List Viterbi Algorithm Aided Low-Latency OSD

Xihao Li† and Li Chen†‡
†School of Electronics and Information Technology, Sun Yat-sen University, Guangzhou, China
‡Guangdong Province Key Laboratory of Information Security Technology, Guangzhou, China
Email: lixh98@mail2.sysu.edu.cn, chenli55@mail.sysu.edu.cn

2025 IEEE International Symposium on Information Theory (ISIT) | 979-8-3315-4399-0/25/$31.00 ©2025 IEEE | DOI: 10.1109/ISIT63088.2025.11195613

***Abstract*—Ordered statistics decoding (OSD) requires Gaussian elimination (GE) to obtain the systematic generator matrix (SGM) for yielding codeword candidates, resulting in an uncompromised latency. Recently, low-latency OSD (LLOSD) has been proposed for BCH codes to avoid GE by computing the SGM of a Reed-Solomon (RS) code. Since BCH codes are binary subcodes of the RS codes, the BCH codeword candidates can be yielded with the RS SGM. To further facilitate the LLOSD, this paper proposes a local constraint-based LLOSD (LC-LLOSD). In particular, the RS SGM is converted into a binary BCH parity-check matrix, whose submatrix can be used to specify a trellis. The serial list Viterbi algorithm (SLVA) can be applied to generate extended test messages (TMs). Further, it can be facilitated by incorporating the TM generation scheme of the LLOSD. Since the SLVA generates the TMs in decreasing likelihood, the LC-LLOSD can yield better decoding performance while re-encoding far less TMs compared to the LLOSD. Simulation results show the LC-LLOSD's complexity advantage over the LLOSD.**

***Index Terms*—BCH codes, ordered statistics decoding (OSD), serial list Viterbi algorithm (SLVA)**

I. Introduction

Good performing short-to-median length channel codes are crucial for ultra-reliable low-latency communication (URLLC) in future networks [1]. Recent research showed that ordered statistics decoding (OSD) [2] of BCH codes approaches the finite-length transmission limit [1], [3], revealing BCH codes are one of the competed candidates for substantiating URLLC.

In OSD of binary linear block code, received symbols are first sorted based on their reliability. Gaussian elimination (GE) is then performed to obtain a systematic generator matrix (SGM) of the code, whose identity submatrix is constituted by the columns that correspond to the most reliable basis (MRB). By flipping the hard decisions at the MRB, multiple test messages (TMs) are generated, which are subsequently re-encoded into codeword candidates. Note that in OSD, the Hamming distance between the hard decisions at the MRB and the TMs is limited to the decoding order, denoted by τ. For a binary linear block code with dimension k and minimum Hamming distance d, the OSD requires an order of $\tau = \lceil d/4 - 1 \rceil$ to approach the code's maximum likelihood (ML) decoding performance [2]. However, there are $\sum_{\eta=0}^{\tau} \binom{k}{\eta}$ TMs to be generated and re-encoded, leading to a complexity that grows exponentially with τ. Addressing this issue, both skipping the unpromising TMs [4], [5] and terminating the re-encoding early [6]–[8] can facilitate OSD. Meanwhile, the OSD complexity can also be reduced by using the information out of the MRB, such as the box-and-match techniques [9], [10] and the multi-basis methods [11], [12]. Recently, the local constraint-based OSD (LC-OSD) [13], [14] was proposed, which introduces constraints for the extended TMs by applying a parity-check submatrix of the code. The TMs can be generated by the serial list Viterbi algorithm (SLVA) [15] that functions over the trellis specified by the submatrix.

On the other hand, GE of the OSD is a serial process with an uncompromised latency. There exist several approaches to alleviate the challenge, including pre-computing the SGMs [16], applying a skipping rule for GE [17], and updating the basis of an SGM [18]. Recently, in decoding BCH codes, the low-latency OSD (LLOSD) [19] lifts the need of performing GE through exploring the code's algebraic property of being the binary subcodes of Reed-Solomon (RS) codes. Consequently, the BCH codeword candidates can be produced by SGM of the mother RS code. Entries of the SGM can be determined in a fully parallel manner. However, the LLOSD inherits a larger worst-case complexity as it requires a higher decoding order for maintaining the same decoding performance as the OSD.

In this paper, we propose the local constraint-based LLOSD (LC-LLOSD) to facilitate the LLOSD. In particular, the RS SGM is converted into a binary BCH parity-check matrix, whose submatrix introduces some constraints for the (extended) TMs. With the trellis specified by the submatrix, the SLVA can be applied to generate the TMs. The SLVA can be simplified by using some rows of the submatrix to specify the trellis and other rows to validate the TMs. Furthermore, it can be facilitated by generating additional TMs through flipping hard decisions at the extended MRB (similar to the order-1 LLOSD). Since the SLVA generates the TMs in decreasing order of likelihood, the LC-LLOSD can yield better decoding performance while re-encoding far less TMs compared to the LLOSD. Simulation results show that the number of finite field operations required by the LC-LLOSD is far fewer than the binary operations required by the OSD and the LC-OSD. Also, compared to the LLOSD, the LC-LLOSD requires fewer finite field operations and offers better decoding performance.

II. Preliminaries

A. System Model

Let m be an integer greater than two. Let $\mathbb{F}_2$ and $\mathbb{F}_{2^m}$ denote the finite fields of sizes 2 and 2^m, respectively. Additionally, let $(x)_n$ denote an x-sequence $(x_0, x_1, \ldots, x_{n-1})$ and let $[n]$ denote an integer set $\{0, 1, \ldots, n-1\}$. $\mathcal{C}(n, k, d)$ denotes a binary BCH code with length $n = 2^m - 1$, dimension k, and designed Hamming distance d. Its codeword $\boldsymbol{c} = (c)_n \in \mathbb{F}_2^n$ is modulated using binary phase shift keying (BPSK) and

transmitted over the additive white Gaussian noise (AWGN) channel. The received vector is written as $\boldsymbol{y} = (y)_n$, where its entries $y_j = (-1)^{c_j} + w_j$ for $j \in [n]$, and variables w_j are i.i.d. Gaussian random variables with zero mean and variance $N_0/2$. The log-likelihood ratio (LLR) of y_j is defined as $L_j = \ln \frac{P(y_j \mid c_j=0)}{P(y_j \mid c_j=1)} = \frac{4y_j}{N_0}$, where $P(y_j \mid c_j = 0)$ and $P(y_j \mid c_j = 1)$ denote the channel observations of c_j. The hard-decision vector $\boldsymbol{z} = (z)_n \in \mathbb{F}_2^n$ is determined by letting its entries $z_j = 0$ if $L_j \geq 0$, and $z_j = 1$ otherwise. The reliability vector $\boldsymbol{r} = (r)_n$ can be further obtained by letting $r_j = |L_j|$. A greater r_j indicates the decision z_j is more reliable.

