Random Linear Streaming Codes in the Finite Memory Length and Decoding Deadline Regime — Part I: Exact Analysis

Pin-Wen Su, *Graduate Student Member, IEEE,* Yu-Chih Huang, *Member, IEEE,*
Shih-Chun Lin, *Senior Member, IEEE,* I-Hsiang Wang, *Member, IEEE,*
and Chih-Chun Wang, *Senior Member, IEEE,*

***Abstract*—*Streaming codes* take a string of source symbols as input and output a string of coded symbols in real time, which eliminate the queueing delay of traditional *block codes* and are thus especially appealing for delay sensitive applications. Existing works on streaming code performance either focused on the asymptotic error-exponent analyses, or on the optimal code construction under *deterministic adversarial channel models*. In contrast, this work analyzes the exact error probability of *random linear streaming codes* (RLSCs) in the large field size regime over the stochastic i.i.d. symbol erasure channel model. A closed-form expression of the error probability of large-field-size RLSCs is derived under, simultaneously, the finite memory length and decoding deadline constraints. The result is then used to examine the intricate tradeoff between memory length (complexity), decoding deadline (delay), code rate (throughput), and error probability (reliability). Numerical evaluation shows that under the same code rate and error probability requirements, the end-to-end delay of RLSCs is 40–48% of that of the optimal block codes (i.e., MDS codes). This implies that switching from block codes to streaming codes not only eliminates the queueing delay completely (which accounts for the initial 50% of the delay reduction) but also improves the reliability (which accounts for the additional 2–10% delay reduction).**

***Index Terms*—streaming codes, erasure channels, finite length analysis, reliability functions, random walk analysis**

I. Introduction

For the fifth generation (5G) mobile communication, the International Telecommunication Union (ITU) has classified its services into three categories [3]. Among these three categories, the design of the ultra-reliable and low latency communication (URLLC) may be the most challenging. It requires the end-to-end delay to be within 1 ms and, meanwhile, the reliability should be at least 0.99999 [4]. Such stringent criteria are hard to meet through the classical ARQ schemes, or even hybrid ARQ used in 4G LTE systems and it is thus critical to actively explore different technologies/architectures that have not been utilized in the existing 4G systems.

One such example is the concept of *streaming codes* that are fundamentally different from the block codes that have been used ubiquitously in wireless communications for the past decades. Specifically, streaming codes are a class of sequential coding for which the encoder receives a string of source packets sequentially and outputs a string of coded packets in real time. Streaming codes can thus be viewed as generalizing the basic encoding unit of the *convolutional codes* from "bit" to "packet" and synchronizing the operation of the shift registers in the encoder with the actual arrival, encoding, and transmission of the packets.[1] Recall that for classical block codes, the encoder has to wait until a certain number of messages is received before it can start encoding, and the waiting time is called the *queueing delay*. In contrast, by synchronizing the convolutional encoder with the sequential packet arrival, streaming codes will *start sending coded packets immediately after each packet arrival,* which thus eliminates the concept of queueing delay in block coding. Nonetheless, having shorter overall delay does not necessarily make streaming codes a superior choice since the low latency performance of a scheme can only be evaluated by jointly considering the tuple of throughput, delay, and error probability (reliability). The apparent delay advantage of streaming codes over block codes could be at the cost of worse reliability. To rigorously evaluate the low latency performance of streaming codes, it is thus critical to resolve the following conjecture:

> Conjecture: For the same code rate, the unique encoding/decoding architecture of streaming codes does not negatively impact the error probability when compared to the classical block codes.

Such a conjecture, if proved affirmatively, will make streaming codes an appealing candidate for the delay sensitive applications such as tele-/video conferencing, online gaming, live TV, and various other URLLC services in 5G [6]. This work answers this conjecture by first finding a closed-form error probability expression of random linear streaming

This work was supported by NSF under Grants CCF-1422997, CCF-1618475, CCF-1816013, CCF-2008527 and CNS-2107363; by MOST Taiwan under Grants 107-2628-E-011-003-MY3, 110-2636-E-009-016 and 110-2222-E-002-009; and by National Taiwan University under Grant NTU-CC-111L894402. Part of the results were presented at the 2020 and the 2021 IEEE International Symposium on Information Theory (ISIT) [1], [2].

Pin-Wen Su and Chih-Chun Wang are with the Elmore Family School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN 47907, USA (e-mail: su173@purdue.edu; chihw@purdue.edu).

Yu-Chih Huang is with the Institute of Communications Engineering, National Yang Ming Chiao Tung University, Hsinchu 300, Taiwan (e-mail: jerryhuang@nctu.edu.tw).

Shih-Chun Lin and I-Hsiang Wang are with the Department of Electrical Engineering and the Graduate Institute of Communication Engineering, National Taiwan University, Taipei 10617, Taiwan (e-mail: sclin2@ntu.edu.tw; ihwang@ntu.edu.tw).

[1] A more precise statement of the definition of streaming codes used in this work is provided in Section II. It is worth noting that other definitions of streaming codes can be found in [5].

codes (RLSCs) with finite memory length and deadline/delay constraints, assuming a sufficiently large finite field size is used. By numerically comparing the streaming code error probability derived in this work and the seminal finite length results of block codes in [7], we resolve this conjecture affirmatively for the memoryless symbol erasure channel.

A. Existing works based on adversarial channel models

There is a growing body of literature that aims to find the optimal streaming code rate and code construction. An early work was proposed by Martinian and Sundberg [8], which presented a new class of systematic streaming codes that can perfectly correct any bursty B-erasure within the predetermined decoding delay[2]. Since the performance metric/criterion is to design a streaming code with maximal code rate while guaranteeing *error-free* decoding under *any* burst erasure pattern, we term such a setting the *adversarial channel model*, which is different from the stochastic channel models (e.g., i.i.d. BSC, AWGNC, etc) commonly used in the physical-layer communication literature [9].

Some follow-up works on streaming codes over adversarial channel models can be found in [5], [10]–[17]. For example, the (adversarial) channel error patterns being considered were expanded from only bursty-erasure [10]–[12] to both bursty-erasure and isolated-erasures [5], [13]–[15]. Along these adversarial channel settings, both the converse results (the largest rate beyond which error-free decoding is impossible) and the matching achievability schemes have been derived. Specific contributions include new bounding techniques and innovative capacity-achieving streaming code constructions. In [16] and [17], the authors considered variable-sized messages for the adversarial erasure channel and analyzed the largest code rate that can still admit error-free decoding.

B. Existing works based on stochastic channel models

In contrast with the existing adversarial settings, this work takes a probabilistic approach and aims to analyze the error probability of large-field-size RLSCs under the i.i.d. erasure channel model. Broadly speaking, we use a stochastic channel model and focus on the probabilistic average, while the existing works [5], [8], [10]–[17] focused on the worst-case performance over a predefined deterministic set of erasure patterns.

While most existing works are based on adversarial channels, a few of them are based on the same/similar stochastic channel model studied in this work. Specifically, [18] considered the following setting. For any fixed ratio of the memory length α over the decoding deadline Δ that satisfies $\beta \triangleq \frac{\alpha}{\Delta} \geq 1$, [18] studied the error exponent when α and Δ jointly go to infinity while β is fixed. A critical finding is that if β exceeds a certain threshold $\beta^* \geq 1$, the error exponent stops improving and a finite ratio $\frac{\alpha}{\Delta} = \beta^*$ is as good as the infinite ratio $\frac{\alpha}{\Delta} = \infty$. In contrast, this work does not impose any constraint on the ratio of the memory length over the deadline. That is, we characterize the exact large-field-size error probability under any arbitrary finite α and Δ combinations while [18] studied the error exponent (decay rate) when both α and Δ are asymptotically large.

Almost all existing error probability analyses of sequential coding followed the basic ideas in the seminal tree-code analysis [19], which first derived a genie-aided error-probability lower bound, then derived an achievable error-probability upper bound using the *union bound techniques*, and finally showed that they share the same error exponent and are thus asymptotically (exponentially) tight. Nonetheless, while the genie-aided relaxation and the union bound do not alter the error exponent (decay rate) of the error probability, they are ill-suited when used to bracket the error probability for arbitrary finite α and Δ. More specifically, the bounds are usually of the form $e^{-\alpha E(R)+o(\alpha)}$ for which the little-o term $o(\alpha)$ can still dramatically alter the value of the expression for both small and large α values.

C. Contributions of this work

This work studies the exact large-finite-field error probability of random linear streaming codes (RLSCs) with arbitrarily given finite memory length $\alpha < \infty$ and decoding deadline $\Delta < \infty$ while focusing exclusively on the i.i.d. *symbol erasure channels*. The contributions of this work include:

(i) A new definition of *information debt* that handles the finite memory and finite deadline setting, a generalization of the infinite-memory-based definition in [20, Chapter 9]. The idea of information debt was first proposed by Martinian in 2004 [20]. It can be viewed as how much additional information we still need to receive about the current messages before we can decode them successfully. However, in the original work [20], the memory length α was assumed to be infinity and there is no discussion about the impact of finite decoding deadline Δ. For comparison[3], this work proposes a new definition of information debt that further takes into account the finite memory length and can be readily used to quantify the impact of finite decoding deadline.

(ii) A new sufficient and necessary characterization of the error events of large-finite-field RLSCs for any finite memory length $\alpha < \infty$ and any finite decoding deadline $\Delta < \infty$ based on the new generalized definition of information debt.

(iii) A new random-walk-based analysis framework, which can be of independent research interest. By employing both the standard stopping-time analysis and some new non-stopping-time analysis, we have obtained a closed-form expression of the exact error probability for the finite α and Δ regime, which complements the finite-length block code analysis in [7] for the symbol erasure channel and is a significant improvement over the existing error-exponent-only results.

[2]In the field of streaming codes, decoding delay refers to the time needed before the receiver can accumulate enough observed packets for successful decoding. Under this definition, decoding delay does not include any encoder/decoder processing time nor propagation delay.

[3]Contributions (ii) to (iv) are brand new developments that have no similar counterparts in [20].

(iv) The resulting closed-form error-probability expression is used to examine the intricate tradeoff between memory length (complexity), decoding deadline (delay), code rate (throughput), and error probability (reliability) of RLSCs over a sufficiently large finite field size regime. For example, numerical evaluation shows that under the same code rate and same error probability requirement, the end-to-end delay of RLSCs is 40–48% of that of the MDS block codes in general. This implies that switching from block codes to streaming codes not only eliminates the queueing delay completely (thus 50%) but also improves the error probability.

In a companying paper "Part II: Asymptotic Constants, Powers and Decay Rates," we will build upon the finite-length results in this work, analyze the asymptotic properties (e.g., the error exponent versus the memory length α and versus the decoding deadline Δ), and derive new approximation formulas that are sufficiently tight even for small finite lengths. For comparison, the exact analysis in this work directly quantifies the performance of RLSCs in practical application scenarios. The asymptotic analysis in Part II would tradeoff some precision at the small α and Δ values in exchange for further analytical insights about the intricate dynamics among memory length, decoding deadline, and error probability.

The rest of this paper is organized as follows. Section II describes the system model and the problem formulation. Section III introduces a new and more general information debt definition, and characterizes the error events in the finite memory length and decoding deadline regime. Section IV studies the connection between the exact error probability analysis and a new random-walk analysis with respect to information debt. By exploiting the Markov property, we derive a transition-matrix-based procedure that computes the exact error probability of RLSCs. In Section V, we provide a conceptually simpler but numerically unstable computation based on reversing the Markov Chain. We present the numerical evaluation results in Section VI, and conclude our work in Section VII.

II. Problem Description

A. Basic Notations

The boldface lower and upper letters denote column vectors and 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 $\mathbf{s}_a^b \triangleq \left[\mathbf{s}^\top(a), \mathbf{s}^\top(a+1), \ldots, \mathbf{s}^\top(b)\right]^\top$, which vertically concatenates each $\mathbf{s}(t)$ with the one of the smallest time index at the top. We define the projection operator $(\cdot)^+ \triangleq \max(0, \cdot)$. Matrix $\mathbf{I}_n$ stands for the $n \times n$ identity matrix. We use $\vec{\delta}_k$ to denote a column vector for which only the k-th entry is one and all other entries are zero. If the size/dimension of $\vec{\delta}_k$ is not clear in the context, it will be explicitly stated. Finally, we use $\vec{\mathbf{1}}$ to denote the column vector of all 1s.

B. System Model

This work considers a slotted streaming-code system such that in every time slot, the encoder takes the latest source symbols plus those received in the previous α time slots and outputs a bunch of coded symbols. The coded symbols then pass through a symbol erasure channel. At the decoder side, we assume the optimal decoder, which is usually implemented via Gaussian elimination. We require that each source symbol to be successfully decoded by a hard deadline constraint Δ. Once the deadline is passed, any not-yet-decoded symbol is considered in error and will be counted in the error probability analysis.

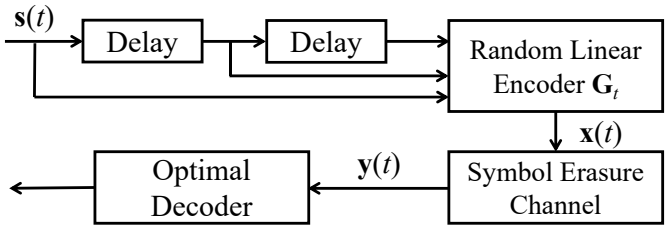

Fig. 1: The block diagram of the random linear streaming codes with $\alpha = 2$.

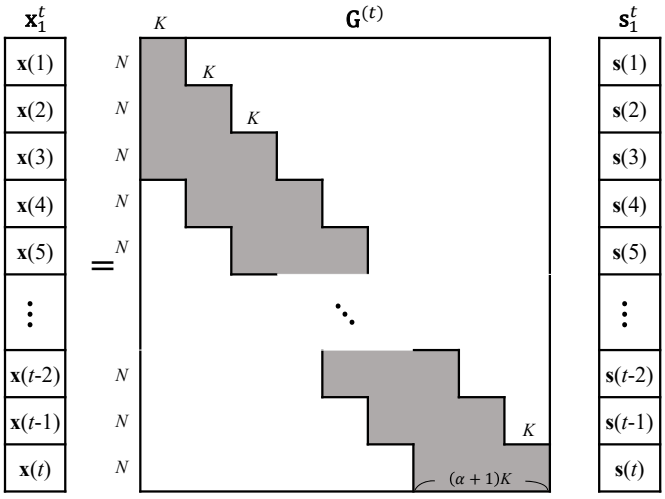

Fig. 2: The illustration of the cumulative generator matrix $\mathbf{G}^{(t)}$ in (2) with $\alpha = 2$. The gray area shows the non-zero entries. The rest of the area are all zeros.

A detailed description of the system is provided as follows and the corresponding illustrations are in Figs. 1 to 3.

Encoder: In every time slot $t \geq 1$, the encoder receives K source symbols, denoted by $\mathbf{s}(t) = [s_1(t), s_2(t), \ldots, s_K(t)]^\top$ where each symbol $s_k(t)$ is a scalar of q bits and is drawn independently and uniformly randomly from the finite field $\mathsf{GF}(2^q)$. The encoder also stores the $\alpha \cdot K$ symbols in the previous α slots $\{\mathbf{s}(\tau) : \tau \in [t-\alpha, t)\}$, where α is the *memory length* and is assumed to be a finite integer. Jointly, it uses the $(\alpha+1)K$ symbols as input and outputs N coded symbols $\mathbf{x}(t) = [x_1(t), \ldots, x_N(t)]^\top \in (\mathsf{GF}(2^q))^N$, see Fig. 1. Throughout the paper, all the encoding/decoding operations are defined over $\mathsf{GF}(2^q)$. Since we focus exclusively on linear codes, define $\mathbf{G}_t$ as the N-by-$(\min(\alpha+1, t) \cdot K)$ *generator matrix* for slot t, and we have

$$\mathbf{x}(t) = \mathbf{G}_t \mathbf{s}_{\max(t-\alpha,1)}^t. \tag{1}$$

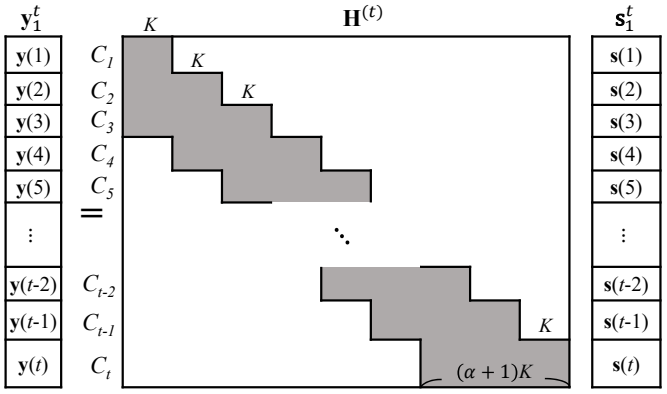

Fig. 3: The illustration of the cumulative receiver matrix $\mathbf{H}^{(t)}$ in (4) with $\alpha = 2$.

It is convenient to also consider $\mathbf{x}_1^t$, the stacked version of all $\mathbf{x}(\cdot)$ vectors until time t. That is, we can properly shift and stack the instantaneous matrix $\mathbf{G}_t$ to create its cumulative representation $\mathbf{G}^{(t)}$, which we term the *cumulative generator matrix*. The cumulative encoding process can then be represented by

$$\mathbf{x}_1^t = \mathbf{G}^{(t)}\mathbf{s}_1^t \tag{2}$$

where $\mathbf{G}^{(t)}$ is an $(\alpha+1)$-block-diagonal matrix. See Fig. 2 for illustration. When $\alpha \to \infty$, matrix $\mathbf{G}^{(t)}$ becomes a lower triangular matrix.

Symbol Erasure Channel: In each time slot t, the source transmits all N symbols in $\mathbf{x}(t)$. A random subset of these N symbols, denoted by $\mathcal{C}_t \subseteq \{1, 2, \cdots, N\}$, will arrive at the decoder perfectly and the complement of which is corrupted heavily and thus considered as erasure. The random set $\mathcal{C}_t$ is i.i.d. across t. We define $C_t \triangleq |\mathcal{C}_t|$ as the number of successfully received symbols and define $P_i \triangleq \Pr(C_t = i)$ as the probability of receiving i symbols successfully, for $i \in \{0, 1, 2, \cdots, N\}$.

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

$$\mathbf{y}(t) = \mathbf{H}_t \mathbf{s}^t_{\max(t-\alpha,1)} \tag{3}$$

where $\mathbf{H}_t$ is the projection of $\mathbf{G}_t$ onto the random (row index) set $\mathcal{C}_t$. Like before, we define the *cumulative receiver matrix* $\mathbf{H}^{(t)}$, and the stacked received signal vector over t time slots can be represented by

$$\mathbf{y}_1^t = \mathbf{H}^{(t)}\mathbf{s}_1^t. \tag{4}$$

See Fig. 3 for illustration.

Decodability: We assume exclusively the optimal decoder at the destination, which is generally implemented via Gaussian elimination. Specifically, for any $t \geq 1$, $k \in [1, K]$, and a deadline constraint $0 \leq \Delta < \infty$, the symbol $s_k(t)$ is decodable if its value can be computed error-freely (no ambiguity) by the cumulative observation $\mathbf{y}_1^{t+\Delta}$ until and including time $t + \Delta$. Otherwise, $s_k(t)$ is viewed as an erasure, or say in error. To algebraically characterize the decodability of the system, we first use $\vec{\delta}_{(t-1)K+k}$ to denote the *location vector* of symbol $s_k(t)$ at time $t + \Delta$, which is a $((t+\Delta)K)$-dimensional column vector for which the $((t-1)K+k)$-th entry is one and all other entries are zero. We then have the following self-explanatory definitions.

Definition 1. A symbol $s_k(t)$ is decodable by time $t + \Delta$ if and only if the transposed vector $\vec{\delta}^\top_{(t-1)K+k}$ is in the row space of $\mathbf{H}^{(t+\Delta)}$.

Definition 2. The vector $\mathbf{s}(t)$ is decodable by time $t+\Delta$ if all $\{s_k(t) : k \in [1, K]\}$ are decodable by time $t + \Delta$. Otherwise, $\mathbf{s}(t)$ is not decodable.

Note that our work focuses on an erasure channel model, and the decoder knows the location of the erasures and thus has full knowledge of the random matrix $\mathbf{H}^{(t)}$. For comparison, in a deletion channel model [21], the erasure positions are unknown at the destination.

Thus far, we describe a general streaming code setup without specifying how the generator matrix $\mathbf{G}_t$ is generated for each t. In this work, we focus exclusively on

Random linear streaming codes (RLSCs): We assume that each entry of $\mathbf{G}_t$ is chosen uniformly and randomly from $\mathsf{GF}(2^q)$, excluding 0. RLSCs are known to be *capacity-achieving* when (α, Δ) both approach infinity simultaneously, though could be strictly suboptimal if under an adversarial channel model with finite deadline $\Delta < \infty$ [20].

The goal of this work is to compute the exact value of the long term *slot error probability* of RLSCs under arbitrarily given finite memory length α and finite deadline Δ. Specifically, we define the slot error probability by:

$$p_{e,[1,T]}^{\text{RLSC}(q)} = \frac{1}{T}\sum_{t=1}^{T} \Pr\left(\mathbf{s}(t) \text{ is not decodable by time } t+\Delta\right). \tag{5}$$

We are interested in quantifying the long term slot error probability under the asymptotically large finite field $\mathsf{GF}(2^q)$, which is defined by

$$p_e \triangleq \lim_{q\to\infty}\lim_{T\to\infty} p_{e,[1,T]}^{\text{RLSC}(q)}. \tag{6}$$

The slot error probability is also used as the main performance metric in the error exponent analysis of [18]. Also see the discussion in Section I.

Remark 1: Another metric that could be of interest in practice is the long-term *symbol error probability* $p_e^{[\text{sym}]}$

$$p_e^{[\text{sym}]} \triangleq \lim_{q\to\infty} \lim_{T\to\infty} \frac{1}{TK}\sum_{t=1}^{T}\sum_{k=1}^{K} \Pr\left(s_k(t) \text{ is not decodable by time } t+\Delta\right). \tag{7}$$

By definition, $p_e^{[\text{sym}]} \leq p_e \leq K \cdot p_e^{[\text{sym}]}$. However, the symbol error probability is not the focus of this work.

