGN) channel.

B. Definitions

Definition 1: A linear block code is a linear mapping $g : \mathbb{F}_2^k \rightarrow \mathbb{F}_2^n$. To characterise any linear block code, there exists a $k \times n$ matrix $\boldsymbol{G}$ called generator matrix. The generator matrix is in standard form if it is represented as $\boldsymbol{G} = [I^k|P]$ where $\boldsymbol{I}^k$ is the identity matrix of size $k \times k$ and $\boldsymbol{P}$ is an $k \times n - k$ matrix.

Definition 2: Hard Demodulation thresholds the received channel signals :

$$\hat{\boldsymbol{y}}_i = 0 \quad ; \quad \boldsymbol{y}_i > 0. \tag{1}$$

$$\hat{\boldsymbol{y}}_i = 1 \quad ; \quad \boldsymbol{y}_i < 0. \tag{2}$$

Algorithm 1: Partial Ordered Statistics Decoding

Input : $\boldsymbol{y}$, $\boldsymbol{r}$, $\boldsymbol{G}$, n
Output: $\hat{\boldsymbol{u}}$

1 $W_{min} \leftarrow \infty$;
2 $\boldsymbol{ind} \leftarrow Sorted\_Indices(\boldsymbol{y}_{1:k})$;
3 $\hat{\boldsymbol{y}} \leftarrow Hard\_Demodulate(\boldsymbol{y})$;
4 $\boldsymbol{TEPs} \leftarrow Generate\_Test\_Error\_Patterns(\boldsymbol{ind})$;
5 $\boldsymbol{c^0} \leftarrow \hat{\boldsymbol{y}}_{1:k} \times \boldsymbol{G}$;
6 **for** $\boldsymbol{e}$ *in* $\boldsymbol{TEPs}$ **do**
7   **if** $WHD(\boldsymbol{e}, \boldsymbol{r}_{1:k}) < W_{min}$ **then**
8         $\boldsymbol{z} \leftarrow \boldsymbol{e} \times \boldsymbol{G}$;
9         $\boldsymbol{v} \leftarrow \boldsymbol{z} \oplus \boldsymbol{c^0}$;
10   **if** $WHD(\boldsymbol{v}, \boldsymbol{r}_{1:n}) < W_{min}$ **then**
11             $\boldsymbol{u}^* \leftarrow \boldsymbol{e} \oplus \hat{\boldsymbol{y}}_{1:k}$;
12             $W_{min} \leftarrow WHD(\boldsymbol{v}, \boldsymbol{r}_{1:n})$;
13   **end**
14   **end**
15 **end**
16 **return** $\boldsymbol{u}^*$;

Definition 3: The reliability of a bit ($\boldsymbol{r}_i$) of the received channel signal $\boldsymbol{y}$ is the absolute value of the log-likelihood ratio associated with that bit where $\boldsymbol{c}_i$ is the encoded codeword bit at transmitter side:

$$\boldsymbol{r}_i = \left| ln\left(\frac{Pr(\boldsymbol{c}_i = 0|\boldsymbol{y}_i)}{Pr(\boldsymbol{c}_i = 1|\boldsymbol{y}_i)}\right)\right|. \tag{3}$$

Given this definition of reliability, it can be observed that it is always a non-negative quantity.

Definition 4: The WHD of a binary vector $\boldsymbol{v}$ of length n is the sum of the reliability of the nonzero elements. The probability of correct decoding is a decreasing function of WHD [12]:

$$WHD(\boldsymbol{v}, \boldsymbol{r}) = \sum_{i=1}^{n} \boldsymbol{r}_i * \boldsymbol{v}_i = \boldsymbol{r} \cdot \boldsymbol{v}. \tag{4}$$

The WHD of a vector can also be represented as the sum of the WHD of its subvectors:

$$WHD(\boldsymbol{v}, \boldsymbol{r}) = WHD(\boldsymbol{v}_{1:k}, \boldsymbol{r}_{1:k}) + WHD(\boldsymbol{v}_{k+1:n}, \boldsymbol{r}_{k+1:n}). \tag{5}$$

III. Partial Ordered Statistics Decoding

In this section, we will introduce the POSD algorithm and the TEPs that can be used with it. The complexity of POSD is evaluated and compared to the complexity of GRAND.

