GRAND-EDGE: A Universal, Jamming-resilient Algorithm with Error-and-Erasure Decoding

Furkan Ercan[†], Kevin Galligan[*], David Starobinski[†], Muriel Médard[§], Ken R. Duffy[*], Rabia Tugce Yazicigil[†]
[†]Department of Electrical and Computer Engineering, Boston University, Boston, MA, USA
[§]Department of Electrical Engineering and Computer Science, MIT, Cambridge, MA, USA
[*]Hamilton Institute, Maynooth University, Ireland

Abstract—**Random jammers that overpower transmitted signals are a practical concern for many wireless communication protocols. As such, wireless receivers must be able to cope with standard channel noise and jamming (intentional or unintentional). To address this challenge, we propose a novel method to augment the resilience of the recent family of universal error-correcting GRAND algorithms. This method, called Erasure Decoding by Gaussian Elimination (EDGE), impacts the syndrome check block and is applicable to any variant of GRAND. We show that the proposed EDGE method naturally reverts to the original syndrome check function in the absence of erasures caused by jamming. We demonstrate this by implementing and evaluating GRAND-EDGE and ORBGRAND-EDGE. Simulation results, using a Random Linear Code (RLC) with a code rate of $105/128$, show that the EDGE variants lower both the Block Error Rate (BLER) and the computational complexity by up to five order of magnitude compared to the original GRAND and ORBGRAND algorithms. We further compare ORBGRAND-EDGE to Ordered Statistics Decoding (OSD), and demonstrate an improvement of up to three orders of magnitude in the BLER.**

I. Introduction

What do a Wi-Fi router, a Bluetooth speaker, and a leaky microwave oven have in common? They all broadcast wireless signals in the same frequency range. Modern wireless technologies ensure reliable communication by employing techniques, such as subcarrier frequency hopping [1] or cyclic prefixing [2]. Yet, powerful interference, such as a leaky microwave oven operating at a nearby frequency, may disrupt communication between devices, e.g., by rendering a substantial portion of the subcarrier frequencies unusable. Under such adversarial conditions, although the received signal strength indicator (RSSI) may alert the receiver about channel anomalies [3], the retrieved data that is overpowered by interference is practically lost.

The purpose of this work is to add resilience against such jamming events. We consider a channel model in which a powerful jammer impacts the transmitted data randomly, at a bit-level. In this case, the individual non-jammed bits are subject to the additive white Gaussian noise (AWGN) of the channel, and the jammed bits are impacted by a more extreme, additive noise. We aim to develop an error correction algorithm that provides data recovery capabilities under these adversarial conditions. We propose doing so in conjunction with Guessing Random Additive Noise Decoding (GRAND), a recently proposed error correction decoder algorithm that can work with any codebook [4]. Besides the original hard-information GRAND algorithm, soft-information-based variants are also available [5]–[7]. Among them, the Ordered Reliability Bits GRAND (ORBGRAND) [8] lends itself to practical hardware implementations while maintaining a near maximum-likelihood (ML) decoding performance [9].

In this work, we propose a jamming-resilient algorithm based on GRAND algorithm and its variants. Our method seeks to perform error-correction on the non-jammed bits of the received frame, and perform erasure-correction on the jammed bits. It does so by:

1) Identifying jammed bits through RSSI observation;
2) Performing error correction on non-jammed bits. As jammed bits occur randomly in any part of the received codeword, a challenge is to error-correct a partial code that changes on each communication. This challenge requires a universal decoding approach, for which we use hard- and soft-information variants of the GRAND algorithm;
3) Having error-corrected the non-jammed bits, determining the values of the jammed bits through Gaussian elimination.

We achieve this capability by empowering the syndrome check function of any GRAND-based algorithm with the ability of restoring the erased bits in a received frame. This upgraded syndrome check method is called *Erasure Decoding by Gaussian Elimination*, or EDGE in short.

In general, most existing error-and-erasure decoding (EED) are based on specific decoding schemes. Some EED schemes are designed to retrieve corrupted *frames* rather than *bits* in the presence of erasures. This includes schemes, such as random linear network coding (RLNC) [10], [11], product/staircase codes [12] and fountain codes [13]. On the other hand, our proposed EDGE method operates at the bit-level, and can be used with *any* linear code.