B. The LLOSD

Let $\mathcal{C}'(n, k', d)$ denote an RS code defined over $\mathbb{F}_{2^m}$. Given the same designed distance d [1], binary BCH codes are binary subcodes of the RS codes [20], i.e., $\mathcal{C}(n, k, d) = \mathcal{C}'(n, k', d) \cap \mathbb{F}_2^n$. Furthermore, RS codes are maximum distance separable (MDS) codes. It follows that $k' = n - d + 1$ and $k' > k$.

In the LLOSD [19], the reliability vector $\boldsymbol{r}$ is sorted in decreasing order, resulting in $\Pi(\boldsymbol{r}) = (r_{j_0}, r_{j_1}, \ldots, r_{j_{n-1}})$, where Π represents the permutation and $j_0, j_1, \ldots, j_{n-1}$ are the refreshed symbol indices. Let Θ denote the extended MRB $\{j_0, j_1, \ldots, j_{k'-1}\}$ and α denote a primitive element of $\mathbb{F}_{2^m}$. For the RS code $\mathcal{C}'(n, k', d)$ with code locators $1, \alpha, \ldots, \alpha^{n-1}$, its SGM that is associated with Θ can be computed as

$$\mathbf{G}^{(\mathrm{RS})} = \begin{bmatrix} \mathcal{L}_{j_0}(1) & \mathcal{L}_{j_0}\left(\alpha^1\right) & \ldots & \mathcal{L}_{j_0}\left(\alpha^{n-1}\right) \\ \mathcal{L}_{j_1}(1) & \mathcal{L}_{j_1}\left(\alpha^1\right) & \ldots & \mathcal{L}_{j_1}\left(\alpha^{n-1}\right) \\ \vdots & \vdots & \ddots & \vdots \\ \mathcal{L}_{j_{k'-1}}(1) & \mathcal{L}_{j_{k'-1}}\left(\alpha^1\right) & \ldots & \mathcal{L}_{j_{k'-1}}\left(\alpha^{n-1}\right) \end{bmatrix}, \tag{1}$$

where for $i \in [k']$ and $p \in [n]$,

$$\mathcal{L}_{j_i}(x) = \prod_{s \in \Theta, s \neq j_i} \frac{x - \alpha^s}{\alpha^{j_i} - \alpha^s} \tag{2}$$

is the Lagrange interpolation polynomial w.r.t. code locator α^{j_i}. Note that $\mathbf{G}^{(\mathrm{RS})}$ is in systematic form, since

$$\mathcal{L}_{j_i}\left(\alpha^p\right) = \begin{cases} 1, & p = j_i, \\ 0, & p \in \Theta, p \neq j_i, \\ \frac{\prod_{s \in \Theta}(\alpha^p - \alpha^s)}{\left(\alpha^p - \alpha^{j_i}\right)\prod_{s \in \Theta, s \neq j_i}\left(\alpha^{j_i} - \alpha^s\right)}, & p \in [n] \setminus \Theta. \end{cases} \tag{3}$$

Subsequently, the columns of $\mathbf{G}^{(\mathrm{RS})}$ are permuted according to Π, yielding $\widetilde{\mathbf{G}}^{(\mathrm{RS})} = \Pi\left(\mathbf{G}^{(\mathrm{RS})}\right) = \left[\mathbf{I}_{k'} \quad \widetilde{\mathbf{P}}^{(\mathrm{RS})}\right]$, where $\mathbf{I}_{k'}$ is a $k' \times k'$ identity submatrix and $\widetilde{\mathbf{P}}^{(\mathrm{RS})}$ is the RS parity submatrix. We also define $\widetilde{\boldsymbol{y}} = (\widetilde{y})_n = \Pi(\boldsymbol{y})$, $\widetilde{\boldsymbol{z}} = (\widetilde{z})_n = \Pi(\boldsymbol{z})$, and $\widetilde{\boldsymbol{r}} = (\widetilde{r})_n = \Pi(\boldsymbol{r})$. For a length-$n$ vector, the subscripts B and P are used to denote the first k' and the remaining $n - k'$ positions, respectively. E.g., $\widetilde{\boldsymbol{z}} = (\widetilde{\boldsymbol{z}}_{\mathrm{B}}, \widetilde{\boldsymbol{z}}_{\mathrm{P}})$. Given a test error pattern (TEP) $\boldsymbol{e} \in \mathbb{F}_2^{k'}$, an (extended) TM is generated by flipping $\widetilde{\boldsymbol{z}}_{\mathrm{B}}$ to $\widetilde{\boldsymbol{z}}_{\mathrm{B}} + \boldsymbol{e}$. A (permuted) RS codeword candidate $\widetilde{\boldsymbol{c}}^{(\boldsymbol{e})} = \left(\widetilde{c}^{(\boldsymbol{e})}\right)_n$ can be generated by re-encoding the TM as

$$\widetilde{\boldsymbol{c}}^{(\boldsymbol{e})} = (\widetilde{\boldsymbol{z}}_{\mathrm{B}} + \boldsymbol{e})\widetilde{\mathbf{G}}^{(\mathrm{RS})} = \left(\widetilde{\boldsymbol{z}}_{\mathrm{B}} + \boldsymbol{e}, \widetilde{\boldsymbol{z}}_{\mathrm{B}}\widetilde{\mathbf{P}}^{(\mathrm{RS})} + \boldsymbol{e}\widetilde{\mathbf{P}}^{(\mathrm{RS})}\right). \tag{4}$$

[1]Note that for RS codes, it is the minimum Hamming distance.

