Random Linear Streaming Codes in the Finite Memory Length and Decoding Deadline Regime

Pin-Wen Su*, Yu-Chih Huang†, Shih-Chun Lin‡, I-Hsiang Wang§, and Chih-Chun Wang*
*School of ECE, Purdue University, USA, {su173, chihw}@purdue.edu
†Institute of Communications Engineering, National Yang Ming Chiao Tung University, Taiwan, jerryhuang@nctu.edu.tw
‡Department of ECE, National Taiwan University of Science and Technology, Taiwan, sclin@ntust.edu.tw
§Department of EE, National Taiwan University, Taiwan, ihwang@ntu.edu.tw

***Abstract*—*Streaming codes* take a string of source symbols as input and output a string of coded symbols in real time, which effectively eliminate the queueing delay and are regarded as a promising scheme for low latency communications. Aiming at quantifying the fundamental latency performance of *random linear streaming codes* (RLSCs) over i.i.d. symbol erasure channels, this work derives the exact error probability under, simultaneously, the finite memory length and finite decoding deadline constraints. The result is then used to examine the tradeoff among memory length (complexity), decoding deadline (delay), and error probability (reliability) of RLSCs for the first time in the literature. Two critical observations are made: (i) *Too much memory can adversely impact the performance under a finite decoding deadline constraint*, a surprising finding not captured by the traditional wisdom that large memory length monotonically improves the performance in the asymptotic regime; (ii) The end-to-end delay of the RLSC is roughly 50% of that of the MDS block code when under identical code rate and error probability requirements. This implies that switching from block codes to RLSCs not only eliminates the queueing delay (thus 50%) but also has little negative impact on the error probability.**

I. Introduction

The design goal of next-generation low-latency communication schemes [1] is to minimize the end-to-end (E2E) delay while attaining the predefined *reliability* and *throughput* requirements. Two major sources of the E2E delay are the queueing delay and the transmission delay. The former describes the time for *the source to accumulate enough data* before it can start transmission, and the latter denotes the time for *the destination to receive enough symbols* before decoding becomes possible. Other sources of delay, such as encoder/decoder processing time and propagation delay, are mostly determined by hardware and environmental conditions and are beyond the scope of this work.

Streaming codes are a promising scheme for low latency communications [2]. Their encoder receives a string of source symbols sequentially and outputs a string of coded symbols in real time. While streaming codes eliminate the queueing delay of traditional block-code schemes, it is not clear whether the "encoding-on-the-fly" architecture offers the same level of error protection as that of the traditional block codes that "accumulate[1] and then encode."

A fair comparison between these two types of codes is highly non-trivial due to their fundamentally different constructions. Specifically, in block coding, a single parameter "block length" simultaneously controls: (a) the encoding/decoding complexity; (b) how much data the source must accumulate before start encoding, hence the queueing delay; and (c) how much time it takes to finish transmission of the entire block, hence the transmission delay. In contrast, the complexity of a streaming code is controlled by its memory length, denoted by α. There is no queueing delay at all since it encodes on the fly. Furthermore, there is no such concept of "finishing transmission of a block of symbols" in streaming codes. Instead, streaming codes have the notion of *decoding deadline* Δ such that the destination must decode the time-t message by time $t+\Delta$. For example, with the same streaming encoder at the source, an aggressive destination may choose a small Δ that minimizes the transmission delay, while a relaxed destination may use a large Δ that incurs longer transmission delay but could further reduce the error probability.

This work studies *random linear streaming codes* (RLSCs) over i.i.d. symbol erasure channels, and derives the exact average error probability with arbitrary finite memory length α and finite decoding deadline Δ constraints. The result is then used to examine the tradeoff among memory length (complexity), decoding deadline (delay), and error probability (reliability) of RLSCs. When paired with the finite-length block code analysis [3], our results show that the E2E delay of the RLSC is roughly 50% of that of the MDS block code when both are under identical code rate and error probability requirements. This implies that the encoding-on-the-fly architecture eliminates the queueing delay completely (thus 50% delay reduction) without sacrificing reliability when compared to the block codes.

Our results also establish that *too much memory can adversely impact the performance of RLSCs when there exists a finite decoding deadline constraint*, a phenomenon that may

This work was supported in parts by NSF under Grants CCF-1422997, CCF-1618475, CCF-1816013 and CCF-2008527; and by MOST Taiwan under Grants 110-2636-E-009-016 and 107-2628-E-011-003-MY3.

[1] Intuitively, block codes spread all information bits to *all coded symbols within a block*, which maximizes the error protection of each bit. In contrast, streaming codes spread the current information bits *only to the future coded symbols*. The "extent of spreading" is also limited by its memory length. Due to these differences, whether streaming codes offer the same level of error protection is unknown until our results resolve this question affirmatively.

seem counter-intuitive, given the traditional wisdom that large memory length monotonically improves the performance of convolutional codes in the asymptotic regime.

A. Comparison to existing results

There are numerous existing results studying the *error exponents* of streaming codes [4], [5]. While the error exponent analyses provide valuable insight to the asymptotic error decay rate, they are ill-suited to quantify the exact error probability with finite (α, Δ), see the discussion in [6]. A middle ground between the error exponent analysis and the arbitrary finite length analysis in this work is the *moderate deviation* regime, which is studied by [7] under an infinite memory setting.