We introduce two variants of the EDGE subroutine: one with hard-information (GRAND-EDGE) and the other with soft-information (ORBGRAND-EDGE) decoding. Simulation results demonstrate that the EDGE subroutine improves both the block-error rate (BLER) performance and the computational complexity by up to five orders of magnitude, under adversarial channel conditions. We also compare ORBGRAND-EDGE with the Ordered Statistics Decoding (OSD) algorithm [14] which is also based on Gaussian elimination. We show that the proposed ORBGRAND-EDGE algorithm improves the

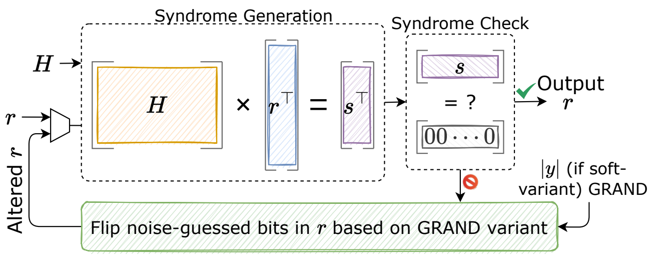

Fig. 1. A component-level description of the GRAND algorithm family, with in-detail syndrome generation process.

BLER by up to three orders of magnitude while achieving lower complexity compared to OSD.

The rest of the paper is organized as follows. The background on GRAND is detailed Section II. In Section II, the EDGE subroutine and its applications with universal hard- and soft-decoding variants, GRAND-EDGE and ORBGRAND-EDGE, are presented. The benefits of the proposed universal error-and-erasure decoding via simulation results are presented in Section IV, followed by concluding remarks in Section V.

II. The GRAND Algorithm

Guessing Random Additive Noise Decoding (GRAND) [4] is a recently introduced universal algorithm capable of decoding any code, using codebook membership checks. For a received (hard-information) frame vector $\mathbf{r}$, the membership (syndrome) check is performed as

$$\mathbf{H}\cdot\mathbf{r}^\top, \tag{1}$$

where $\mathbf{H}$ is the codebook-specific parity-check matrix of a linear code. Unlike traditional decoding algorithms, GRAND focuses on the noise component of the received frame. If (1) is not equal to an all-zero vector ($\mathbf{0}$), then the received sequence $\mathbf{r}$ is not a member of the codebook due to noise-corrupted bits. The GRAND algorithm trials putative error sequences, represented by $\mathbf{e}$, in maximum likelihood order. It subtracts each of them from $\mathbf{r}$ until it finds one that satisfies

$$\mathbf{H}\cdot(\mathbf{r}\oplus\mathbf{e})^\top=\mathbf{0}. \tag{2}$$

where $\oplus$ is the modulo-2 sum operator.

Different ordering of guessing noise sequences lead to different variations of GRAND. A high-level description of the GRAND algorithm family is depicted in Fig. 1. Among the variants, Ordered Reliability Bits GRAND (ORBGRAND) [7] is a soft-information decoding algorithm that orders the putative noise sequences based on the *logistic weight* (LW) of their sorted LLR magnitudes, and is amenable to hardware implementation [9].

III. The GRAND-EDGE Algorithm

In this Section, the channel model is characterized first. Then the EDGE subroutine is explained in detail, followed by a closer look at the Gaussian elimination process. Finally, the proposed GRAND-EDGE algorithm is described.

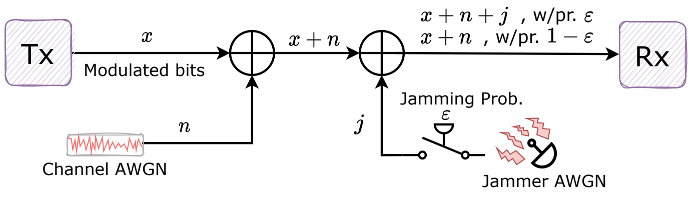

Fig. 2. The channel model.

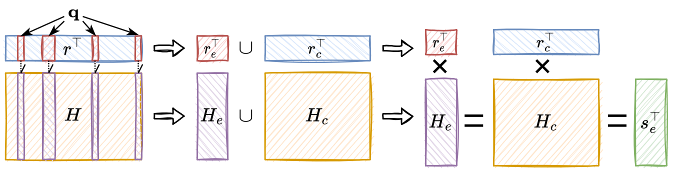

Fig. 3. Isolation of erasures ($\mathbf{r_e}$) from the received (hard-decision) codeword and corresponding parity-check columns ($\mathbf{H_e}$) from the H-matrix, followed by the calculation of erasure syndrome ($\mathbf{s_e}$).

A. Channel Model

In this work, we consider an AWGN channel model that is randomly disrupted by a jammer, as depicted in Fig. 2. The additive channel noise instance, represented by n, is added to the modulated signal x carried over a specific frequency. A powerful jammer instance j, activated with a probability ϵ may be added to the transmitted signal. We assume that the probability of jamming ϵ is on the bit-level rather than the frame-level. Therefore, the received signal y can be expressed as

$$y=\begin{cases}x+n+j & \text{with probability } \epsilon;\\ x+n & \text{otherwise.}\end{cases} \tag{3}$$

The jammer is also modeled as AWGN, but with a variance that is far greater than that of the channel AWGN. Therefore, if the received signal magnitude is suspiciously stronger than expected, then it is assumed that the signal is jammed and its value is invalidated.