A. Algorithm Description

The POSD algorithm, presented in Algorithm 1, is a list-based decoding method that is code agnostic and does not require the use of Gaussian elimination. The algorithm begins by sorting the first k bits of the received channel signal in ascending order of bit reliability (Line 2). Then, the ordered set of indices which are sorted by increasing reliability, denoted by $\boldsymbol{ind}$, can be utilized to generate the TEPs (Line 4). The various methods of generating the TEPs will be discussed in Section III-B in greater detail.

The algorithm proceeds to a re-encoding phase in which the test error patterns are re-encoded using the generator matrix in standard form to generate $\boldsymbol{z}$ (Line 8). This is followed by XOR-ing the result of re-encoding the error signal to the result of re-encoding the first k bits from the channel signal to obtain the current codeword estimate $\boldsymbol{v}$ (Line 9). If the resulting codeword estimate has a WHD that is less than the lowest WHD of the previously queried codewords (W_{min}) (Line 10), the algorithm updates the optimal message ($\boldsymbol{u}^*$) and the current W_{min} (Line 11-12). Once all test error patterns have been queried, the algorithm returns the optimal message as the decoded output (Line 16).

B. Error Patterns

This section will discuss the different test error patterns that will be used in our analysis. It is important to note that, to the best of our knowledge, only the Hamming weight order and a partition-based order has been used with POSD. The contribution of this work is the analysis of the performance of the logistic weight order, improved logistic weight order and empirical order TEPs with POSD:

1) Hamming Weight Order (HW): The HW order entails generating test error patterns by increasing Hamming weight defined in (6).

$$hw(\boldsymbol{e}) = \sum_{i=1}^{n} \boldsymbol{e}_i. \tag{6}$$

These error patterns can be generated without any knowledge of the soft channel information.

2) Logistic Weight Order (LW): The LW order [5] generates test error patterns by increasing logistic weight using the ordered set of indices. The logistic weight of any pattern can be calculated using (7).

$$lw(\boldsymbol{e}) = \sum_{i=1}^{n} \boldsymbol{ind}_i \times \boldsymbol{e}_i. \tag{7}$$

For example, the error pattern which results in flipping the least reliable bit has a logistic weight of 1 while the error pattern which results in flipping the three least reliable bits has a logistic weight of $1+2+3=6$. Highly efficient parallelized hardware implementations using integer partitioning can be used to produce these TEPs as in [13], [14].

3) Improved Logistic Weight Order (ILW): The improved logistic weight order [6] generates TEPs by increasing improved logistic weight. The improved logistic weight of any pattern can be calculated using (8) where $\boldsymbol{im}$ represents the ordered vector of indices of the nonzero entries of the error pattern ($\boldsymbol{im} = \boldsymbol{ind}[\boldsymbol{e}[i] == 1]$):

$$ilw(\boldsymbol{e}) = \sum_{i=1}^{hw(\boldsymbol{e})} \boldsymbol{im}_i \times i. \tag{8}$$

4) Empirical Error Patterns (EEP): Similar to what was discussed in [7], TEPs can be generated based on their likelihood of occurring. To that end, after sorting the received channel signal in increasing reliability, we ran 10^9 Monte-Carlo simulations at $\frac{E_b}{N_0} = 7$ dB and collected the most frequent TEPs that occur on the sorted channel signal. This set of TEPs is stored in memory and used with either GRAND, as in [7], or with POSD for the entire range of $\frac{E_b}{N_0}$. For a high MQ, the storage requirements of the EEP TEPs become a large concern.