If $\boldsymbol{c}^{(\boldsymbol{e})}$ is a binary vector, it is also a valid BCH codeword candidate. Otherwise, it should be discarded. In the order-τ LLOSD, the TEPs are enumerated with Hamming weight increasing from 0 to τ. In total, there are $\sum_{\eta=0}^{\tau}\binom{k'}{\eta}$ TMs. The optimal codeword candidate $\widetilde{\boldsymbol{c}}^{(\mathrm{opt})}$ is identified by selecting the candidate $\widetilde{\boldsymbol{c}}^{(\boldsymbol{e})}$ that minimizes the correlation distance to the received vector $\widetilde{\boldsymbol{y}}$, which is defined as $\mathcal{D}\left(\widetilde{\boldsymbol{c}}^{(\boldsymbol{e})}, \widetilde{\boldsymbol{y}}\right) = \sum_{j:\widetilde{c}_j^{(\boldsymbol{e})} \neq \widetilde{z}_j} \widetilde{r}_j$. The decoding output $\hat{\boldsymbol{c}}$ is obtained as $\hat{\boldsymbol{c}} = \Pi^{-1}\left(\widetilde{\boldsymbol{c}}^{(\mathrm{opt})}\right)$, where Π^{-1} is the inverse of permutation Π.

Note that in practice, the LLOSD requires a higher order to achieve the same decoding performance as the OSD. This leads to a higher worst-case complexity. Addressing this issue, the ML stopping condition [21] is used to terminate the re-encoding process early. For a codeword candidate $\widetilde{\boldsymbol{c}}^{(\boldsymbol{e})}$, let $d_{\boldsymbol{e}}$ denote its Hamming distance to $\widetilde{\boldsymbol{z}}$. Let the position set $\left\{j : \widetilde{c}_j^{(\boldsymbol{e})} = \widetilde{z}_j\right\}$ be re-written as $\{l_0, l_1, \ldots, l_{n-d_{\boldsymbol{e}}-1}\}$, where $l_0 < l_1 < \cdots < l_{n-d_{\boldsymbol{e}}-1}$. The ML stopping condition is the follows. If

$$\mathcal{D}\left(\widetilde{\boldsymbol{c}}^{(\boldsymbol{e})}, \widetilde{\boldsymbol{y}}\right) \leq \sum_{j=n-d}^{n-d_{\boldsymbol{e}}-1} \widetilde{r}_{l_j}, \tag{5}$$

the decoding terminates and outputs $\Pi^{-1}\left(\widetilde{\boldsymbol{c}}^{(\boldsymbol{e})}\right)$.

Lemma 1. ([19]) The complexity of the LLOSD is

$$\Omega_{\mathrm{L}} = \underbrace{O(n \log n)}_{\text{sorting } \boldsymbol{r}} + \underbrace{O(n(n-k'))}_{\text{obtaining } \mathbf{G}^{(\mathrm{RS})}} + \underbrace{O\left(k'(n-k') + \tau k'^{\tau}\right)}_{\text{re-encoding the TMs}}. \tag{6}$$

Despite the LLOSD requires a higher order, only very few BCH codeword candidates are generated, as most of the re-encoded codewords are invalid (being non-binary). Therefore, the complexity of computing the correlation distances and checking the ML stopping condition are minor compared to the terms of (6). Hence, they are dropped.

III. The LC-LLOSD

This section proposes the LC-LLOSD, which generates the TMs by applying the SLVA [15] over the trellis that is specified by the parity-check submatrix of the BCH code. To further reduce the complexity, two approaches are also proposed.

A. The Proposed Decoding

After obtaining $\widetilde{\mathbf{G}}^{(\mathrm{RS})} = \left[\mathbf{I}_{k'} \quad \widetilde{\mathbf{P}}^{(\mathrm{RS})}\right]$ through (1) and permutation Π, the RS parity-check matrix can be derived as

$$\widetilde{\mathbf{H}}^{(\mathrm{RS})} = \left[\left(\widetilde{\mathbf{P}}^{(\mathrm{RS})}\right)^{\mathrm{T}} \quad \mathbf{I}_{n-k'}\right], \tag{7}$$

where T denotes transposition of the matrix. Given an element $\gamma \in \mathbb{F}_{2^m}$, it can be represented by a length-m binary column vector that is written as $\left(\gamma^{(0)}, \gamma^{(1)}, \ldots, \gamma^{(m-1)}\right)^{\mathrm{T}}$, such that

$$\gamma = \left(1, \alpha, \ldots, \alpha^{m-1}\right) \cdot \left(\gamma^{(0)}, \gamma^{(1)}, \ldots, \gamma^{(m-1)}\right)^{\mathrm{T}}. \tag{8}$$

Hence, replacing each element of $\widetilde{\mathbf{H}}^{(\mathrm{RS})}$ with a length-m binary column yields a binary parity-check matrix of the BCH

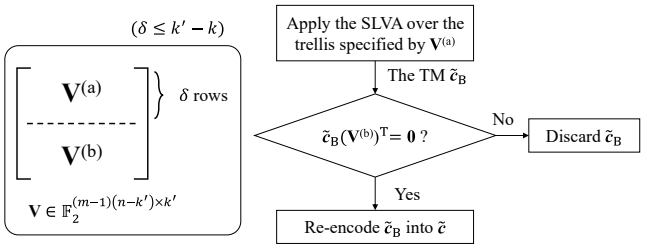

Fig. 1. Generating and re-encoding TMs with limited trellis states in SLVA.

code $\mathcal{C}(n,k,d)$ [22]. Further, its rows are permuted to form an identity submatrix in its top-right corner as

$$\widetilde{\mathbf{H}}^{(\mathrm{BCH})} = \begin{bmatrix} \mathbf{U} & \mathbf{I}_{n-k'} \\ \mathbf{V} & \mathbf{0} \end{bmatrix}, \tag{9}$$

where $\mathbf{0}$ is an all zero matrix, $\mathbf{U} \in \mathbb{F}_2^{(n-k')\times k'}$ and $\mathbf{V} \in \mathbb{F}_2^{(m-1)(n-k')\times k'}$. Note that $\widetilde{\mathbf{H}}^{(\mathrm{BCH})}$ has a rank of $n-k$. Any (permuted) BCH codeword candidate $\widetilde{\boldsymbol{c}} = (\widetilde{\boldsymbol{c}}_{\mathrm{B}}, \widetilde{\boldsymbol{c}}_{\mathrm{P}})$ should satisfy $\widetilde{\boldsymbol{c}}_{\mathrm{B}}\mathbf{U}^{\mathrm{T}} = \widetilde{\boldsymbol{c}}_{\mathrm{P}}$ and $\widetilde{\boldsymbol{c}}_{\mathrm{B}}\mathbf{V}^{\mathrm{T}} = \mathbf{0}$.