For frame-level jamming or erasures, such as lost frames due to undecodable preambles, can also be converted to bit-level erasures by simple interleaving techniques, such as in [15].

B. The EDGE Subroutine

Let $\mathbf{p}=\{0,1,\cdots N-1\}$ represent the total set of 0-indexed indices of the received vector of length N, and let $\mathbf{q}$ represent the set of imputed (erased) indices in the received vector, such that $\mathbf{q}$ has e elements, $\mathbf{q}\subseteq\mathbf{p}$, and $q_i<q_{i+1},\forall i$. As shown in Fig. 3, given $\mathbf{q}$, let us split the received vector $\mathbf{r}$ and the parity-check matrix $\mathbf{H}$

$$\begin{aligned}\mathbf{r}&=\mathbf{r_e}\cup\mathbf{r_c},\\ \mathbf{H}&=\mathbf{H_e}\cup\mathbf{H_c},\end{aligned} \tag{4}$$

where $\mathbf{r_e}$ and $\mathbf{r_c}$ represent the erased and non-erased subsets of $\mathbf{r}$ with sizes e and $N-e$, respectively. Similarly, $\mathbf{H_e}$ is a $(N-k)\times e$ matrix that contains the columns of $\mathbf{H}$ which correspond to the erased bits, and $\mathbf{H_c}$ contains the remaining

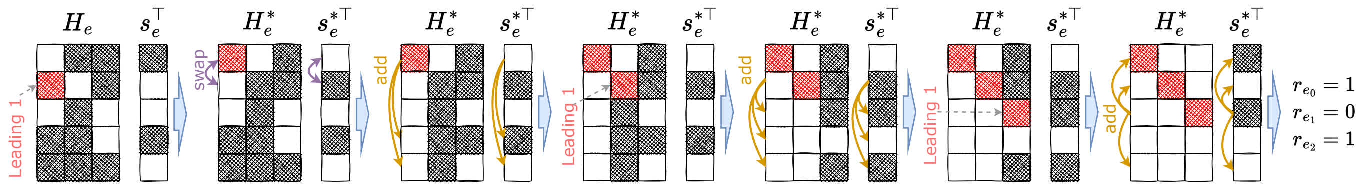

Fig. 4. Gaussian elimination example using two elementary row operations to transform $\mathbf{H_e}$ into reduced row echelon form (RREF), to find $\mathbf{r_e}$ from $\mathbf{s_e^*}$. 1s and 0s in the matrix and the vector are indicated by black and white, respectively. Leading 1s at each column of $\mathbf{H_e}$ are represented by red.

Algorithm 1: EDGE Subroutine Initialization

```
Inputs  : r, H, q, $N-k$, $e$
Outputs: $\mathbf{r_c}$, $\mathbf{r_e}$, $\mathbf{H_c}$, $\mathbf{E}$
1 $\mathbf{r_e} \leftarrow \mathbf{r}[\mathbf{q}]$,   $\mathbf{r_c} \leftarrow \mathbf{r} \setminus \mathbf{r_c}$
2 $\mathbf{H_e} \leftarrow \mathbf{H}[\mathbf{q}]$,   $\mathbf{H_c} \leftarrow \mathbf{H} \setminus \mathbf{H_c}$
3 if $N-k < e$ then
4     no unique solution, terminate decoding
5 end
6 $\mathbf{E} \leftarrow$ GaussianElimination($\mathbf{H_e}$)
```

Algorithm 2: The EDGE Subroutine

```
Inputs  : $\mathbf{r_c}$, $\mathbf{r_e}$, $\mathbf{H_c}$, $\mathbf{E}$, $\mathbf{q}$, $N-k$, $e$
Outputs: r, success
1 $\mathbf{s_e} = \mathbf{E} \cdot \mathbf{H_c} \cdot \mathbf{r_c}$
2 if $\mathbf{s_e}[e : N-k-1] \neq \mathbf{0}$ then
3     success = 0, terminate subroutine
4 end
5 $\mathbf{r_e} = \mathbf{s_e}[0 : e-1]$
6 $\mathbf{r} \leftarrow \mathbf{r_c} \cup \mathbf{r_e}$ using $\mathbf{q}$
7 success = 1
```

columns. Note that the original sets in (4) can be reconstructed from the separated subsets, using $\mathbf{q}$. The parity-check equation described in (1) can be expanded as

$$\mathbf{H_e} \cdot \mathbf{r_e}^\top \oplus \mathbf{H_c} \cdot \mathbf{r_c}^\top = \mathbf{0}. \tag{5}$$

Here, all the components are known on the receiver side except $\mathbf{r_e}$. Let us define the *erasure syndrome*, $\mathbf{s_e}$, as

$$\mathbf{s_e}^\top = \mathbf{H_c} \cdot \mathbf{r_c}^\top, \tag{6}$$

and transfer it to the right-hand side of (5) in the binary domain, to obtain

$$\mathbf{H_e} \cdot \mathbf{r_e}^\top = \mathbf{s_e}^\top. \tag{7}$$

as visualized in Fig. 3.

