

Code-Weight Sphere Decoding

Yubeen Jo, Geon Choi, *Student Member, IEEE*, Yongjune Kim, *Member, IEEE*,
and Namyoony Lee, *Senior Member, IEEE*

Abstract—Ultra-reliable low-latency communications (URLLC) demand high-performance error-correcting codes and decoders in the finite blocklength regime. This letter introduces a novel two-stage near-maximum likelihood (near-ML) decoding framework applicable to any linear block code. Our approach first employs a low-complexity initial decoder. If this initial stage fails a cyclic redundancy check, it triggers a second stage: the proposed code-weight sphere decoding (WSD). WSD iteratively refines the codeword estimate by exploring a localized sphere of candidates constructed from pre-computed low-weight codewords. This strategy adaptively minimizes computational overhead at high signal-to-noise ratios while achieving near-ML performance, especially for low-rate codes. Extensive simulations demonstrate that our two-stage decoder provides an excellent trade-off between decoding reliability and complexity, establishing it as a promising solution for next-generation URLLC systems.

Index Terms—URLLC, finite blocklength, near-ML decoding, two-stage decoding, code-weight sphere decoding.

I. INTRODUCTION

ULTRA-RELIABLE low-latency communications (URLLC) necessitate efficient error-correcting codes at finite blocklengths and low code rates, as well as developing decoding algorithms that offer both low complexity and high performance [1]–[7]. While significant research has focused on various suboptimal decoding methods such as syndrome decoding [8], ordered statistics decoding (OSD) [9], guessing random additive noise decoding (GRAND) [10]–[12], and list decoding techniques such as successive cancellation list (SCL) decoding [13] including linearity-enhanced serial list decoding approaches that use sphere-based concepts [14], achieving near-ML decoding performance with optimized complexity at low code rates remains an open challenge.

In this letter, we propose a novel two-stage near-ML decoding method specifically tailored for short blocklength and low-rate codes. The decoding process begins with an initial stage where a preliminary codeword is obtained using an arbitrary low-complexity suboptimal decoding algorithm. This initial decoder's primary objective is to efficiently identify a valid codeword while minimizing computational complexity and decoding latency. If the decoded codeword passes the cyclic redundancy check (CRC) validation, the process terminates. Otherwise, the algorithm proceeds to the second stage: our introduced code-weight sphere decoding (WSD).

The core concept of WSD lies in leveraging the fundamental properties of linear block codes, specifically their low-weight

codewords. By constructing a Hamming sphere centered on the initial codeword, WSD exploits the characteristic that all codewords within this sphere can be represented as the initial codeword plus a low-weight codeword. Instead of an exhaustive search over the entire codebook, WSD constructs a Hamming sphere around the initial estimate and iteratively explores this confined neighborhood to find a more reliable codeword. This greedy, step-by-step refinement allows the decoder to navigate the solution space efficiently, progressively improving the estimate. A distinguishing feature of the proposed WSD decoder is its universal applicability to any linear code and any initial decoding method. Simulation results demonstrate that it achieves near-ML decoding performance across various low-rate codes in the short blocklength regime.

II. PRELIMINARIES

In this section, we define the notation and fundamental concepts used throughout this work. The cardinality of a set \mathcal{A} is denoted by $|\mathcal{A}|$. The Euclidean norm of a vector \mathbf{v} is denoted by $\|\mathbf{v}\|$. \mathbb{F}_2 represents the Galois field of two elements, and $\mathbf{1}_N$ denotes an all-one vector of length N . Let $d_H(\mathbf{c}_1, \mathbf{c}_2)$ denote the Hamming distance between two binary vectors $\mathbf{c}_1, \mathbf{c}_2 \in \mathbb{F}_2^N$, and $w_H(\mathbf{c})$ denote the Hamming weight of a binary vector $\mathbf{c} \in \mathbb{F}_2^N$.