Thereby, to generate $\widetilde{\boldsymbol{c}}$, we can first generate the TM $\widetilde{\boldsymbol{c}}_{\mathrm{B}}$ that satisfies $\widetilde{\boldsymbol{c}}_{\mathrm{B}}\mathbf{V}^{\mathrm{T}} = \mathbf{0}$ and then re-encode it as $\widetilde{\boldsymbol{c}} = \left(\widetilde{\boldsymbol{c}}_{\mathrm{B}}, \widetilde{\boldsymbol{c}}_{\mathrm{B}}\mathbf{U}^{\mathrm{T}}\right)$. For this, the SLVA that functions over the trellis specified by $\mathbf{V}$ can be applied to generate the TMs. However, since the submatrix $\begin{bmatrix}\mathbf{U} & \mathbf{I}_{n-k'}\end{bmatrix}$ has a rank of $n-k'$, $\mathbf{V}$ has a rank of $(n-k)-(n-k') = k'-k$. Hence, the trellis specified by $\mathbf{V}$ has at most $2^{k'-k}$ states [23], leading to a high SLVA complexity. In order to further reduce the complexity of generating the TMs, the following two approaches are proposed.

B. Limitation of Trellis States

To limit the trellis states in the SLVA, Fig. 1 illustrates the proposed approach for generating and re-encoding the TMs. Let δ denote an integer not greater than $k'-k$. The matrix $\mathbf{V}$ is partitioned into two submatrices as $\mathbf{V} = \left[\left(\mathbf{V}^{(\mathrm{a})}\right)^{\mathrm{T}} \quad \left(\mathbf{V}^{(\mathrm{b})}\right)^{\mathrm{T}}\right]^{\mathrm{T}}$, where $\mathbf{V}^{(\mathrm{a})} \in \mathbb{F}_2^{\delta\times k'}$ and $\mathbf{V}^{(\mathrm{b})} \in \mathbb{F}_2^{((m-1)(n-k')-\delta)\times k'}$. In particular, $\mathbf{V}^{(\mathrm{a})}$ is formed by the first δ rows of $\mathbf{V}$. The remaining rows of $\mathbf{V}$ form $\mathbf{V}^{(\mathrm{b})}$. Generating the TM $\widetilde{\boldsymbol{c}}_{\mathrm{B}}$ that satisfies $\widetilde{\boldsymbol{c}}_{\mathrm{B}}\mathbf{V}^{\mathrm{T}} = \mathbf{0}$ can be accomplished in two steps. First, the SLVA functions over the trellis that is specified by $\mathbf{V}^{(\mathrm{a})}$, yielding the TM $\widetilde{\boldsymbol{c}}_{\mathrm{B}}$ such that $\widetilde{\boldsymbol{c}}_{\mathrm{B}}\left(\mathbf{V}^{(\mathrm{a})}\right)^{\mathrm{T}} = \mathbf{0}$. Secondly, the condition $\widetilde{\boldsymbol{c}}_{\mathrm{B}}\left(\mathbf{V}^{(\mathrm{b})}\right)^{\mathrm{T}} = \mathbf{0}$ is checked. If it is met, $\widetilde{\boldsymbol{c}}_{\mathrm{B}}$ will satisfy $\widetilde{\boldsymbol{c}}_{\mathrm{B}}\mathbf{V}^{\mathrm{T}} = \mathbf{0}$ and will be re-encoded into $\widetilde{\boldsymbol{c}}$. Otherwise, it will be discarded. Note that the rank of $\mathbf{V}^{(\mathrm{a})}$ is at most δ, which limits the number of trellis states to at most 2^{δ}. However, a smaller δ will lead to the generation of more TMs. Section IV will discuss the optimized selection of δ.

Note that the aforementioned process of generating and re-encoding the TMs can be further optimized. Let $l_{\max}$ denote the maximum number of TMs to be generated. For $h \in [l_{\max}]$, let $\widetilde{\boldsymbol{c}}_{\mathrm{B}}^{(h)}$ denote the $(h+1)$-th TM that is generated by the SLVA. The TEP can also be written as $\boldsymbol{e}^{(h)} = \widetilde{\boldsymbol{c}}_{\mathrm{B}}^{(h)} + \widetilde{\boldsymbol{z}}_{\mathrm{B}}$. Hence, the condition $\widetilde{\boldsymbol{c}}_{\mathrm{B}}^{(h)}\left(\mathbf{V}^{(\mathrm{b})}\right)^{\mathrm{T}} = \mathbf{0}$ is equivalent to $\boldsymbol{e}^{(h)}\left(\mathbf{V}^{(\mathrm{b})}\right)^{\mathrm{T}} = \widetilde{\boldsymbol{z}}_{\mathrm{B}}\left(\mathbf{V}^{(\mathrm{b})}\right)^{\mathrm{T}}$. When validating the condition, $\widetilde{\boldsymbol{z}}_{\mathrm{B}}\left(\mathbf{V}^{(\mathrm{b})}\right)^{\mathrm{T}}$ can be computed once and reused. Moreover, for re-encoding $\widetilde{\boldsymbol{c}}_{\mathrm{B}}^{(h)}$ as $\widetilde{\boldsymbol{c}}^{(h)} = (\widetilde{\boldsymbol{c}}_{\mathrm{B}}^{(h)}, \widetilde{\boldsymbol{c}}_{\mathrm{B}}^{(h)}\mathbf{U}^{\mathrm{T}})$, the vector $\widetilde{\boldsymbol{c}}_{\mathrm{B}}^{(h)}\mathbf{U}^{\mathrm{T}}$ can be computed as $\widetilde{\boldsymbol{z}}_{\mathrm{B}}\mathbf{U}^{\mathrm{T}} + \boldsymbol{e}^{(h)}\mathbf{U}^{\mathrm{T}}$, where $\widetilde{\boldsymbol{z}}_{\mathrm{B}}\mathbf{U}^{\mathrm{T}}$ can again be computed once.