The closest existing result is [6], which derived the exact error probability of RLSCs under the finite memory $\alpha < \infty$ but infinite deadline $\Delta \to \infty$ setting. While the settings look similar, the generalization from infinite to finite Δ in this work is highly non-trivial since one has to quantify the *joint impact* of (α, Δ). The finiteness of either of them alone would have significantly affected the performance. To address the associated challenges, this work characterizes the *earliest decoding time* (EDT) of RLSCs, a new latency-centric concept that is neither defined nor explored in [6] since [6] focuses exclusively on the no-deadline setting $\Delta \to \infty$. By explicitly considering finite Δ, our result is later used to quantify the tradeoff among the tuple (complexity, delay, reliability), a fundamental contribution that is not viable in [6].

Streaming codes have also been widely studied under the *adversarial channel models* [8]–[18], a sharp departure from the i.i.d. channel model in this work. The goal therein is to design a streaming code with maximal code rate while guaranteeing *error-free* decoding under any erasure pattern selected from a predefined subset. See [6], [17], [18] for further discussion. The works in [19]–[21] also consider the error-free setting, but focus on a new metric of *in-order* delay under i.i.d. channels. In contrast, this paper studies squarely the classical error probability metric/model [22].

II. System Model

Notations: The boldface lower/upper letters denote column vectors/matrices, respectively, e.g., $\mathbf{s}(t)$ denotes a column vector indexed by t. We use $\mathbf{s}_a^b$ to represent the *cumulative* column vector $\mathbf{s}_a^b \triangleq \left[\mathbf{s}^\top(a), \mathbf{s}^\top(a+1), \ldots, \mathbf{s}^\top(b)\right]^\top$. $(\cdot)^+ \triangleq \max(0, \cdot)$ is the projection operator; $\mathbf{I}_n$ is the $n \times n$ identity matrix; $\vec{\delta}_k$ is a column vector for which the k-th entry is one and all other entries are zero; $\vec{\mathbf{1}}$ is the column vector of all 1s.

Encoder: In every time slot $t \geq 1$, the encoder receives K source symbols $\mathbf{s}(t) = [s_1(t), s_2(t), \ldots, s_K(t)]^\top$ where each symbol $s_k(t)$ is drawn independently and uniformly randomly from GF(2^q). The encoder also stores the $\alpha \cdot K$ symbols in the previous α slots $\{\mathbf{s}(\tau) : \tau \in [t-\alpha, t)\}$, where $\alpha < \infty$ is the *memory length.* Jointly, it uses the $(\alpha+1)K$ symbols as input and outputs N coded GF(2^q) symbols $\mathbf{x}(t) = [x_1(t), \ldots, x_N(t)]^\top$, see Fig. 1. Define $\mathbf{G}_t$ as the N-by-$(\min(\alpha+1, t) \cdot K)$ *generator matrix* for slot t, i.e., $\mathbf{x}(t) = \mathbf{G}_t \mathbf{s}_{\max(t-\alpha,1)}^t$. (1)

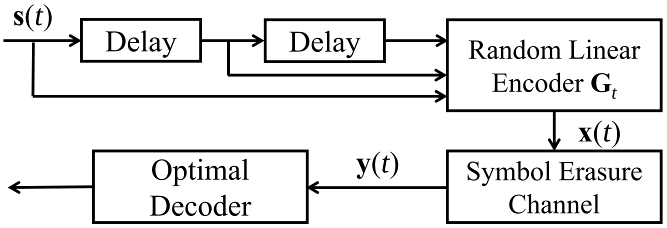

Fig. 1: The block diagram of the RLSCs with $\alpha = 2$.

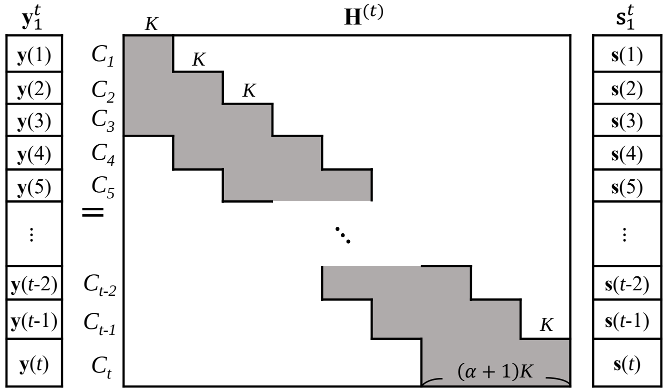

Fig. 2: The illustration of $\mathbf{H}^{(t)}$ in (3) with $\alpha = 2$.

Symbol Erasure Channel: In each time slot t, the source transmits all N symbols in $\mathbf{x}(t)$. A random subset of them, denoted by $\mathcal{C}_t \subseteq \{1, 2, \cdots, N\}$, will arrive at the destination perfectly and the complement of which is "erased" completely. $\mathcal{C}_t$ is i.i.d. across t. We define $C_t \triangleq |\mathcal{C}_t|$ as the number of successfully received symbols and define $P_i \triangleq \Pr(C_t = i)$.

Received Symbols: The C_t received symbols at time t are denoted by $\mathbf{y}(t) = [y_1(t), \ldots, y_{C_t}(t)]^\top$. We write

$$\mathbf{y}(t) = \mathbf{H}_t \mathbf{s}_{\max(t-\alpha,1)}^t \tag{2}$$

where $\mathbf{H}_t$ is the projection of $\mathbf{G}_t$ onto the random (row index) set $\mathcal{C}_t$. The following notations of the *cumulative generator and receiver matrices* turn out to be very useful:

$$\mathbf{x}_1^t = \mathbf{G}^{(t)} \mathbf{s}_1^t \qquad \text{and} \qquad \mathbf{y}_1^t = \mathbf{H}^{(t)} \mathbf{s}_1^t \tag{3}$$

where we properly shift and stack the instantaneous matrices $\mathbf{G}_t$ and $\mathbf{H}_t$ to create their cumulative representation $\mathbf{G}^{(t)}$ and $\mathbf{H}^{(t)}$, respectively. See Fig. 2 for illustration.