The EDGE subroutine performs Gaussian elimination on the *linear set of equations* in (7) to find $\mathbf{r_e}$. To find a unique solution for $\mathbf{r_e}$, the number of equations must not be smaller than the number of variables. In other words, the number of rows of $\mathbf{H_e}$ must be equal to or greater than its columns, therefore, we must first satisfy $e \leq N-k$.

The initialization procedure for the EDGE subroutine is described in Algorithm 1, where $\mathbf{r_c}$, $\mathbf{r_e}$, $\mathbf{H_c}$ are prepared (lines 1-2) and whether the number of erasures could be recovered by the code is determined (lines 3-5). This is followed by the Gaussian elimination process using $\mathbf{H_e}$, to store the required operations in an *elimination matrix*, $\mathbf{E}$ (line 6). These stored Gaussian elimination operations are later used to reduce the erasure restoration complexity, explained in Section III-C.

The EDGE subroutine is described in Algorithm 2. First, the erasure syndrome is calculated $\mathbf{s_e}$ using $\mathbf{H_c}$ and $\mathbf{r_c}$, and the Gaussian elimination matrix $\mathbf{E}$. If the resulting $\mathbf{s_e}$ is *error-free* (lines 2-4), then the erased sequence $\mathbf{r_e}$ is substituted from $\mathbf{s_e}$ (line 5) and the complete received sequence is restored (line 6). Otherwise, the subroutine is terminated with no success (line 3).

C. The Gaussian Elimination Process

The Gaussian elimination process reduces the linear set of equations into a form from which the variables can be directly obtained. For our purpose, we review the binary matrix case of Gaussian elimination [16]. To achieve this form, the $\mathbf{H_e}$ matrix should be modified into a reduced row-echelon form (RREF), with the particular structure of $[\mathbf{I}|\mathbf{0}]^\top$ where $\mathbf{I}$ and $\mathbf{0}$ represent identity and all-zero matrices, respectively. Note that there is a unique RREF for any matrix. If the RREF of a matrix yields an all-zero column, there is no unique solution for $\mathbf{r_e}$.

An example of the Gaussian elimination process is depicted in Fig. 4. Starting from the leftmost column, a leading 1 is first identified for each column. If the row index of the leading 1 does not match its column index, an elementary swap operation takes place to place the leading 1 cell to the diagonal of the matrix. This row with the leading 1 is then subtracted from the other rows that also hold a 1 on the subject column, to ensure there is a single 1 remaining. The process is sequentially continued for all the columns in $\mathbf{H_e}$. The equivalent swap and add operations are also applied to the transposed $\mathbf{s_e}^\top$, to obtain $\mathbf{s_e}^{*\top}$ at the end. Changes to the $\mathbf{H_e}$ and $\mathbf{s_e}^\top$ are denoted with an asterisk (*).

In practice, Gaussian elimination is costly with a complexity of $O(n^3)$. To reduce its impact, it can be performed only once at the initialization (refer to Algorithm 1) and all the operations towards obtaining the RREF $\mathbf{H_e^*}$ can be stored in an *elimination matrix*, $\mathbf{E}$. This way, the final $\mathbf{s_e}^{*\top}$ can be obtained by

$$\mathbf{s_e}^{*\top} = \mathbf{E} \cdot \mathbf{s_e}, \tag{8}$$

which has the same effective complexity as the syndrome check operation of GRAND, see (1).

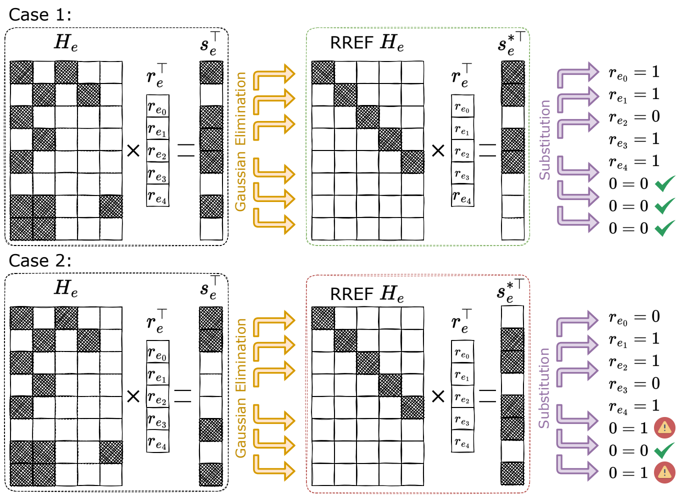

Fig. 5. Two examples are shown for erased bits restoration using the EDGE algorithm, with 5 erased bits and $n - k = 8$. In both cases same indices are erased from the same codebook, yielding the same H_e. The result of Case 1 is consistent and is therefore valid. In Case 2, the result is inconsistent due to incurred errors and therefore is invalid.