A. Channel Coding System

We employ a binary linear block code $\mathcal{C}(N, K)$ for channel coding, which has a codeword length N and a message length K . The code is defined by its generator matrix $\mathbf{G} \in \mathbb{F}_2^{K \times N}$, or equivalently, by its parity-check matrix $\mathbf{H} \in \mathbb{F}_2^{(N-K) \times N}$. An information block $\mathbf{m} = [m_0, m_1, \dots, m_{K-1}] \in \mathbb{F}_2^K$ is mapped to a codeword $\mathbf{c} = [c_0, c_1, \dots, c_{N-1}] \in \mathbb{F}_2^N$ via the relation $\mathbf{c} = \mathbf{m}\mathbf{G}$.

For modulation, we use binary phase shift keying (BPSK), which maps a binary codeword \mathbf{c} to a real-valued symbol vector $\mathbf{x} \in \mathbb{R}^N$ according to $\mathbf{x} = \mathbf{1}_N - 2\mathbf{c}$. The modulated vector \mathbf{x} is transmitted over an additive white Gaussian noise (AWGN) channel. The received signal vector is given by $\mathbf{y} = \mathbf{x} + \mathbf{n}$, where \mathbf{n} is an independent and identically distributed (i.i.d.) Gaussian noise vector with zero-mean and variance $\sigma^2 = N_0/2$, i.e., $\mathbf{n} \sim \mathcal{N}(0, \sigma^2 \mathbf{I}_N)$. To enhance error detection, CRC precoding is employed. The CRC-encoded message, denoted as $\mathbf{v} \in \mathbb{F}_2^{K+K_{\text{crc}}}$, is treated as a new information vector for the subsequent channel encoding process. Here, K_{crc} is the length of the appended CRC bits. The generator and parity-check matrices for the CRC code are denoted as $\mathbf{G}_{\text{crc}} \in \mathbb{F}_2^{K_{\text{crc}} \times (K+K_{\text{crc}})}$ and $\mathbf{H}_{\text{crc}} \in \mathbb{F}_2^{(K+K_{\text{crc}}) \times (K+K_{\text{crc}})}$, respectively.

Yubeen Jo is with the School of Electrical Engineering, Korea University, Seoul 02841, Republic of Korea (e-mail: jybin00@korea.ac.kr).

Geon Choi, Yongjune Kim and Namyoony Lee are with the Department of Electrical Engineering, POSTECH, Pohang 37673, Gyeongbuk, Republic of Korea (e-mail: geon.choi@postech.ac.kr; yongjune@postech.ac.kr; nylee@postech.ac.kr).

The CRC-encoded vector \mathbf{v} is generated from the original message vector $\mathbf{m} \in \mathbb{F}_2^K$ as $\mathbf{v} = \mathbf{m}\mathbf{G}_{\text{crc}}$. Consequently, the final codeword is given by $\mathbf{c} = \mathbf{v}\mathbf{G}$, where the channel code's generator matrix is modified to $\mathbf{G} \in \mathbb{F}_2^{(K+K_{\text{crc}}) \times N}$ to accommodate the expanded message length.

B. Hamming Sphere Construction from Code Weights

In a linear block code, we can efficiently construct Hamming spheres by leveraging the properties of code weights. For any codeword $\mathbf{c} \in \mathcal{C}(N, K)$, the set of all codewords at a Hamming distance d_ℓ from \mathbf{c} is given by

$$\mathcal{C}_\ell(\mathbf{c}) = \{\mathbf{c}' \in \mathcal{C}(N, K) : d_H(\mathbf{c}, \mathbf{c}') = d_\ell\} \quad (1)$$

and d_ℓ represents the ℓ -th distinct weight in the code's weight spectrum.

Suppose the support of code's weight spectrum consists of $L+1$ distinct elements arranged in increasing order, such that $d_0 = 0 < d_1 (= d_{\min}) < \dots < d_L$. Here, d_{\min} denotes the minimum distance of the code $\mathcal{C}(N, K)$. The Hamming sphere centered at codeword \mathbf{c} with a radius up to $d_r (\leq d_L)$ is then defined as the union of all codewords up to the distance d_r , given by:

$$\mathcal{S}_r(\mathbf{c}) = \bigcup_{\ell=0}^r \mathcal{C}_\ell(\mathbf{c}). \quad (2)$$

Furthermore, the entire codeword set $\mathcal{C}(N, K)$ can be expressed as a Hamming sphere of radius d_L , that is, $\mathcal{C}(N, K) = \mathcal{S}_L(\mathbf{c})$. A specific example of this construction is illustrated in Fig. 1.