Remark 2: Another existing metric is called the *block error probability* [19], which, as suggested by its name, is significantly different from the slot error probability p_e in this work. A detailed comparison between p_e and the block error probability in [19] is provided in Appendix A.

C. Symbols, Slots, Packets, and Their Connections to Actual Implementation

While our model is motivated by packet-level communication networks, we deliberately refrain from using the term *packet* in our system model. Instead, we use *symbol* in $\mathsf{GF}(2^q)$ as the basic unit of incoming messages, see the **Encoder** description in Section II-B. Symbols are also the basic unit of erasure, such that each symbol is either perfectly received or completely erased. See the **Symbol Erasure Channel** description in Section II-B where the unit of each channel delivery $\mathcal{C}_t$ is also symbols. The term *slot* is used for the basic time unit that measures the time interval between two consecutive arrivals of the messages.

The reason that we use *symbols* as our basic information unit is that the term *packet* has a very clear definition in network protocols as the basic unit of transport-layer control, network routing protocols, etc., is thus highly dependent on the actual implementation, and can be quite confusing if used as part of the abstract model.

The following are two examples illustrating how to relate our (symbol, slot)-based model to actual network communication problems.

Example 1. Suppose each packet contains 1000 bytes, specified by the underlying network protocol, and the source can transmit a new packet every 10 ms. We also suppose that every 10 ms, the application layer at the source node would like to transmit a small piece of message of 550 bytes (say a measurement of a remote sensor) and we hope to deliver that message to the destination within 100 ms. Suppose we also know that each packet has 40% chance to be completely erased and 60% chance to be perfectly received. The question to answer is what is the success probability of delivering a single 550-byte message within its 100 ms deadline.

The abstract model for this "practical scenario" is as follows. Each slot is chosen to be 10 ms since we have a new message arriving every 10 ms. We choose the basic message unit to be 1 symbol $= 50$ bytes, which is the greatest common divider of 1000 and 550 bytes. As a result, at every slot, the source will receive $K = 11$ symbols (since each message has 550 bytes), and use them, plus the past message symbols in the last α slots, to generate $N = 20$ symbols for transmissions (since each packet has 1000 bytes). To model the 40% chance of complete packet erasure, the symbol erasure channel model has $\Pr(C_t = 20) = 0.6$ and $\Pr(C_t = 0) = 0.4$. The deadline for decoding each message is set to $\Delta = 100/10 = 10$ slots.

Example 2. This example is identical to Example 1 except that a 550-byte message arrives every 50 ms, rather than 10 ms. The question to answer is still what is the success probability of delivering a single 550-byte message within its 100 ms deadline but in this scenario the messages are arriving less frequently. The abstract model for Example 2 is as follows. Each slot is chosen to be 50 ms since we have a new message arriving every 50 ms. During each slot, we can send 5 packets, which, jointly, consist of 5000 bytes. We choose the basic message unit to be 50 bytes, which is the greatest common divider of 5000 and 550 bytes. As a result, at every slot, the source will receive $K = 11$ symbols (since the message is of 550 bytes), and use them, plus the past message symbols in the last α slots, to generate $N = 100$ symbols for transmissions (since in each slot we can send 5 packets for a total of 5000 bytes). To model the 40% chance of complete packet erasure, the symbol erasure channel model has $\Pr(C_t = 20 \cdot b) = \binom{5}{b} 0.6^b 0.4^{5-b}$, where $b \in \{0, 1, \cdots, 5\}$ is the number of successful packet deliveries within 1 slot (50 ms). The deadline for decoding each message is set to $\Delta = 100/50 = 2$ slots.

As can be seen in Examples 1 and 2, the definitions of symbols and slots provide great flexibility when modeling different packet-level communication scenarios. In real world scenarios, not every quantity is an integer multiple of the other. However, we can define the slots and symbols that give the closest integer approximations of the original problem. The results of our abstract model thus give a close approximation of the real world performance of any streaming code application.

D. From RLSCs to *GMDS* Codes

Our work focuses exclusively on the sufficiently large finite field size regime, see (6). In this subsection, we first define the generalized MDS condition (**GMDS**), and show that any code that satisfies **GMDS** will have the same slot error probability:

$$p_{e,[1,T]}^{\text{GMDS}(T)} \triangleq \frac{1}{T}\sum_{t=1}^{T} \Pr\big(\mathbf{s}(t) \text{ is not decodable by time } t+\Delta \;\big|\; \text{the code satisfies GMDS}(T)\big) \quad (8)$$

where T is an arbitrary finite integer. (The above definition will be carefully elaborated in the subsequent paragraphs.) We then prove that

$$p_e \triangleq \lim_{q\to\infty}\lim_{T\to\infty} p_{e,[1,T]}^{\text{RLSC}(q)} = \lim_{T\to\infty} p_{e,[1,T]}^{\text{GMDS}(T)}. \quad (9)$$

As a result, instead of assuming RLSCs with sufficiently large $\mathsf{GF}(2^q)$, we can just assume the code satisfies **GMDS**. In this way, all the randomness is a result of random channel realization, not the random code construction. This shift of focus from large-finite-field RLSCs to **GMDS** codes greatly simplifies the statements and the proofs for the rest of the paper.

The Generalized MDS Condition (GMDS) consists of two sub-conditions:

- Sub-condition 1: For any time slot t, all entries in the *per-slot generator matrix* $\mathbf{G}_t$ are non-zero. Also see the definition in (1).
- Sub-condition 2: For any finite sequence of pairs $\{(i_l, j_l) : l \in [1, L]\}$ where L can be any positive integer, define the corresponding row index set $S_R = \{i_l : l \in [1, L]\}$ and the column index set $S_C = \{j_l : l \in [1, L]\}$ and define $\mathbf{M}$ as the submatrix of the *cumulative generator matrix* $\mathbf{G}^{(t)}$ induced by S_R and S_C. The matrix $\mathbf{M}$ is invertible for any t and any $\{(i_l, j_l) : l \in [1, L]\}$ satisfying
  (i) $i_{l_1} \neq i_{l_2}$ and $j_{l_1} \neq j_{l_2}$ for any $l_1 \neq l_2$, and
  (ii) the (i_l, j_l)-th entry of $\mathbf{G}^{(t)}$ are non-zero for all $l \in [1, L]$.

Example 3. Suppose (part of) the cumulative generator matrix $\mathbf{G}^{(t)}$ of an RLSC looks like

$$\begin{bmatrix} 1 & 2 & 3 & 4 \\ 4 & 1 & 2 & 1 \\ 5 & 1 & 1 & 3 \\ 2 & 4 & 1 & 3 \end{bmatrix}. \tag{10}$$

One can easily verify that such a $\mathbf{G}^{(t)}$ is of full rank (assuming the real-field is used). However, this $\mathbf{G}^{(t)}$ does *not* satisfy **GMDS** since if we choose $\{(i_l, j_l)\} = \{(1,2),(2,3),(3,4)\}$, the induced submatrix in the upper-right corner becomes

$$\begin{bmatrix} 2 & 3 & 4 \\ 1 & 2 & 1 \\ 1 & 1 & 3 \end{bmatrix}, \tag{11}$$

which is not invertible.

Example 4. Suppose (part of) the $\mathbf{G}^{(t)}$ matrix of an RLSC looks like

$$\begin{bmatrix} 1 & 2 & 3 & 4 \\ 0 & 0 & 3 & 1 \\ 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 4 \end{bmatrix} \tag{12}$$

where some entries are zero due to the finite memory length constraint, also see the zeros in Fig. 2. It is clear that $\mathbf{G}^{(t)}$ is *not* of full rank. However, by exhaustively examining all the sequences of $\{(i_l, j_l)\}$ satisfying (i) and (ii) in the end of **GMDS**, which corresponds to 26 distinct submatrices and by verifying that all 26 such submatrices are of full rank, we can conclude that $\mathbf{G}^{(t)}$ *does* satisfy **GMDS**.

Remark 3: Suppose we replace $\mathbf{G}^{(t)}$ by a traditional (column-based) $n \times k$ generator matrix $\mathbf{G}$ such that $\mathbf{x} = \mathbf{G} \cdot \mathbf{s}$ and we assume that all the entries are non-zero, e.g., a Reed-Solomon code. We now argue that any code that satisfies **GMDS** also satisfies the traditional MDS condition. That is, we first define $j_1 = 1, j_2 = 2, \cdots, j_k = k$. Then for any set of k distinct rows $S_R = \{i_1, i_2, \cdots, i_k\}$, the k pairs $\{(i_l, j_l) : l \in [1,k]\}$ satisfy (i) each i_l (resp. j_l) is unique and (ii) all (i_l, j_l) elements are non-zero because all entries are non-zero. Since the matrix $\mathbf{G}$ satisfies **GMDS**, the induced $k \times k$ submatrix must be invertible. This shows that any set of k row vectors of $\mathbf{G}$ must be of full rank, which satisfies the traditional MDS condition. **GMDS** essentially requires that *the matrix $\mathbf{G}^{(t)}$ plus all its submatrices are of the largest possible rank, even though some portion of $\mathbf{G}^{(t)}$ is hardwired to zero due to the memory length constraint*, see Figs. 2 and 3. Intuitively speaking, **GMDS** ensures that all symbols arriving at the destination carry as much information as possible for *any subset* of the participating source symbols.

We now formalize the discussion around (8) and (9). We first note that **GMDS** does not place any limit on the time index t being considered. Therefore, it is possible that regardless how large the finite field $\mathsf{GF}(2^q)$ is, one cannot find a code that satisfies **GMDS** for infinitely many $t \in [1, \infty)$.

To circumvent this technical difficulty, we define a less restrictive condition GMDS(T) that only requires the two sub-conditions of **GMDS** hold for all $t \leq T + \Delta$ but may or may not hold for $t > T + \Delta$. The input argument (T) restricts our focus to a finite cumulative generator matrix $\mathbf{G}^{(T+\Delta)}$, and **GMDS** defined earlier can thus be viewed as GMDS(∞). While the condition GMDS(∞) may be too restrictive, the condition GMDS(T) is not. In fact, because of the bounded scope of interest in the GMDS(T) condition, we have the following simple lemma.

Lemma 1. *When the order of the finite field* $\mathsf{GF}(2^q)$ *approaches infinity, the probability of RLSCs satisfying* GMDS(T) *approaches one. That is,*

$$\lim_{q\to\infty} \Pr\left(\text{RLSCs under } \mathsf{GF}(2^q) \text{ satisfy GMDS}(T)\right) = 1 \tag{13}$$

for all finite T.

Proof: See the discussion of the Schwartz-Zippel theorem in [22, Theorems 3 and 4]. □

Later in Propositions 1 and 2 of Section III we show that if a code satisfies the GMDS(∞) condition, the error event is only determined by the random channel realization, not by the actual code structure $\mathbf{G}^{(t)}$. Since the error event "$\mathbf{s}(t)$ is not decodable by time $t+\Delta$" only depends on the cumulative generator matrix $\mathbf{G}^{(t+\Delta)}$ and the channel realization from time 1 to $t+\Delta$, Propositions 1 and 2 show that for any fixed T value, any code that satisfies GMDS(T) will have the same average slot error probability $p_{e,[1,T]}^{\text{GMDS}(T)}$ defined in (8).

The above arguments immediately imply that

$$\lim_{q\to\infty} p_{e,[1,T]}^{\text{RLSC}(q)} = p_{e,[1,T]}^{\text{GMDS}(T)}, \ \forall\ T < \infty. \tag{14}$$

In other words, over any finite time duration $[1, T+\Delta]$, the performance of RLSCs with sufficiently large q is indistinguishable from any code that satisfies the GMDS(T) condition. One can easily see that (14) implies

$$\lim_{T\to\infty}\lim_{q\to\infty} p_{e,[1,T]}^{\text{RLSC}(q)} = \lim_{T\to\infty} p_{e,[1,T]}^{\text{GMDS}(T)}. \tag{15}$$

To further strengthen the relationship between RLSCs and **GMDS** codes, we prove the following lemma that swaps the limits of T and q in (15):

Lemma 2. *For any arbitrary but fixed* α, Δ, *and channel distribution* $\{P_i : i \in [0, N]\}$, *the equation* (9) *is true.*

Our proof is constructed by carefully quantifying the impact of the error event propagation over the time horizon. The details are provided in Appendix B.

By Lemma 2, we can quantify p_e defined in (6), the error probability of large-finite-field RLSCs, by assuming GMDS(∞) holds (or just "**GMDS** holds" as shorthand) even though technically we are only assuming GMDS(T) for any arbitrarily given T. One side benefit of assuming **GMDS** (i.e., GMDS(∞)) is that the results of this work are not limited to RLSCs. One can construct the streaming codes in any algebraic/deterministic fashion. As long as it satisfies **GMDS**, our analysis holds.

Remark 4: With **GMDS**, RLSCs assumed in this work can be viewed as a class of MDS convolutional codes [23], [24]. However, the main metric discussed by the literature on traditional MDS convolutional codes is the free distance, the minimum Hamming distance between two legitimate (infinite-length) codewords. Those works thus did not involve the

notion of the error events of each (uncoded) message slot. Also, while the Hamming weight is loosely connected to the decoding deadline, most of the works did not have the explicit notion of decoding deadline. For strongly-MDS convolutional codes [24], which aims to achieve the maximal free distance profile, one may only look at the distance within a finite-size window and hence have a given finite decoding deadline. However, only analyzing the minimum distance under a truncated window cannot characterize the error probability accurately. For example, it is shown in [24] that the free distance increases with the memory length α. A blind application of such a result would assume that the error probability always gets better (smaller) when α becomes larger. However, numerical evaluation of our exact error probability analysis in Section VI-A will show that it is not the case when we have a finite Δ. This again shows the significant difference between the distance profile analysis in [24] and our error probability analysis.

III. Main Result 1: Error-Event Characterization

A. Information-Debt under Finite Memory

The first main result of this work is to generalize the concept of *information debt* $I_d(t)$ originally defined in [20], which considered exclusively the $(\alpha, \Delta) \to (\infty, \infty)$ setting. After providing a new and more general $I_d(t)$ definition, we use it to characterize the error events for the $\alpha < \infty$ and $\Delta \to \infty$ setting in Section III-B and for the $\alpha < \infty$ and $\Delta < \infty$ setting in Section III-D, respectively.

Definition 3. For any arbitrarily given $\alpha < \infty$ (regardless of the Δ value), define a constant $\zeta \triangleq \alpha K + 1$ and initialize $I_d(0) \triangleq 0$. For any $t \geq 1$, we iteratively compute

$$\widehat{I}_d(t) \triangleq (K - C_t + \min\left(I_d(t-1), \alpha K\right))^+ \tag{16}$$

$$I_d(t) \triangleq \min\left(\zeta,\ \widehat{I}_d(t)\right) \tag{17}$$

according to the random channel realization $\{C_t : t\}$.

Broadly speaking, the information debt $I_d(t)$ indicates how much more information, or equivalently, how many more linear equations the receiver needs before it can start to decode successfully. The detailed intuition behind Definition 3 is as follows. The *debt* cannot be negative, hence the $(\cdot)^+$ in (16). Also, since the memory length is α, the maximum debt one can "carry forward" is at most αK, hence the $\min(\cdot, \alpha K)$ operation in (16). The constant $\zeta \triangleq \alpha K + 1$ defines the absolute "ceiling" of the information debt, hence the $\min(\zeta, \cdot)$ in (17). The difference between $\zeta = \alpha K + 1$ and the maximum allowable debt αK is that the former represents the event that the information debt exceeds the maximum allowable debt, i.e., go bankrupt, while the latter represents the event of touching the maximum allowable debt but still maintaining good standing. We introduce two minimum operations, one in (16) and one in (17), to capture the subtle distinction between the two.

Remark 5: When $\alpha \to \infty$, the above definition is equivalent to the original infinite-memory-based definition in [20].

Remark 6: The iterative definition of $I_d(t)$ does not depend on what the code is being used. The results in [20, Lemma 6] use $I_d(t)$ to state the following converse statement on decodability for any codes: If $I_d(t) > 0$, regardless the code construction, at least one $\mathbf{s}(\tau)$ with $\tau \in [1, t]$ is not decodable by time t. In this work, we show that if we focus exclusively on the codes satisfying **GMDS**, then we can significantly strengthen the results and use $I_d(t)$ to describe the exact error events and characterize the exact error probability. In addition to the high-level debt/ceiling-based interpretation, a more formally stated connection between $I_d(t)$ and the structure of the cumulative receiver matrix $\mathbf{H}^{(t)}$ is provided in Appendix C for the interested readers, particularly in the description around Fig. 14 and Lemma 15 therein.

Before proceeding, we notice that $I_d(t)$ is a random process indexed by t. The following two sequences of hitting times $\{t_i : i \in [0, \infty)\}$ and $\{\tau_j : j \in [0, \infty)\}$ will be useful in our subsequent discussion.

Definition 4. Initialize $t_0 \triangleq 0$ and $\tau_0 \triangleq 0$, and define iteratively

$$t_i \triangleq \inf\{t' : t' > t_{i-1}, I_d(t') = 0\} \tag{18}$$

$$\tau_j \triangleq \inf\{t' : t' > \tau_{j-1}, I_d(t') = \zeta\} \tag{19}$$

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

B. The Case of $\Delta \to \infty$

We now describe how to use the information debt in Definition 3 and the corresponding hitting times $\{t_i\}$ and $\{\tau_j\}$ in Definition 4 to characterize the error events of the RLSCs when there is no deadline constraint, i.e., $\Delta \to \infty$.

Proposition 1. *Assume* **GMDS** *holds. For any fixed $i_0 \geq 0$, if there exists no $\tau_j \in (t_{i_0}, t_{i_0+1})$, then $\mathbf{s}(t)$ is decodable by time t_{i_0+1} for all $t \in (t_{i_0}, t_{i_0+1}]$. If there exists a $\tau_j \in (t_{i_0}, t_{i_0+1})$, define τ_{j^*} as the largest τ_j within the interval (t_{i_0}, t_{i_0+1})[4]. Then $\mathbf{s}(t)$ is decodable by t_{i_0+1} for all $t \in (\tau_{j^*} - \alpha, t_{i_0+1}]$.*

Proposition 2. *Continuing from the second case of Proposition 1, none of $\{\mathbf{s}(t) : t \in (t_{i_0}, \tau_{j^*} - \alpha]\}$ is decodable by time T, regardless how large we set the decoding time T.*

See Appendix C for the proofs of Propositions 1 and 2. A byproduct of the proof of Proposition 2 is provided below with the corresponding proof also provided in Appendix C.

Lemma 3. *Assume* **GMDS** *holds. The slot error probability p_e in* (6) *and the symbol error probability $p_e^{[sym]}$ in* (7) *are always identical.*

The essence of Lemma 3 is that under **GMDS** the only way to decode one symbol $s_k(t)$ sent during time slot t is to decode the entire vector $\mathbf{s}(t)$ for the same time slot. It is not possible to selectively decode a subset of symbols within the vector $\mathbf{s}(t)$.

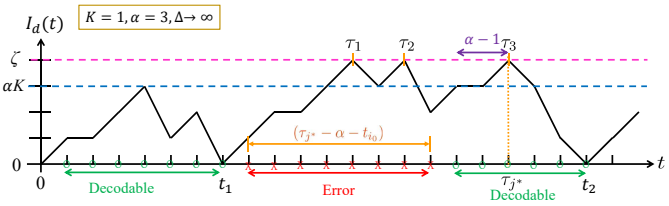

Fig. 4: Error-event characterization for $\Delta \to \infty$.

C. Illustration and Intuition of Propositions 1 and 2

Fig. 4 illustrates an example of the above error-event characterization. In Fig. 4, $I_d(t)$ never hits ζ during the interval (t_0, t_1), i.e., all $I_d(t)$ are within the maximum allowable debt αK in this interval. By Proposition 1, $\mathbf{s}(t)$ is decodable by time t_1 for all $t \in (t_0, t_1]$. On the other hand, $I_d(t)$ hits ζ thrice during the interval (t_1, t_2) and the last time it hits ζ before going back to 0 at time t_2 is at τ_3. By Propositions 1 and 2, $\mathbf{s}(t)$ is decodable by t_2 for all $t \in (\tau_3 - 3, t_2]$ and is not decodable for $t \in (t_1, \tau_3 - 3]$. Note that Propositions 1 and 2 focus exclusively on whether the symbols are *eventually* decodable. They thus can be viewed as a no-deadline setting $\Delta \to \infty$.

The intuition behind is straightforward. Whenever $I_d(t)$ hits 0 at time t_{i_0+1}, it means that we have observed enough linear equations, i.e., we have experienced a random realization with a bunch of large $\{C_t : t\}$ in (16), and can thus start decoding from $\mathbf{s}(t_{i_0+1}), \mathbf{s}(t_{i_0+1} - 1), \cdots$, in a backward fashion. However, if $I_d(t)$ ever hits the bankrupt ceiling ζ during (t_{i_0}, t_{i_0+1}), say at time τ_{j^*}, then the temporal coupling between the earlier symbols $\{\mathbf{s}(t) : t \le \tau_{j^*} - \alpha\}$ and the later symbols $\{\mathbf{s}(t) : t > \tau_{j^*} - \alpha\}$ is severed. The backward decoding thus cannot proceed beyond $\tau_{j^*} - \alpha$, see Propositions 1 and 2. The earlier symbols $\{\mathbf{s}(t) : t \le \tau_{j^*} - \alpha\}$ are forever "stranded" and cannot be decoded.

While the intuition is clear, the proofs of Propositions 1 and 2 require careful use of **GMDS**. The proof of the achievability result Proposition 1 is relatively straightforward as explained below. Recall that the decodability definition in Definition 1 is based on whether the location vector of $s_k(t)$ is in the row space of the cumulative receiver matrix $\mathbf{H}^{(T)}$ or not. Because of **GMDS**, one can easily prove that the submatrix corresponding to the $t \in (t_{i_0}, t_{i_0+1}]$ or $t \in (\tau_{j^*} - \alpha, t_{i_0+1}]$ is of *full rank*. Therefore all location vectors must be in the row space and the proof is complete. See Appendix C-A for details.