D. *The GRAND-EDGE Algorithm Family*

Refer to Fig. 5, where two arbitrary Gaussian elimination examples are illustrated using the same $\mathbf{H_e}$, with $e = 5$ and $N - k = 8$. In the first case, $\mathbf{s_e}$ does not incur any channel errors. Therefore, the transformed $\mathbf{s_e^*}$ is in the form of $[\mathbf{r_e}|\mathbf{0}]$, in other words, the last $N-k-e$ equations with 0 coefficients naturally yield 0. In the second case, the $\mathbf{s_e}$ is infused with channel errors that yield an $\mathbf{s_e^*}$ with nonzero entries in its last $N - k - e$ indices. For the EDGE subroutine, this indicates that there are errors in the channel. When errors are involved, erased indices cannot be accurately restored, and an additional error-correction decoder is required.

The process of the proposed GRAND-EDGE algorithm is described in Algorithm 3. Note that the EDGE subroutine (line 6) replaces the syndrome check function of GRAND. If the EDGE subroutine fails, then the GRAND algorithm generates the next putative error pattern and combines it with the received part of the codeword (lines 6-7, followed by line 4). As discussed in Section II, the agenda of the error pattern generation depends on the GRAND variant. For simplicity, inputs and parameters for soft-information GRAND variants are not shown in the algorithm, instead, we refer to [5], [8] for details.

When there are no erasures in the received codeword, then $\mathbf{r_e} = \mathbf{H_e} = \varnothing$, with $\mathbf{r_c} = \mathbf{r}$ and $\mathbf{H_c} = \mathbf{H}$. Hence, the expression in (6) becomes the same as in (1), and GRAND-EDGE reverts to GRAND algorithm. Therefore, the proposed GRAND-EDGE does not present a computational burden to the original algorithm in the absence of erasures in the channel.

IV. Performance Assessment

Simulations are carried over the channel model described in Fig. 2, where the channel and jammer AWGN are generated through independent Gaussian processes. The overpowered jammer SNR is set to -100 dB. The probability of jamming, ϵ, is generated through an independent Bernoulli process for each transmitted symbol. Prior to decoding, a threshold is required to decide whether the received signal is jammed. For that purpose, we follow the empirical rule; any signal that is observed within 3 standard deviations of the modulated signal space is considered non-jammed, and is jammed otherwise. Note that different approaches for determining jamming in the channel can also be used with the proposed algorithm, but are beyond the scope of this paper. For all evaluations, a random linear code (RLC) with code length $N = 128$ with $k = 105$ information bits is featured, which is generated using the simulator on-the-fly. The abandonment threshold [4] is set to a Hamming weight of 3 for both GRAND and GRAND-EDGE. The logistic weight threshold [7] is set to 104 for ORBGRAND and ORBGRAND-EDGE.

A. *GRAND-EDGE Performance*

Fig. 6(a) presents the error correction performance for the proposed GRAND-EDGE algorithm against the GRAND algorithm. The x-axis represents the channel SNR, and ϵ is fixed for each simulation and is represented in the legend. We observed that GRAND-EDGE outperforms GRAND by up to five orders of magnitude, because it has the capability of restoring erased bits whereas GRAND attempts to guess their values iteratively.

Fig. 6(e) shows the average number of iterations (codebook queries) for each BLER curve. With improving AWGN SNR conditions, the GRAND algorithm gets stuck in guessing the jammed bits in an attempt to find the correct codeword. This yields a high average number of iterations, even at high SNRs. On the other hand, the efficient handling of the erased bits by the EDGE routine helps the GRAND-EDGE algorithm reduces the avarage number of iterations compared to GRAND, by up to more than five orders of magnitude.

Fig. 6(b) and Fig. 6(f) present the GRAND-EDGE performance against GRAND over a simulated set of bit jamming probability values ϵ. The channel SNR values are fixed and are represented in the legend. Similar to Fig. 6(a), the GRAND-EDGE algorithm achieves superior BLER performance as

Algorithm 3: The GRAND-EDGE Algorithm

```
   Inputs  : $\mathbf{r}, \mathbf{H}, \mathbf{G}^{-1}, \mathbf{q}$
   Outputs: $\hat{\mathbf{u}}$
1  $\mathbf{e} \leftarrow \mathbf{0}$
2  $iter = 0$
3  $\{\mathbf{r_c}, \mathbf{r_e}, \mathbf{H_c}, \mathbf{E}\} \leftarrow$ EDGE_Init$(\mathbf{r}, \mathbf{H}, \mathbf{q}, N-k, e)$
4  while $success = 0 \land iter \neq maxIters$ do
5      $\mathbf{r_c^*} \leftarrow \mathbf{r_c} \oplus \mathbf{e}$
6      $\{success, \mathbf{r}^*\} \leftarrow$EDGE$(\mathbf{r_c^*}, \mathbf{r_e}, \mathbf{H_c}, \mathbf{E}, \mathbf{q}, N-k, e)$
7      $\mathbf{e} \leftarrow$NextErrorPattern$(iter)$
8      $iter = iter + 1$
9  end
10 $\hat{\mathbf{u}} \leftarrow \mathbf{r}^* \cdot \mathbf{G}^{-1}$
```