A key property of linear codes is that the set of codewords at distance d_ℓ from any codeword $\hat{\mathbf{c}}$ can be expressed as a coset of the set of codewords with weight d_ℓ . Specifically, since $d_H(\hat{\mathbf{c}}, \mathbf{c}) = d_H(\mathbf{0}, \mathbf{c} + \hat{\mathbf{c}}) = w_H(\mathbf{c} + \hat{\mathbf{c}})$, it holds that

$$\begin{aligned} \mathcal{C}_\ell(\hat{\mathbf{c}}) &= \{\mathbf{c} \in \mathcal{C}(N, K) : d_H(\hat{\mathbf{c}}, \mathbf{c}) = d_\ell\} \\ &= \{\hat{\mathbf{c}} + \mathbf{c} : \mathbf{c} \in \mathcal{C}_\ell(\mathbf{0})\} \\ &= \hat{\mathbf{c}} + \mathcal{C}_\ell(\mathbf{0}) \end{aligned} \quad (3)$$

where $\mathcal{C}_\ell(\mathbf{0})$ denotes the set of all codewords with Hamming weight exactly d_ℓ . This relationship shows that the elements within a Hamming sphere centered at any codeword $\hat{\mathbf{c}}$ can be efficiently generated by simply adding the set of low-weight codewords to $\hat{\mathbf{c}}$. This coset relationship holds due to the group property of linear codes, where the mapping $\hat{\mathbf{c}} \mapsto \hat{\mathbf{c}} + \mathbf{c}$ is a bijection over $\mathcal{C}(N, K)$ that preserves Hamming distances. This property significantly reduces the computational complexity of sphere construction in our decoding algorithm, as we only need to maintain a precomputed database of low-weight codewords.

III. A TWO-STAGE NEAR-ML DECODER

In this section, we present a two-stage decoding method that leverages both a low-complexity decoder and code-weight sphere decoding.

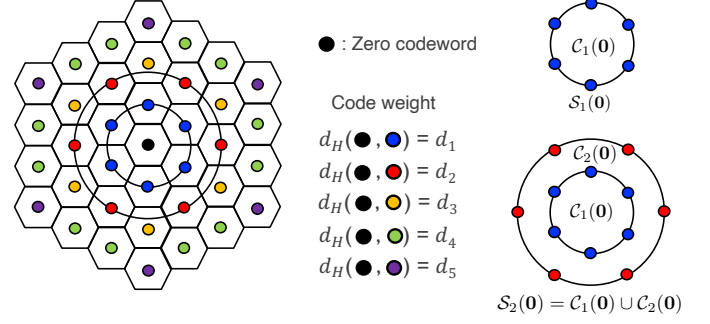


Fig. 1. Illustration of the code lattice and Hamming sphere construction. The set of all codewords forms a lattice. The Hamming sphere $\mathcal{S}_r(\mathbf{c})$ around a codeword \mathbf{c} is formed by the union of shells $\mathcal{C}_\ell(\mathbf{c})$ containing codewords at specific Hamming distance d_ℓ .

A. Initial Low-Complexity Decoding Phase

In the initial phase, a low-complexity decoder processes the received signal \mathbf{y} to generate a preliminary codeword estimate $\hat{\mathbf{c}}^{(-1)}$. From this estimated codeword $\hat{\mathbf{c}}^{(-1)}$, the CRC-encoded message vector $\hat{\mathbf{v}}^{(-1)}$ is extracted. If the CRC validation, (i.e., $\mathbf{H}_{\text{crc}}(\hat{\mathbf{v}}^{(-1)})^\top = \mathbf{0}$) passes, the decoding process terminates. However, if the CRC check fails (i.e., $\mathbf{H}_{\text{crc}}(\hat{\mathbf{v}}^{(-1)})^\top \neq \mathbf{0}$), indicating a mismatch with the transmitted codeword, further steps are initiated. In this event, $\hat{\mathbf{m}}^{(-1)}$ is extracted from $\hat{\mathbf{v}}^{(-1)}$ and is re-encoded to form $\hat{\mathbf{c}}^{(0)} = (\hat{\mathbf{m}}^{(-1)}\mathbf{G}_{\text{crc}})\mathbf{G}$. Subsequently, a reliability measure $M^{(0)} = \|\mathbf{y} - \mathbf{x}(\hat{\mathbf{c}}^{(0)})\| \propto -\log P(\mathbf{y}|\mathbf{x}(\hat{\mathbf{c}}^{(0)}))$ is calculated. This early termination mechanism significantly reduces the average computational complexity that would otherwise be introduced by the subsequent boosting phase.