Algorithm 1: LC-LLOSD

Input: $\boldsymbol{y}$, δ, $l_{\max}$
Output: $\hat{\boldsymbol{c}}$

Obtain the hard-decision vector $\boldsymbol{z}$ and the reliability vector $\boldsymbol{r}$ based on $\boldsymbol{y}$;
Sort $\boldsymbol{r}$ in decreasing order, yielding permutation Π;
Obtain $\mathbf{G}^{(\mathrm{RS})}$ as in (1) and let $\widetilde{\mathbf{G}}^{(\mathrm{RS})} = \Pi(\mathbf{G}^{(\mathrm{RS})})$;
For $\eta = 0, 1$ **do**
  For each TEP $\boldsymbol{e} \in \mathbb{F}_2^{k'}$ of weight η **do**
    Generate the RS codeword $\widetilde{\boldsymbol{c}}^{(\boldsymbol{e})}$ as in (4);
    If $\widetilde{\boldsymbol{c}}^{(\boldsymbol{e})}$ is binary **and** the ML stopping condition of (5) is satisfied **then**
      Return $\hat{\boldsymbol{c}} = \Pi^{-1}(\widetilde{\boldsymbol{c}}^{(\boldsymbol{e})})$;
Obtain $\widetilde{\mathbf{H}}^{(\mathrm{BCH})}$ as in (9);
Form $\mathbf{V}^{(\mathrm{a})}$ by the frist δ rows of $\mathbf{V}$;
Form $\mathbf{V}^{(\mathrm{b})}$ by the remaining rows of $\mathbf{V}$;
Initialize $\mathcal{D}\left(\widetilde{\boldsymbol{c}}^{(\mathrm{opt})}, \widetilde{\boldsymbol{y}}\right) = \infty$;
For $h = 0, 1, \ldots, l_{\max} - 1$ **do**
  Apply the SLVA over the trellis specified by $\mathbf{V}^{(\mathrm{a})}$ to generate the $(h+1)$-th TM $\widetilde{\boldsymbol{c}}_{\mathrm{B}}^{(h)}$;
  If $\widetilde{\boldsymbol{c}}_{\mathrm{B}}^{(h)}(\mathbf{V}^{(\mathrm{b})})^{\mathrm{T}} = \mathbf{0}$ **then**
    Recover $\widetilde{\boldsymbol{c}}^{(h)} = (\widetilde{\boldsymbol{c}}_{\mathrm{B}}^{(h)}, \widetilde{\boldsymbol{c}}_{\mathrm{B}}^{(h)}\mathbf{U}^{\mathrm{T}})$;
    If $\mathcal{D}(\widetilde{\boldsymbol{c}}^{(h)}, \widetilde{\boldsymbol{y}}) < \mathcal{D}(\widetilde{\boldsymbol{c}}^{(\mathrm{opt})}, \widetilde{\boldsymbol{y}})$ **then**
      Let $\widetilde{\boldsymbol{c}}^{(\mathrm{opt})} = \widetilde{\boldsymbol{c}}^{(h)}$;
  If the stopping condition of (10) is satisfied **then**
    Break;
Return $\hat{\boldsymbol{c}} = \Pi^{-1}\left(\widetilde{\boldsymbol{c}}^{(\mathrm{opt})}\right)$;

Similar to the LLOSD, the optimal codeword candidate $\widetilde{\boldsymbol{c}}^{(\mathrm{opt})}$ is identified as the one that minimizes the correlation distance to the received vector $\widetilde{\boldsymbol{y}}$. The decoding output is obtained as $\Pi^{-1}\left(\widetilde{\boldsymbol{c}}^{(\mathrm{opt})}\right)$. Instead of the ML stopping condition of (5), a more efficient stopping condition of [14] can be applied to the SLVA, as the TMs are generated in decreasing order of likelihood. That is, after obtaining the TM $\widetilde{\boldsymbol{c}}_{\mathrm{B}}^{(h)}$, if

$$\mathcal{D}\left(\widetilde{\boldsymbol{c}}^{(\mathrm{opt})}, \widetilde{\boldsymbol{y}}\right) < \mathcal{D}\left(\widetilde{\boldsymbol{c}}_{\mathrm{B}}^{(h)}, \widetilde{\boldsymbol{y}}_{\mathrm{B}}\right) + \sum_{j=k'}^{n-1} \frac{\widetilde{r}_j}{1+\exp(4\widetilde{r}_j/N_0)}, \tag{10}$$

the decoding terminates and yields $\Pi^{-1}\left(\widetilde{\boldsymbol{c}}^{(\mathrm{opt})}\right)$ as the output.

C. Combinatorial Generation of TMs

Before applying the SLVA, additional TMs can be generated by flipping no more than one hard decisions at the extended MRB (like the order-1 LLOSD) and re-encoded into codeword candidates. If the ML stopping condition of (5) is satisfied, the decoding can be terminated without applying the SLVA. Note that the stopping condition of (10) is not applied to these TMs, as their likelihoods are not ensured to be in decreasing order.

When the channel condition is sufficiently good, the stopping condition of (5) will be satisfied very frequently, leading

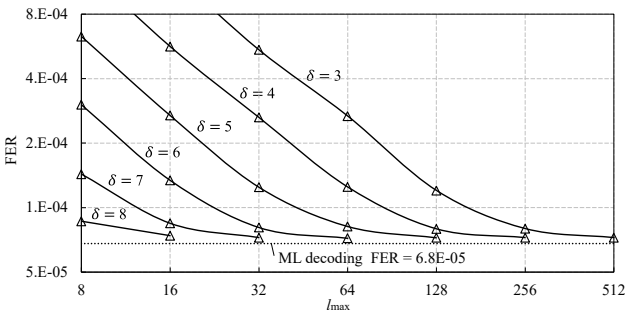

Fig. 2. FER of the LC-LLOSD with different values of δ and $l_{\max}$ in decoding the $(63, 45, 7)$ BCH code at the SNR of 5 dB.

to the removal of running the SLVA and the significantly reduced LC-LLOSD complexity. *Algorithm* 1 summarizes the proposed LC-LLOSD by incorporating the complexity reducing techniques mentioned in the above two subsections.