TABLE I
COMPUTATIONAL COMPLEXITY OF POSD AND ORBGRAND

Decoder	Operation	BOPs	FLOPs
POSD	Sorting Re-Encoding WHD Calculation	 $(k+(2k-1)(n-k))Q$ $5*q*a*Q$	$k\log(k)$
ORBGRAND	Sorting Syndrome Calculation	 $(2n-1)(n-k)Q$	$n\log(n)$

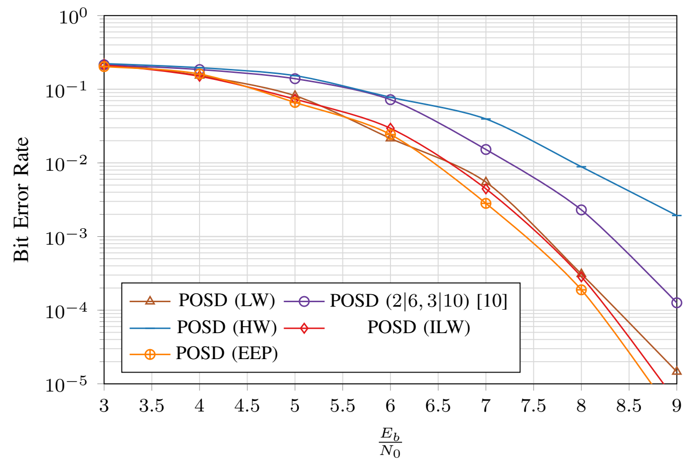

Fig. 2. BER performance of Various Error Patterns on eBCH (128,64). Maximum Number of Queries =135

5) Partion-based Error Patterns ($I_1|K_1, I_2|K_2$): The original version of POSD with partition-based error patterns [10] sorts the received channel signal by increasing reliability and it takes into account two least reliable partitions. This method involves optimizing the size of the first partition (K_1), the size of the second partition (K_2), the maximum Hamming weight of the TEP in the first partition (I_1), and the maximum Hamming weight of the TEP in the second partition (I_2). This method flips up to I_1 bits in the first partition ($\boldsymbol{ind}_{1:K_1}$) and then goes on to flip I_2 bits in the second partition ($\boldsymbol{ind}_{K_1+1:K_1+K_2}$). Hence, unlike the aforementioned TEP schedulings, it does not generate error patterns across the partitions.

C. Complexity Analysis

Table I shows the computational complexity of the major steps in GRAND and POSD. The computational complexity can be discussed in terms of Binary Operations (BOPs), Floating Point Operations (FLOPs) and number of queries per frame (Q).

FLOPs: Contrary to the soft-input variants of POSD, which necessitate sorting only the first k received bits, the soft-input variants of GRAND mandate sorting the reliabilities of the n received bits in increasing reliabilities which results in