B. Code-Weight Sphere Decoding Phase

The WSD phase encompasses multiple iterations designed to refine the initial codeword estimate $\hat{\mathbf{c}}^{(0)}$. During this phase, the decoder leverages the code-weight sphere set centered at $\hat{\mathbf{c}}^{(0)}$, denoted as $\mathcal{S}_r(\hat{\mathbf{c}}^{(0)})$. In the first boosting round, the decoder refines $\hat{\mathbf{c}}^{(0)}$ by selecting a candidate codeword from the code-weight sphere set $\mathcal{S}_r(\hat{\mathbf{c}}^{(0)})$ that maximizes likelihood. This selection is accomplished by solving the following optimization problem:

$$\hat{\mathbf{c}}^{(1)} = \arg \min_{\mathbf{c} \in \mathcal{S}_r(\hat{\mathbf{c}}^{(0)})} \|\mathbf{y} - \mathbf{x}(\mathbf{c})\| \quad (4)$$

The reliability measure for this first boosting round is then recorded as:

$$M^{(1)} = \|\mathbf{y} - \mathbf{x}(\hat{\mathbf{c}}^{(1)})\|. \quad (5)$$

If this reliability measure $M^{(1)}$ shows improvement over the initial measure $M^{(0)}$, (specifically when $M^{(1)} < M^{(0)}$), the boosting process advances to the next round. The reliability measure for this round is computed as $M^{(i)} = \|\mathbf{y} - \mathbf{x}(\hat{\mathbf{c}}^{(i)})\|$. If $M^{(i)} > M^{(i-1)}$, indicating a degradation in reliability, the decoding process terminates and returns $\hat{\mathbf{c}}^{(i-1)}$. Figure 2 provides a comprehensive summary of the entire decoding procedure.

Remark 1 (Decoding refinement): By construction, the proposed boosting decoder progressively enhances the reliability

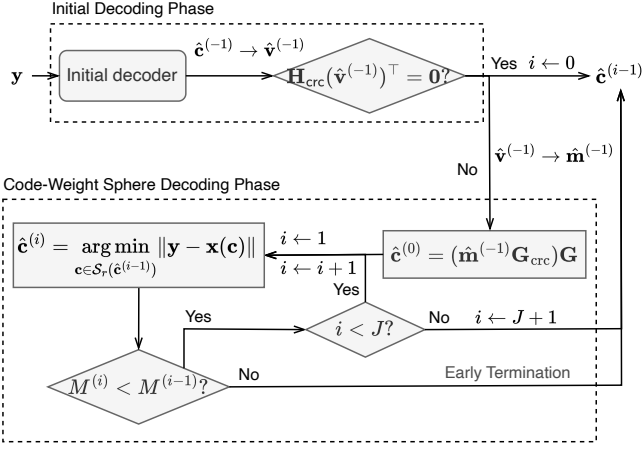


Fig. 2. Flowchart of the proposed two-stage decoding algorithm, detailing the initial decoding phase and the code-weight sphere decoding (WSD) phase.

measure through an iterative refinement process, ensuring that $M^{(0)} > M^{(1)} > M^{(2)} > \dots > M^{(J)}$.

Remark 2 (Difference from sphere decoding): A soft-decision sphere decoder efficiently identifies the codeword closest to the received vector within a predefined Euclidean radius D , reducing the search space while maintaining optimality in decoding. The fundamental challenge in implementing sphere decoding lies in efficiently enumerating all codewords that satisfy the sphere constraint, namely those within a distance D from the received vector.

Recent research has demonstrated the remarkable effectiveness of applying sphere decoding to polar codes, achieving significant improvements in short blocklength regimes [15]–[18]. Nevertheless, these techniques still face key challenges, particularly related to the exhaustive search process and computational complexity inherent to sphere decoding.