On the other hand, the proof of the converse in Proposition 2 is much more involved. In particular, it is easy to argue that the submatrix corresponding to the $t \in (t_{i_0}, \tau_{j^*} - \alpha]$ is *not* of full rank. This shows that *at least* one location vector of $s_k(t)$, $t \in (t_{i_0}, \tau_{j^*} - \alpha]$, is not in the row space of the cumulative receiver matrix $\mathbf{H}^{(T)}$. This is how [20, Lemma 6] proves that *at least* one $\mathbf{s}(t)$ is not decodable regardless code construction, see Remark 6. Nonetheless, it has not ruled out the possibility that *some* location vectors of $s_k(t)$ could still be in the row space of $\mathbf{H}^{(T)}$. To prove Proposition 2, one must show that *none* of the location vectors of $s_k(t)$, $t \in (t_{i_0}, \tau_{j^*} - \alpha]$ can possibly be in the row space of $\mathbf{H}^{(T)}$, which requires more in-depth analysis of the random process $I_d(t)$ and the additional condition of **GMDS**. The majority of the proofs in Appendix C-B is dedicated to establishing the converse.

D. The Case of Arbitrary $\Delta < \infty$

Propositions 1 and 2 have jointly answered the following question: *which set of slots* $\mathbf{s}(t)$ *is decodable and which set of slots is not decodable if there is no decoding deadline requirement, i.e.,* $\Delta \to \infty$. We now consider the case of finite $\Delta < \infty$. For simplicity, we use the statement that $\mathbf{s}(t)$ *is* Δ*-decodable* as shorthand for "$\mathbf{s}(t)$ is decodable by time $t+\Delta$". Our goal is to answer the question:

> Which set of slots $\mathbf{s}(t)$ is Δ-decodable and which set of slots is not?

By Proposition 1, we quickly have

Corollary 1. *Assume* **GMDS** *holds. For any fixed* $i_0 \ge 0$*, if there exists no* $\tau_j \in (t_{i_0}, t_{i_0+1})$*, then* $\mathbf{s}(t)$ *is* Δ*-decodable for all* t *satisfying*

$$t \in [\max(t_{i_0} + 1, t_{i_0+1} - \Delta), t_{i_0+1}]. \tag{20}$$

If there exists a $\tau_j \in (t_{i_0}, t_{i_0+1})$*, define* τ_{j^*} *as the one with the largest* j*. Then* $\mathbf{s}(t)$ *is* Δ*-decodable for all* t *satisfying*

$$t \in [\max(\tau_{j^*} - \alpha + 1, t_{i_0+1} - \Delta), t_{i_0+1}]. \tag{21}$$

Proof: The proof is straightforward. Specifically, Proposition 1 implies that those $\mathbf{s}(t)$ can be decoded by time t_{i_0+1}, which is earlier than their individual deadline $(t+\Delta)$ since (20) and (21) also imply $t + \Delta \ge t_{i_0+1}$. □

Similarly, Proposition 2 implies

Corollary 2. *Continuing from Corollary 1, if* τ_{j^*} *exists, then slot* $\mathbf{s}(t)$ *is* not Δ-decodable *for all* t *satisfying*

$$t \in (t_{i_0}, \tau_{j^*} - \alpha]. \tag{22}$$

Proof: The proof is straightforward since Proposition 2 implies that those symbols are not decodable even if we set $\Delta \to \infty$. Therefore, they are not Δ-decodable within a finite deadline $\Delta < \infty$. □

Comparing Corollaries 1 and 2, one can see that the characterization for finite Δ is not complete since some t value satisfies none of (20) to (22). To close the gap, we strengthen Corollary 2 by the following proposition.

Proposition 3. *Assume* **GMDS** *holds. For any fixed* $i_0 \ge 0$*, if there exists no* $\tau_j \in (t_{i_0}, t_{i_0+1})$*, then* $\mathbf{s}(t)$ *is* not Δ-decodable *for all* t *satisfying*

$$t \in (t_{i_0}, t_{i_0+1} - \Delta). \tag{23}$$

If there exists a $\tau_j \in (t_{i_0}, t_{i_0+1})$ *and* τ_{j^*} *being the one with the largest* j*, then* $\mathbf{s}(t)$ *is* not Δ-decodable *for all* t *satisfying*

$$t \in (t_{i_0}, \max(\tau_{j^*} - \alpha + 1, t_{i_0+1} - \Delta)). \tag{24}$$

[4]By definition, $\tau_{j^*} < t_{i_0+1}$. Furthermore, we always have $t_{i_0} + \alpha < \tau_{j^*}$ since by (16) for each time slot $I_d(t)$ can increase by at most K and it takes at least $\alpha + 1$ slots for $I_d(t)$ to start from $I_d(t_{i_0}) = 0$ to reach $I_d(\tau_{j^*}) = \zeta = \alpha K + 1$. Jointly it ensures that the interval $(\tau_{j^*} - \alpha, t_{i_0+1}]$ is a subset between two consecutive 0-hitting times $(t_{i_0}, t_{i_0+1}]$ and is always non-empty.

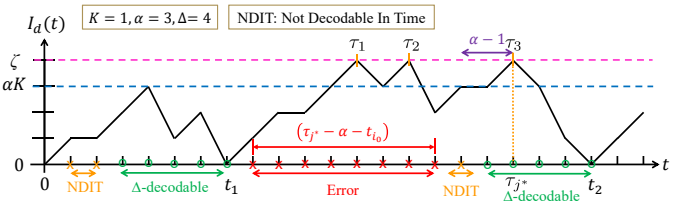

Fig. 5: Error-event characterization for fixed $\Delta < \infty$. NDIT stands for *not decodable in time* and refers to the time slots $\mathrm{s}(t)$ that are eventually decodable, but not before the preset deadline $t+\Delta$.

Proposition 3 strictly subsumes Corollary 2. Jointly Proposition 3 and Corollary 1 close the gap and tightly characterize the Δ-decodability for finite $\Delta < \infty$. See Fig. 5 for illustration, in which we use NDIT as shorthand for *not decodable in time*. Essentially, Proposition 3 shows that all those symbols that are decodable at time t_{i_0+1} according to Proposition 1 cannot be decoded at a strictly earlier time $\tau < t_{i_0+1}$.

See Appendix G for the proof of Proposition 3.

IV. Main Result 2: Exact Slot Error Probability Computation

The iterative definition of $I_d(t)$ in (16) and (17), and the assumption of i.i.d. C_t imply that $I_d(t)$ is a time-homogeneous Markov chain with the state space being $\mathcal{S} = \{0, 1, \cdots, \zeta\}$.

The ergodicity assumption: Without loss of generality, we assume that the Markov chain is ergodic. It is possible that some channel distribution C_t may lead to a reducible Markov chain. For example, suppose $K=2$, $N=4$, and $C_t = 0, 2$ and 4 with probabilities 0.25, 0.5 and 0.25, respectively. The corresponding Markov chain is reducible since $I_d(t)$ only hops on the "even values" except for the ceiling state ζ. We can convert it to an equivalent problem of $K=1$, $N=2$, $C_t = 0, 1$ and 2 with probabilities 0.25, 0.5 and 0.25, respectively. For most practical channel distribution C_t, the ergodicity of $I_d(t)$ holds naturally. For ease of description, all the following lemmas (Lemmas 4 to 8) implicitly assume ergodicity.

Besides, we notice that t_i defined in (18) is a Markov renewal process. To take advantage of the properties of Markov renewal processes, we define the slots between $(t_i, t_{i+1}]$ as the i-th round. For any fixed $i_0 \geq 0$, if there exists no $\tau_j \in (t_{i_0}, t_{i_0+1})$, define the interval $(t_{i_0}, t_{i_0+1}]$ as a *good round*. On the contrary, if there exists a $\tau_j \in (t_{i_0}, t_{i_0+1})$, define the interval $(t_{i_0}, t_{i_0+1}]$ as a *bad round* since the information debt $I_d(t)$ hits the ceiling ζ during this round.

Propositions 1 to 3 imply that for **GMDS** codes, the slot error event is equivalent to some specific event of the Markov chain $I_d(t)$ where the distribution of the random process $I_d(t)$ depends only on the channel realization C_t, not on the actual code structure. As a result, the large-finite-field, long term average slot error probability of RLSCs in (6) can be computed by the following lemma.

Lemma 4. *Assuming operating within the capacity, i.e., $K < \mathbb{E}\{C_t\}$, we have*

$$p_e = \lim_{q\to\infty}\lim_{T\to\infty} p_{e,[1,T]}^{\mathrm{RLSC}(q)} = \lim_{T\to\infty} p_{e,[1,T]}^{\mathrm{GMDS}(T)} \tag{25}$$

$$= \frac{\mathbb{E}\{L_G + L_B\}}{\mathbb{E}\{t_{i_0+1} - t_{i_0}\}} \tag{26}$$

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

$$L_G \triangleq \mathbb{1}_{\{no\ \tau_j \in (t_{i_0}, t_{i_0+1})\}} \cdot (t_{i_0+1} - \Delta - 1 - t_{i_0})^+, \tag{27}$$

$$L_B \triangleq \mathbb{1}_{\{\exists \tau_j \in (t_{i_0}, t_{i_0+1})\}} \cdot (\max(\tau_{j^*} - \alpha,\ t_{i_0+1} - \Delta - 1) - t_{i_0}), \tag{28}$$

τ_{j^*} is the largest τ_j within the interval (t_{i_0}, t_{i_0+1}), and $\mathbb{1}_{\{\cdot\}}$ is the indicator function. The subscripts G and B denote a good and a bad round, respectively.

Proof: The equalities in (25) follow directly from Lemma 2. From Propositions 1 to 3, the number of erroneous slots of **GMDS** codes in a good round is $(t_{i_0+1} - \Delta - 1 - t_{i_0})^+$ while the number of erroneous slots in a bad round is $\max(\tau_{j^*} - \alpha,\ t_{i_0+1} - \Delta - 1) - t_{i_0}$. Also, because t_i defined in (18) is a Markov renewal process, we have Lemma 4 by [25, Theorem 3.3] for any arbitrary fixed constant $i_0 \geq 0$. □

To continue, we further rewrite $\mathbb{E}\{L_B\}$ by the following realization-based summation:

$$L_B = L_{B_1} + L_{B_2} \tag{29}$$

$$\text{where } L_{B_1} \triangleq \mathbb{1}_{\{\exists \tau_j \in (t_{i_0}, t_{i_0+1})\}} \cdot (\tau_{j^*} - t_{i_0}) \tag{30}$$

$$L_{B_2} \triangleq \mathbb{1}_{\{\exists \tau_j \in (t_{i_0}, t_{i_0+1})\}} \cdot (\max(-\alpha,\ t_{i_0+1} - \Delta - 1 - \tau_{j^*})). \tag{31}$$

The rest of this section is dedicated to deriving the exact formulas of $\mathbb{E}\{t_{i_0+1} - t_{i_0}\}$, $\mathbb{E}\{L_G\}$, $\mathbb{E}\{L_{B_1}\}$ and $\mathbb{E}\{L_{B_2}\}$. Note that these quantities are based on the random process $I_d(t)$ and are thus functions of the channel distribution $P_i = \Pr(C_t = i)$. They do not depend on the underlying code structure.

Since the state space of the (time-homogeneous) Markov chain $I_d(t)$ is $\{0, 1, \cdots, \zeta\}$, its transition matrix is of dimension $(\zeta+1)$-by-$(\zeta+1)$ and we denote it by $\Gamma = [\gamma_{i,j}]$. To slightly abuse the notation, we assume the subscripts $i, j \in [0, \zeta]$, rather than the traditional range of $[1, \zeta+1]$. The value $\gamma_{i,j}$, the intersection of the i-th row and j-th column of Γ, is the transition probability from state i to state j, i.e., $\gamma_{i,j} = \Pr(I_d(t+1) = j \mid I_d(t) = i)$. The actual value of $\gamma_{i,j}$ can be easily computed by the encoder parameters N, K, the channel distribution $\{P_i\}$, and the iterative update rules of $I_d(t)$ in (16) and (17). One such example will be provided in Section VI.

Define $\boldsymbol{\phi} = [1, 2, \ldots, \zeta-1]$ as the collection of the *interior* states, i.e., excluding the boundary states $I_d(t) = 0$ and $I_d(t) = \zeta$. We partition the transition matrix Γ into 9 submatrices:

$$\Gamma = \begin{bmatrix} \Gamma_{0,0} & \Gamma_{0,\boldsymbol{\phi}} & \Gamma_{0,\zeta} \\ \Gamma_{\boldsymbol{\phi},0} & \Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}} & \Gamma_{\boldsymbol{\phi},\zeta} \\ \Gamma_{\zeta,0} & \Gamma_{\zeta,\boldsymbol{\phi}} & \Gamma_{\zeta,\zeta} \end{bmatrix} \tag{32}$$

where $\Gamma_{\mathbf{x},\mathbf{y}} = [\ \gamma_{i,j}\ ], \forall i \in \mathbf{x}$ and $j \in \mathbf{y}$. Additionally, we denote $\mathbf{A} \triangleq (\mathbf{I}_{\zeta-1} - \Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}})^{-1}$, which is an $(\zeta-1)$-by-$(\zeta-1)$ square matrix. Define $\mathbf{M}_1$ the $(\zeta+1)$-by-$(\zeta+1)$ matrix that hardwires the first column of Γ to zeros:

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

All our formulas will be based on these three matrices Γ, $\mathbf{A}$, $\mathbf{M}_1$, and sometimes their inverse as well. The existence of the inverses is guaranteed by the ergodicity assumption.

Lemma 5. *Recall that $\vec{\delta}_1$ is a delta-vector for which only the first entry is 1 and all other entries are zero. Then,*

$$\mathbb{E}\{t_{i_0+1} - t_{i_0}\} = \vec{\delta}_1^{\top} (\mathbf{I}_{\zeta+1} - \mathbf{M}_1)^{-1} \vec{\mathbf{1}}. \tag{33}$$

Proof: It can be computed directly by Markov chain analysis [25, Chapter 4]. Specifically, because state-0 is positive recurrent, for any fixed $i_0 \in [0,\infty)$ we have $\Pr(t_{i_0} = \infty) = 0$. Therefore, there is no need to consider the corner case of $t_{i_0} = \infty$ in the rest of the paper. Hence, we have

$$\Pr(t_{i_0+1} - t_{i_0} = k \mid t_{i_0} < \infty) = \begin{cases} \Gamma_{0,0} & \text{if } k = 1, \\ \begin{bmatrix} \Gamma_{0,\boldsymbol{\phi}} & \Gamma_{0,\zeta} \end{bmatrix} \left(\begin{bmatrix} \Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}} & \Gamma_{\boldsymbol{\phi},\zeta} \\ \Gamma_{\zeta,\boldsymbol{\phi}} & \Gamma_{\zeta,\zeta} \end{bmatrix} \right)^{k-2} \begin{bmatrix} \Gamma_{\boldsymbol{\phi},0} \\ \Gamma_{\zeta,0} \end{bmatrix} & \\ & \text{if } k \geq 2. \end{cases} \tag{34}$$

Denote

$$\mathbf{Q} = \begin{bmatrix} \Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}} & \Gamma_{\boldsymbol{\phi},\zeta} \\ \Gamma_{\zeta,\boldsymbol{\phi}} & \Gamma_{\zeta,\zeta} \end{bmatrix}.$$

Since $\mathbf{Q}$ is a submatrix of the (probability) transition matrix Γ, we can always slightly increase the values of some entries of $\mathbf{Q}$ to find a new probability transition matrix $\mathbf{P}$ such that (i) $\mathbf{Q}$ and $\mathbf{P}$ are of the same dimension; (ii) the summation of each row of $\mathbf{P}$ is 1; (iii) every entry of $\mathbf{Q}$ is no larger than that of $\mathbf{P}$, and (iv) $\mathbf{Q} \neq \mathbf{P}$. By the Perron-Frobenius Theorem, the maximum eigenvalue of $\mathbf{P}$ is $\lambda_{\max} = 1$, and every eigenvalue σ of $\mathbf{Q}$ satisfies $|\sigma| < \lambda_{\max} = 1$. Hence,

$$\lim_{k' \to \infty} \left(\begin{bmatrix} \Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}} & \Gamma_{\boldsymbol{\phi},\zeta} \\ \Gamma_{\zeta,\boldsymbol{\phi}} & \Gamma_{\zeta,\zeta} \end{bmatrix} \right)^{k'} = \mathbf{0}. \tag{35}$$

As a result,

$$\mathbb{E}\{t_{i_0+1} - t_{i_0}\} = \mathbb{E}\{t_{i_0+1} - t_{i_0} \mid t_{i_0} < \infty\} = \Gamma_{0,0} + \sum_{k=2}^{\infty} k \cdot \begin{bmatrix} \Gamma_{0,\boldsymbol{\phi}} & \Gamma_{0,\zeta} \end{bmatrix} \left(\begin{bmatrix} \Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}} & \Gamma_{\boldsymbol{\phi},\zeta} \\ \Gamma_{\zeta,\boldsymbol{\phi}} & \Gamma_{\zeta,\zeta} \end{bmatrix} \right)^{k-2} \begin{bmatrix} \Gamma_{\boldsymbol{\phi},0} \\ \Gamma_{\zeta,0} \end{bmatrix}, \tag{36}$$

which can be simplified to the expression in (33) by basic matrix operations and by (35). See Appendix H for details. □

Lemma 6. *The average number of errors in a good round in* (26) *in Lemma 4 can be derived by*

$$\mathbb{E}\{L_G\} = \Gamma_{0,\boldsymbol{\phi}} (\mathbf{A})^2 (\Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}})^{\Delta} \Gamma_{\boldsymbol{\phi},0}. \tag{37}$$

Proof: We note that

$$\Pr(t_{i_0+1} - t_{i_0} = k,\ \text{no } \tau_j \in (t_{i_0}, t_{i_0+1}) \mid t_{i_0} < \infty) = \begin{cases} \Gamma_{0,0} & \text{if } k = 1, \\ \Gamma_{0,\boldsymbol{\phi}} (\Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}})^{k-2} \Gamma_{\boldsymbol{\phi},0} & \text{if } k \geq 2. \end{cases} \tag{38}$$

By similar reasons as when proving (35), we have

$$\lim_{k' \to \infty} (\Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}})^{k'} = \mathbf{0}. \tag{39}$$

Rewriting the expectation of (27) as a summation where the index $k = t_{i_0+1} - t_{i_0}$, we have

$$\mathbb{E}\{L_G\} = \mathbb{E}\{L_G \mid t_{i_0} < \infty\} = \sum_{k=2}^{\infty} (k - \Delta - 1)^+ \Gamma_{0,\boldsymbol{\phi}} (\Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}})^{k-2} \Gamma_{\boldsymbol{\phi},0} \tag{40}$$

where we notice that when $k = 1$, we always have $(k - \Delta - 1)^+ = 0$. Simplifying (40) using (39) will give us (37). □

The computation of $\mathbb{E}\{L_{B_1}\}$ and $\mathbb{E}\{L_{B_2}\}$ is much more involved since τ_{j^*} is the *last time* $I_d(t)$ hits ζ before hitting 0, which is not a stopping time and hence requires careful treatment. For any t, we use the standard hitting time definition $H_t(x)$ of x

$$H_t(x) \triangleq \inf\{\tau > 0 : I_d(t+\tau) = x\}. \tag{41}$$

We then define the following random variable

$$\Lambda_t \triangleq \sup\{\tau \geq 0 : I_d(t+\tau) = \zeta \text{ and } \tau \leq H_t(0)\}, \tag{42}$$

which denotes the *last time* $I_d(\cdot)$ hits ζ before hitting 0. One can see that Λ_t is not a stopping time.

For any fixed time slot t, conditioning on the event $\{I_d(t) = \zeta\}$ we have $\Lambda_t \geq 0$ since the supremum in (42) is over a non-empty set. By the Markov property, the value of $\mathbb{E}\{\Lambda_t \mid I_d(t) = \zeta\}$, the average number of time slots it takes for $I_d(\cdot)$ going from ζ and hitting the last ζ before hitting 0, is not a function of t. Note that (42) implies $\Lambda_t \leq H_t(0)$ always holds, which implies the probability $\Pr(\Lambda_t = \infty) = 0$. We can thus compute $\mathbb{E}\{\Lambda_t \mid I_d(t) = \zeta\}$ by the following recursive equation

$$\mathbb{E}\{\Lambda_t \mid I_d(t) = \zeta\} = \Gamma_{\zeta,\zeta} \cdot (1 + \mathbb{E}\{\Lambda_t \mid I_d(t) = \zeta\}) + \sum_{k=2}^{\infty} (k + \mathbb{E}\{\Lambda_t \mid I_d(t) = \zeta\}) \Gamma_{\zeta,\boldsymbol{\phi}} (\Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}})^{k-2} \Gamma_{\boldsymbol{\phi},\zeta} \tag{43}$$

where the index k represents how many time slots it takes for $I_d(\cdot)$ to hit ζ for yet another time without hitting the value 0. Once $I_d(\cdot)$ hits ζ, we can recursively use the quantity $\mathbb{E}\{\Lambda_t \mid I_d(t) = \zeta\}$ when computing the expectation. Specifically, the first term of (43) calculates the scenario of $I_d(t) = \zeta$ and in the next time slot $I_d(t+1) = \zeta$ is true again. The summation term of (43) uses the index $k \geq 2$ to count the number of slots for $I_d(\cdot)$ to hit ζ again.