Decodability: Since the destination can use the observation $\mathbf{y}_1^{t+\Delta}$ and the knowledge of $\mathbf{H}^{(t+\Delta)}$ to decode $\mathbf{s}(t)$, we have

Definition 1. *A symbol $s_k(t)$ is decodable by time $t+\Delta$ if the transposed vector $\vec{\delta}_{(t-1)K+k}^\top$ is in the row space of $\mathbf{H}^{(t+\Delta)}$, where $\vec{\delta}_{(t-1)K+k}$ is the* location vector *of $s_k(t)$ at time $t+\Delta$ such that its $((t-1)K+k)$-th entry is one and all other $(t+\Delta)K-1$ entries are zero.*

Definition 2. *A vector $\mathbf{s}(t)$ is decodable by time $t+\Delta$ if all $\{s_k(t) : k \in [1, K]\}$ are decodable by time $t+\Delta$.*

Definition 3. *The* earliest decoding time (EDT) *of $\mathbf{s}(t)$ is*

$$\mathsf{EDT}(\mathbf{s}(t)) \triangleq \inf\{\tau \geq t : \mathbf{s}(t) \textit{ is decodable by time } \tau\}. \tag{4}$$

Using the convention $\inf\emptyset = \infty$*, "*$\mathsf{EDT}(\mathbf{s}(t)) = \infty$*" implies* $\mathbf{s}(t)$ *is not decodable by time* τ *regardless how large* τ *is.*

Our goal is to compute the *slot error probability* p_e:

$$p_e \triangleq \lim_{T\to\infty} \frac{1}{T}\sum_{t=1}^{T} \Pr\left(\mathbf{s}(t) \text{ is not decodable by time } t+\Delta\right). \tag{5}$$

To avoid some corner cases, our analysis assumes:

The Less-than-Capacity (LC) condition: The code rate is strictly less than the capacity, i.e., $0 < \frac{K}{N} < \frac{\mathbb{E}\{C_t\}}{N}$.

Random linear streaming codes (RLSCs): Each entry of $\mathbf{G}_t$ is chosen uniformly and randomly from GF(2^q), excluding 0. In this work, instead of quantifying the probabilistic behavior of RLSCs, we simply assume that the resulting encoder satisfies the following condition.

The Generalized MDS Condition (GMDS): For any t and any finite sequence of pairs $\{(i_l, j_l) : l \in [1, L]\}$ satisfying (a) $i_{l_1} \neq i_{l_2}$ and $j_{l_1} \neq j_{l_2}$ for any $l_1 \neq l_2$ and (b) the (i_l, j_l)-th entry of $\mathbf{G}^{(t)}$ is non-zero for all $l \in [1, L]$, define the corresponding row and column index sets $S_R \triangleq \{i_l : l \in [1, L]\}$ and $S_C \triangleq \{j_l : l \in [1, L]\}$. The GMDS condition requires that the submatrix of the cumulative generator matrix $\mathbf{G}^{(t)}$ induced by S_R and S_C is always invertible.

Remark 1: If the transmission only lasts for a bounded duration, the probability of RLSCs satisfying **GMDS** approaches one when $q \to \infty$, i.e., our result focuses on the typical behavior of RLSCs when $q \to \infty$. Also see the Schwartz-Zippel theorem in [23, Theorems 3 and 4]. Further research is needed to explicitly quantify the performance gap between finite GF(2^q) and $q \to \infty$, though most existing results [23] show that the gap decreases at rate $O(2^{-q})$.

III. Main Results

A. Characterizing the EDT for Finite (α, Δ)

We reuse the following definitions, first introduced in [6], to describe our new EDT characterization results.

Definition 4. *Define a constant* $\zeta \triangleq \alpha K + 1$ *and initialize* $I_d(0) \triangleq 0$*. For any* $t \geq 1$*, we iteratively compute the information debt* $I_d(t)$ *at time* t *by*

$$\hat{I}_d(t) \triangleq \left(K - C_t + \min\left(I_d(t-1), \alpha K\right)\right)^+ \tag{6}$$

$$I_d(t) \triangleq \min\left(\zeta,\ \hat{I}_d(t)\right). \tag{7}$$

Definition 5. *Define* $t_0 \triangleq 0$ *and* $\tau_0 \triangleq 0$*, and define iteratively*

$$t_i \triangleq \inf\{t' : t' > t_{i-1}, I_d(t') = 0\} \tag{8}$$

$$\tau_i \triangleq \inf\{t' : t' > \tau_{i-1}, I_d(t') = \zeta\} \tag{9}$$

as the i*-th time that* $I_d(t)$ *hits* 0 *and* ζ*, respectively.*

Proposition 1. *Assume* **GMDS***. For any* $i_0 \geq 0$*, consider two cases. Case 1: there exists no* $\tau_j \in (t_{i_0}, t_{i_0+1})$*. In this case,*

$$\mathsf{EDT}(\mathbf{s}(t)) = t_{i_0+1},\ \forall t \in (t_{i_0}, t_{i_0+1}]. \tag{10}$$

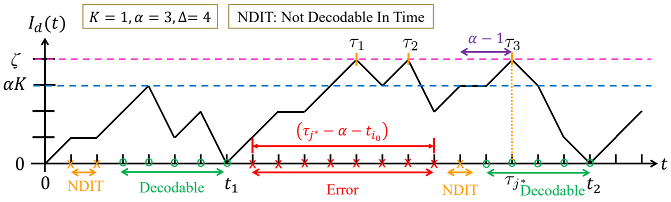

Fig. 3: Error-event characterization for $\Delta < \infty$.