The WSD approach diverges from conventional sphere decoding in its fundamental search paradigm. Sphere decoding initiates a single, exhaustive search within a conventional Euclidean sphere centered on the received vector \mathbf{y} . In contrast, WSD operates directly on the discrete code lattice. It begins with an initial codeword estimate and then progressively refines the solution by hopping from the current codeword to a more likely neighbor within a localized sphere. This iterative process transforms the decoding problem from a comprehensive search to a guided, step-by-step improvement on the code lattice.

C. Decoding Complexity

The computational complexity of the proposed WSD decoder varies significantly with signal-to-noise ratio (SNR) conditions. The WSD phase represents an additional computational cost compared to conventional methods and only activates when the initial decoding fails the CRC check.

At high SNR, where the CRC failure probability of the initial decoding, given by

$$P_{e,\text{crc}} = \mathbb{P}[\mathbf{H}_{\text{crc}}(\hat{\mathbf{v}}^{(-1)})^T \neq \mathbf{0}] \quad (6)$$

is low, the WSD phase rarely engages, minimizing computational overhead. Conversely, at low SNR where the CRC failure probability is high, the WSD phase activates more frequently, leading to an increase in average computational complexity. This adaptive behavior is evidenced in Fig. 5 and Fig. 6.

When triggered, each boosting round requires $N \times |\mathcal{S}_r(\mathbf{0})|$ operations to evaluate all candidate codewords and $|\mathcal{S}_r(\mathbf{0})|$ operations to find the most reliable codeword. The overall worst-case complexity is therefore bounded by

$$P_{e,\text{crc}} \times (J \times (N + 1) |\mathcal{S}_r(\mathbf{0})|). \quad (7)$$

This formulation demonstrates how WSD naturally adjusts computational effort based on channel conditions—conserving resources when conditions are favorable and applying additional computation only when necessary to improve decoding performance.

IV. SIMULATION RESULTS

We compare the block error rate (BLER) performances of the proposed decoding algorithms under a binary-input AWGN channel.

A. Simulation Settings and Benchmarks

- **CRC-aided (CA) polar codes with successive cancellation list (SCL) decoding:** As a primary benchmark, we employ CA-polar codes, a powerful and widely adopted variant. The information sets are constructed according to the 5G-NR reliability sequence [19]. We utilize CRC precoding with the generator polynomial $g(x) = 1 + x^5 + x^9 + x^{10} + x^{11}$. Standard successive cancellation list (SCL) decoders with list size $L = 32$ are employed.
- **CRC-aided deep polar (CA-DP) codes with SCL-backpropagation parity check (SCL-BPC) decoding** [6]: CA-deep-polar codes, a novel class of pre-transformed polar codes with three transformation layers, are used. We employ the SCL-BPC decoder [6], adopting the 5G-NR reliability sequence for information bit selection and CRC precoding with the polynomial $g(x) = 1 + x^5 + x^6$. The SCL-BPC decoder has a list size of $L = 32$.
- **Code-weight sphere decoder (WSD):** WSD leverages the code-weight sphere set $\mathcal{S}_r(\mathbf{0})$, exhaustively generated for low-rate codes due to the high complexity for large K . All set elements are pre-stored. The maximum number of boosting iterations, J , is 3, balancing complexity and performance; decoding may terminate earlier if no improvement is observed. WSD(r) denotes WSD leveraging $\mathcal{S}_r(\mathbf{0})$. Simulation parameters are summarized in Table I.
- **Maximum-likelihood decoder (MLD):** The MLD serves as a performance benchmark, providing the theoretical lower bound on BLER. It operates by searching the entire codeword space to find the codeword closest to the received vector in terms of Euclidean distance. Though optimal, its exponential complexity ($N \times 2^K$) makes it impractical for large codes, thus included for reference.

- **Theoretical bounds:** To evaluate the performance gap to theoretical limits, we reference the random coding union (RCU) bound [20] and the meta-converse bound [21]. These provide insights into fundamental communication limits.