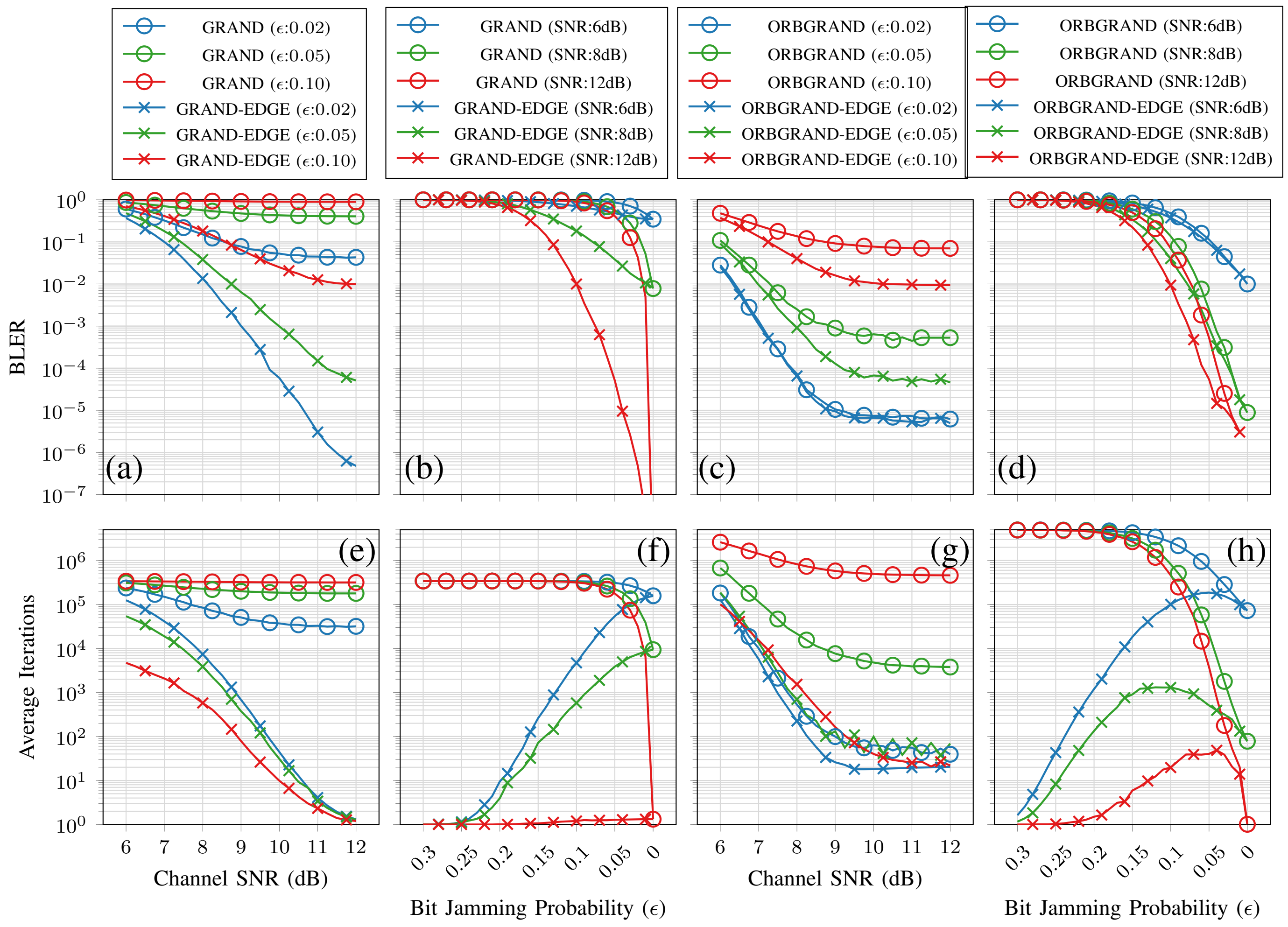

Fig. 6. Top: BLER performance comparison of the proposed GRAND-EDGE and ORBGRAND-EDGE algorithms against their GRAND-only counterparts, using RLC[128,105]. The comparisons are carried out with various channel AWGN SNR and bit-jamming probabilities as indicated in the x-axis labels and the legends. Bottom: Average number of iterations for each evaluated algorithm with matching legends.

the channel conditions improve. On the other hand, the average number of iterations of GRAND-EDGE increases as ϵ decreases. This is because GRAND-EDGE abandons decoding if the number of erasures is beyond its capability (if $e > N - k$). Nonetheless, the average number of iterations of GRAND-EDGE is always smaller or equal to that of GRAND, sometimes by up to five orders of magnitude less. Finally, we note that both the BLER performance and the average number of iterations of GRAND-EDGE meet the performance of GRAND at $\epsilon = 0$. This is because the GRAND-EDGE algorithm reverts to the GRAND algorithm when there is no erasure in the channel, as mentioned previously in Section III-D.