We are now ready to compute $\mathbb{E}\{L_{B_1}\}$. By similar arguments, $\mathbb{E}\{L_{B_1}\}$ and $\mathbb{E}\{\Lambda_t \mid I_d(t)=\zeta\}$ are related through the following equation:

$$\begin{aligned}\mathbb{E}\{L_{B_1}\} &= \mathbb{E}\{L_{B_1} \mid t_{i_0}<\infty\}\\ &= \Gamma_{0,\zeta}\cdot(1+\mathbb{E}\{\Lambda_t \mid I_d(t)=\zeta\})\\ &\quad+\sum_{k=2}^{\infty}(k+\mathbb{E}\{\Lambda_t \mid I_d(t)=\zeta\})\,\Gamma_{0,\boldsymbol{\phi}}\left(\Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}}\right)^{k-2}\Gamma_{\boldsymbol{\phi},\zeta} \quad (44)\end{aligned}$$

where the first term of (44) calculates the scenario of $I_d(t_{i_0})=0$ and in the next time slot $I_d(t_{i_0}+1)=\zeta$. In this case, the expectation of $\mathbb{E}\{L_{B_1}\}$ is linked to $\mathbb{E}\{\Lambda_t \mid I_d(t)=\zeta\}$ afterwards. The summation term of (44) uses the index $k\geq 2$ to count the number of slots for $I_d(\cdot)$ to hit ζ after $I_d(t_{i_0})=0$. After $I_d(\cdot)$ hits ζ, the expectation of $\mathbb{E}\{L_{B_1}\}$ is linked to $\mathbb{E}\{\Lambda_t \mid I_d(t)=\zeta\}$ once again.

Lemma 7. *The expectation $\mathbb{E}\{L_{B_1}\}$ in* (28) *to* (30)*, which counts how many time slots it takes to go from state-*0 *to the* last time $I_d(t)$ *hits ζ before hitting* 0 *again, can be computed as follows.*

$$\begin{aligned}\mathbb{E}\{L_{B_1}\} &= (\Gamma_{0,\zeta}+\Gamma_{0,\boldsymbol{\phi}}\mathbf{A}\Gamma_{\boldsymbol{\phi},\zeta})\left(\frac{1+\Gamma_{\zeta,\boldsymbol{\phi}}\left(\mathbf{A}\right)^2\Gamma_{\boldsymbol{\phi},\zeta}}{1-\Gamma_{\zeta,\zeta}-\Gamma_{\zeta,\boldsymbol{\phi}}\mathbf{A}\Gamma_{\boldsymbol{\phi},\zeta}}\right)\\ &\quad+\Gamma_{0,\boldsymbol{\phi}}\left(\mathbf{A}\right)^2\Gamma_{\boldsymbol{\phi},\zeta}. \quad (45)\end{aligned}$$

Proof: By (39) and (43), we have

$$\mathbb{E}\{\Lambda_t \mid I_d(t)=\zeta\} = \frac{\Gamma_{\zeta,\zeta}+\Gamma_{\zeta,\boldsymbol{\phi}}\left(\mathbf{A}+\left(\mathbf{A}\right)^2\right)\Gamma_{\boldsymbol{\phi},\zeta}}{1-\Gamma_{\zeta,\zeta}-\Gamma_{\zeta,\boldsymbol{\phi}}\mathbf{A}\Gamma_{\boldsymbol{\phi},\zeta}}. \quad (46)$$

Substituting (46) into (44), we have (45). □

The most involved computation is $\mathbb{E}\{L_{B_2}\}$ due to its complicated expression in (31) and its involvement of τ_{j^*}, the last time $I_d(\cdot)$ hits ζ before hitting 0, which is not a stopping time.

Lemma 8. *Define* $\psi \triangleq (\Delta-\alpha-1)^+$.

$$\begin{aligned}\mathbb{E}\{L_{B_2}\} &= (\Gamma_{0,\zeta}+\Gamma_{0,\boldsymbol{\phi}}\mathbf{A}\Gamma_{\boldsymbol{\phi},\zeta})\cdot(\Gamma_{\zeta,0}+\Gamma_{\zeta,\boldsymbol{\phi}}\mathbf{A}\Gamma_{\boldsymbol{\phi},0})^{-1}\\ &\cdot\Bigg(-\min(\Delta,\alpha)\,\Gamma_{\zeta,0}-\alpha\Gamma_{\zeta,\boldsymbol{\phi}}\mathbf{A}\Gamma_{\boldsymbol{\phi},0}\\ &\quad+\Gamma_{\zeta,\boldsymbol{\phi}}\left(\left(\mathbf{A}\right)^2+(\alpha+\psi-\Delta)\,\mathbf{A}\right)\left(\Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}}\right)^{\psi}\Gamma_{\boldsymbol{\phi},0}\Bigg). \quad (47)\end{aligned}$$

Proof: Using the hitting time $H_t(x)$ definition in (41), for any fixed constant t we define four associated terms as follows.

$$\mathsf{term}_1 \triangleq \Pr(H_t(0)>H_t(\zeta) \mid I_d(t)=0) \quad (48)$$

$$\begin{aligned}\mathsf{term}_2 \triangleq \mathbb{E}\{&\mathbb{1}_{\{H_t(0)<H_t(\zeta)\}}\cdot\max(-\alpha,H_t(0)-\Delta-1)\\ &\mid I_d(t)=\zeta\} \quad (49)\end{aligned}$$

$$\mathsf{term}_3 \triangleq \Pr(H_t(0)<H_t(\zeta) \mid I_d(t)=\zeta) \quad (50)$$

$$\begin{aligned}\mathsf{term}_4 \triangleq \mathbb{E}\{&\max(-\alpha,H_t(0)-\Delta-1)\\ &\mid I_d(t)=\zeta, H_t(0)<H_t(\zeta)\}. \quad (51)\end{aligned}$$

Because $I_d(\cdot)$ is Markov, the values of term_1 to term_4 do not depend on the given constant $t\geq 0$. Also, by basic probability computation we have $\mathsf{term}_4=\mathsf{term}_2/\mathsf{term}_3$.

For any fixed finite constant $i_0\in[0,\infty)$, we now define

$$\begin{aligned}\mathsf{term}_5 \triangleq \mathbb{E}\{&\max(-\alpha,t_{i_0+1}-\tau_{j^*(i_0)}-\Delta-1)\\ &\mid t_{i_0+1}<\infty, \exists\tau_j\in(t_{i_0},t_{i_0+1})\} \quad (52)\end{aligned}$$

where we append the parentheses '(i_0)' to j^* to emphasize that the value of the *last ζ-hitting time* $j^*(i_0)\triangleq\max\{j:\tau_j\in(t_{i_0},t_{i_0+1})\}$ depends on i_0. It is worth pointing out that term_5 is defined for a fixed constant round-index $i_0\geq 0$ while term_1 to term_4 are defined for a fixed constant time-index $t\geq 0$.

To continue, we have to prove that $\mathsf{term}_4=\mathsf{term}_5$. Intuitively, we note that the conditional event in (51) 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 $t_{i_0+1}=\tau_{j^*}+H_t(0)$. Jointly, we thus have $\mathsf{term}_4=\mathsf{term}_5$. See Appendix I for a rigorous proof.

Next, because $I_d(t)$ is strong Markov, for any fixed $i_0\in[0,\infty)$, by substituting the stopping time $t=t_{i_0}$ into (48), we have

$$\begin{aligned}\mathsf{term}_1 &= \Pr(H_{t_{i_0}}(0)>H_{t_{i_0}}(\zeta) \mid t_{i_0}<\infty)\\ &= \Pr(H_{t_{i_0}}(0)>H_{t_{i_0}}(\zeta) \mid t_{i_0+1}<\infty) \quad (53)\\ &= \Pr(\exists\tau_j\in(t_{i_0},t_{i_0+1}) \mid t_{i_0+1}<\infty) \quad (54)\end{aligned}$$

where (53) follows from state-0 being positive recurrent and thus $\Pr(t_{i_0+1}=\infty)=0$.

For any fixed $i_0\in[0,\infty)$, we then have

$$\begin{aligned}\mathbb{E}\{L_{B_2}\} &= \mathbb{E}\{L_{B_2} \mid t_{i_0+1}<\infty\} \quad (55)\\ &= \mathsf{term}_1\cdot\mathsf{term}_5 \quad (56)\\ &= \mathsf{term}_1\cdot\mathsf{term}_4 \quad (57)\\ &= \mathsf{term}_1\cdot\frac{\mathsf{term}_2}{\mathsf{term}_3} \quad (58)\end{aligned}$$

where (55) follows from $\Pr(t_{i_0+1}=\infty)=0$; (56) follows from (31), (52), (54) and basic probability computation; (57) follows from our previous proof of $\mathsf{term}_4=\mathsf{term}_5$; and (58) follows from the definitions in (49) to (51).

The rest of the computation of $\mathbb{E}\{L_{B_2}\}$ is to compute the values of term_1 to term_3 via standard Markov chain analysis [25]. Specifically we have

$$\mathsf{term}_1 = \Gamma_{0,\zeta}+\Gamma_{0,\boldsymbol{\phi}}\mathbf{A}\Gamma_{\boldsymbol{\phi},\zeta} \quad (59)$$

$$\begin{aligned}\mathsf{term}_2 = &-\min(\Delta,\alpha)\,\Gamma_{\zeta,0}-\alpha\Gamma_{\zeta,\boldsymbol{\phi}}\mathbf{A}\Gamma_{\boldsymbol{\phi},0}\\ &+\Gamma_{\zeta,\boldsymbol{\phi}}\left(\left(\mathbf{A}\right)^2+(\alpha+\psi-\Delta)\,\mathbf{A}\right)\left(\Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}}\right)^{\psi}\Gamma_{\boldsymbol{\phi},0} \quad (60)\end{aligned}$$

$$\mathsf{term}_3 = \Gamma_{\zeta,0}+\Gamma_{\zeta,\boldsymbol{\phi}}\mathbf{A}\Gamma_{\boldsymbol{\phi},0}. \quad (61)$$

See Appendix J for detailed derivation of (59) to (61). Substituting (59) to (61) into (58) gives us the final expression of $\mathbb{E}\{L_{B_2}\}$ in (47). □

Theorem 1. *Assume we operate within the capacity, i.e., $K<\mathbb{E}\{C_t\}$. For any finite (α,Δ), the large-finite-field slot error probability p_e can be computed by* (62)*, shown at the bottom of the next page, where $\psi\triangleq(\Delta-\alpha-1)^+$,*

$$\begin{aligned}Z = &-\min(\Delta,\alpha)\,\Gamma_{\zeta,0}-\alpha\Gamma_{\zeta,\boldsymbol{\phi}}\mathbf{A}\Gamma_{\boldsymbol{\phi},0}\\ &+\Gamma_{\zeta,\boldsymbol{\phi}}\left(\left(\mathbf{A}\right)^2+(\alpha+\psi-\Delta)\,\mathbf{A}\right)\left(\Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}}\right)^{\psi}\Gamma_{\boldsymbol{\phi},0}, \quad (63)\end{aligned}$$

and the definitions of the matrices Γ, $\mathbf{A}$, *and* $\mathbf{M}_1$ *can be found in the discussion around* (32).

Proof: Directly assembling Lemmas 4 to 8 and (29) to (31) gives us (62). □

By Definitions 1 and 2, (5) and (6), it is clear that p_e is monotonically decreasing versus the deadline Δ. By the monotone convergence theorem, for any fixed memory length α, the no-deadline slot error probability ($\Delta \to \infty$) can be computed by

Corollary 3. *For any given* α, *we have*

$$\lim_{\Delta\to\infty} \mathbb{E}\{L_G\} = 0 \tag{64}$$

$$\lim_{\Delta\to\infty} \mathbb{E}\{L_{B_2}\} = -\alpha \cdot \mathsf{term}_1. \tag{65}$$

Since $\mathbb{E}\{L_{B_1}\}$ *does not depend on* Δ, *the no-deadline error probability* $\lim_{\Delta\to\infty} p_e$ *from* (26) *can be computed by*

$$\lim_{\Delta\to\infty} p_e = \frac{\mathbb{E}\{L_{B_1}\} - \alpha \cdot \mathsf{term}_1}{\mathbb{E}\{t_{i_0+1} - t_{i_0}\}} \tag{66}$$

where $\mathbb{E}\{t_{i_0+1} - t_{i_0}\}$, $\mathbb{E}\{L_{B_1}\}$, *and* term_1 *can be calculated by* (33), (45), *and* (59), *respectively.*

The proof of this corollary can be found in Appendix K.

V. A Conceptually Simpler But Numerically Unstable Computation

Significant efforts of the p_e derivation are made to analyze the last ζ-hitting time τ_{j^*} within (t_{i_0}, t_{i_0+1}), which is not a stopping time. While the computation is involved, the resulting formulas in Theorem 1 are numerically stable. For example, one could compute small $p_e \approx 10^{-14}$ with $\Delta = 500$ and the transition matrix Γ of dimension $\alpha K + 2 = 20 \times 5 + 2 = 102$ and $p_e \approx 10^{-6}$ with $\Delta = 394$ and the transition matrix Γ of dimension $\alpha K + 2 = 60 \times 39 + 2 = 2342$, see the numerical evaluation for Case (b) in Section VI-A and the numerical evaluation in Section VI-D.

In this section, we describe an algebraically equivalent computation that is simpler in concept. The main idea of the new computation is based on *reversing the Markov chain*. Comparing the formulas in Theorem 1 and the formulas in this section may lead to new matrix equalities between forward and backward transition matrices. However, in its current form, the formula is not numerically stable for small p_e, and further development is still needed. For example, while the computation of Section IV is numerically stable and can be applied to Case (b) in Section VI-A with $\alpha = 20$ and $\Delta = 500$, if we apply the computation formulas described in this section, one would get a negative error probability because of the numerical precision error.

The new derivation is as follows. Recall that τ_{j^*} is the last time $I_d(t)$ hits the ceiling ζ before going back to state-0 during (t_{i_0}, t_{i_0+1}). Considering the corresponding time-reversed Markov chain of $I_d(t)$, denoted as $\overleftarrow{I_d}(t)$, τ_{j^*} becomes the *first* ζ-hitting time during (t_{i_0}, t_{i_0+1}), which is a stopping time with respect to the time-reversed Markov chain $\overleftarrow{I_d}(t)$. Any expectation/probability computation that involves τ_{j^*} thus becomes a standard stopping time analysis of the reversed Markov chain $\overleftarrow{I_d}(t)$.

Following this idea, we re-formulate the slot error probability in a way similar to Lemma 4:

Lemma 9. *Assuming* $K < \mathbb{E}\{C_t\}$, *we have*

$$p_e = 1 - p_c = 1 - \frac{\mathbb{E}\left\{\overline{L}_G + \overline{L}_B\right\}}{\mathbb{E}\{t_{i_0+1} - t_{i_0}\}} \tag{67}$$

where p_c *represents the probability that a slot is* Δ-decodable *under the large-finite-field regime and*

$$\overline{L}_G = \mathbb{1}_{\{no\ \tau_j \in (t_{i_0}, t_{i_0+1})\}} \cdot (\min(t_{i_0+1} - t_{i_0},\ \Delta + 1)) \tag{68}$$

$$\overline{L}_B = \mathbb{1}_{\{\exists \tau_j \in (t_{i_0}, t_{i_0+1})\}} \cdot (\min(t_{i_0+1} - (\tau_{j^*} - \alpha),\ \Delta + 1)) \tag{69}$$

are the numbers of Δ-decodable slots *in a good round and in a bad round, respectively.*

The proof of Lemma 9 is self-explanatory by describing the complement events of those in Lemma 4.

The denominator $\mathbb{E}\{t_{i_0+1} - t_{i_0}\}$ can be found by Lemma 5. Since the expression of $\overline{L}_G$ in (68) does not involve τ_{j^*}, we can derive its value in ways similar to Lemma 6:

Lemma 10. *The value of* $\mathbb{E}\left\{\overline{L}_G\right\}$ *in* (67) *of Lemma 9 can be derived by*

$$\begin{aligned}\mathbb{E}\left\{\overline{L}_G\right\} =& \Gamma_{0,0} + \Gamma_{0,\boldsymbol{\phi}} (\mathbf{A})^2 \left(\mathbf{I}_{\zeta-1} - (\Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}})^{\Delta-1}\right) \Gamma_{\boldsymbol{\phi},0} \\ &+ \Gamma_{0,\boldsymbol{\phi}} \mathbf{A} \left(\mathbf{I}_{\zeta-1} + (\Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}})^{\Delta-1}\right) \Gamma_{\boldsymbol{\phi},0}.\end{aligned} \tag{70}$$

Proof: From (38), we rewrite the expectation of (68) as

$$\begin{aligned}&\mathbb{E}\left\{\overline{L}_G\right\} = \mathbb{E}\left\{\overline{L}_G \mid t_{i_0} < \infty\right\} \\ &= 1 \cdot \Gamma_{0,0} + \sum_{k=2}^{\Delta} k \Gamma_{0,\boldsymbol{\phi}} (\Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}})^{k-2} \Gamma_{\boldsymbol{\phi},0} \\ &\quad + (\Delta + 1) \sum_{k=\Delta+1}^{\infty} \Gamma_{0,\boldsymbol{\phi}} (\Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}})^{k-2} \Gamma_{\boldsymbol{\phi},0}\end{aligned} \tag{71}$$

where the index $k = t_{i_0+1} - t_{i_0}$. Simplifying the above equation gives us (70). □

We now describe how to compute $\mathbb{E}\left\{\overline{L}_B\right\}$. Recall that Γ is the transition matrix of the Markov chain $I_d(t)$. We denote

$$p_e = \frac{\Gamma_{0,\boldsymbol{\phi}} (\mathbf{A})^2 \left((\Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}})^{\Delta} \Gamma_{\boldsymbol{\phi},0} + \Gamma_{\boldsymbol{\phi},\zeta}\right) + (\Gamma_{0,\zeta} + \Gamma_{0,\boldsymbol{\phi}} \mathbf{A} \Gamma_{\boldsymbol{\phi},\zeta}) \left(\frac{1 + \Gamma_{\zeta,\boldsymbol{\phi}} (\mathbf{A})^2 \Gamma_{\boldsymbol{\phi},\zeta}}{1 - \Gamma_{\zeta,\zeta} - \Gamma_{\zeta,\boldsymbol{\phi}} \mathbf{A} \Gamma_{\boldsymbol{\phi},\zeta}} + \frac{Z}{\Gamma_{\zeta,0} + \Gamma_{\zeta,\boldsymbol{\phi}} \mathbf{A} \Gamma_{\boldsymbol{\phi},0}}\right)}{\vec{\delta}_1^{\top} (\mathbf{I}_{\zeta+1} - \mathbf{M}_1)^{-1} \vec{\mathbf{1}}} \tag{62}$$

the stationary distribution of the transition matrix Γ as $\boldsymbol{\pi} = [\pi_0, \pi_1, \ldots, \pi_\zeta]^\top$, which satisfies

$$\boldsymbol{\pi}^\top \Gamma = \boldsymbol{\pi}^\top \text{ and } \boldsymbol{\pi}^\top \cdot \vec{\mathbf{1}} = 1. \tag{72}$$

Assuming that we are already in the stationary distribution, the transition probability of the reversed Markov chain $\overleftarrow{I_d}(\cdot)$ can then be computed by

$$\begin{aligned} \overleftarrow{\gamma}_{i,j} &= \Pr(\overleftarrow{I_d}(\overleftarrow{t}+1) = j | \overleftarrow{I_d}(\overleftarrow{t}) = i) \\ &= \Pr(I_d(t-1) = j | I_d(t) = i) = \pi_j \cdot \gamma_{j,i}/\pi_i. \end{aligned} \tag{73}$$

Eq. (73) can be written in a matrix form, i.e., the transition matrix $\overleftarrow{\Gamma} = [\ \overleftarrow{\gamma}_{i,j}\ ]$ of the reversed chain can be found by

$$\overleftarrow{\Gamma} = [\ \overleftarrow{\gamma}_{i,j}\ ] = \text{diag}\left(\frac{1}{\boldsymbol{\pi}}\right) \cdot \Gamma^\top \cdot \text{diag}\left(\boldsymbol{\pi}\right) \tag{74}$$

where $\frac{1}{\boldsymbol{\pi}} \triangleq (\frac{1}{\pi_0}, \cdots, \frac{1}{\pi_\zeta})^\top$. In sum, knowing the forward transition matrix Γ, which is defined based on the i.i.d. channel distribution, we can numerically compute the reversed transition matrix $\overleftarrow{\Gamma}$ by first solving $\boldsymbol{\pi}$ via (72) and then plugging it into (74). The computation of $\mathbb{E}\left\{\overline{L}_B\right\}$ is then based on the numerically computed $\overleftarrow{\Gamma}$.

Similar to (32), we partition the transition matrix $\overleftarrow{\Gamma}$ into 9 sub-matrices:

$$\overleftarrow{\Gamma} = \begin{bmatrix} \overleftarrow{\Gamma}_{0,0} & \overleftarrow{\Gamma}_{0,\boldsymbol{\phi}} & \overleftarrow{\Gamma}_{0,\zeta} \\ \overleftarrow{\Gamma}_{\boldsymbol{\phi},0} & \overleftarrow{\Gamma}_{\boldsymbol{\phi},\boldsymbol{\phi}} & \overleftarrow{\Gamma}_{\boldsymbol{\phi},\zeta} \\ \overleftarrow{\Gamma}_{\zeta,0} & \overleftarrow{\Gamma}_{\zeta,\boldsymbol{\phi}} & \overleftarrow{\Gamma}_{\zeta,\zeta} \end{bmatrix}, \tag{75}$$

and denote $\overleftarrow{\mathbf{A}} \triangleq (\mathbf{I}_{\zeta-1} - \overleftarrow{\Gamma}_{\boldsymbol{\phi},\boldsymbol{\phi}})^{-1}$.