TABLE I
CARDINALITY OF THE CODE-WEIGHT SPHERE IN SEC. IV

	(N, K)	fig.	$ S_1(\mathbf{0}) $	$ S_2(\mathbf{0}) $	$ S_3(\mathbf{0}) $	$ S_4(\mathbf{0}) $
CA-polar	(64, 16)	3	9	246	4002	-
CA-polar	(128, 16)	3	1	24	1078	12995
CA-polar	(256, 16)	3	1	10	537	6471
CA-DP	(64, 16)	4	8	277	3941	-
CA-DP	(128, 16)	4	610	14490	-	-
CA-DP	(256, 16)	4	214	5670	13222	-

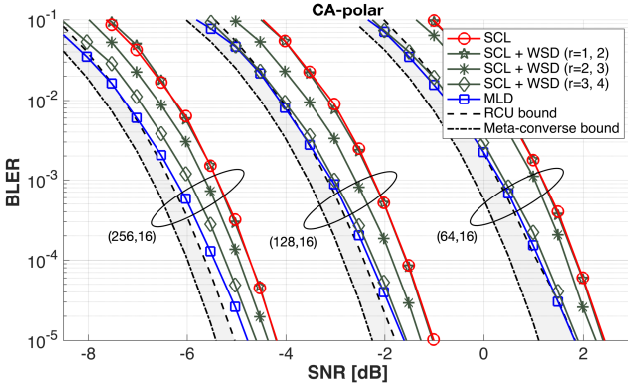


Fig. 3. BLER comparison under SCL, SCL+WSD, and MLD decoding methods.

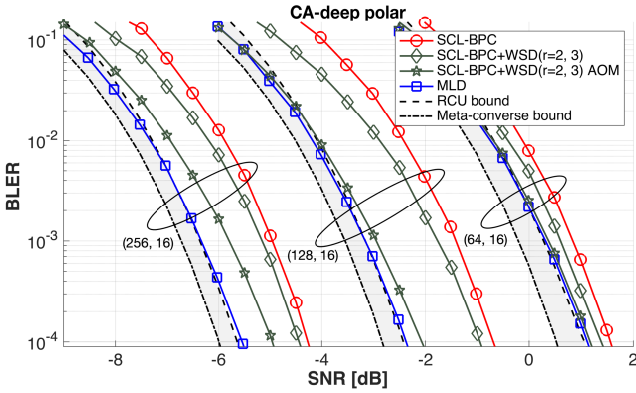


Fig. 4. BLER comparison under SCL-BPC, SCL-BPC+WSD, and MLD decoding methods.

B. Comparison of BLER Performance and Decoding Complexity

We evaluate BLER performance for blocklength $N \in \{64, 128, 256\}$ and code rate $R \in \{\frac{1}{4}, \frac{1}{8}, \frac{1}{16}\}$.

The results for the CA-polar codes, presented in Fig. 3 highlights the capability of WSD to systematically approach the ML bound by increasing the search radius r . The results

are particularly compelling at larger radii: for $N = 64$ and $N = 128$, the performance of SCL+WSD becomes nearly indistinguishable from that of the MLD for $r \geq 3$. Even for the longer blocklength of $N = 256$, the gap to the ML bound is reduced to just 0.2 dB at a BLER of 10^{-5} with $r = 4$.

This near-optimal performance is built upon the consistent gain observed even at smaller radii. For instance, employing a modest search with $r \in \{2, 3\}$ provides a significant gain of at least 0.2 dB over the strong SCL baseline across all configurations. This demonstrates WSD's effectiveness as a powerful enhancement stage, capable of correcting the residual, hard-to-decode error events that persist after an initial high-performance list decoding attempt.

Similarly, Fig. 4 illustrates the BLER performance for CA-deep polar codes, a state-of-the-art construction decoded with its high-performance SCL-BPC decoder. For these tests, we used a search radius of $r = 3$ for $N = \{64, 256\}$ and $r = 2$ for $N = 128$. The standard WSD integration (the green diamond-marked curve) demonstrates a consistent gain over the SCL-BPC baseline.