B. *ORBGRAND-EDGE Performance*

Fig. 6(c) and Fig. 6(d) present the BLER performance evaluation for the proposed ORBGRAND-EDGE algorithm against the ORBGRAND algorithm, *i.e.* when soft-information is involved in decoding the received codeword. Compared to GRAND, the ORBGRAND algorithm can find more errors by taking advantage of the bit-reliability information (LLRs). As a result, ORBGRAND demonstrates better BLER performance compared to GRAND. On the other hand, the improvement in BLER with ORBGRAND-EDGE is not as dramatic compared to the gains in hard-information GRAND-EDGE. This is because ORBGRAND can prioritize jammed indices for bit-flipping as long as their values are beyond the predetermined threshold. Nonetheless, we observe a BLER gain of up to one order of magnitude when ORBGRAND is enhanced with the EDGE subroutine.

Compared to the GRAND-EDGE performance in Fig. 6(a) with $\epsilon = 0.02$, the ORBGRAND-EDGE performance in Fig. 6(c) experiences an error floor. As a result, at high SNR regimes, GRAND-EDGE outperforms ORBGRAND-EDGE. This is due to the differences in the bit-flipping agenda of these two variants. With a logistic weight of 104, only the least reliable 104 indices and a subset of their combinations are considered for bit-flipping in ORBGRAND [8]. Therefore,

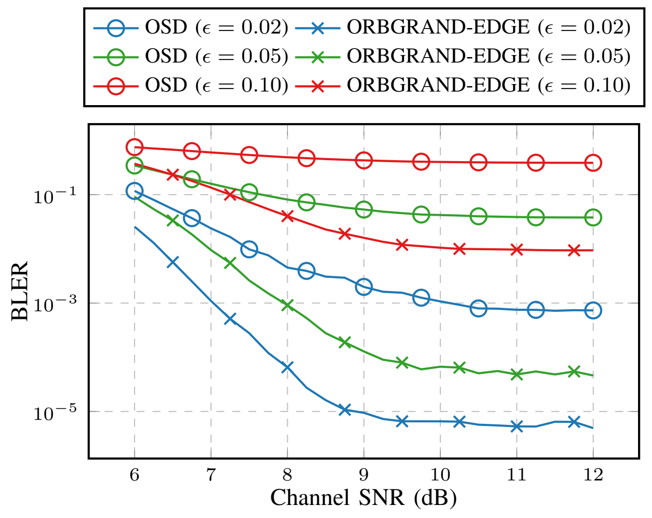

Fig. 7. ORBGRAND-EDGE performance compared to the OSD algorithm, using RLC[128,105]. The bit-level jamming probabilities are set to $\epsilon = \{0.02, 0.05, 0.1\}$ and x-axis represents the channel AWGN SNR.

when a jammed index is overlooked by the jamming detection, ORBGRAND-EDGE does not erase the bit, and it is likely considered an index reliable enough that it is never considered for bit-flipping. On the other hand, even when a jammed index is overseen, the GRAND-EDGE algorithm evaluates all non-erased indices for bit-flipping. As a result, especially when the ϵ is small, GRAND-EDGE can outperform ORBGRAND-EDGE. Although more sophisticated pre-decoding jamming detection patterns can be implemented [17], they are beyond the scope of this work.

Fig. 6(g) and Fig. 6(h) present the computational complexity comparison for ORBGRAND-EDGE and ORBGRAND, with respect to channel SNR and bit-jamming probability, respectively. Compared to GRAND, ORBGRAND has better (less) computational complexity in general, due to the efficient identification of bit-flipping indices. On the other hand, when ORBGRAND is augmented with the EDGE subroutine, the average number of iterations reduces by up to five orders of magnitude. Similar to the observations made in GRAND, ORBGRAND-EDGE reverts to the ORBGRAND algorithm when $\epsilon = 0$. Different than GRAND-EDGE, the computational complexity of ORBGRAND-EDGE begins to reduce as ϵ reduces further. This is because the complexity of the linear set of equations reduces with reducing ϵ, and with the help of soft information ORBGRAND-EDGE can find the erroneous component of the received codeword faster compared to GRAND-EDGE.