IV. Complexity Analysis

Note that the rank of $\mathbf{V}^{(\mathrm{b})}$ is approximately $k' - k - \delta$. Therefore, the condition $\widetilde{\boldsymbol{c}}_{\mathrm{B}} \left(\mathbf{V}^{(\mathrm{b})}\right)^{\mathrm{T}} = \mathbf{0}$ is satisfied with a probability of approximately $1/2^{k'-k-\delta}$. To maintain the LC-LLOSD performance, the number of TMs that satisfy the condition $\widetilde{\boldsymbol{c}}_{\mathrm{B}} \left(\mathbf{V}^{(\mathrm{b})}\right)^{\mathrm{T}} = \mathbf{0}$ should be preserved. Hence, $l_{\max}$ should be set proportional to $2^{k'-k-\delta}$.

As an example, Fig. 2 shows the frame error rate (FER) performance of the LC-LLOSD with different values of δ and $l_{\max}$ in decoding the $(63, 45, 7)$ BCH code at the signal-to-noise ratio (SNR) of 5 dB. For this code, $k' - k = 12$. As shown, $l_{\max}$ should be set proportional to $2^{k'-k-\delta}$ to maintain the performance. This is because a larger number of trellis states leads to a more selective generation of the TMs, thereby reducing the list size. Hence, adjusting δ and $l_{\max}$ trade-offs the decoding performance and complexity. The following lemma characterizes the LC-LLOSD complexity, which will show that $\delta = (k' - k)/2$ offers a good trade-off for the decoding.

Lemma 2. The complexity of the LC-LLOSD is

$$\begin{aligned}\Omega_{\mathrm{C}} = &\underbrace{O(n \log n)}_{\text{sorting } \boldsymbol{r}} + \underbrace{O(n(n-k'))}_{\text{obtaining } \mathbf{G}^{(\mathrm{RS})}} + \underbrace{O\left(2^{\delta} k' + l_{\max} k'\right)}_{\text{applying the SLVA}} \\ &+ \underbrace{O\left(l_{\max} k' + \left(l_{\max}/2^{k'-k-\delta}\right) k'(n-k') \log n\right)}_{\text{checking and re-encoding the TMs generated by the SLVA}}. \end{aligned} \tag{11}$$

Proof: The complexity of sorting $\boldsymbol{r}$ and generating $\mathbf{G}^{(\mathrm{RS})}$ follows from *Lemma* 1. For the TMs generated by flipping no more than one hard decision at the extended MRB, re-encoding them has a complexity of $O(k'(n-k'))$. This complexity is not expressed in (11) since generating $\mathbf{G}^{(\mathrm{RS})}$ has a complexity of $O(n(n-k'))$. Also, the complexity of transforming $\widetilde{\mathbf{G}}^{(\mathrm{RS})}$ into $\widetilde{\mathbf{H}}^{(\mathrm{BCH})}$ is omitted, as it involves no computation. The complexity of computing the correlation distances and checking the stopping condition of (10) is also omitted in (11), as these are part of the SLVA complexity.

Since the trellis specified by $\mathbf{V}^{(\mathrm{a})}$ has at most 2^{δ} states, the SLVA generates the first TM $\widetilde{\boldsymbol{c}}_{\mathrm{B}}^{(0)}$ with a complexity of $O\left(2^{\delta} k'\right)$. For $h > 0$, the SLVA generates each TM $\widetilde{\boldsymbol{c}}_{\mathrm{B}}^{(h)}$ with a further complexity of $O(k')$. Hence, the SLVA has a complexity of $O\left(2^{\delta} k' + l_{\max} k'\right)$ [13], [15].

After the SLVA generates the $(h+1)$-th TM $\widetilde{\boldsymbol{c}}_{\mathrm{B}}^{(h)}$, the condition $\widetilde{\boldsymbol{c}}_{\mathrm{B}}^{(h)} \left(\mathbf{V}^{(\mathrm{b})}\right)^{\mathrm{T}} = \mathbf{0}$ is then checked. Note that the check can be terminated once a parity-check equation specified by a row of $\mathbf{V}^{(\mathrm{b})}$ detects that $\widetilde{\boldsymbol{c}}_{\mathrm{B}}^{(h)}$ violates it. Therefore, if the TM does not satisfy the condition, the check has a complexity of $O(k')$. Otherwise, the check has a complexity of $O(mk'(n-k'))$, and re-encoding the TM $\widetilde{\boldsymbol{c}}_{\mathrm{B}}^{(h)}$ through $\widetilde{\boldsymbol{c}}^{(h)} = \left(\widetilde{\boldsymbol{c}}_{\mathrm{B}}^{(h)}, \widetilde{\boldsymbol{c}}_{\mathrm{B}}^{(h)} \mathbf{U}^{\mathrm{T}}\right)$ has a complexity of $O(k'(n-k'))$. Note that among the $l_{\max}$ TMs generated by the SLVA, only approximately $l_{\max}/2^{k'-k-\delta}$ TMs satisfy the condition and require re-encoding. Therefore, checking and re-encoding all the TMs generated by the SLVA has a complexity of $O\left(l_{\max} k' + \left(l_{\max}/2^{k'-k-\delta}\right) k'(n-k') \log n\right)$. ■

Note that for the TMs generated by the SLVA, the process of checking and re-encoding can be simplified by utilizing the TEP $\boldsymbol{e}^{(h)}$ as outlined in Section III-B. Since the SLVA generates the TMs in decreasing likelihood order and can be efficiently terminated using the stopping condition of (10), it can be assumed that the TEP $\boldsymbol{e}^{(h)}$ has a Hamming weight of at most $2d = 2(n - k' + 1)$. The complexity of checking and re-encoding the TMs can be reduced to $O\left(l_{\max}(n-k') + \left(l_{\max}/2^{k'-k-\delta}\right)(n-k')^2 \log n\right)$ [2].

Since $l_{\max}$ is set proportional to $2^{k'-k-\delta}$, the asymptotic complexity of the SLVA is bounded below by $O\left(2^{(k'-k)/2} k'\right)$ when $\delta = (k'-k)/2$. However, the complexity of the SLVA grows exponentially with $k'-k$. It dominates the overall LC-LLOSD complexity. Compared to the LLOSD complexity that is characterized in (6), the LC-LLOSD has two additional terms associated with the SLVA. Nevertheless, our simulation results will show that the LC-LLOSD can significantly reduce both the number of TMs and the complexity of the LLOSD.