To further isolate and quantify the maximum potential gain offered by the WSD stage, we also introduce an "Always On Mode" (AOM). In this configuration, the WSD phase is intentionally activated for every block, effectively setting the CRC failure probability to one ($P_{e,crc} = 1$) for the purpose of analysis. This allow us to observe the full error correction capability of WSD as a post-processor. Remarkably, the AOM results reveal an even more substantial performance gain, exceeding 1.2 dB in $N = 128$. this highlights that for highly complex codes like CA-deep polar, WSD serves as a powerful enhancement stage, capable of refining the initial high-quality estimates from SCL-BPC to push performance even closer to the ML bound.

TABLE II
COMPLEXITY ANALYSIS

Decoder	Complexity
SCL(L)	$\mathcal{O}(LN \log N)$
SCL(L) + WSD(r)	$\mathcal{O}(LN \log N) + P_{e,crc} \times \mathcal{O}(J \times (N+1) S_r(\mathbf{0}))$
SCL-BPC(L, P)	$\mathcal{O}(LN \log N) + \mathcal{O}(\sum_{\ell=1}^{P-1} N_\ell \log N_\ell)$
SCL-BPC(L, P) + WSD(r)	$\mathcal{O}(LN \log N) + \mathcal{O}(\sum_{\ell=1}^{P-1} N_\ell \log N_\ell) + P_{e,crc} \times \mathcal{O}(J \times (N+1) S_r(\mathbf{0}))$
MLD	$N \times 2^K$

From the complexity perspective, Table II summarizes the theoretical complexities of SCL, SCL+WSD, SCL-BPC, SCL-BPC+WSD, and MLD, while Fig. 5 and 6 present the corresponding experimental complexities. Specifically, the decoding complexities of MLD and WSD are measured by counting the number of Euclidean distance computations per bit performed during the decoding. For SCL decoding with a list size L , the complexity is well known as $\mathcal{O}(LN \log N)$. In the case of SCL-BPC decoding, complexity calculations follow the approach detailed in [6], where we consider three transformation layers as used in our simulations.

Both experimental complexity analyses clearly demonstrate a decreasing trend for SCL+WSD and SCL-BPC+WSD as the SNR increases from low to high. This behavior directly reflects the adaptive nature of our two-stage approach: as the channel quality improves, the initial decoder succeeds more often, thereby reducing the activation frequency of the WSD stage and conserving computational resources precisely when they are least needed.

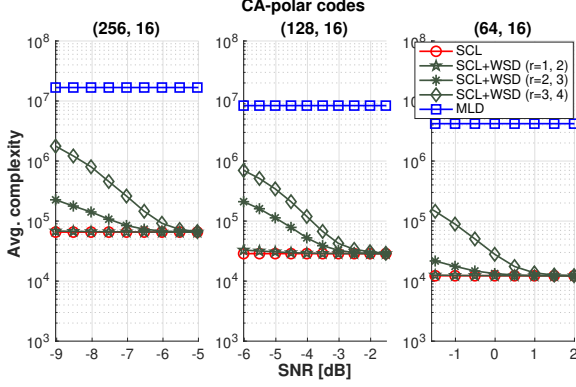


Fig. 5. Average complexity comparison under SCL, SCL+WSD, and MLD decoding methods.

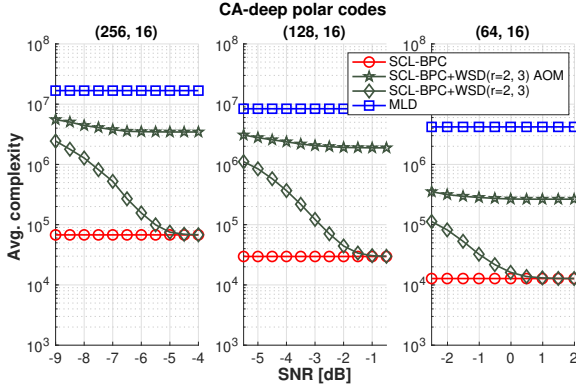


Fig. 6. Average complexity comparison under SCL-BPC, SCL-BPC+WSD, and MLD decoding methods.

V. CONCLUSION

In this letter, we presented a two-stage near-ML decoding method that effectively addresses the stringent reliability and latency challenges in finite blocklength regimes for URLLC applications. Our approach adaptively combines a low-complexity initial decoder with an innovative WSD technique.