Lemma 11. *Define $\psi \triangleq (\Delta - \alpha - 1)^+$. The value of $\mathbb{E}\left\{\overline{L}_B\right\}$ in* (67) *of Lemma 9 can be computed by*

$$\begin{aligned} &\mathbb{E}\left\{\overline{L}_B\right\} \\ &= \min\left(\Delta+1,\ \alpha+1\right) \overleftarrow{\Gamma}_{0,\zeta} + (\alpha+1)\, \overleftarrow{\Gamma}_{0,\boldsymbol{\phi}} \overleftarrow{\mathbf{A}} \overleftarrow{\Gamma}_{\boldsymbol{\phi},\zeta} \\ &\quad + (\Delta - \alpha - \psi)\, \overleftarrow{\Gamma}_{0,\boldsymbol{\phi}} \overleftarrow{\mathbf{A}} \left(\overleftarrow{\Gamma}_{\boldsymbol{\phi},\boldsymbol{\phi}}\right)^{\psi} \overleftarrow{\Gamma}_{\boldsymbol{\phi},\zeta} \\ &\quad + \overleftarrow{\Gamma}_{0,\boldsymbol{\phi}} \left(\overleftarrow{\mathbf{A}}\right)^2 \left(\mathbf{I}_{\zeta-1} - \left(\overleftarrow{\Gamma}_{\boldsymbol{\phi},\boldsymbol{\phi}}\right)^{\psi}\right) \overleftarrow{\Gamma}_{\boldsymbol{\phi},\zeta}. \end{aligned} \tag{76}$$

While the expression of Lemma 11 is long, the proof is relatively straightforward because the τ_{j^*} is now a stopping time in terms of the reversed Markov chain $\overleftarrow{I_d}(\cdot)$. The detailed proof is as follows.

Proof: Define $H_{\zeta\leftarrow 0}$ and $H_{0\leftarrow 0}$ as the (first) ζ-hitting time and the (first) 0-hitting time of the reversed Markov chain $\overleftarrow{I_d}(\cdot)$, respectively, conditioning on starting from state-0. It is clear that

$$\begin{aligned} &\Pr(H_{\zeta\leftarrow 0} = k < H_{0\leftarrow 0} \mid t_{i_0} < \infty) \\ &= \begin{cases} \overleftarrow{\Gamma}_{0,\zeta} & \text{if } k = 1, \\ \overleftarrow{\Gamma}_{0,\boldsymbol{\phi}} \left(\overleftarrow{\Gamma}_{\boldsymbol{\phi},\boldsymbol{\phi}}\right)^{k-2} \overleftarrow{\Gamma}_{\boldsymbol{\phi},\zeta} & \text{if } k \geq 2. \end{cases} \end{aligned} \tag{77}$$

We then note that the event $\{\exists \tau_j \in (t_{i_0}, t_{i_0+1})\}$ is equivalent to the event $\Pr(H_{\zeta\leftarrow 0} < H_{0\leftarrow 0})$ if we focus on the "starting time instant" of $\overleftarrow{I_d}(\cdot)$ to be t_{i_0+1}. Furthermore, we also have $t_{i_0+1} - \tau_{j^*} = H_{\zeta\leftarrow 0}$ under the same starting time instant t_{i_0+1}. Using these two observations and counting the events of different k values as discussed in (77), we have

$$\begin{aligned} &\mathbb{E}\left\{\overline{L}_B\right\} = \mathbb{E}\left\{\overline{L}_B \mid t_{i_0} < \infty\right\} \\ &= \min\left(\Delta+1,\ \alpha+1\right) \overleftarrow{\Gamma}_{0,\zeta} \\ &\quad + \sum_{k=2}^{\psi+1} (\alpha+k)\, \overleftarrow{\Gamma}_{0,\boldsymbol{\phi}} \left(\overleftarrow{\Gamma}_{\boldsymbol{\phi},\boldsymbol{\phi}}\right)^{k-2} \overleftarrow{\Gamma}_{\boldsymbol{\phi},\zeta} \\ &\quad + (\Delta+1) \sum_{k=\psi+2}^{\infty} \overleftarrow{\Gamma}_{0,\boldsymbol{\phi}} \left(\overleftarrow{\Gamma}_{\boldsymbol{\phi},\boldsymbol{\phi}}\right)^{k-2} \overleftarrow{\Gamma}_{\boldsymbol{\phi},\zeta} \end{aligned} \tag{78}$$

where the first term counts the event $\Pr(H_{\zeta\leftarrow 0} = 1 < H_{0\leftarrow 0} \mid t_{i_0} < \infty)$; the first summation counts the events $\Pr(H_{\zeta\leftarrow 0} = k < H_{0\leftarrow 0} \mid t_{i_0} < \infty)$ with $k \leq \psi + 1$; and the second summation counts the rest of the events. Simplifying the above equation gives us (76). □

Theorem 2. *For any finite (α, Δ), the slot error probability p_e of large-finite-field RLSCs can be computed by directly assembling Lemma 5, and Lemmas 9 to 11.*

The formula of computing p_e using Theorem 2 is numerically unstable when p_e is small since p_c in (67) is a value that is very close to 1. It is possible to further simplify the formula to improve stability, which is, however, beyond the scope of this work. Also note that while the concept of reversing the Markov chain is straightforward, converting Γ to $\overleftarrow{\Gamma}$ obscures the near-Toeplitz structure of the original forward transition matrix Γ. This also makes any subsequent analysis more difficult. For example, in [1] the asymptote of the error probability is derived by taking advantage of the near-Toeplitz structure of the forward Γ, an approach not possible with the reversed $\overleftarrow{\Gamma}$.

VI. Numerical Evaluations

Theorem 1 can be used to pinpoint the exact slot error probability p_e of any (α, Δ) and the results can be used by system designers when balancing the RLSC complexity α and latency Δ tradeoff in practical applications. In the first three parts of this section, we use the exact p_e formula to explore three important design tradeoffs: (i) error probability versus memory length, (ii) error probability versus decoding deadline, and (iii) optimal memory length of a given deadline constraint. We consider the following cases in our simulation.

Case (a): $N = 10$, $K = 5$ and C_t be a binomial distribution with $p = \frac{K}{N} + 0.01 = 0.51$, i.e., $C_t \sim B(10, 0.51)$ and $P_i = \binom{10}{i} p^i (1-p)^{10-i}$. This case represents a medium rate $(1/2)$, near-capacity scenario.

Case (b): $N = 10$, $K = 5$ and $C_t \sim B(10, 0.55)$. This case is also a medium rate $(1/2)$ scenario, but the channel is more forgivable and we are operating in a scenario far-away from capacity.

Case (c): $N = 10$, $K = 8$ and $C_t \sim B(10, 0.81)$. This case represents a high rate (0.8), near-capacity scenario.

Case (d): $N = 100$, $K = 50$ and $C_t \sim B(100, 0.51)$. This case is similar to Case (a) but considers the scenario that the source sends 10x more symbols in a single slot. (Or

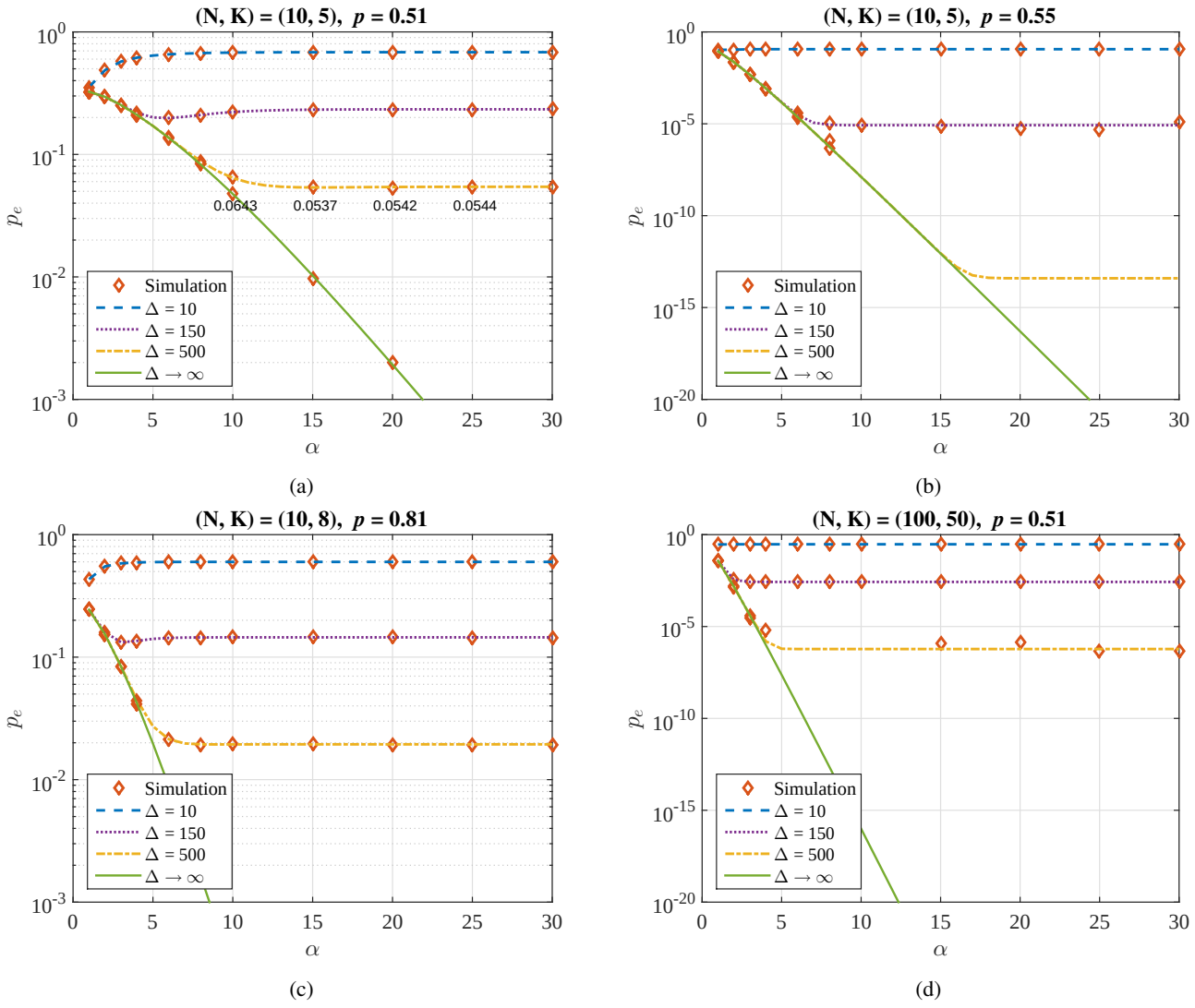

Fig. 6: Slot error probability p_e versus memory length α.

equivalently the duration of the time unit "slot" is lengthened by 10 times.)

For the fourth and the last part of this section, we examine the code rate versus end-to-end delay tradeoff that was considered in the seminal finite-block length work [7]. To ensure direct and fair comparison, the fourth part assumes a simpler packet erasure channel model that is different from the first three parts so that we can reuse the block code results over binary erasure channels (BEC) discussed in [7].

A. Error Probability versus Memory Length Tradeoff

Fig. 6 compares the error probability for different α and Δ values. The diamond markers of "Simulation" are plotted by running the random process $I_d(t)$ from $t = 1$ to 10^8, counting the erroneous slots using Propositions 1 to 3, and dividing by 10^8 to calculate the empirical probability[5]. The continuous curves are obtained by Theorem 1. For those (N, K, α, Δ) and $\{P_i\}$ which yield large p_e (e.g., in Figs. 6a and 6c we have $p_e \geq 10^{-2}$), simulation is very accurate and serves as the ground truth of the analytical results. As expected, the exact error probability computation matches the simulation results, which confirms the correctness of our analysis. For Cases (b) and (d), the p_e values are much smaller ($p_e \leq 10^{-5}$) and simulation markers still center around the computed curves but we start to see random variation since even 10^8 slots do not guarantee enough precision. Regardless how small p_e is, the computation using Theorem 1 can be finished within seconds, which demonstrates the power of having a closed-form probability formula over simple Monte-Carlo simulation.[6]

We first observe that in the operate-near-capacity and small (N, K) settings, Cases (a) and (c), the latency Δ is the limiting

[5]In our simulation, the program is terminated at $t = 10^8$. Those slots which are still unable to determine its decodability at the very end of the simulation are not counted as erroneous slots.

[6]The Monte-Carlo simulation is possible only after the new error event characterization developed in Propositions 1 and 2. If one would like to actually run RLSCs and use naive Gaussian elimination to check decodability, the time it takes would quickly explode and some hardware simulation may be needed.

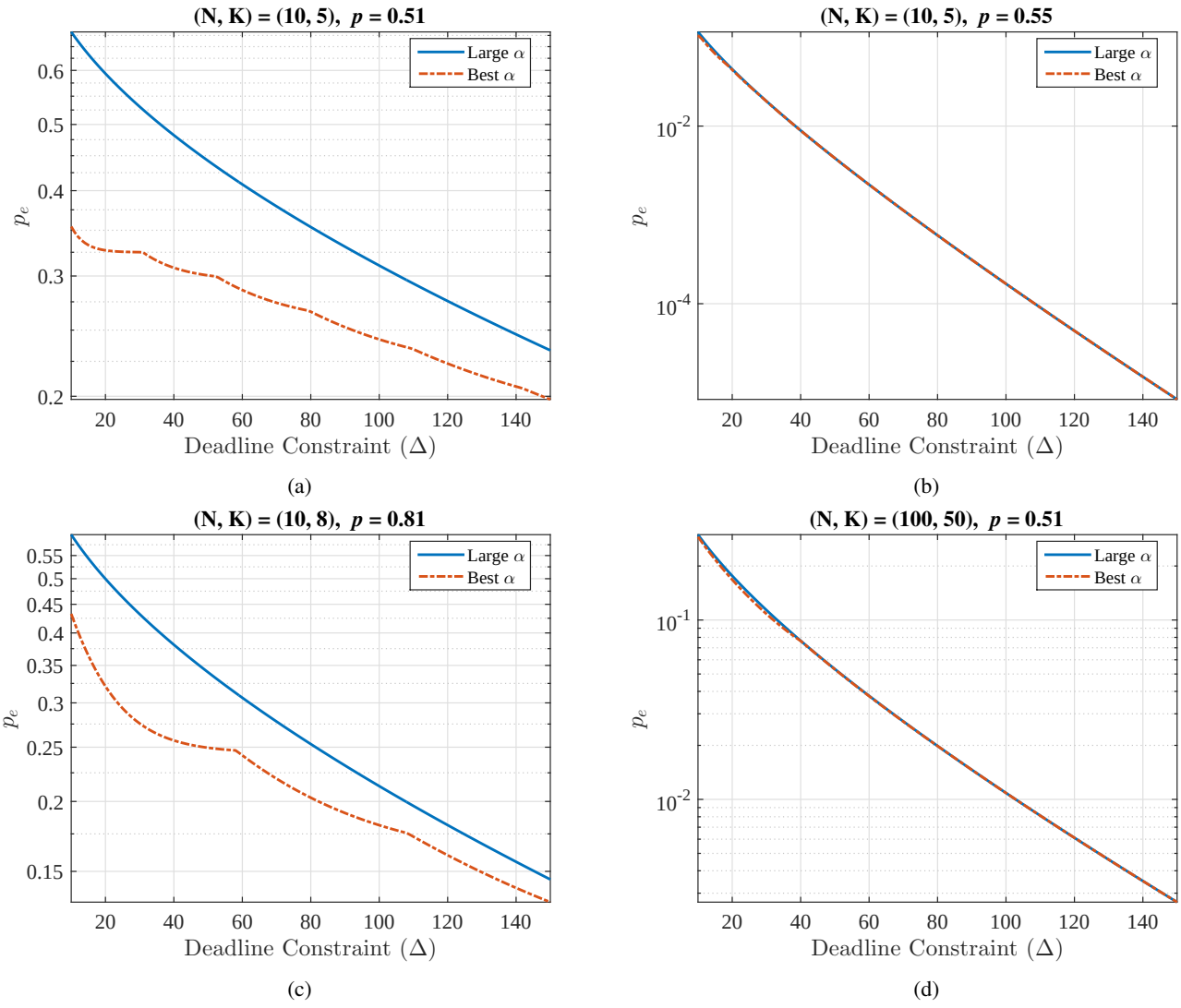

Fig. 7: Slot error probability p_e versus delay constraint Δ.

factor. That is, while the p_e can be made very small when $\Delta \to \infty$, for a finite but large $\Delta = 500$, the error probability p_e remains quite large ($\geq 10^{-2}$) regardless how we set the memory length α, see Figs. 6a and 6c. When we either operate away from capacity (Case (b)) or when many symbols are coded togethers (large N and K in Case (d)), we start to see small error probability $p_e \leq 10^{-5}$ with reasonably small (Δ, α) values.

We also note that if we impose a finite deadline constraint Δ, p_e no longer improves monotonically[7] with α. Take Case (a) for example. When $\Delta = 10$ and 150, the best α is 1 and 6, respectively. When $\Delta = 500$, we wrote the exact p_e values in Fig. 6a for $\alpha = 10, 15, 20$ and 25, respectively. The best p_e is 0.0537 when $\alpha = 15$. In all our evaluations, including others that are not shown, too large the 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 lesson for practical implementation is thus to avoid choosing unnecessarily large memory length α, which is both of higher complexity and also of poorer performance.

[7]Similar finding has been discovered in [18] for a different class of encoder and decoder in finite length regime.

B. Error Probability versus Deadline Constraint Tradeoff

Fig. 7 investigates the effects of deadline constraint Δ on the probability of error. In all our experiments in Fig. 6, the error probability stops improving after $\alpha \geq 50$, we thus use $\alpha = 50$ as the proxy for the case when $\alpha \to \infty$ in our simulation. The blue curves in Figs. 7a to 7d plot p_e versus Δ for the $\alpha \to \infty$ case. As expected, the p_e decreases monotonically as the decoding deadline Δ becomes looser.

In Fig. 6 we observe that for any given Δ, there is an "optimal α" that can minimize p_e. For fixed Δ, we evaluate all p_e values for $\alpha = 1$ to 50 and select the smallest such p_e. The curve "best α" depicts the resulting p_e values using the optimal α. In the large p_e scenarios, Cases (a) and (c),

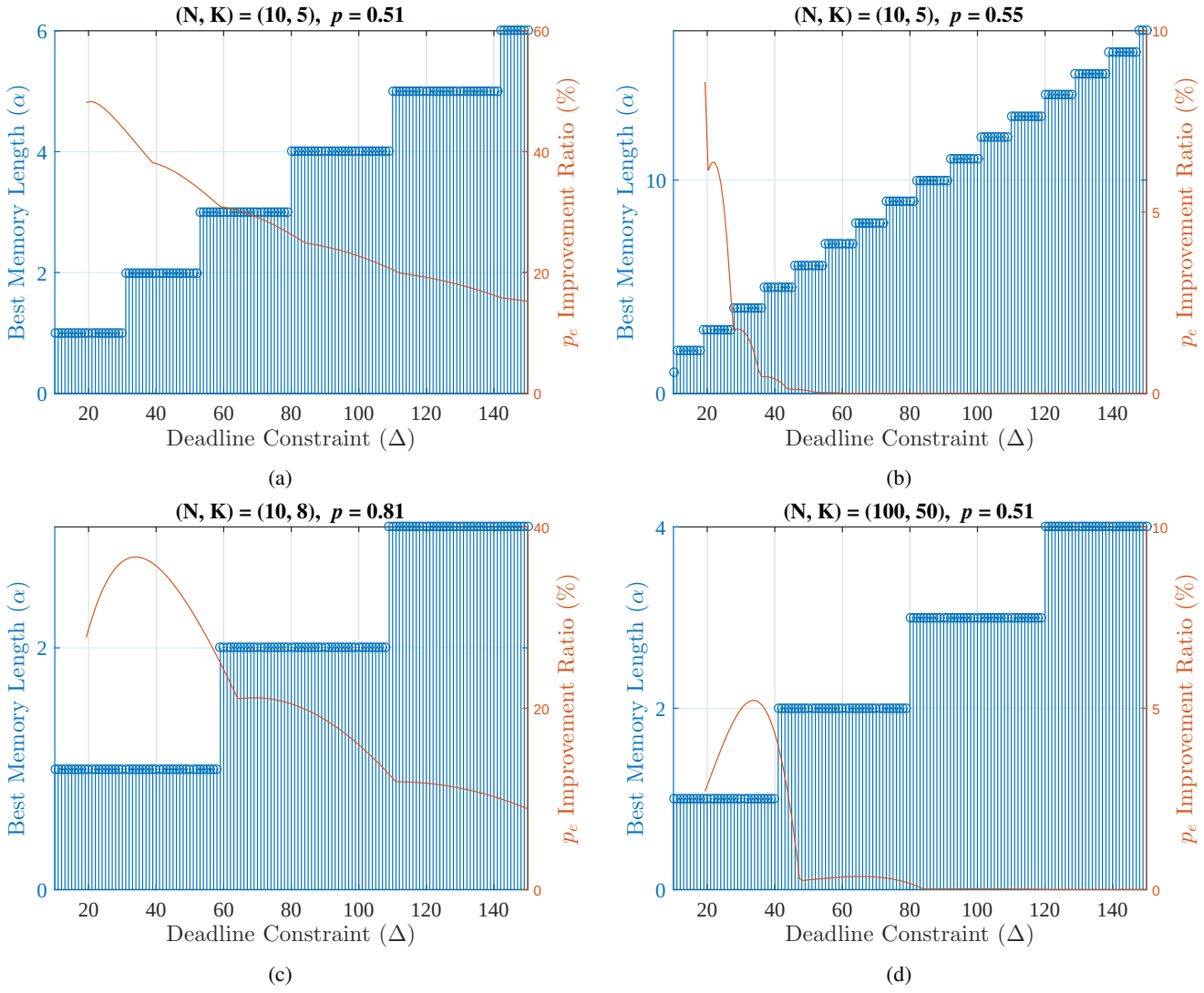

Fig. 8: Best memory length (α) and p_e improvement ratio versus delay constraint Δ.

optimizing α for the given Δ value can further lower p_e by a significant factor. However, in a low-p_e scenario (Cases (b) and (d)), the p_e of the optimal α is indistinguishable to the p_e of large $\alpha \to \infty$. Even though the former is still slightly smaller, the degree of improvement is small. See Figs. 7b and 7d.