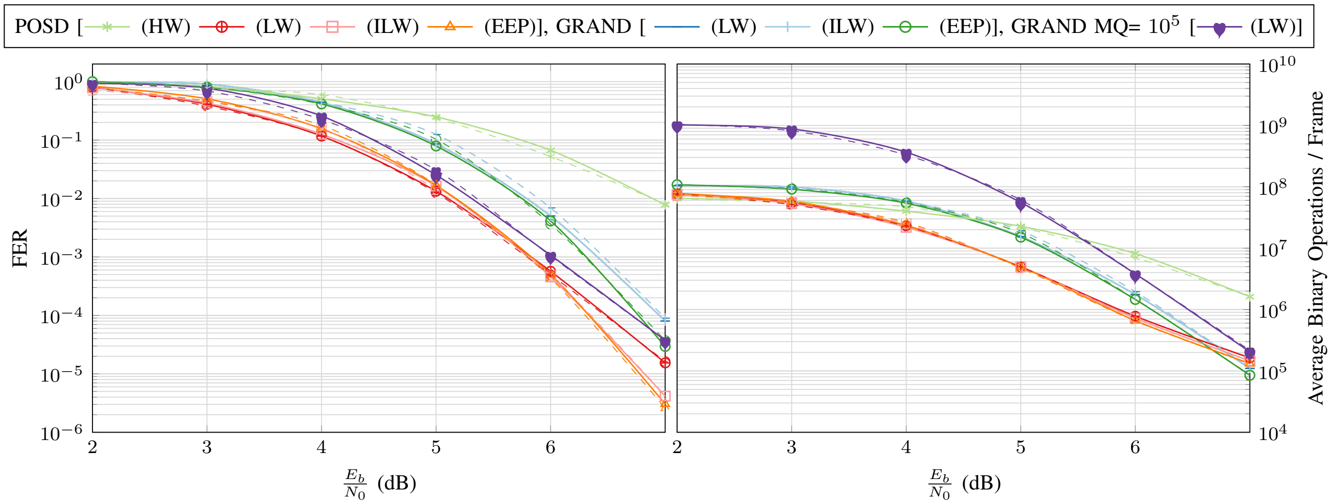

Fig. 3. FER performance of Various Decoders on RLC and BCH (127,85). Maximum Number of Queries= 10^4 with POSD and ORBGRAND and 4-bit quantization is used. Dashed lines correspond to BCH(127,85) and solid lines correspond to RLC(127,85).

a computational complexity of $nlog(n)$ FLOPs/Frame with GRAND compared to $klog(k)$ FLOPs/Frame with POSD.

BOPs: Guessing Random Additive Noise Decoding (GRAND) [3], [5] methods only use a syndrome calculation per query and compare the resulting syndrome to the 0 vector resulting in $(2n-1)(n-k)*Q$ BOPs/Frame. Alternatively, POSD determines the optimality of the candidate codeword through WHD calculations. To find the binary operations used to calculate the WHD, we first quantize the scaled reliabilities ($\frac{\sigma^2}{2}*\boldsymbol{r}$) where σ is the standard deviation of the noise on an AWGN channel. After quantization, we calculate the complexity of the WHD calculation as $5*q*a$ where 5 corresponds to the number of binary operations used by a full adder, q is word size, and a is the number of additions. For example, assuming a 4-bit quantization and $n-1$ additions per candidate codeword, the complexity of WHD calculation would equate to $20*(n-1)*Q$ BOPs/Frame. Additionally, since the other major POSD step is the re-encoding stage, we mandate the use of a Generator matrix in standard form. As such, we only generate the parity bits in Line 8 of Algorithm 1 which results in a re-encoding complexity of $(k+(2k-1)\times(n-k))Q$. For example, with 4-bit quantization and with eBCH (128,64), the complexity of one query with POSD is 10752 BOPs for POSD and 16320 BOPs for GRAND.

To further reduce the average computational complexity of POSD, we first calculate the WHD between the error pattern and the first k bits of the received channel signal before the re-encoding phase [12]. If the resulting quantity is greater than the current minimum WHD, the re-encoding step is skipped as it is impossible that the resulting re-encoded codeword has a lower WHD. Additionally we use an early stopping criterion to terminate decoding when $WHD=0$ since WHD is non-negative.

IV. Performance Evaluation

In this section, we evaluate the decoding performance and complexity of GRAND and POSD with enhanced error patterns on BCH and RLC codes. Unless otherwise specified, the maximum number of queries is set at 10^4 with both GRAND and POSD methods.

Fig. 2 shows the Bit Error Rate (BER) performance of POSD with different error patterns on extended BCH code (128,64) with a maximum number of queries of 135. We can clearly see that the POSD with EEP, ILW and LW error patterns achieve around 1.2 dB gain at a BER of 10^{-4} compared to the POSD with error patterns in HW order. Compared to the partition based method [10], a 0.6 dB gain at a target BER of 10^{-4} can be seen with these improved EEP, ILW and LW error patterns compared to using POSD $(2|6,3|10)$. This gain in performance with EEP, ILW and LW TEPs can be attributed to the generation of error patterns across partitions.

Fig. 3 shows the Frame Error Rate (FER) and Average BOPs per frame for POSD and GRAND with different error patterns on RLC (127,85) and BCH (127,85). We can see that the fastest reduction in FER with respect to $\frac{E_b}{N_0}$ is seen using POSD (EEP) and POSD (ILW). At a target FER 10^{-4} and with the same test error pattern scheduling, we can observe that POSD achieves a 0.5 dB gain in performance compared to GRAND. With $10\times$ the MQ used with POSD, GRAND still sees a degradation of 0.2 dB at target FER 10^{-4}.