By exploiting the inherent lattice structure of linear codes and implementing an iterative refinement strategy, WSD achieves near-ML performance with substantially reduced complexity compared to exhaustive methods. The technique's universal applicability to any linear code enhances its versatility. Simulation results confirm robust performance across various code rates, establishing WSD as a promising architectural foundation for future near-ML decoders in reliability-critical systems.

REFERENCES

- [1] 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, 2019.
- [2] 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, 2023.
- [3] M. Rowshan, M. Qiu, Y. Xie, X. Gu, and J. Yuan, "Channel coding toward 6g: Technical overview and outlook," *IEEE Open Journal of the Communications Society*, vol. 5, pp. 2585–2685, 2024.
- [4] D. Han, J. Park, Y. Lee, H. V. Poor, and N. Lee, "Block orthogonal sparse superposition codes for ultra-reliable low-latency communications," *IEEE Trans. Commun.*, vol. 71, no. 12, pp. 6884–6897, 2023.
- [5] D. Han, B. Lee, M. Jang, D. Lee, S. Myung, and N. Lee, "Block orthogonal sparse superposition codes for L^3 communications: Low error rate, low latency, and low transmission power," *IEEE Journal on Selected Areas in Communications*, vol. 43, no. 4, pp. 1183–1199, 2025.
- [6] G. Choi and N. Lee, "Deep polar codes," *IEEE Trans. Commun.*, vol. 72, no. 7, pp. 3842–3855, 2024.
- [7] —, "Sparsely pre-transformed polar codes for low-latency SCL decoding," *IEEE Trans. Commun.*, pp. 1–1, 2025.
- [8] S. Lin and D. J. Costello, *Error Control Coding: Fundamentals and Applications*. Pearson-Prentice Hall, 2004.
- [9] M. P. 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, 2002.
- [10] 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, 2019.
- [11] K. R. Duffy, W. An, and M. Médard, "Ordered reliability bits guessing random additive noise decoding," *IEEE Trans. Sig. Process.*, vol. 70, pp. 4528–4542, 2022.
- [12] P. Yuan, M. Médard, K. Galligan, and K. R. Duffy, "Soft-output GRAND and iterative decoding to outperform LDPC codes," *IEEE Trans. Wireless Commun.*, vol. 24, no. 4, pp. 3386–3399, 2025.
- [13] I. Tal and A. Vardy, "List decoding of polar codes," *IEEE Trans. Inf. Theory*, vol. 61, no. 5, pp. 2213–2226, 2015.
- [14] W. Sui, B. Towell, Z. Qu, E. Min, and R. D. Wesel, "Linearity-enhanced serial list decoding of linearly expurgated tail-biting convolutional codes," in *2024 IEEE Int. Symp. Info. Theory*, 2024, pp. 1770–1775.
- [15] S. Kahrman and M. E. Çelebi, "Code based efficient maximum-likelihood decoding of short polar codes," in *2012 IEEE Int. Symp. Inf. Theory*, 2012, pp. 1967–1971.
- [16] J. Piao, K. Niu, J. Dai, and C. Dong, "Approaching the normal approximation of the finite blocklength capacity within 0.025 db by short polar codes," *IEEE Wireless Commun. Lett.*, vol. 9, no. 7, pp. 1089–1092, 2020.
- [17] J. Piao, J. Dai, and K. Niu, "CRC-aided sphere decoding for short polar codes," *IEEE Commun. Lett.*, vol. 23, no. 2, pp. 210–213, 2019.
- [18] S. A. Hashemi, C. Condo, and W. J. Gross, "List sphere decoding of polar codes," in *2015 49th Asilomar Conf. Sig., Sys. Comp.*, 2015, pp. 1346–1350.
- [19] 3GPP, "Multiplexing and channel coding," 3rd Generation Partnership Project (3GPP), TS 38.212, 2020.
- [20] G. Durisi and A. Lancho, "Transmitting short packets over wireless channels—an information-theoretic perspective." [Online]. Available: <https://gdurisi.github.io/fbl-notes/>
- [21] Y. Polyanskiy, S. Chen, A. Collins, G. Durisi, T. Erseghe, G. C. Ferrante, V. Kostina, J. Östman, I. Tal, and W. Yang, "SPECTRE: short packet communication toolbox." [Online]. Available: <https://github.com/yp-mit/spectre>