Case 2: there exists a $\tau_j \in (t_{i_0}, t_{i_0+1})$*. In this case, define* τ_{j^*} *as the one with the largest* j*. We have*

$$\mathsf{EDT}(\mathbf{s}(t)) = \begin{cases} t_{i_0+1}, & \forall t \in (\tau_{j^*} - \alpha, t_{i_0+1}] \quad (11) \\ \infty, & \forall t \in (t_{i_0}, \tau_{j^*} - \alpha]. \quad (12) \end{cases}$$

It is worth noting that characterizing the EDT of $\mathbf{s}(t)$ requires not only proving that $\mathbf{s}(t)$ can be decoded by a certain time τ but also proving that $\mathbf{s}(t)$ cannot be decoded by time $\tau - 1$. Take the instance of $i_0 = 0$ for example. In Case 1, it is not hard to prove that when $I_d(t)$ hits 0 again at time t_1, we have observed enough linear equations, i.e., receiving many large C_t in (6), and can thus start decoding from $\mathbf{s}(t_1), \mathbf{s}(t_1 - 1), \cdots, \mathbf{s}(1)$ in a backward fashion. Algebraically, the cumulative receiver matrix $\mathbf{H}^{(t_1)}$ is of full row rank and thus all $\mathbf{s}(t)$ are decodable according to Definitions 1 and 2. However, for the converse, simply proving "$\mathbf{H}^{(t')}$ is not of full row rank when $t' < t_1$" is not enough since it only shows that for *some* $s_k(t)$, its location vector is not in the row space of $\mathbf{H}^{(t')}$. Instead, one must prove that *every one of the* $\mathbf{s}(t)$ *considered in* (10) *is not decodable when examining the location vectors using Definitions 1 and 2.* Each of the EDT statements (10) and (11), is essentially a combination of matched achievability and converse results in the time domain, with the converse proofs being much more involved than a full-rank argument. We omit the proofs due to the space limit.

B. Exact Error Probability Analysis

For any $i_0 \geq 0$, if there exists no $\tau_j \in (t_{i_0}, t_{i_0+1})$, we name the interval $(t_{i_0}, t_{i_0+1}]$ a *good round*; and if there exists a $\tau_j \in (t_{i_0}, t_{i_0+1})$, we name $(t_{i_0}, t_{i_0+1}]$ a *bad round*. By Proposition 1, the slots in a good round can be labeled as:

$$\begin{cases} \mathbf{s}(t) \text{ is decodable by } t + \Delta \\ \qquad \text{if } t \in [\max(t_{i_0} + 1, t_{i_0+1} - \Delta), t_{i_0+1}] \\ \mathbf{s}(t) \text{ is Not Decodable In Time (NDIT)} \\ \qquad \text{if } t \in (t_{i_0}, t_{i_0+1} - \Delta). \end{cases} \tag{13}$$

By (11) and (12), the slots in a bad round can be labeled as:

$$\begin{cases} \mathbf{s}(t) \text{ is decodable by } t + \Delta \\ \qquad \text{if } t \in [\max(\tau_{j^*} - \alpha + 1, t_{i_0+1} - \Delta), t_{i_0+1}] \\ \mathbf{s}(t) \text{ is NDIT} \qquad \text{if } t \in (\tau_{j^*} - \alpha, t_{i_0+1} - \Delta) \\ \mathbf{s}(t) \text{ is in error} \qquad \text{if } t \in (t_{i_0}, \tau_{j^*} - \alpha]. \end{cases} \tag{14}$$

Fig. 3 illustrates the above error-event characterization.

Since p_e in (5) involves a deadline constraint Δ, it is contributed by both the NDIT and the in-error slots in (13) and (14). Noting that $I_d(t)$ is a Markov chain, we have:

Lemma 1. *Assuming the* **LC** *and* **GMDS** *conditions,*

$$p_e = \frac{\mathbb{E}\left\{L_G + L_B\right\}}{\mathbb{E}\left\{t_{i_0+1} - t_{i_0}\right\}} \tag{15}$$

where $i_0 \geq 0$ is any arbitrary but fixed index,

$$L_G \triangleq \mathbb{1}_{\{no\ \tau_j \in (t_{i_0}, t_{i_0+1})\}} \cdot (t_{i_0+1} - \Delta - 1 - t_{i_0})^+ \tag{16}$$

$$L_B \triangleq \mathbb{1}_{\{\exists \tau_j \in (t_{i_0}, t_{i_0+1})\}} \cdot (\max(\tau_{j^*} - \alpha,\ t_{i_0+1} - \Delta - 1) - t_{i_0})$$

and $\mathbb{1}_{\{\cdot\}}$ is the indicator function. The subscripts G and B denote a good and a bad round, respectively.