Fig. 4 shows the FER and Average BOPs per frame for POSD and GRAND with different error patterns with RLC (128,64) and eBCH (128,64). At a target FER of 10^{-3}, we can see more than 1.2 dB gain using POSD (EEP, ILW, LW) compared to GRAND (EEP, ILW, LW) with eBCH (128,64). At high $\frac{E_b}{N_0}s$ we can also see that the EEP error patterns outperform the LW error patterns when used with POSD due to them being a more accurate approximation of the ML pattern.

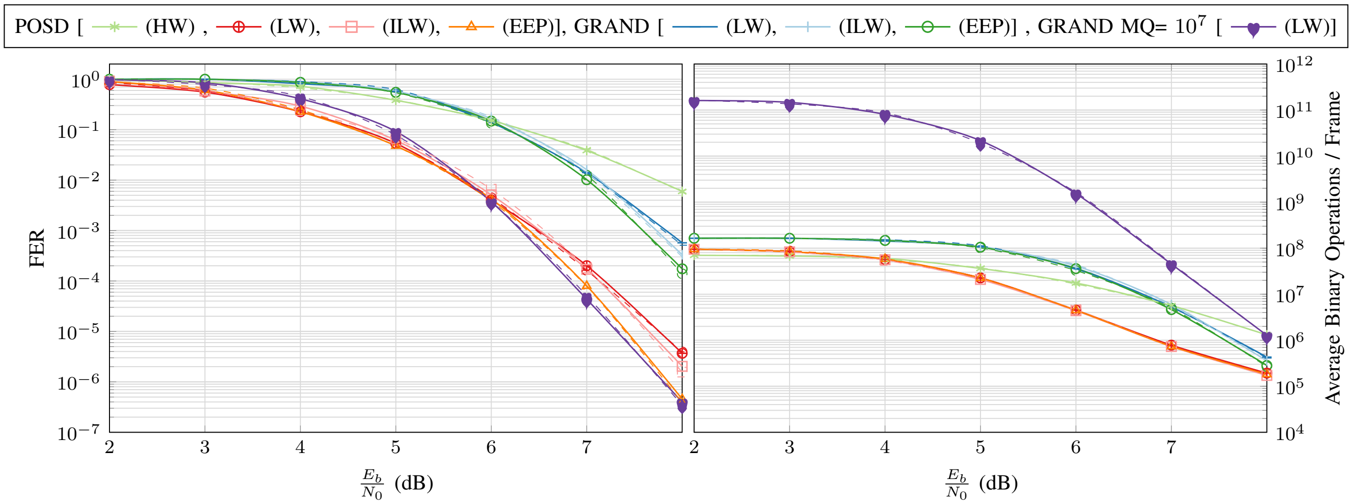

Fig. 4. FER performance of Various Decoders on RLC (128,64) and eBCH (128,64). Unless otherwise specified, the maximum number of queries used with the decoders is 10^4 and 4-bit quantization is used. Dashed lines correspond to eBCH(128,64) and solid lines correspond to RLC(128,64).

After using $1000\times$ the MQ used with POSD, GRAND (LW) can match the decoding performance of POSD (EEP), but with $1698\times \rightarrow 7\times$ the average computational complexity.

Fig. 1 shows the performance of GRAND and POSD with BCH (127,29) and RLC (128,32). POSD with LW and EEP error patterns attain comparable performance but superior to that of ILW error patterns since the ILW penalizes flipping more bits. We can also clearly observe that decoding fails with all GRAND methods at varying $\frac{E_b}{N_0}$ whereby POSD still achieves adequate decoding performance at those code rates. At this low code rate, GRAND is unable to find a codeword close enough to the received signal in the search space using this maximum number of queries. Even by using $1000\times$ the maximum number of queries used by POSD (LW), GRAND (LW) fails to achieve a FER less than 0.1 at $\frac{E_b}{N_0} = 7$. We believe that with POSD, having a lower rate code reduces the search space for TEPs while with GRAND, the search space remains the same irrespective of rate.