C. Optimal Memory Length of a Given Deadline Constraint

The capability of numerically searching for the optimal memory length $\alpha^*(\Delta)$ given any fixed deadline constraint Δ is one of the most interesting achievements unlocked by the exact p_e computation in Theorem 1. It is worth noting that the optimal $\alpha^*(\Delta)$ leads simultaneously to both a smaller p_e and lower complexity for any choice of $\alpha > \alpha^*(\Delta)$. In some scenarios (Figs. 7a and 7c), the p_e reduction benefit is significant. Nonetheless, in the small error probability regime (Figs. 7b and 7d), the p_e reduction benefit of choosing the best $\alpha^*(\Delta)$ becomes negligible. It is worth noting that even in those scenarios, the complexity saving of using $\alpha^*(\Delta)$ is still tangible. That is, the values of $\alpha^*(\Delta)$ inform the designer when we should stop increasing the memory length α since any further increase is a strict waste of complexity.

In this section, we examine the relationship between $\alpha^*(\Delta)$ versus Δ in Fig. 8. We superpose the figure with the *improvement ratio* of the optimal α, which is calculated by

$$\frac{(p_e \text{ given } \alpha \to \infty) - (p_e \text{ given optimal } \alpha)}{(p_e \text{ given } \alpha \to \infty)}. \tag{79}$$

From all four subfigures of Fig. 8, the optimal memory length $\alpha^*(\Delta)$ appears to grow linearly as the deadline Δ increases. A heuristic explanation is that because α is proportional to the number of the future coded symbols influenced by the source symbols in the current time slot, the optimal $\alpha^*(\Delta)$ would maintain a fixed-ratio "balance" between the spread of the source symbols and the number of received symbols used for decoding (the decoding deadline). Another critical observation is that the ratios $\frac{\alpha^*(\Delta)}{\Delta}$ are very small in all four cases Figs. 8a to 8d, with the largest being around 12% in Fig. 8b and the rest being 2% to 5% in Figs. 8a, 8c, and 8d. That is, the optimal memory length $\alpha^*(\Delta)$ is generally

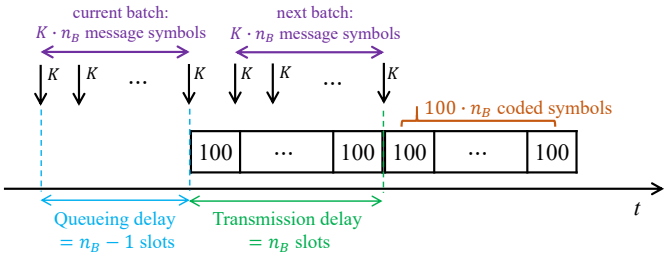

Fig. 9: The illustration of how traditional MDS block codes handle the sequential arrival setting.

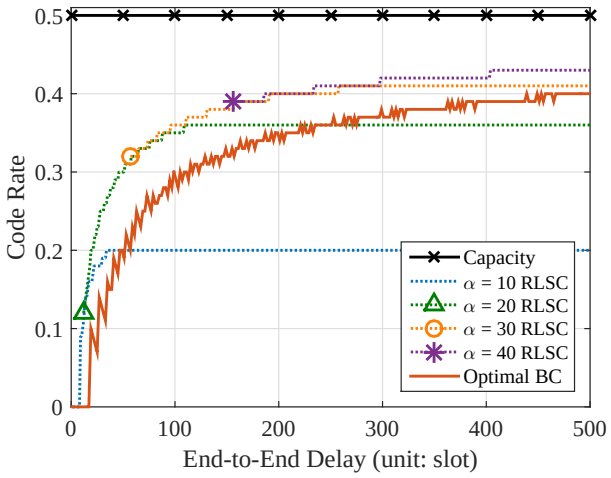

Fig. 10: Rate-delay tradeoff with packet erasure probability $\delta = 0.5$ and maximal error probability $\epsilon = 10^{-3}$. The markers denote the points when a larger memory size should be used to achieve a larger code rate.

much smaller than the decoding deadline Δ. More rigorous analytical investigation is left for future work.

In Cases (a) and (c) for which the error probability larger than 10^{-1}, the p_e improvement ratio of the optimal $\alpha^*(\Delta)$ is around 10% to 50%. In Cases (b) and (d) for which the error probability ranging from 10^{-1} to 10^{-5}, the improvement ratio is much smaller, 0.01% to 5%. It is worth noting that even for the latter two cases, the improvement in the ultra-low latency setting (small Δ) is still relatively significant. See the portion of $\Delta \leq 30$ in Figs. 8b and 8d.

D. Code Rate versus End-to-End Delay Tradeoff

In this subsection, we fix $p_e \leq \epsilon$ and plot the code rate $R = \frac{K}{N}$ versus delay tradeoff, an important setup considered in [7] that is among the most influential metric/plot in the finite-length analysis literature. 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 set $N = 100$ 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 Theorem 1 is still $\leq \epsilon$. The code rate is then defined as $R(\Delta) = \frac{K^*(\Delta)}{N}$.

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 $K \cdot n_B$ message symbols. The equivalent memory length is thus $n_B - 1$ slots. The queueing 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 end-to-end delay is $\Delta_{\text{MDS}}(n_B) = 2n_B - 1$. See Fig. 9 for illustration. For every n_B value, we find the largest K that satisfies the error probability

$$p_{e,\text{MDS}} = \Pr\left(n_B \cdot K > \textstyle\sum_{t=1}^{n_B} C_t\right) \leq \epsilon \tag{80}$$

and plot the curve $(\Delta_{\text{MDS}}(n_B), R_{\text{MDS}}(n_B))$ by varying $n_B \in [1, 250]$. (The zigzagging behavior in Figs. 10 and 11 is quite common in finite-length analysis [7, Fig. 5].) Note that (80) is identical to [7, Theorem 38] under the assumption of q approaches infinity.

Figs. 10 and 11 compare the rate-delay tradeoff between RLSCs and MDS codes, where the former upper bounds the allowable p_e by 10^{-3} and the latter upper bounds p_e by 10^{-6}. In Fig. 10, we plot the four different RLSC constructions with memory length $\alpha = 10$ to 40. Because we are focusing on the small error probability regime $p_e \leq 10^{-3}$, in all our computation, the difference between p_e of optimal $\alpha^*(\Delta)$ versus p_e of $\alpha \to \infty$ is negligible (improvement ratio $< 0.1\%$). Therefore, one can (falsely) "assume" that the larger α, the better performance when examining the figure. For example, we note that the curve of $\alpha = 10$ in Fig. 10 never exceeds $R = 0.2$. That is, with a small memory length $\alpha = 10$, any code rate $\frac{K}{N} > 0.2$ will never achieve $p_e \leq 10^{-3}$ no matter how large we set the deadline Δ. Similarly, the green curve for $\alpha = 20$ in Fig. 10 can be as high as $R = 0.36$ when the deadline $\Delta \geq 109$. However, to support code rate higher than 0.36 (recall that the capacity is $1 - \delta = 0.5$), one must use even larger α with relatively relaxed delay constraint $\Delta \geq 111$. In Fig. 10, we use a marker to describe when a larger α starts to strictly outperform a smaller α. For example, the green triangle marker on the curve of $\alpha = 20$ at the location of $(\text{rate}, \text{delay}) = (0.12, 12)$ signifies that for any rate $R > 0.12$, the delay performance of $\alpha = 20$ would be strictly better (shorter) than the delay performance of $\alpha = 10$ though the curve of $\alpha = 10$ is still rising after the location $(\text{rate}, \text{delay}) = (0.12, 12)$. An equivalent interpretation is that for any application with deadline constraint $\Delta > 12$, the construction with $\alpha = 20$ can support higher rate than those with $\alpha = 10$ while meeting the same $p_e \leq 10^{-3}$ constraint.

Similarly, there is an orange circle marker on the $\alpha = 30$ curve at the location $(\text{rate}, \text{delay}) = (0.32, 57)$. This shows that for any rate $R > 0.32$, the delay performance of $\alpha = 30$

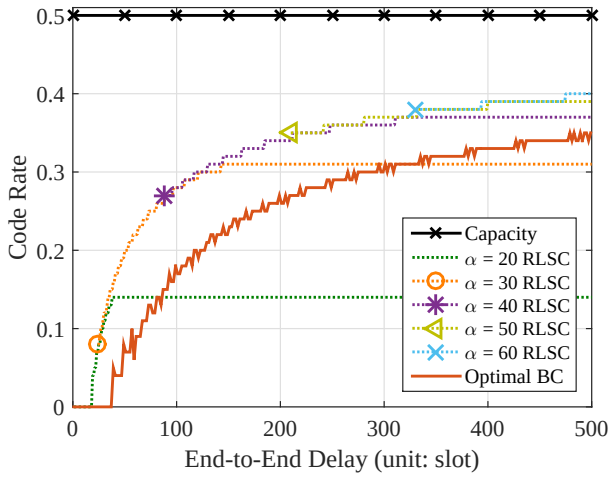

Fig. 11: Rate-delay tradeoff with packet erasure probability $\delta = 0.5$ and maximal error probability $\epsilon = 10^{-6}$. The markers denote the points when a larger memory size should be used to achieve a larger code rate.

would be strictly better (shorter) than the delay performance of $\alpha = 20$ even though the curve of $\alpha = 30$ is still rising at that location. One way of using Fig. 10 for practical designs is that if we aim to operate at rate $R = 0.3$, a reasonable choice[8] of memory length is $\alpha = 20$ because, at that rate, $\alpha = 20$ strictly outperforms $\alpha = 10$ while the benefit of $\alpha = 30$ will not be visible until $(\text{rate}, \text{delay}) = (0.32, 57)$. Fig. 11 is similar to Fig. 10 except for focusing on stricter error probability $p_e \leq 10^{-6}$ and we plot $\alpha = 20$ to 60. If we compare Figs. 10 and 11, the RLSC with code rate 0.3 and memory length $\alpha = 30$, which was an *overkill* for the $p_e \leq 10^{-3}$, is now *under-performing* for the $p_e \leq 10^{-6}$ when compared to the $\alpha = 40$.

The results of both figures show that under the same code rate and the same p_e requirements, the end-to-end delay of RLSCs, assuming the best α is used, is less than 50% of that of the MDS block codes in general. We list a few delay reduction values for different code rates in TABLE I. For example, when constrained on $p_e \leq 10^{-3}$, the shortest end-to-end delay of RLSCs with code rate 0.3 is 46 slots while the shortest end-to-end delay of the MDS block codes with the same code rate is 99 slots. The delay of RLSCs is thus only 46.46% of that of MDS block codes. Similarly, when constrained on $p_e \leq 10^{-6}$, to achieve code rate $R = 0.3$, the shortest end-to-end delay of RLSCs is 117 slots while the shortest end-to-end delay of the MDS block codes is 265 slots. The delay of RLSCs is thus 44.15% of that of MDS block codes. The only exception is the ultra-low delay instance with code rate $R = 0.1$ and $\epsilon = 10^{-3}$. The delay of RLSCs is $11/19 = 57.89\%$ of that of MDS block codes, a number that is higher than 50%, which is likely due to the integer effects in such a short delay/block length example.

[8]The absolutely optimal memory length $\alpha^*(\Delta)$ can be found by numeric search using Theorem 1. In this figure, we only consider α being a multiple of 10 and do not care about finding the optimal $\alpha^*(\Delta)$.

	$\epsilon = 10^{-3}$			$\epsilon = 10^{-6}$		
R	Op. SC	Op. BC	SC/BC	Op. SC	Op. BC	SC/BC
0.1	11	19	57.89%	27	57	47.37%
0.2	19	47	40.43%	50	117	42.74%
0.3	46	99	46.46%	117	265	44.15%
0.4	186	439	42.40%	475	1089	43.62%

TABLE I: The shortest end-to-end delay (unit: slot) and its ratio given the code rate R and the maximal allowable error probability ϵ by using optimal RLSCs and optimal MDS codes.

Even though we only report the results for the $\delta = 0.5$ scenario, similar to the reported results in [7], this phenomenon persists in all our numerical experiments with different δ values, including $\delta = 0.2$, 0.4, and 0.8. In general, our numerical evaluation implies that the encoding-on-the-fly structure of RLSCs not only eliminates the queueing delay completely (thus 50%) but also achieves slightly better slot error probability, which allows RLSCs to further shorten the delay by another 2% to 10% while meeting the same error probability requirement. One possible explanation of this small gain is that, in RLSCs, the information spreading factor (memory length) and the decoding delay are controlled by two separated parameters while they are determined by a single parameter (block length) in block coding. This flexibility may benefit to both encoding and decoding process, and hence we can have additional latency reduction.

VII. Conclusion

In this paper, we have studied random linear streaming codes (RLSCs) over symbol erasure channels. We have proposed a new information debt definition and used it to characterize the error events of RLSCs in the finite memory length and finite decoding deadline regime. A new random-walk-based analysis has been derived, which computes the exact slot error probability via a series of matrix computations based on the transition matrix of the information debt. The resulting closed-form error-probability expression is then used to examine the intricate tradeoff between memory length (complexity), decoding deadline (delay), code rate (throughput), and error probability (reliability) of RLSCs. Numerical evaluation, when paired with the seminal work on the finite-length block code analysis, shows that under the same code rate and the same error probability requirement, the end-to-end delay of RLSCs is 40–48% of that of the MDS block codes in general. This implies that switching from (random) linear block codes to random linear streaming codes not only eliminates the queueing delay completely (thus 50%) but also achieves strictly better erasure recovery under the same latency requirement. The exact error probability formula also enables new in-depth examination of optimal memory length under any given deadline constraint.

Appendix A
Detailed Comparison to the Block Error Probability Metric

Unlike the metrics p_e in (6) and $p_e^{[\text{sym}]}$ in (7) that are based on (α, Δ) pair, the block error probability depends on the

(T, α) pair, where T is the *block length* (unit: slot) and α is the memory length. To rigorously define the block error probability, for any given (T, α), consider a finite sequence $\mathbf{s}(1)$ to $\mathbf{s}(T)$ of T slots. Each slot $\mathbf{s}(t)$ contains K symbols $s_1(t)$ to $s_K(t) \in \mathsf{GF}(2^q)$, the same formulation as discussed previously in **Encoder** in Section II. The first main difference is that *the last α slots are hardwired to zero.* That is, $s_k(t) = 0$ for all $t \in (T-\alpha, T]$ and $k \in [1, K]$. For any fixed (T, α) value, the block error probability is defined as

$$p_e^{[\text{blk}]} \triangleq \Pr\left(\exists t \in [1, T] \text{ such that } \mathbf{s}(t) \text{ is not decodable by time } T\right). \tag{81}$$

Block error probability is the metric used in [19], which characterized the error exponent of the block error probability when α increases to infinity, while assuming the block length T satisfying

$$\lim_{\alpha\to\infty} \frac{T}{\alpha} = \infty, \text{ and } \lim_{\alpha\to\infty} \frac{\log(T)}{\alpha} = 0. \tag{82}$$

The block error probability metric was used in the early days of error correcting code designs, of which the goal was to use convolutional codes as a mechanism of designing block codes and thus the focus on the block lengths and the memory lengths (complexity). In contrast, the slot error probability in (5) is tailored for modern streaming applications in which we synchronize the shift-register encoding operations with the actual symbol arrivals. The operation continues indefinitely with no zero-padding and no "block" to be considered.

There are multiple fundamental differences between block error and slot error probabilities. Firstly, one is a block-based setting (similar to the frame error probability in coding-theory terminology) and the other is a slot-based setting (similar to the bit error probability).

Secondly, in the finite-length regime, i.e., when (T, α) are finite, the code rate in the block-error-probability setting is $\frac{(T-\alpha)\cdot K}{T\cdot N}$, which depends on both T and α since the last α slots are zero-padded. In contrast, for slot/symbol-error-probability setting, the code rate is always $\frac{K}{N}$. With different code rates, it is difficult to compare the two different metrics for the finite (T, α) and finite (α, Δ) regimes even if they share the same (K, N) and the same channel statistics $\{P_i\}$.[9]

Thirdly, in block error probability computation, because of zero-padding at the *end* of the input sequence $\mathbf{s}(1)$ to $\mathbf{s}(T)$, different slots may experience different levels of protection and the block error probability, focusing on the union of the slot error events, serves only as the upper bound of the potentially varying slot error probabilities. For comparison, in slot/symbol error probability computation, see (5) and (6), there is no zero-padding and the definitions in (6) and (7) naturally focus on the (stable) long-term slot error probability.

Finally, there is no concept of a uniform (per slot) decoding deadline Δ in the block-error-probability setting. In particular, the very first slot $\mathbf{s}(1)$ will not be decoded until time T, thus experiencing a delay of $T-1$ slots. The last slot $\mathbf{s}(T-\alpha)$ will also be decoded at time T, thus experiencing a delay of α slots. As a result, the deadline/delay varies among $\mathbf{s}(t)$ for different t. For comparison, the decoding deadline in the slot-error-probability setting is set universally for all t, which is thus a more direct metric for modern delay-oriented applications.

[9] Almost all existing works consider exclusively the asymptotic block error probability. When (T, α) are large and satisfy (82), the code rate $\frac{(T-\alpha)\cdot K}{T\cdot N}$ converges to $\frac{K}{N}$.

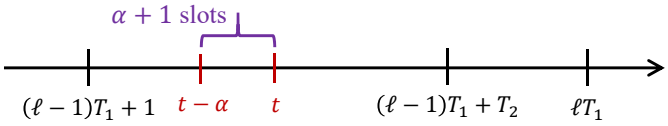

Fig. 12: The illustration of the random set $\mathcal{T}_{\ell,T_1,T_2}^{\text{RLSC}(q)}$.

Appendix B
Proof of Lemma 2

We prove the equality in Lemma 2 by showing

$$\lim_{q\to\infty}\lim_{T\to\infty} p_{e,[1,T]}^{\text{RLSC}(q)} \geq \lim_{T\to\infty} p_{e,[1,T]}^{\text{GMDS}(T)} \tag{83}$$

and

$$\lim_{q\to\infty}\lim_{T\to\infty} p_{e,[1,T]}^{\text{RLSC}(q)} \leq \lim_{T\to\infty} p_{e,[1,T]}^{\text{GMDS}(T)}. \tag{84}$$

To that end, we introduce the following lemmas.

Lemma 12. *For any finite integer $T_1 > 0$,*

$$\lim_{T\to\infty} p_{e,[1,T]}^{\text{RLSC}(q)} \geq p_{e,[1,T_1]}^{\text{RLSC}(q)}. \tag{85}$$

Proof:

$$\lim_{T\to\infty} p_{e,[1,T]}^{\text{RLSC}(q)}$$

$$\geq \lim_{L\to\infty} \frac{1}{L}\sum_{\ell=1}^{L} \frac{1}{T_1} \sum_{t=(\ell-1)T_1+1}^{\ell T_1} \Pr\big(\mathbf{s}(t) \text{ is not decodable by time } t+\Delta \text{ WHILE a genie gives the values of } \mathbf{s}(\tau) \text{ for } \tau \leq (\ell-1)\,T_1 \text{ as side info.}\big) \tag{86}$$

$$= \lim_{L\to\infty} \frac{1}{L}\sum_{\ell=1}^{L} p_{e,[1,T_1]}^{\text{RLSC}(q)} = p_{e,[1,T_1]}^{\text{RLSC}(q)}. \tag{87}$$

We have the inequality in (86) since the additional side information given by the genie always improves the decodability of $\mathbf{s}(t)$. Note that knowing the side information, the values of $\mathbf{s}(\tau)$ for $\tau \leq (\ell-1)\,T_1$, breaks the temporal dependence between those $\mathbf{s}(\tau)$ for $\tau \leq (\ell-1)\,T_1$ and those $\mathbf{s}(\tau)$ for $\tau \geq (\ell-1)\,T_1 + 1$. Therefore, at time $\tau = (\ell-1)T_1 + 1$, it is as if the encoder/decoder has been "reset" and the decoder is facing the same environment as if at time $\tau = 1$. Also see the first three paragraphs of Appendix C-A for a more formal discussion about this argument.

Because the channel is i.i.d., the equality from (86) to (87) holds. The second equality in (87) holds because $p_{e,[1,T_1]}^{\text{RLSC}(q)}$ is not a function of ℓ. □

Definition 5. For any integer $T_1 > T_2 > 0$ and any integer $\ell > 0$, define a random set

$$\mathcal{T}^{\text{RLSC}(q)}_{\ell,T_1,T_2} \triangleq \big\{ t \in [(\ell-1)\,T_1+1, (\ell-1)\,T_1+T_2] \text{ such that } \mathbf{s}(t') \text{ is decodable by time } t \text{ for all } t' \in [t-\alpha, t] \big\}. \quad (88)$$

That is, $\mathcal{T}^{\text{RLSC}(q)}_{\ell,T_1,T_2}$ focuses on $[(\ell-1)\,T_1+1, (\ell-1)\,T_1+T_2]$, a sub-interval of the length-T_1 partition $[(\ell-1)\,T_1+1, \ell T_1]$. See Fig. 12 for illustration. Then we let $\mathcal{T}^{\text{RLSC}(q)}_{\ell,T_1,T_2}$ contains the slots such that that slot t and its previous α slots are all decodable at time t. Obviously $\mathcal{T}^{\text{RLSC}(q)}_{\ell,T_1,T_2}$ is a random set that depends on the realization of the RLSCs (which affects the level of protection experienced by each slot) and the realization of the erasure pattern (which affects the level of noise each slot faces).