To continue, we further rewrite $\mathbb{E}\left\{L_B\right\}$ as:

$$\mathbb{E}\left\{L_B\right\} = \mathbb{E}\left\{L_{B_1}\right\} + \mathbb{E}\left\{L_{B_2}\right\} \tag{17}$$

where $L_{B_1} \triangleq \mathbb{1}_{\{\exists \tau_j \in (t_{i_0}, t_{i_0+1})\}} \cdot (\tau_{j^*} - t_{i_0})$ (18)

$$L_{B_2} \triangleq$$

$$\mathbb{1}_{\{\exists \tau_j \in (t_{i_0}, t_{i_0+1})\}} \cdot \left(\max\left(-\alpha,\ t_{i_0+1} - \Delta - 1 - \tau_{j^*}\right)\right). \tag{19}$$

We now describe the ingredients needed when evaluating (15). We first note that $I_d(t)$ is a Markov chain with the state space $\{0, 1, \cdots, \zeta\}$. We denote its $(\zeta+1)$-by-$(\zeta+1)$ transition matrix as $\Gamma = [\gamma_{i,j}]$ where $\gamma_{i,j} \triangleq \Pr\left(I_d(t) = j \mid I_d(t-1) = i\right)$. Γ can be explicitly written from the distribution of the channel C_t and the iterative formulas (6) and (7). We then define $\boldsymbol{\phi} \triangleq \{1, 2, \ldots, \zeta - 1\}$ as the collection of non-boundary states and partition Γ into 9 sub-matrices:

$$\Gamma = \begin{bmatrix} \Gamma_{0,0} & \Gamma_{0,\boldsymbol{\phi}} & \Gamma_{0,\zeta} \\ \Gamma_{\boldsymbol{\phi},0} & \Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}} & \Gamma_{\boldsymbol{\phi},\zeta} \\ \Gamma_{\zeta,0} & \Gamma_{\zeta,\boldsymbol{\phi}} & \Gamma_{\zeta,\zeta} \end{bmatrix} \tag{20}$$

where $\Gamma_{\mathbf{x},\mathbf{y}} \triangleq [\gamma_{i,j}], \forall i \in \mathbf{x}$ and $j \in \mathbf{y}$. Additionally, we denote $\mathbf{A} \triangleq (\mathbf{I}_{\zeta-1} - \Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}})^{-1}$ and define two $(\zeta+1)$-by-$(\zeta+1)$ matrices $\mathbf{M}_1$ and $\mathbf{M}_2$ that hardwire parts of Γ to zeros:

$$\mathbf{M}_1 \triangleq \begin{bmatrix} 0 & \Gamma_{0,\boldsymbol{\phi}} & \Gamma_{0,\zeta} \\ 0 & \Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}} & \Gamma_{\boldsymbol{\phi},\zeta} \\ 0 & \Gamma_{\zeta,\boldsymbol{\phi}} & \Gamma_{\zeta,\zeta} \end{bmatrix}, \mathbf{M}_2 \triangleq \begin{bmatrix} 0 & \Gamma_{0,\boldsymbol{\phi}} & \Gamma_{0,\zeta} \\ 0 & \Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}} & \Gamma_{\boldsymbol{\phi},\zeta} \\ 0 & 0 & 0 \end{bmatrix}.$$

Since $\mathbb{E}\left\{t_{i_0+1} - t_{i_0}\right\}$, see (8), and $\mathbb{E}\left\{L_{B_1}\right\}$ in (18) do not involve Δ, we can reuse the results in [6]. Hence, we have

Lemma 2.

$$\mathbb{E}\left\{t_{i_0+1} - t_{i_0}\right\} = \vec{\delta}_1^{\top}(\mathbf{I}_{\zeta+1} - \mathbf{M}_1)^{-1}\vec{\mathbf{1}} \tag{21}$$

$$\begin{aligned}\mathbb{E}\left\{L_{B_1}\right\} = {} & (\Gamma_{0,\zeta} + \Gamma_{0,\boldsymbol{\phi}}\mathbf{A}\Gamma_{\boldsymbol{\phi},\zeta})\left(\frac{1 + \Gamma_{\zeta,\boldsymbol{\phi}}\left(\mathbf{A}\right)^2\Gamma_{\boldsymbol{\phi},\zeta}}{1 - \Gamma_{\zeta,\zeta} - \Gamma_{\zeta,\boldsymbol{\phi}}\mathbf{A}\Gamma_{\boldsymbol{\phi},\zeta}}\right) \\ & + \Gamma_{0,\boldsymbol{\phi}}\left(\mathbf{A}\right)^2\Gamma_{\boldsymbol{\phi},\zeta}.\end{aligned} \tag{22}$$

In addition to the new EDT characterization theorem in Proposition 1, another critical innovation versus [6] is the following formulas that compute $\mathbb{E}\left\{L_G\right\}$ and $\mathbb{E}\left\{L_{B_2}\right\}$.

Lemma 3. *Define $\psi \triangleq (\Delta - \alpha - 1)^+$.*

$$\mathbb{E}\left\{L_G\right\} = \sum_{k=2}^{\infty}(k - \Delta - 1)^+\Gamma_{0,\boldsymbol{\phi}}\left(\Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}}\right)^{k-2}\Gamma_{\boldsymbol{\phi},0} \tag{23}$$

$$= \Gamma_{0,\boldsymbol{\phi}}\left(\mathbf{A}\right)^2\left(\Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}}\right)^{\Delta}\Gamma_{\boldsymbol{\phi},0} \tag{24}$$

$$\begin{aligned}\mathbb{E}\left\{L_{B_2}\right\} = {} & \vec{\delta}_1^{\top}(\mathbf{I}_{\zeta+1} - \mathbf{M}_2)^{-1}\vec{\delta}_{(\zeta+1)} \cdot \left(\Gamma_{\zeta,0} + \Gamma_{\zeta,\boldsymbol{\phi}}\mathbf{A}\Gamma_{\boldsymbol{\phi},0}\right)^{-1} \\ & \cdot \Bigg(- \min\left(\Delta, \alpha\right)\Gamma_{\zeta,0} - \alpha\Gamma_{\zeta,\boldsymbol{\phi}}\mathbf{A}\Gamma_{\boldsymbol{\phi},0} \\ & + \Gamma_{\zeta,\boldsymbol{\phi}}\left(\left(\mathbf{A}\right)^2 + \left(\alpha + \psi - \Delta\right)\mathbf{A}\right)\left(\Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}}\right)^{\psi}\Gamma_{\boldsymbol{\phi},0}\Bigg).\end{aligned} \tag{25}$$