V. Simulation Results

This section shows the performance and complexity of different decoding algorithms, including the OSD [2], the LC-OSD [13], the LLOSD [19], and the proposed LC-LLOSD. For simplicity, let OSD (τ) and LLOSD (τ) denote the order-τ OSD and order-τ LLOSD, respectively. Let LC-OSD $(\delta, l_{\max})$ denote the LC-OSD with constraints δ and a maximum list size $l_{\max}$. Similarly, let LC-LLOSD $(\delta, l_{\max})$ denote the parameterized LC-LLOSD. To ensure fairness, the stopping conditions of (5) and (10) are applied to the OSD and the LC-OSD, respectively. The ML decoding performance is from [24].

Fig. 3 shows the FER performance of different algorithms in decoding the $(63, 45, 7)$ BCH code. It can be seen that the LC-LLOSD performs almost the same as the LC-OSD and

[2]Based on the Plotkin bound, BCH codes with a dimension $k \geq m$ satisfy $d \leq 2^{m-1}$. Their corresponding mother RS codes satisfy $k' > n/2$.

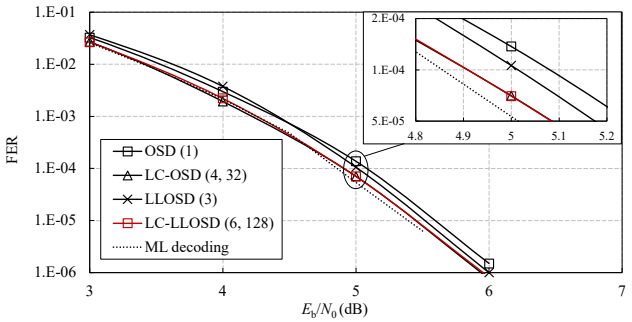

Fig. 3. FER of different algorithms in decoding the $(63, 45, 7)$ BCH code.

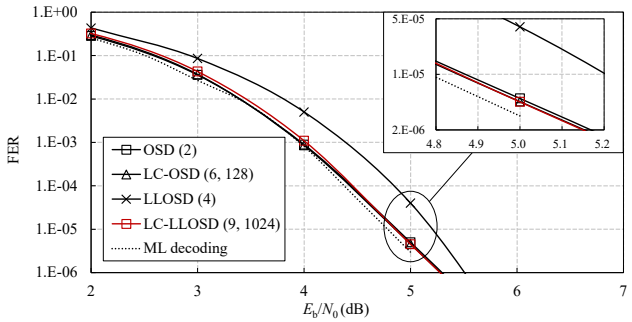

Fig. 4. FER of different algorithms in decoding the $(127, 99, 9)$ BCH code.

TABLE I
COMPLEXITY OF ALGORITHMS IN DECODING THE $(63, 45, 7)$ BCH CODE

A. Average number of generated TMs.

SNR	OSD	LC-OSD	LLOSD	LC-LLOSD
4 dB	8.4×10^0	1.6×10^0	5.0×10^3	1.2×10^1
5 dB	2.2×10^0	1.4×10^0	8.3×10^2	3.1×10^0
6 dB	1.1×10^0	1.3×10^0	4.1×10^1	1.2×10^0

B. Average number of required operations and latency at the SNR of 5 dB.

		OSD	LC-OSD	LLOSD	LC-LLOSD
Oper. Type	$\mathbb{F}_2$	6.7×10^3	1.4×10^4	0	3.6×10^2
	FLOPs	5.0×10^2	2.1×10^3	4.9×10^2	6.5×10^2
	$\mathbb{F}_{64}$	0	0	4.5×10^3	2.0×10^3
Latency (μs)		24.4	79.2	52.2	15.6

TABLE II
COMPLEXITY OF ALGORITHMS IN DECODING THE (127,99,9) BCH CODE

A. Average number of generated TMs.

SNR	OSD	LC-OSD	LLOSD	LC-LLOSD
3 dB	3.6×10^3	5.1×10^0	6.1×10^6	1.5×10^2
4 dB	1.5×10^3	1.9×10^0	2.5×10^6	4.6×10^1
5 dB	2.2×10^2	1.4×10^0	3.6×10^5	8.4×10^0

B. Average number of required operations and latency at the SNR of 5 dB.

		OSD	LC-OSD	LLOSD	LC-LLOSD
Oper. Type	$\mathbb{F}_2$	4.9×10^4	8.9×10^4	0	1.4×10^4
	FLOPs	4.6×10^3	1.1×10^4	1.1×10^3	5.6×10^3
	$\mathbb{F}_{128}$	0	0	1.5×10^6	5.3×10^3
Latency (μs)		164.1	224.8	33423.0	108.9

it approaches the ML decoding performance. At the FER of 10^{-4}, the LC-LLOSD achieves a performance gain of 0.2 dB and 0.1 dB over the OSD and the LLOSD, respectively. Note that we select $\delta = (k' - k)/2 = 6$ and $l_{\max} = 2^{\delta+1} = 128$ to contain the LC-LLOSD complexity. Compared with the LC-OSD, despite the LC-LLOSD introduces two more constraints for the TMs in applying the SLVA (with the parameter δ being greater by two), it still requires a larger $l_{\max}$. This is because, in the LC-LLOSD, the TM $\widetilde{\boldsymbol{c}}_{\mathrm{B}}^{(h)}$ generated by the SLVA must satisfy the condition $\widetilde{\boldsymbol{c}}_{\mathrm{B}}^{(h)} \left(\mathbf{V}^{(\mathrm{b})}\right)^{\mathrm{T}} = \mathbf{0}$ in order to be re-encoded, which results in some TMs being discarded.

Table I shows the complexity of these algorithms in decoding the $(63, 45, 7)$ BCH code with parameters mentioned above. In particular, Table I-A shows the average number of generated TMs. Thanks to the SLVA and the stopping condition of (10), the LC-LLOSD generates significantly fewer TMs than the LLOSD. Compared with the OSD and the LC-OSD, the LC-LLOSD generates more TMs. This comes from the LC-LLOSD also generating the TMs by flipping hard decisions at the extended MRB. Table I-B shows the average number of required operations and latency at the SNR of 5 dB. It shows the LC-LLOSD significantly reduces the $\mathbb{F}_{64}$ operations over the LLOSD. Meanwhile, the number of $\mathbb{F}_{64}$ operations required by the LLOSD and LC-LLOSD is smaller than that of binary operations required by the OSD and the LC-OSD. For the LC-LLOSD, the required number of floating point operations (FLOPs) and binary operations are minor in comparison with the number of the $\mathbb{F}_{64}$ operations. This is because in the shown SNR regime, the SLVA is applied with a very low probability. As a result, the LC-LLOSD complexity mainly arises from obtaining $\mathbf{G}^{(\mathrm{RS})}$ and re-encoding the first TM $\widetilde{\boldsymbol{z}}_{\mathrm{B}}$. The latency results are measured using simulation time on an Intel Core i5-8400 CPU. In the LLOSD and the LC-LLOSD, we assume that $\mathbf{G}^{(\mathrm{RS})}$ is generated in parallel. As shown, the LC-LLOSD inherits an advantage in decoding latency, since it avoids GE and generates only a few TMs.