Fig. 7 compares the BLER performance of the proposed ORBGRAND-EDGE algorithm against the Ordered Statistics Decoding (OSD) algorithm [14], using RLC[128,105]. Similar to EDGE, OSD performs Gaussian elimination to find the most likely transmitted codeword. However, the Gaussian elimination in OSD is always performed over k columns, whereas in EDGE it is only performed over up to $N - k$ columns. Moreover, OSD requires multiple permutations which adds to its implementation complexity. Nonetheless, it can be observed from Fig. 7 that the ORBGRAND-EDGE outperforms OSD by up to three orders of magnitude in BLER.

V. Conclusion

In this work, we introduced an adversarial model, whereby a jammer randomly overpowers bits of a transmitted signal, effectively causing erasures. To address this adversarial channel condition, we generalized the syndrome check component of the universal GRAND algorithm family to support erasure decoding. The proposed GRAND-EDGE algorithm and its variants address bit-level errors and erasures simultaneously. The erasure decoding component (i.e., the EDGE subroutine) presents no additional computational complexity when there is no detected erasure in the channel, reverting to the syndrome check function of the GRAND algorithm in that case. The proposed EDGE algorithm can be used with both hard and soft variants of GRAND, which we demonstrated through the implementation of GRAND-EDGE and ORBGRAND-EDGE. Compared to their original counterparts, the EDGE-enhanced GRAND algorithms achieve up to five orders of magnitude improvement both in terms of error-correction performance in terms of computational complexity under the considered adversarial model.

Acknowledgements

This work was partially supported by Defense Advanced Research Projects Agency Contract number HR00112120008 and by National Science Foundation ECCS Award numbers 2128517 and 2128555. The content of the information does not necessarily reflect the position or the policy of the US Government, and no official endorsement should be inferred. This publication has emanated from research supported in part by a grant from Science Foundation Ireland under grant number 18/CRT/6049. The opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Science Foundation Ireland.

References

[1] D. Torrieri, *Principles of Spread-Spectrum Communication Systems*. Cham: Springer International Publishing, 2018, pp. 155–212.

[2] D. Tse and P. Viswanath, *Fundamentals of Wireless Communication*. Cambridge: Cambridge University Press, 2005.

[3] R.-H. Wu, Y.-H. Lee, H.-W. Tseng, Y.-G. Jan, and M.-H. Chuang, "Study of characteristics of RSSI signal," in *2008 IEEE International Conference on Industrial Technology*, 2008, pp. 1–3.

[4] K. R. Duffy, J. Li, and M. Médard, "Capacity-achieving guessing random additive noise decoding," *IEEE Transactions on Information Theory*, vol. 65, no. 7, pp. 4023–4040, 2019.

[5] A. Solomon, K. R. Duffy, and M. Médard, "Soft maximum likelihood decoding using GRAND," in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, 2020, pp. 1–6.

[6] K. R. Duffy, M. Médard, and W. An, "Guessing random additive noise decoding with symbol reliability information (SRGRAND)," *IEEE Transactions on Communications*, vol. 70, no. 1, pp. 3–18, 2022.

[7] K. R. Duffy, "Ordered reliability bits guessing random additive noise decoding," in *IEEE ICASSP*, 2021, pp. 8268–8272.

[8] 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.

[9] 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 Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 30, no. 6, pp. 681–693, 2022.

[10] R. Koetter and M. Médard, "An algebraic approach to network coding," *IEEE/ACM Transactions on Networking*, vol. 11, no. 5, pp. 782–795, 2003.

[11] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft, "XORs in the air: Practical wireless network coding," *IEEE/ACM Transactions on Networking*, vol. 16, no. 3, pp. 497–510, 2008.

[12] L. Rapp and L. Schmalen, "Error-and-erasure decoding of product and staircase codes," *IEEE Transactions on Communications*, vol. 70, no. 1, pp. 32–44, 2022.

[13] V. Manthamkarn, U. Tuntoolavest, N. Sriratanapochai, and S. Chuenjit, "Methods to convert error & error-erasure channel to erasure channel for fountain code decoding," in *2022 International Electrical Engineering Congress (iEECON)*, 2022, pp. 1–4.

[14] M. Fossorier and S. Lin, "Soft-decision decoding of linear block codes based on ordered statistics," *IEEE Transactions on Information Theory*, vol. 41, no. 5, pp. 1379–1396, 1995.

[15] A. Riaz, A. Solomon, F. Ercan, M. Médard, R. T. Yazicigil, and K. R. Duffy, "Interleaved noise recycling using GRAND," in *ICC 2022 - IEEE International Conference on Communications*, 2022, pp. 2483–2488.

[16] G. Strang, *Introduction to Linear Algebra, 5th Edition*. Wellesley, MA: Wellesley - Cambridge Press, 2016.

[17] F. Ercan, K. Galligan, K. R. Duffy, M. Médard, D. Starobinski, and R. T. Yazicigil, "A general security approach for soft-information decoding against smart bursty jammers," in *2022 IEEE Globecom Workshops (GC Wkshps)*, 2022, pp. 1–6.