Eq. (23) follows by directly rewriting the expectation of (16) as a summation where the index $k = t_{i_0+1} - t_{i_0}$ and by noticing that $(k - \Delta - 1)^+ = 0$ when $k = 1$. Eq. (24) follows from (23) by simple algebra. Eq. (25) is the most involved since τ_{j^*}, the *last* hitting time within the interval (t_{i_0}, t_{i_0+1}), is not a stopping time and (19) admits a complicated expression. To derive (25), we first define the hitting time $\Lambda_t(x)$ of x and five associated terms as follows.

$$\Lambda_t(x) \triangleq \inf\{\tau > 0 : I_d(t + \tau) = x\} \tag{26}$$

$$\mathsf{term}_1 \triangleq \Pr(\Lambda_t(0) > \Lambda_t(\zeta) \mid I_d(t) = 0) \tag{27}$$

$$\begin{aligned}\mathsf{term}_2 \triangleq {} & \mathbb{E}\{\mathbb{1}_{\{\Lambda_t(0) < \Lambda_t(\zeta)\}} \cdot \max(-\alpha, \Lambda_t(0) - \Delta - 1) \\ & \mid I_d(t) = \zeta\}\end{aligned} \tag{28}$$

$$\mathsf{term}_3 \triangleq \Pr(\Lambda_t(0) < \Lambda_t(\zeta) \mid I_d(t) = \zeta) \tag{29}$$

$$\begin{aligned}\mathsf{term}_4 \triangleq {} & \mathbb{E}\{\max(-\alpha, \Lambda_t(0) - \Delta - 1) \\ & \mid I_d(t) = \zeta, \mathbb{1}_{\{\Lambda_t(0) < \Lambda_t(\zeta)\}} = 1\}\end{aligned} \tag{30}$$

$$\begin{aligned}\mathsf{term}_5 \triangleq {} & \mathbb{E}\{\max(-\alpha, t_{i_0+1} - \tau_{j^*} - \Delta - 1) \\ & \mid \exists \tau_j \in (t_{i_0}, t_{i_0+1})\}\end{aligned} \tag{31}$$

We note that the conditional event in (30) means that t *is the last time $I_d(\cdot)$ hits ζ before hitting 0,* which is equivalent to t being the last ζ-hitting time $\tau_{j^*} \in (t_{i_0}, t_{i_0+1})$ for some i_0. Furthermore, it also implies $t = \tau_{j^*}$ and $t_{i_0+1} = \tau_{j^*} + \Lambda_t(0)$. Jointly, we thus have $\mathsf{term}_4 = \mathsf{term}_5$. More rigorous arguments are omitted due to the space limit.

By basic probability computation, we have $\mathsf{term}_4 = \mathsf{term}_2/\mathsf{term}_3$. Because $I_d(t)$ is strong Markov, we also have

$$\begin{aligned}\mathsf{term}_1 & = \Pr(\Lambda_{t_{i_0}}(0) > \Lambda_{t_{i_0}}(\zeta) \mid I_d(t_{i_0}) = 0) \\ & = \Pr(\exists \tau_j \in (t_{i_0}, t_{i_0+1})).\end{aligned} \tag{32}$$

By (19) we have $\mathsf{term}_5 = \mathbb{E}\left\{L_{B_2}\right\}/\mathsf{term}_1$. Jointly we have

$$\begin{aligned}\mathbb{E}\left\{L_{B_2}\right\} & = \mathsf{term}_1 \cdot \mathsf{term}_5 = \mathsf{term}_1 \cdot \mathsf{term}_4 \\ & = \mathsf{term}_1 \cdot \mathsf{term}_2/\mathsf{term}_3.\end{aligned} \tag{33}$$

All the values of term_1 to term_3 can be easily computed via standard Markov chain analysis [24]. Putting them together in (33) gives us (25).

Proposition 2. *For any finite (α, Δ), the slot error probability p_e can be computed by assembling Lemmas 1 to 3.*

IV. Numerical Evaluation

A. Error Probability versus Memory Length Tradeoff

We choose $N = 8$, $K = 5$ and C_t being a binomial distribution with $p = \frac{K}{N} + 0.01 = 0.635$, i.e., $P_i = \binom{8}{i}p^i(1-p)^{8-i}$. Fig. 4 compares the error probability for different α and Δ values. The "Simulation" is plotted by running the random process $I_d(t)$ from $t = 1$ to 10^8, counting the number of

NDIT and in-error slots using (13) and (14), and dividing by 10^8 to calculate the empirical probability. The other curves are obtained by Proposition 2. As expected, our exact error probability computation matches the simulation curve. We deliberately choose the (N, K, p) values inducing $p_e \approx 10^{-1}$ so that the simulation can provide extremely accurate ground truth. This exact match between simulation and our analytical results also holds for other experiments with much smaller p_e.