Lemma 13. *Based on Definition 5, we have*

$$\frac{1}{T_1}\sum_{t=(\ell-1)T_1+1}^{\ell T_1} \Pr\left(\mathbf{s}(t) \textit{ is not decodable by time } t+\Delta\right) \le \frac{T_2}{T_1} + \max_{\tau\in[1,T_2]} p^{\text{RLSC}(q)}_{e,[1,T_1-\tau]} + \Pr\left(\mathcal{T}^{\text{RLSC}(q)}_{\ell,T_1,T_2} = \emptyset\right). \quad (89)$$

The intuition of (89) is as follows. Whenever $\mathcal{T}^{\text{RLSC}(q)}_{\ell,T_1,T_2} \neq \emptyset$, the RLSCs can decode $\mathbf{s}(t-\alpha), \cdots, \mathbf{s}(t)$ by time t for some $t \in [(\ell-1)\,T_1+1, (\ell-1)\,T_1+T_2]$. Similar to the previous genie-based discussion, the temporal dependence between those $\mathbf{s}(\tau)$ for $\tau \le t$ and those $\mathbf{s}(\tau)$ for $\tau \ge t+1$ is then broken. Therefore, at time $\tau = t+1$, one can treat that the encoder/decoder has been "reset" and the decoder is facing the same environment as if at time $\tau = 1$. However, because we do not know how large/small such $t \in \mathcal{T}^{\text{RLSC}(q)}_{\ell,T_1,T_2}$ would be, we take the maximum over $\tau \in [1, T_2]$ when evaluating the error probability in the second term of (89). The first and the third terms of (89) are based on union bounds that consider all other scenarios. Specifically, the first term T_2/T_1 in (89) assumes all $\mathbf{s}(t)$ are not decodable by time $t+\Delta$ for $t \in [(\ell-1)\,T_1+1, (\ell-1)\,T_1+T_2]$. The third term counts the probability that $\mathcal{T}^{\text{RLSC}(q)}_{\ell,T_1,T_2}$ is an empty set, i.e., no such $\mathbf{s}(t)$ exists that breaks the temporal dependence of RLSC decoding.

A detailed and more rigorous proof is as follows.

Proof:

$$\frac{1}{T_1}\sum_{t=(\ell-1)T_1+1}^{\ell T_1} \Pr\left(\mathbf{s}(t) \text{ is not decodable by time } t+\Delta\right)$$
$$\le \frac{1}{T_1}\sum_{t=(\ell-1)T_1+1}^{\ell T_1} \Bigg(\sum_{\tau=1}^{T_2} \Pr\Big(\inf \mathcal{T}^{\text{RLSC}(q)}_{\ell,T_1,T_2} = (\ell-1)\,T_1+\tau \Big) \cdot \Pr\Big(\mathbf{s}(t) \text{ is not decodable by time } t+\Delta \Big| \inf \mathcal{T}^{\text{RLSC}(q)}_{\ell,T_1,T_2} = (\ell-1)\,T_1+\tau \Big) + \Pr\Big(\mathcal{T}^{\text{RLSC}(q)}_{\ell,T_1,T_2} = \emptyset\Big) \Bigg) \quad (90)$$

$$\le \Pr\Big(\mathcal{T}^{\text{RLSC}(q)}_{\ell,T_1,T_2} = \emptyset\Big) + \frac{T_2}{T_1} + \sum_{\tau=1}^{T_2} \Pr\Big(\inf \mathcal{T}^{\text{RLSC}(q)}_{\ell,T_1,T_2} = (\ell-1)\,T_1+\tau \Big) \cdot \Bigg(\frac{1}{T_1}\sum_{t=(\ell-1)T_1+T_2+1}^{\ell\cdot T_1} \Pr\Big(\mathbf{s}(t) \text{ is not decodable by time } t+\Delta \Big| \inf \mathcal{T}^{\text{RLSC}(q)}_{\ell,T_1,T_2} = (\ell-1)\,T_1+\tau \Big) \Bigg) \quad (91)$$

where (90) follows from the Bayes rule and the union bound, and (91) upper bounds the terms corresponding to $t = (\ell-1)T_1+1$ to $(\ell-1)T_1+T_2$ by one. We now further upper bound the terms corresponding to $t = (\ell-1)T_1+T_2+1$ to ℓT_1. For any arbitrarily given $\tau \in [1, T_2]$, we have

$$\frac{1}{T_1}\sum_{t=(\ell-1)T_1+\tau+1}^{\ell T_1} \Pr\Big(\mathbf{s}(t) \text{ is not decodable by time } t+\Delta \Big| \inf \mathcal{T}^{\text{RLSC}(q)}_{\ell,T_1,T_2} = (\ell-1)\,T_1+\tau \Big) \quad (92)$$
$$= \frac{1}{T_1}\Big((T_1-\tau)\, p^{\text{RLSC}(q)}_{e,[1,T_1-\tau]} \Big) \le p^{\text{RLSC}(q)}_{e,[1,T_1-\tau]}. \quad (93)$$

The equality in (93) can be obtained as follows. The (indicator of the) event $\{\inf \mathcal{T}^{\text{RLSC}(q)}_{\ell,T_1,T_2} = (\ell-1)\,T_1+\tau\}$ is a *stopping time*. Since the encoder/decoder has been "reset" at time $(\ell-1)\,T_1+\tau$ and since both the RLSC construction and the erasure channel are memoryless, by the strong Markov property, the (random) decoding behavior of time interval $[(\ell-1)\,T_1+\tau+1, \ell T_1]$ is identical to the time interval $[1, T_1-\tau]$. The summation of the conditional error probabilities in (92) is thus $(T_1-\tau)p^{\text{RLSC}(q)}_{e,[1,T_1-\tau]}$, which implies the equality in (93).

We then note that (i) the left-hand side of (93) covers a wider range $[(\ell-1)\,T_1+\tau+1, \ell T_1]$ than the range $[(\ell-1)\,T_1+T_2+1, \ell T_1]$ used in (91), and (ii) for any random variable X and any non-negative function $f(x)$, we always have $\sum \Pr(X=x) f(x) \le \max_x f(x)$. Jointly, these two facts plus (91) and (93) imply (89).

□

Lemma 14. *There exists a function $f_{\text{UB}}(T_2, q)$ such that*

$$\Pr\left(\mathcal{T}^{\text{RLSC}(q)}_{\ell,T_1,T_2} = \emptyset\right) \le f_{\text{UB}}(T_2, q) \quad \textit{for all } \ell,\ T_1 \quad (94)$$

and

$$\lim_{T_2\to\infty}\lim_{q\to\infty} f_{\mathrm{UB}}(T_2,q)=0. \tag{95}$$

Proof: For any integer $T_1 > T_2 > 0$ and any integer $\ell > 0$, recall that $\mathbf{G}^{((\ell-1)T_1+T_2)}$ is the cumulative generator matrix at time $(\ell-1)\,T_1+T_2$. Define a submatrix $\overline{\mathbf{G}}^{\mathrm{RLSC}(q)}_{\ell,T_1,T_2}$ as follows.

$$\begin{aligned}\overline{\mathbf{G}}^{\mathrm{RLSC}(q)}_{\ell,T_1,T_2}:&\text{ the submatrix induced by rows corresponding}\\ &\text{to the coded symbols during time interval}\\ &[(\ell-1)\,T_1+1,(\ell-1)\,T_1+T_2]\end{aligned} \tag{96}$$

By definition, $\overline{\mathbf{G}}^{\mathrm{RLSC}(q)}_{\ell,T_1,T_2}$ is a fixed-size matrix but its entries are randomly chosen according to RLSCs. We now define two random events:

$$\mathcal{A}^{\mathrm{RLSC}(q)}_{\ell,T_1,T_2}=\left\{\overline{\mathbf{G}}^{\mathrm{RLSC}(q)}_{\ell,T_1,T_2}\text{ satisfies }\mathbf{GMDS}\right\} \tag{97}$$

$$\begin{aligned}\mathcal{B}_{\ell,T_1,T_2}=&\Big\{\exists\widetilde{t}\in[(\ell-1)\,T_1+1,(\ell-1)\,T_1+T_2]\text{ such that}\\ &\sum_{t'=(\ell-1)T_1+1}^{\widetilde{t}} C_{t'}\geq\left(\widetilde{t}-((\ell-1)\,T_1)+\alpha\right)K\Big\}\end{aligned} \tag{98}$$

The main idea is that $\mathcal{A}^{\mathrm{RLSC}(q)}_{\ell,T_1,T_2}$ is a random event of the RLSC construction being "good", i.e., satisfying **GMDS**; and $\mathcal{B}_{\ell,T_1,T_2}$ is a random event about the channel realization being "good", i.e., there exists an interval $[(\ell-1)T_1+1,\widetilde{t}\,]$ such that the total number of observations C_t exceeds K times the sum of the number of slots $\widetilde{t}-(\ell-1)\,T_1$ plus additional α.

We now argue that

$$\mathcal{A}^{\mathrm{RLSC}(q)}_{\ell,T_1,T_2}\cap\mathcal{B}_{\ell,T_1,T_2}\subseteq\left\{\mathcal{T}^{\mathrm{RLSC}(q)}_{\ell,T_1,T_2}\neq\emptyset\right\}. \tag{99}$$

The intuition is straightforward. That is, when both the code structure and the channel realization are good, one can decode all $\mathbf{s}(\tau')$ for $\tau'\in[\,\widetilde{t}-\alpha,\widetilde{t}\,]$ where $\widetilde{t}$ is the $\widetilde{t}$ satisfying $\mathcal{B}_{\ell,T_1,T_2}$. The reason is that the good code structure plus the good channel realization guarantees that the corresponding generator matrix being considered is of full rank. Therefore, such $\widetilde{t}\in\mathcal{T}^{\mathrm{RLSC}(q)}_{\ell,T_1,T_2}$. A more rigorous argument can be derived by using the terminologies described in the proofs of and right after Lemma 16 in Appendix C-A.

To prove Lemma 14, we define

$$f_{\mathrm{UB}}(T_2,q)\triangleq\Pr\left(\overline{\mathcal{A}^{\mathrm{RLSC}(q)}_{\ell,T_1,T_2}}\right)+\Pr\left(\overline{\mathcal{B}_{\ell,T_1,T_2}}\right) \tag{100}$$

where $\overline{\mathcal{A}^{\mathrm{RLSC}(q)}_{\ell,T_1,T_2}}$ and $\overline{\mathcal{B}_{\ell,T_1,T_2}}$ are the complements of $\mathcal{A}^{\mathrm{RLSC}(q)}_{\ell,T_1,T_2}$ and $\mathcal{B}_{\ell,T_1,T_2}$, respectively. Because the construction of RLSCs is stationary, $\Pr\left(\overline{\mathcal{A}^{\mathrm{RLSC}(q)}_{\ell,T_1,T_2}}\right)$ is a function of q and T_2, but not a function of ℓ and T_1. Similarly, assuming the channel is i.i.d., $\Pr\left(\overline{\mathcal{B}_{\ell,T_1,T_2}}\right)$ is a function of T_2, but not a function of ℓ and T_1. That is why (100) is only a function of (T_2,q), not a function of (ℓ,T_1).

Ineq. (94) is then a simple result of (99) and the union bound. To prove (95), we notice that

$$\lim_{q\to\infty}\Pr\left(\overline{\mathcal{A}^{\mathrm{RLSC}(q)}_{\ell,T_1,T_2}}\right)=0 \tag{101}$$

for any fixed T_2 because of Lemma 1. Therefore, proving (95) is equivalent to proving

$$\lim_{T_2\to\infty}\Pr\left(\overline{\mathcal{B}_{\ell,T_1,T_2}}\right)=0. \tag{102}$$

To that end, define a random walk

$$\begin{cases}W((\ell-1)\,T_1)\triangleq\alpha K\\ W(t)=(W(t-1)-C_t+K)^+,\ \forall t>(\ell-1)\,T_1.\end{cases} \tag{103}$$

One can easily verify by definitions (98) and (103) that $\mathcal{B}_{\ell,T_1,T_2}=\{W(t)=0,\text{ for some }t\leq(\ell-1)T_1+T_2\}$.

Recall that we are interested in the within-capacity regime, i.e., $0<K<\mathbb{E}\{C_t\}$. Therefore, $W(t)$ has a negative drift and will return to 0 within a finite number of slots almost surely. This implies

$$\begin{aligned}&\lim_{T_2\to\infty}\Pr\left(\mathcal{B}_{\ell,T_1,T_2}\right)\\ &=\lim_{T_2\to\infty}\Pr\big(\inf\{t\geq(\ell-1)\,T_1+1:W(t)=0\}\\ &\qquad\leq(\ell-1)\,T_1+T_2\big)\qquad(104)\\ &=1.\qquad(105)\end{aligned}$$

As a result, we have (102), which completes the proof. □

With Lemmas 12 to 14, we are ready to prove Lemma 2. We first prove (83) as follows.

Proof: For any arbitrary but fixed T_1, we have

$$\lim_{q\to\infty}\lim_{T\to\infty}p^{\mathrm{RLSC}(q)}_{e,[1,T]}\geq\lim_{q\to\infty}p^{\mathrm{RLSC}(q)}_{e,[1,T_1]}=p^{\mathrm{GMDS}(T_1)}_{e,[1,T_1]} \tag{106}$$

where the inequality is by Lemma 12 and the equality is by Lemma 1. By letting $T_1\to\infty$, we have (83). □

Next, we prove (84).

Proof: For any arbitrary but fixed $T_1>T_2$, we have

$$\begin{aligned}&\lim_{q\to\infty}\lim_{T\to\infty}p^{\mathrm{RLSC}(q)}_{e,[1,T]}\\ &=\lim_{q\to\infty}\lim_{L\to\infty}\frac{1}{L}\sum_{\ell=1}^{L}\frac{1}{T_1}\sum_{t=(\ell-1)T_1+1}^{\ell T_1}\\ &\qquad\Pr\left(\mathbf{s}(t)\text{ is not decodable by time }t+\Delta\right)\qquad(107)\\ &\leq\lim_{q\to\infty}\lim_{L\to\infty}\frac{1}{L}\sum_{\ell=1}^{L}\left(\frac{T_2}{T_1}+\max_{\tau\in[1,T_2]}p^{\mathrm{RLSC}(q)}_{e,[1,T_1-\tau]}+f_{\mathrm{UB}}(T_2,q)\right)\qquad(108)\\ &=\frac{T_2}{T_1}+\max_{\tau\in[1,T_2]}p^{\mathrm{GMDS}(T_1-\tau)}_{e,[1,T_1-\tau]}+\lim_{q\to\infty}f_{\mathrm{UB}}(T_2,q)\qquad(109)\end{aligned}$$

where (107) follows from partitioning the average and summation into sub-intervals of length T_1; (108) follows from applying Lemmas 13 and 14 to each of the sub-interval; (109) follows from first noting that the expression of (108) does not depend on ℓ so we can first remove the averaging operation over ℓ and then we take the limit of $q\to\infty$, which, when combined with Lemma 1, gives us (109).

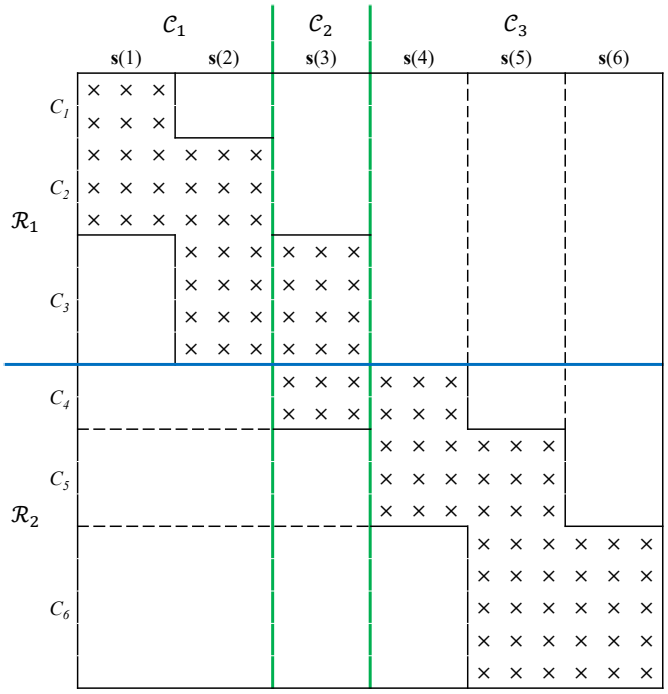

Fig. 13: An example of $\mathbf{H}^{(t_{i^\circ+1})}$ with $N = 8$, $K = 3$, and $\alpha = 1$. The channel realizations are $(C_1, C_2, \cdots, C_6) = (2, 3, 4, 2, 3, 5)$; and the corresponding $I_d(t)$ in (17) are $(1, 1, 0, 1, 1, 0)$. The cross mark "$\times$" shows the non-zero entries. In this case, $t_{i^\circ} = 3$ and $t_{i^\circ+1} = 6$.

Since (109) holds for all finite $T_1 > T_2$, for any fixed finite T_2 we have

$$\lim_{q\to\infty}\lim_{T\to\infty} p_{e,[1,T]}^{\text{RLSC}(q)}$$
$$\le \lim_{T_1\to\infty}\left(\frac{T_2}{T_1} + \max_{\tau\in[1,T_2]} p_{e,[1,T_1-\tau]}^{\text{GMDS}(T_1-\tau)} + \lim_{q\to\infty} f_{\text{UB}}(T_2,q)\right) \tag{110}$$
$$= \lim_{T_1\to\infty} p_{e,[1,T_1]}^{\text{GMDS}(T_1)} + \lim_{q\to\infty} f_{\text{UB}}(T_2,q) \tag{111}$$

where (111) follows from (i) the limit of $p_{e,[1,T]}^{\text{GMDS}(T)}$ exists; and (ii) $\lim_{q\to\infty} f_{\text{UB}}(T_2, q)$ does not depend on the value of T_1. Finally, we note that the limit $\lim_{T_1\to\infty} p_{e,[1,T_1]}^{\text{GMDS}(T_1)}$ does not depend on the T_2 value. Further taking the limit of $T_2 \to \infty$, (111) becomes

$$\lim_{q\to\infty}\lim_{T\to\infty} p_{e,[1,T]}^{\text{RLSC}(q)}$$
$$\le \lim_{T_1\to\infty} p_{e,[1,T_1]}^{\text{GMDS}(T_1)} + \lim_{T_2\to\infty}\lim_{q\to\infty} f_{\text{UB}}(T_2,q). \tag{112}$$

By Lemma 14, we have proved (84). □

Appendix C
Proofs of Error Events

A. Proof of Proposition 1

We prove Proposition 1 by induction. Proposition 1 holds for $i_0 = -1$ since we only start sending $\mathbf{s}(t)$ for $t \ge 1$ and any $\mathbf{s}(t)$ with $t \le t_{i_0+1} = 0$ is automatically considered to be decodable.

Suppose Proposition 1 holds for all $i_0 < i^\circ$, where i° is a non-negative integer. We now consider $i_0 = i^\circ$ and would like to decode the slots $\{\mathbf{s}(t) : t \in (t_{i^\circ}, t_{i^\circ+1}]\}$ at time $t_{i^\circ+1}$. Considering the cumulative receiver matrix $\mathbf{H}^{(t_{i^\circ+1})}$, we divide the rows of $\mathbf{H}^{(t_{i^\circ+1})}$ into two groups

$$\mathcal{R}_1 : \text{rows corresponding to } \{\mathbf{y}(t) : t \in (0, t_{i^\circ}]\} \tag{113}$$
$$\mathcal{R}_2 : \text{rows corresponding to } \{\mathbf{y}(t) : t \in (t_{i^\circ}, t_{i^\circ+1}]\} \tag{114}$$

and divide the columns of $\mathbf{H}^{(t_{i^\circ+1})}$ into three groups

$$\mathcal{C}_1 : \text{columns corresponding to } \{\mathbf{s}(t) : t \in (0, t_{i^\circ} - \alpha]\} \tag{115}$$
$$\mathcal{C}_2 : \text{columns corresponding to } \{\mathbf{s}(t) : t \in (t_{i^\circ} - \alpha, t_{i^\circ}]\} \tag{116}$$
$$\mathcal{C}_3 : \text{columns corresponding to } \{\mathbf{s}(t) : t \in (t_{i^\circ}, t_{i^\circ+1}]\}. \tag{117}$$

See Fig. 13 for illustration. Our decodability/achievability scheme uses only the $\mathcal{R}_2$ rows and ignore the $\mathcal{R}_1$ rows. Since the memory length is α, the intersection of $\mathcal{C}_1$ columns and $\mathcal{R}_2$ rows must be all-zero.

Per our induction assumption, all symbols corresponding to $\mathcal{C}_2$ (or use the term "$\mathcal{C}_2$-symbols" as shorthand) can be successfully decoded using the $\mathcal{R}_1$ rows. We can thus remove the impact of $\mathcal{C}_2$-symbols from the equations corresponding to the $\mathcal{R}_2$ rows. In the end, we need to focus only on the intersection of the $\mathcal{C}_3$ columns and $\mathcal{R}_2$ rows. See Fig. 13. It is as if the timeline has been reset, with t_{i° being shifted back to 0. As a result, to prove that Proposition 1 holds for general i_0, we only need to prove that Proposition 1 holds for $i_0 = 0$.

We now consider the cumulative receiver matrix $\mathbf{H}^{(t_1)}$, where t_1 is the first time $I_d(t)$ hits back 0 after time 0. For ease of exposition, we denote $\mathbf{H}^{(t_1)}$ by $\mathbf{A}$. We then introduce a subroutine called *Forward Diagonal Labeling* (FDL), which is described by the following pseudocode.