V. Conclusion

In conclusion, we demonstrate the effectiveness of utilizing enhanced error patterns with POSD. To attain the same performance of POSD, GRAND requires $1698\times \rightarrow 7\times$ more BOPs/Frame with codes of rate 0.5. GRAND also fails to achieve an equivalent decoding performance with codes of rate 0.25 even by using $1000\times$ the maximum number of queries of POSD. As such, the rate adaptability of POSD makes it a more versatile code-agnostic decoder for short codes compared to GRAND. However, the method's reliance on sorting poses a challenge in developing an efficient and parallel hardware architecture. As such, future studies should explore the possibility of using partial sorting to overcome this challenge.

References

[1] 3GPP, "NR Multiplexing and channel coding," Technical Specification (TS) 3GPP.38.212, 3rd Generation Partnership Project (3GPP), Nov. 2021.

[2] S. R. Pokhrel, J. Ding, J. Park, O.-S. Park, and J. Choi, "Towards Enabling Critical mMTC: A Review of URLLC Within mMTC," *IEEE Access*, vol. 8, pp. 131796–131813, 2020.

[3] K. R. Duffy, J. Li, and M. Médard, "Capacity-Achieving Guessing Random Additive Noise Decoding," *IEEE Transactions on Information Theory*, vol. 65, pp. 4023–4040, July 2019.

[4] A. Valembois and M. Fossorier, "An improved method to compute lists of binary vectors that optimize a given weight function with application to soft-decision decoding," *IEEE Communications Letters*, vol. 5, pp. 456–458, Nov. 2001.

[5] K. R. Duffy, W. An, and M. Médard, "Ordered Reliability Bits Guessing Random Additive Noise Decoding," *IEEE Transactions on Signal Processing*, vol. 70, pp. 4528–4542, 2022.

[6] C. Condo, V. Bioglio, and I. Land, "High performance low-complexity error pattern generation for ORBGRAND decoding," in *2021 IEEE Globecom Workshops (GC Wkshps)*, pp. 1–6, IEEE, 2021.

[7] C. Condo, "A fixed latency ORBGRAND decoder architecture with LUT-aided error-pattern scheduling," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 69, no. 5, pp. 2203–2211, 2022.

[8] M. P. Fossorier and S. Lin, "Soft decision decoding of linear block codes based on ordered statistics," *IEEE International Symposium on Information Theory - Proceedings*, vol. 41, no. 5, p. 395, 1994.

[9] M. Fossorier and S. Lin, "Soft-decision decoding of linear block codes based on ordered statistics," *IEEE Transactions on Information Theory*, vol. 41, pp. 1379–1396, Sept. 1995.

[10] S. E. A. Alnawayseh and P. Loskot, "Ordered statistics-based list decoding techniques for linear binary block codes," *EURASIP Journal on Wireless Communications and Networking*, vol. 2012, p. 314, Oct. 2012.

[11] S. E. A. Alnawayseh and P. Loskot, "Low-complexity soft-decision decoding techniques for linear binary block codes," in *2009 International Conference on Wireless Communications Signal Processing*, pp. 1–5, 2009.

[12] A. Valembois and M. Fossorier, "Box and match techniques applied to soft-decision decoding," *IEEE Transactions on Information Theory*, vol. 50, pp. 796–810, May 2004.

[13] S. M. Abbas, T. Tonnellier, F. Ercan, M. Jalaleddine, and W. J. Gross, "High-Throughput VLSI Architecture for Soft-Decision Decoding with ORBGRAND," in *ICASSP*, pp. 8288–8292, 2021.

[14] S. M. Abbas, T. Tonnellier, F. Ercan, M. Jalaleddine, and W. J. Gross, "High-Throughput and Energy-Efficient VLSI Architecture for Ordered Reliability Bits GRAND," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 30, pp. 681–693, June 2022.