We observe that if we impose a finite Δ, p_e no longer improves monotonically with α. When $\Delta = 10$ and 150, the best α are 1 and 5, respectively. When $\Delta = 500$, we wrote the exact p_e values in Fig. 4 for $\alpha = 10, 15, 20$ and 25, respectively. The best p_e is 0.0688 when $\alpha = 15$. In all our evaluations, including others that are not shown, too large memory length always makes the p_e *strictly worse*, sometimes by a large degree (see $\Delta = 10$ and 150) and sometimes just slightly (see $\Delta = 500$). The intuition is that larger α means that *information is spread over a longer horizon*, which makes it harder to decode a single $\mathbf{s}(t)$ before the deadline $t+\Delta$ since *we now have too many other source symbols* $\mathbf{s}(t')$, $t' \neq t$, *that are fully mixed within the interval* $[t, t+\Delta]$. A practical implication is thus to avoid choosing unnecessarily large memory length α, which is both of higher complexity and also of poorer performance.

B. Code Rate versus Delay Tradeoff

In this subsection, we fix $p_e = 0.001$ and plot the code rate versus delay tradeoff, an important setup considered in [3]. We consider a packet erasure channel with erasure probability $\delta = 0.5$ and assume that each packet has 100 coded symbols.

When cast under the framework in Section II (i.e., define the duration of a slot to be the time it takes to transmit a packet of 100 symbols), we choose $N = 100$ and $C_t = 0$ and 100 with probability 0.5 and 0.5, respectively. We assume a new message of K symbols arrives for each *slot* and the RLSC encoder immediately turns them into a packet of $N = 100$ coded symbols that will be transmitted in the current time slot. For each $\Delta \in [1, 500]$, we find the largest $K^*(\Delta)$ such that the p_e computed by Proposition 2 is still ≤ 0.001. The code rate is then defined as $R(\Delta) = \frac{K^*(\Delta)}{N}$, shown in Fig. 5.

We now describe how traditional MDS block codes handle this sequential arrival setting. The MDS encoder first queues n_B messages to collect a total of $n_B \cdot K$ message symbols, which leads to a queueing delay of $n_B - 1$ slots since we assume that each new message $\mathbf{s}(t)$ arrives in the beginning of the slot. Those message symbols are then encoded into $100 \cdot n_B$ coded symbols and are sent in the next n_B slots. The transmission delay is thus n_B slots. The code rate of MDS codes is $R_{\text{MDS}}(n_B) = \frac{n_B \cdot K}{n_B \cdot N} = \frac{K}{100}$, and the E2E delay is $\Delta_{\text{MDS}}(n_B) = 2n_B - 1$. For every n_B value, we find the largest K that satisfies the error probability

$$p_{e,\text{MDS}} = \Pr\left(n_B \cdot K > \textstyle\sum_{t=1}^{n_B} C_t\right) \leq 0.001 \qquad (34)$$

and plot the curve $(\Delta_{\text{MDS}}(n_B), R_{\text{MDS}}(n_B))$ in Fig. 5 by varying $n_B \in [1, 250]$. (The zigzagging behavior is quite common in finite-length analysis [3, Fig. 5].)

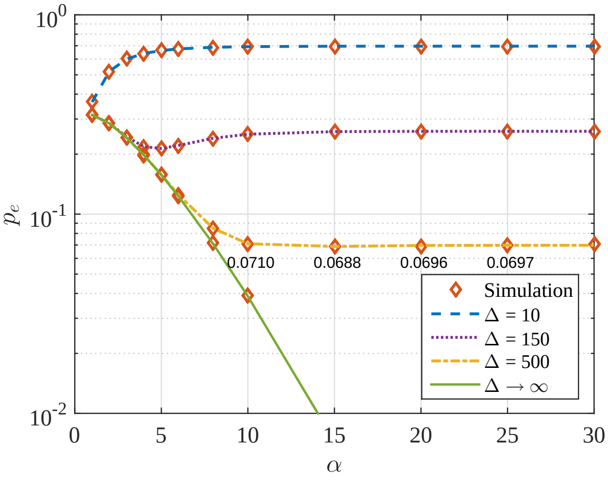

Fig. 4: p_e vs α when $N = 8$, $K = 5$ and $C_t \sim B(8, 0.635)$.

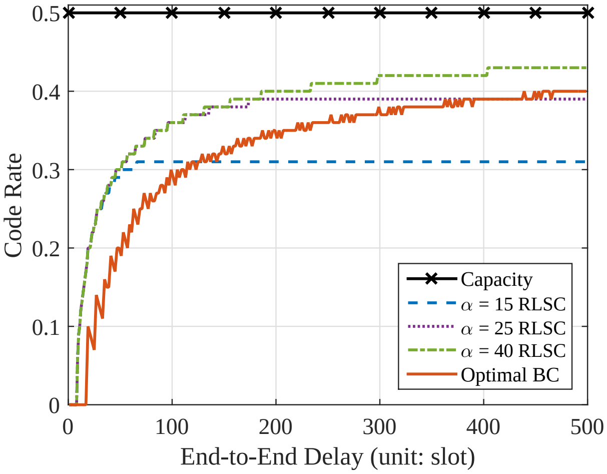

Fig. 5: Rate-delay tradeoff with packet erasure probability $\delta = 0.5$ and maximal error probability $p_e = 10^{-3}$.

Fig. 5 compares the rate-delay tradeoff between RLSCs and MDS codes. The results show that under the same code rate and the same p_e requirements, the E2E delay of RLSCs, assuming the best α is used, is $\approx 50\%$ of that of the MDS block codes. The gain mainly follows from eliminating the queueing delay completely (thus 50%). Since this comparison imposes the same p_e value, it establishes that the encoding-on-the-fly structure of RLSCs has little negative impact on p_e when compared to the accumulate-&-then-encode structure of MDS codes.