Fig. 4 and Table II further show the FER performance and the complexity, respectively, of different algorithms in decoding the $(127, 99, 9)$ BCH code. For this code, $k' - k = 20$. The parameters of the LC-LLOSD are $\delta = 9$ and $l_{\max} = 1024$. Fig. 4 shows that the LC-LLOSD approaches the ML decoding performance and achieves a 0.3 dB gain over the LLOSD at the FER of 10^{-5}. Table II-A illustrates that the LC-LLOSD generates far fewer TMs than both the OSD and the LLOSD. Table II-B shows that the numbers of binary operations, FLOPs, and $\mathbb{F}_{128}$ operations required by the LC-LLOSD are of the same order of magnitude, indicating that the complexity of the SLVA is comparable to that of generating $\mathbf{G}^{(\mathrm{RS})}$. Compared to the LLOSD, the number of $\mathbb{F}_{128}$ operations required by the LC-LLOSD is fewer by two orders of magnitude. Also, the latency of the LC-LLOSD is much lower. The LC-LLOSD decodes longer BCH codes more effectively than the LLOSD.

ACKNOWLEDGMENT

This work is supported in part by the National Natural Science Foundation of China (NSFC) with project ID 62471503; and in part by the Natural Science Foundation of Guangdong Province with project ID 2024A1515010213.

REFERENCES

[1] M. Shirvanimoghaddam *et al.*, "Short block-length codes for ultra-reliable low latency communications," *IEEE Commun. Mag.*, vol. 57, no. 2, pp. 130–137, Feb. 2019.

[2] M. P. C. 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.

[3] 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.

[4] Y. Wu and C. Hadjicostis, "Soft-decision decoding using ordered recodings on the most reliable basis," *IEEE Trans. Inf. Theory*, vol. 53, no. 2, pp. 829–836, Feb. 2007.

[5] C. Yue, M. Shirvanimoghaddam, B. Vucetic, and Y. Li, "A revisit to ordered statistics decoding: Distance distribution and decoding rules," *IEEE Trans. Inf. Theory*, vol. 67, no. 7, pp. 4288–4337, Jul. 2021.

[6] W. Jin and M. P. C. Fossorier, "Probabilistic sufficient conditions on optimality for reliability based decoding of linear block codes," in *Proc. IEEE Int. Symp. Inform. Theory (ISIT)*, Seattle, USA, Jul. 2006, pp. 2235–2239.

[7] C. Choi and J. Jeong, "Fast and scalable soft decision decoding of linear block codes," *IEEE Commun. Lett.*, vol. 23, no. 10, pp. 1753–1756, Oct. 2019.

[8] X. Li, W. Chen, L. Chen, Y. Li, and H. Zhang, "Order skipping ordered statistics decoding and its performance analysis," in *Proc. IEEE Inf. Theory Workshop (ITW)*, Shenzhen, China, Nov. 2024, pp. 448–453.

[9] A. Valembois and M. P. C. Fossorier, "Box and match techniques applied to soft-decision decoding," *IEEE Trans. Inf. Theory*, vol. 50, no. 5, pp. 796–810, May 2004.

[10] Y. Wu and M. P. C. Fossorier, "Soft-decision decoding using time and memory diversification," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jul. 2008, pp. 76–80.

[11] W. Jin and M. P. C. Fossorier, "Reliability-based soft-decision decoding with multiple biases," *IEEE Trans. Inf. Theory*, vol. 53, no. 1, pp. 105–120, Jan. 2007.

[12] S. Bitzer and M. Bossert, "On multibasis information set decoding," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Espoo, Finland, Jun. 2022, pp. 2780–2784.

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

[14] 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.

[15] N. Seshadri and C.-E. W. Sundberg, "List Viterbi decoding algorithms with applications," *IEEE Trans. Commun.*, vol. 42, no. 2/3/4, pp. 313–323, Feb. 1994.

[16] C. Choi and J. Jeong, "Fast soft decision decoding algorithm for linear block codes using permuted generator matrices," *IEEE Commun. Lett.*, vol. 25, no. 12, pp. 3775–3779, Dec. 2021.

[17] C. Yue, M. Shirvanimoghaddam, B. Vucetic, and Y. Li, "Ordered-statistics decoding with adaptive Gaussian elimination reduction for short codes," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Rio de Janeiro, Brazil, Dec. 2022, pp. 492–497.

[18] X. Li, W. Chen, L. Chen, and H. Zhang, "Iterative basis update for ordered statistics decoding of linear block codes," *IEEE Commun. Lett.*, vol. 28, no. 9, pp. 1981–1985, 2024.

[19] L. Yang and L. Chen, "Low-latency ordered statistics decoding of BCH codes," in *Proc. IEEE Inf. Theory Workshop (ITW)*, Mumbai, India, Nov. 2022, pp. 404–409.

[20] P. Delsarte, "On subfield subcodes of modified Reed-Solomon codes," *IEEE Trans. Inf. Theory*, vol. 21, no. 5, pp. 575–576, Sep. 1975.

[21] D. J. Taipale and M. B. Pursley, "An improvement to generalized-minimum-distance decoding," *IEEE Trans. Inf. Theory*, vol. 37, no. 1, pp. 167–172, Jan. 1991.

[22] F. J. MacWilliams and N. J. A. Sloane, *The theory of error correcting codes*. Amsterdam: Elsevier, 1977, vol. 16.

[23] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inf. Theory*, vol. 20, no. 2, pp. 284–287, Mar. 1974.

[24] M. Helmling *et al.*, "Database of Channel Codes and ML Simulation Results," www.uni-kl.de/channel-codes, 2019.