V. Conclusion

We have derived the closed-form slot error probability of random linear streaming codes (RLSCs) over i.i.d. symbol erasure channels in the finite memory length and finite decoding deadline regime, and demonstrated the superior performance of RLSCs over block codes under a fair comparison of the tuple of (code rate, delay, error probability), the first of such results in the literature.

References

[1] M. Series, “IMT Vision–Framework and overall objectives of the future development of IMT for 2020 and beyond,” *Recommendation ITU*, vol. 2083, Sep. 2015.

[2] M. N. Krishnan, V. Ramkumar, M. Vajha, and P. V. Kumar, “Simple streaming codes for reliable, low-latency communication,” *IEEE Communications Letters*, vol. 24, no. 2, pp. 249–253, 2020.

[3] Y. Polyanskiy, H. V. Poor, and S. Verdu, “Channel coding rate in the finite blocklength regime,” *IEEE Transactions on Information Theory*, vol. 56, no. 5, pp. 2307–2359, May 2010.

[4] A. Viterbi, “Error bounds for convolutional codes and an asymptotically optimum decoding algorithm,” *IEEE Transactions on Information Theory*, vol. 13, no. 2, pp. 260–269, Apr. 1967.

[5] S. C. Draper and A. Khisti, “Truncated tree codes for streaming data: Infinite-memory reliability using finite memory,” in *2011 8th International Symposium on Wireless Communication Systems*, Nov. 2011, pp. 136–140.

[6] P. W. Su, Y. C. Huang, S. C. Lin, I. H. Wang, and C. C. Wang, “Error rate analysis for random linear streaming codes in the finite memory length regime,” in *2020 IEEE International Symposium on Information Theory (ISIT)*, 2020, pp. 491–496.

[7] S.-H. Lee, V. Y. F. Tan, and A. Khisti, “Exact moderate deviation asymptotics in streaming data transmission,” *IEEE Transactions on Information Theory*, vol. 63, no. 5, pp. 2726–2736, 2017.

[8] E. Martinian and C.-E. W. Sundberg, “Burst erasure correction codes with low decoding delay,” *IEEE Transactions on Information Theory*, vol. 50, no. 10, pp. 2494–2502, Oct. 2004.

[9] E. Martinian and M. Trott, “Delay-optimal burst erasure code construction,” in *2007 IEEE International Symposium on Information Theory*, 2007, pp. 1006–1010.

[10] A. Khisti and J. P. Singh, “On multicasting with streaming burst-erasure codes,” in *2009 IEEE International Symposium on Information Theory*, Jun. 2009, pp. 2887–2891.

[11] A. Badr, A. Khisti, and E. Martinian, “Diversity embedded streaming erasure codes (DE-SCo): Constructions and optimality,” *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 5, pp. 1042–1054, May 2011.

[12] A. Badr, A. Khisti, W. Tan, and J. Apostolopoulos, “Streaming codes with partial recovery over channels with burst and isolated erasures,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, no. 3, pp. 501–516, Apr. 2015.

[13] A. Badr, P. Patil, A. Khisti, W. Tan, and J. Apostolopoulos, “Layered constructions for low-delay streaming codes,” *IEEE Transactions on Information Theory*, vol. 63, no. 1, pp. 111–141, Jan. 2017.

[14] S. L. Fong, A. Khisti, B. Li, W. Tan, X. Zhu, and J. Apostolopoulos, “Optimal streaming codes for channels with burst and arbitrary erasures,” *IEEE Transactions on Information Theory*, vol. 65, no. 7, pp. 4274–4292, Jul. 2019.

[15] M. Rudow and K. V. Rashmi, “Streaming codes for variable-size arrivals,” in *2018 56th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, Oct. 2018, pp. 733–740.

[16] ——, “Online versus offline rate in streaming codes for variable-size messages,” in *2020 IEEE International Symposium on Information Theory (ISIT)*, 2020, pp. 509–514.

[17] D. Dudzicz, S. L. Fong, and A. Khisti, “An explicit construction of optimal streaming codes for channels with burst and arbitrary erasures,” *IEEE Transactions on Communications*, vol. 68, no. 1, pp. 12–25, 2020.

[18] M. N. Krishnan, D. Shukla, and P. V. Kumar, “Rate-optimal streaming codes for channels with burst and random erasures,” *IEEE Transactions on Information Theory*, vol. 66, no. 8, pp. 4869–4891, 2020.

[19] M. Karzand and D. J. Leith, “Low delay random linear coding over a stream,” in *2014 52nd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 2014, pp. 521–528.

[20] G. Joshi, Y. Kochman, and G. W. Wornell, “The effect of block-wise feedback on the throughput-delay trade-off in streaming,” in *2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2014, pp. 227–232.

[21] A. Cohen, D. Malak, V. B. Bracha, and M. Mdard, “Adaptive causal network coding with feedback,” *IEEE Transactions on Communications*, vol. 68, no. 7, pp. 4325–4341, 2020.

[22] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, 2nd ed. Wiley-Interscience, 2006.

[23] T. Ho, M. Medard, R. Koetter, D. R. Karger, M. Effros, J. Shi, and B. Leong, “A random linear network coding approach to multicast,” *IEEE Transactions on Information Theory*, vol. 52, no. 10, pp. 4413–4430, Oct. 2006.

[24] R. Durrett, *Essentials of Stochastic Processes*, 3rd ed. Springer, 2016.