Procedure 1 Forward Diagonal Labeling (FDL)

Input: A matrix $\mathbf{A}$ with the number of rows and columns denoted by rows($\mathbf{A}$) and cols($\mathbf{A}$), respectively. We use $A_{x,y}$ to denote the (x, y)-th entry of $\mathbf{A}$.
Initialization: $i = j = 0$, $S_L = \emptyset$
while $i <$ rows($\mathbf{A}$) **do**
  $i \leftarrow i + 1$
  if $\{j' : j < j' \le \text{cols}(\mathbf{A}) \text{ and } A_{i,j'} \ne 0\} \ne \emptyset$ **then**
    $j \leftarrow \min\{j' : j < j' \le \text{cols}(\mathbf{A}) \text{ and } A_{i,j'} \ne 0\}$
    $S_L = S_L \cup \{(i, j)\}$.
  else
    break
  end if
end while
Output: S_L

See Fig. 14 for illustration. In Fig. 14, we consider an example of $N = 8$, $K = 3$, and $\alpha = 1$. The first 7 channel realizations are $(C_1, C_2, \cdots, C_7) = (1, 2, 1, 0, 3, 2, 8)$. The corresponding $(I_d(1), I_d(2), \cdots, I_d(7))$ are $(2, 3, 4, 4, 3, 4, 0)$ and the intermediate values $\widehat{I}_d(t)$ in (16) are $(2, 3, 5, 6, 3, 4, 0)$. The first 0-hitting time is $t_1 = 7$. Since the ceiling $\zeta =$

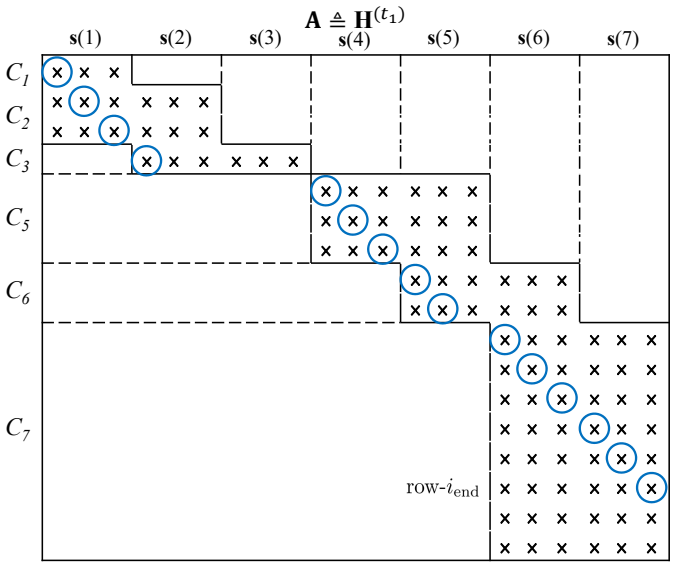

Fig. 14: An example of $\mathbf{A} \triangleq \mathbf{H}^{(t_1)}$ with $N = 8$, $K = 3$, and $\alpha = 1$. The channel realizations are $(C_1, C_2, \cdots, C_7) = (1, 2, 1, 0, 3, 2, 8)$; the corresponding $I_d(t)$ in (17) are $(2, 3, 4, 4, 3, 4, 0)$; the corresponding $\widehat{I}_d(t)$ in (16) are $(2, 3, 5, 6, 3, 4, 0)$; the 0-hitting time is $t_1 = 7$; and the ζ-hitting time are $\tau_1 = 3$, $\tau_2 = 4$ and $\tau_3 = 6$. The cross mark "$\times$" shows the non-zero entries. The blue circles denote the output set S_L of the FDL subroutine.

$\alpha K + 1 = 4$, there are three ζ-hitting times $\tau_1 = 3$, $\tau_2 = 4$ and $\tau_3 = 6$ in the interval (t_0, t_1). The blue circles denote the S_L set, the output of FDL. The intuition of the FDL subroutine is to label "the non-zero diagonal elements" of $\mathbf{A}$. Whenever the diagonal labeling of FDL hits a zero-entry, the subroutine shifts the diagonal line to the right and continues such labeling. The labeling stops once it hits either the row limit of the matrix (Line 3), or when there is no non-zero entries on the right of the next line of the latest element (i, j) of S_L (Line 8). That is, even if we allow for shifting the diagonal line further to the right, we still cannot proceed any further.

Using the FDL set S_L, we can define the following functions. Recall that $\mathbf{A} \triangleq \mathbf{H}^{(t_1)}$ has exactly $\text{cols}(\mathbf{A}) = t_1 K$ columns, which are labeled as 1 to $t_1 K$, respectively, and has exactly $\text{rows}(\mathbf{A}) = \text{rows}\left(\mathbf{H}^{(t_1)}\right) = \sum_{t'=1}^{t_1} |\mathbf{y}(t')| = \sum_{t'=1}^{t_1} C_{t'}$ rows, which are labeled as 1 to $\sum_{t'=1}^{t_1} C_{t'}$, respectively. Define $j_{\mathsf{FDL}}(0) \triangleq 0$. For any $i \in [1, \text{rows}(\mathbf{A})]$, define

$$j_{\mathsf{FDL}}(i) \triangleq \begin{cases} j & \text{if } (i, j) \in S_L, \\ \infty & \text{if no such } j \text{ satisfying } (i, j) \in S_L. \end{cases} \tag{118}$$

Note that because of the construction of FDL, we either have a unique j such that $(i, j) \in S_L$ or no such j. As a result, $j_{\mathsf{FDL}}(i)$ is well-defined and is essentially a mapping from i to j based on S_L. Also, for any i, define the *rightmost non-zero element* $j_{\text{nz.max}}(i)$ as follows

$$j_{\text{nz.max}}(i) \triangleq \max\{j \in [1, t_1 K] : A_{i,j} \neq 0\} \tag{119}$$

where we use the special convention $\max \emptyset = \infty$. For any arbitrary row index i, according to the structure of the cumulative generator matrix in Fig. 2, we have

$$j_{\text{nz.max}}(i) = tK \tag{120}$$

where t is the unique value that satisfies

$$\sum_{t'=1}^{t-1} C_{t'} < i \leq \sum_{t'=1}^{t} C_{t'}. \tag{121}$$

The intuition of (120) and (121) is straightforward. For the row corresponding to $\mathbf{y}(t)$, see (121), the largest non-zero entry must be at the (tK)-th because the latest source symbol vector that can participate in $\mathbf{y}(t)$ is $\mathbf{s}(t)$. We thus have (120). In both (118) and (119), we use ∞ to represent the non-existence case. Finally, define the very last row of those entries in S_L by

$$i_{\text{end}} \triangleq \max\{i : (i, j) \in S_L\}. \tag{122}$$

With the above definitions of S_L, $j_{\mathsf{FDL}}(\cdot)$, $j_{\text{nz.max}}(\cdot)$, and i_{end}, we can now establish the relationship between $I_d(t)$ and the structure of the cumulative receiver matrix $\mathbf{A} = \mathbf{H}^{(t_1)}$.

Lemma 15. *Consider any arbitrarily given $t \in [1, t_1)$, the intermediate step of the information debt $\widehat{I}_d(t)$ defined in* (16) *satisfies*

$$\widehat{I}_d(t) = tK - \max\left((t - \alpha - 1)K, j_{\mathsf{FDL}}\left(\sum_{t'=1}^{t} C_{t'}\right)\right) > 0. \tag{123}$$

Furthermore, the following (in)equalities always hold:

$$\sum_{t'=1}^{t_1 - 1} C_{t'} < i_{\text{end}} \leq \sum_{t'=1}^{t_1} C_{t'}; \tag{124}$$

$$j_{\mathsf{FDL}}(i_{\text{end}}) = j_{\text{nz.max}}(i_{\text{end}}) = t_1 K < \infty; \tag{125}$$

$$j_{\mathsf{FDL}}(i) < j_{\text{nz.max}}(i) \leq t_1 K, \quad \forall i \in [1, i_{\text{end}}). \tag{126}$$

The proof of this lemma is a straightforward exercise of mathematical induction but the detailed steps are rather technical. We thus relegate the proof to Appendix D. This lemma is the most critical part of the proofs of Propositions 1 and 2. In the sequel, we further elaborate the physical meanings behind this lemma.

Ineqs. (124) and (125) form a pair and characterize how the FDL subroutine ends when applied to the matrix $\mathbf{A} = \mathbf{H}^{(t_1)}$. Specifically, (124) implies that when the FDL subroutine ends, the last row (the i_{end}-th row) must correspond to $\mathbf{y}(t_1)$, the rows received during the last time instant t_1. Hence, by (120) and (121), we immediately have $j_{\text{nz.max}}(i_{\text{end}}) = t_1 K$. The equality $j_{\mathsf{FDL}}(i_{\text{end}}) = j_{\text{nz.max}}(i_{\text{end}})$ in (125) further implies that the FDL subroutine ends because the corresponding $j_{\mathsf{FDL}}(i)$ value finally hits the last column. In our example of Fig. 14, the very last blue circle indeed corresponds to a row of $\mathbf{y}(7)$, since $t_1 = 7$, and the last column (the $t_1 K = 21$-th column). This phenomenon is characterized rigorously by (124) and (125).

Ineq. (126) implies that before the last row i_{end}, the FDL subroutine will never hit the "max-boundary" of the non-zero entries $j_{\text{nz.max}}$. Such a phenomenon can be observed in Fig. 14, in which there is always at least one cross mark on the right of each blue circle, except for the very last blue circle that coincides with the very last cross mark.

The physical meaning of (123) is more subtle. For any $t \in [1, t_1)$, we discuss the case when $C_t > 0$ and omit the corner case discussion of $C_t = 0$. Suppose $C_t > 0$ for an arbitrarily given t. Eq. (123) can be reduced to

$$\widehat{I}_d(t) = tK - j_{\mathsf{FDL}}(\sum_{t'=1}^{t} C_{t'}) \tag{127}$$

$$= j_{\text{nz.max}}(\sum_{t'=1}^{t} C_{t'}) - j_{\mathsf{FDL}}(\sum_{t'=1}^{t} C_{t'}) > 0. \tag{128}$$

The reason is that with $C_t > 0$, the rows with their row indices inside $(\sum_{t'=1}^{t-1} C_{t'}, \sum_{t'=1}^{t} C_{t'}]$ correspond to the observation $\mathbf{y}(t)$. Since the $\mathbf{y}(t)$ is generated by $\mathbf{s}(t-\alpha)$ to $\mathbf{s}(t)$, see (3), the very first non-zero column of the rows of $\mathbf{y}(t)$ must be the $((t-\alpha-1)K+1)$-th column. Since $j_{\mathsf{FDL}}(\cdot)$ always outputs a non-zero entry, we thus have

$$j_{\mathsf{FDL}}(\sum_{t'=1}^{t} C_{t'}) \geq (t-\alpha-1)K + 1. \tag{129}$$

Eq. (123) thus becomes (127). The next equality (128) then follows from (120) and (121). The new (128) (focusing exclusively on the case of $C_t > 0$) shows that when focusing on the last row of each $\mathbf{y}(t)$, the value of $\widehat{I}_d(t)$ is exactly the distance between the max-boundary and location of the FDL entry $j_{\mathsf{FDL}}(\cdot)$. Such a phenomenon can be observed in Fig. 14. For example, consider the time slot $t = 5$. Since $(C_1, C_2, C_3, C_4, C_5) = (1, 2, 1, 0, 3)$, the 7-th row corresponds to the last observed symbol in time 5. In this example, we have $j_{\mathsf{FDL}}(7) = 12$ and $j_{\text{nz.max}}(7) = 15$. Their difference is indeed $\widehat{I}_d(5) = 3$ as predicted by (128).

Remark 7: The condition "$C_t > 0$" is to ensure that the received symbols $\mathbf{y}(t)$ are not empty at the time of interest t. As for the case of $C_t = 0$, i.e., no received symbol $\mathbf{y}(t)$ at time t, the corresponding rows become "hidden/invisible" from the cumulative receiver matrix $\mathbf{H}^{(t)}$. See $t = 4$ in Fig. 14. That is why we need an additional $\max$ operation in (123).

An immediate result of Lemma 15 is

Lemma 16. *If $I_d(t)$ does not hit ζ during the interval $(0, t_1]$, i.e., $\{j : \tau_j \in (0, t_1]\} = \emptyset$, the FDL set S_L labels an uninterrupted diagonal line connecting the two locations $(1, 1)$ and (t_1K, t_1K). That is,*

$$S_L = (1,1) \dashrightarrow (t_1K, t_1K)$$
$$\triangleq \{(1,1), (2,2), \cdots, (t_1K, t_1K)\}. \tag{130}$$

Otherwise, let τ_{j^*} denote the last *ζ-hitting time during the interval $(0, t_1]$. The following uninterrupted diagonal line connecting $(i_{\text{start}}, j_{\text{start}})$ and $(i_{\text{end}}, j_{\text{end}})$ is part of the FDL output set S_L, where*

$$(i_{\text{start}}, j_{\text{start}}) \dashrightarrow (i_{\text{end}}, j_{\text{end}})$$
$$\triangleq \{(i_{\text{start}}, j_{\text{start}}), (i_{\text{start}}+1, j_{\text{start}}+1), \cdots, (i_{\text{end}}, j_{\text{end}})\} \tag{131}$$

and

$$i_{\text{start}} = \textstyle\sum_{t'=1}^{\tau_{j^*}} C_{t'} + 1;$$
$$j_{\text{start}} = (\tau_{j^*} - \alpha)\, K + 1;$$
$$j_{\text{end}} = t_1 K;$$
$$i_{\text{end}} = \max\{i : (i,j) \in S_L\} = i_{\text{start}} + j_{\text{end}} - j_{\text{start}}. \tag{132}$$

Note that the term i_{end} was first defined in (122), which is the last row index in S_L. The second half of Lemma 16 thus describes the *last* uninterrupted diagonal line segment in S_L, for which the starting point must be the first observation (row) after time τ_{j^*}, see i_{start}, and its first non-zero element, see j_{start}, and the ending point must be the very last point in S_L. The phenomenon described in Lemma 16 can be observed in Fig. 14. Specifically, in the example of Fig. 14, we have $\tau_{j^*} = \tau_3 = 6$. Lemma 16 predicts the last uninterrupted diagonal line segment to be the one connecting $(i_{\text{start}}, j_{\text{start}}) = (10, 16)$ and $(i_{\text{end}}, j_{\text{end}}) = (15, 21)$, which is consistent with the observation in Fig. 14. We leave its proof to Appendix E.

The rest of the decodability proof is straightforward. If $\{j : \tau_j \in (0, t_1]\}$ is empty, then there are exactly t_1K rows and t_1K columns in the diagonal line defined in (130). We consider the submatrix of $\mathbf{H}^{(t_1)}$ induced by the rows of 1 to t_1K and denote it by $\mathbf{B}$. By **GMDS**, $\mathbf{B}$ is invertible and we can decode all symbols in $\{\mathbf{s}(t) : t \in (0, t_1]\}$ using the observations corresponding to rows 1 to t_1K.

Similarly, if $\{j : \tau_j \in (0, t_1]\}$ is not empty, by the expressions of j_{start} and j_{end} in (132) there are exactly $(t_1 - (\tau_{j^*} - \alpha))K$ rows and $(t_1 - (\tau_{j^*} - \alpha))K$ columns in the diagonal line defined in (132). We consider the submatrix of $\mathbf{H}^{(t_1)}$ induced by the rows of i_{start} to i_{end} defined in (132) and denote it by $\mathbf{B}$. Because the memory length is α, the j_{start} in (132) is the first non-zero entry for the i_{start}-th row in $\mathbf{H}^{(t_1)}$. Therefore, all the columns of $\mathbf{B}$ that is strictly before j_{start} must be all-zero. As a result, the last uninterrupted diagonal line in (132) spans across all non-zero columns of $\mathbf{B}$. By **GMDS**, we can decode all symbols $\{\mathbf{s}(t) : t \in (\tau_{j^*} - \alpha, t_1]\}$ (those symbols corresponding to all the non-zero columns in $\mathbf{B}$) using the observations corresponding to rows $\sum_{t'=1}^{\tau_{j^*}} C_{t'} + 1$ to $\sum_{t'=1}^{\tau_{j^*}} C_{t'} + (t_1 - (\tau_{j^*} - \alpha))\, K$ (all the rows in $\mathbf{B}$). The proof of Proposition 1 is complete. In our running example of Fig. 14, we can decode $\mathbf{s}(6)$ and $\mathbf{s}(7)$, i.e., columns 16 to 21, using rows 10 to 15 since there exists an uninterrupted diagonal line connecting locations $(10, 16)$ to $(15, 21)$.

B. Proof of Proposition 2

The proof of Proposition 2 is much more involved since we have to prove that some $\mathbf{s}(t)$ is not decodable no matter how many observations $\mathbf{y}(t)$ (and thus how many rows) we have. To that end, we first perform the following set of lossless simplifications.

1) A Series of Lossless Simplifications:

Lemma 17. *Proposition 2 holds for general $i_0 \geq 0$ if and only if Proposition 2 holds for $i_0 = 0$.*

Proof: The proof is similar to the arguments used in the beginning of Appendix C-A. For any given $i_0 > 0$, we first

rename i_0 as i° so that it is compatible to the discussion in Appendix C-A. Suppose an arbitrarily given decoder would like to start decoding those $\mathbf{s}(t)$ in $(t_{i^\circ}, t_{i^\circ+1}]$ by some large time T satisfying $T \geq t_{i^\circ+1}$. We again divide the rows into two groups $\mathcal{R}_1$ and $\mathcal{R}_2$, where $\mathcal{R}_1$ is defined as (113) but we lightly modified $\mathcal{R}_2$ in (114) to the following definition:

$$\mathcal{R}_2: \text{ rows corresponding to } \{\mathbf{y}(t) : t \in (t_{i^\circ}, T]\}. \quad (133)$$

That is, we expand the $\mathcal{R}_2$ set since we are trying to decode at time T instead of at time $t_{i^\circ+1}$. Similarly, we divide the columns into three groups $\mathcal{C}_1$, $\mathcal{C}_2$, and $\mathcal{C}_3$, where $\mathcal{C}_1$ and $\mathcal{C}_2$ are defined as (115) and (116), respectively, but we lightly modified $\mathcal{C}_3$ in (117) to the following definition:

$$\mathcal{C}_3: \text{ columns corresponding to } \{\mathbf{s}(t) : t \in (t_{i^\circ}, T]\} \quad (134)$$

where we again expand the column set $\mathcal{C}_3$ because of the focus on time T. See Fig. 13 for illustration.

Having proved that Proposition 1 holds for general $i^\circ \geq 0$, all the vectors in the time slots $(t_{i^\circ} - \alpha, t_{i^\circ}]$ must be decodable by time t_{i°. As a result, when decoding any $t \in (t_{i^\circ}, t_{i^\circ+1}]$ at time T, we can first decode those symbols in $\{\mathbf{s}(t') : t' \in (t_{i^\circ} - \alpha, t_{i^\circ}]\}$. We then note that removing the impact of those symbols is equivalent to removing the corresponding columns in $\mathcal{C}_2$. We now analyze the intersection of $(\mathcal{R}_1, \mathcal{R}_2)$ rows and $(\mathcal{C}_1, \mathcal{C}_3)$ columns.

Since the memory length is $\alpha < \infty$, the intersection between the $\mathcal{R}_2$ rows and $\mathcal{C}_1$ columns must be all-zero. Also, since we consider causal systems, the intersection between the $\mathcal{R}_1$ rows and the $\mathcal{C}_3$ columns must be all-zero. As a result, the only intersection is between $\mathcal{R}_1$ and $\mathcal{C}_1$ and between $\mathcal{R}_2$ and $\mathcal{C}_3$. After removing the $\mathcal{C}_2$ columns, the cumulative receiver matrix $\mathbf{H}^{(T)}$ is thus separated into a block diagonal form

$$\begin{bmatrix} \mathbf{H}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_2 \end{bmatrix} \quad (135)$$

where $\mathbf{H}_1$ (resp. $\mathbf{H}_2$) is the intersection of $(\mathcal{R}_1, \mathcal{C}_1)$ (resp. $(\mathcal{R}_2, \mathcal{C}_3)$). Therefore, the decoder can decode $\mathbf{s}(t)$, $t \in (t_{i^\circ}, t_{i^\circ+1}]$ using both $\mathcal{R}_1$ and $\mathcal{R}_2$ if and only if it can decode those $\mathbf{s}(t)$ using exclusively the $\mathcal{R}_2$ rows. However, if we focus only on decoding using $\mathcal{R}_2$ rows, it is as if the timeline has been reset, with t_{i° being shifted back to 0. As a result, to prove Proposition 2 holds for general i°, we only need to prove Proposition 2 holds for $i^\circ = 0$. □

Focusing on $i_0 = 0$, we now prove

Lemma 18. *For any $T \geq t_1$, any $t \in (0, t_1 - \alpha]$ and $k \in [1, K]$, symbol $s_k(t)$ is decodable by time T if and only if $s_k(t)$ is decodable by t_1.*

Proof: For any $T \geq t_1$, by Proposition 1 the vectors $\{\mathbf{s}(t') : t' \in (t_1 - \alpha, t_1]\}$ are decodable by time t_1 and are thus decodable by time T. Using the decoded vectors $\{\mathbf{s}(t') : t' \in (t_1 - \alpha, t_1]\}$ as side information, we can cancel their impact/interference in the observations, which effectively removes the corresponding columns from $\mathbf{H}^{(T)}$. Since the memory length is $\alpha < \infty$, the intersection between the rows corresponding to the observations after time t_1, $\{\mathbf{y}(t') : t' \in (t_1, T]\}$, and the columns corresponding to $\{\mathbf{s}(t') : t' \in (0, t_1 - \alpha]\}$ is all-zero. Besides, since the system is causal, the intersection between the rows corresponding to the observations until time t_1, $\{\mathbf{y}(t') : t' \in (0, t_1]\}$, and the columns corresponding to $\{\mathbf{s}(t') : t' \in (t_1, T]\}$ is all-zero. As a result, after removing the columns corresponding to $\{\mathbf{s}(t') : t' \in (t_1 - \alpha, t_1]\}$, the cumulative receiver matrix $\mathbf{H}^{(T)}$ is separated into a block diagonal form as in (135). Therefore, the decoder can decode $s_k(t)$, $t \in (0, t_1 - \alpha]$ using $\mathbf{y}(t')$, $t' \in [1, T]$ if and only if it can decode $s_k(t)$ using only the rows corresponding to $\{\mathbf{y}(t') : t' \in [1, t_1]\}$. □

Reusing the definitions of $\mathbf{A} = \mathbf{H}^{(t_1)}$ and i_{end} in the proof of Proposition 1, we can further strengthen Lemma 18 to

Lemma 19. *The location vector of $s_k(t)$ is in $\mathbf{A} = \mathbf{H}^{(t_1)}$ if and only if the location vector $s_k(t)$ is in the row space of the first i_{end} rows of $\mathbf{A}$. That is, we can discard the last $(\sum_{t'=1}^{t_1} C_{t'} - i_{\text{end}})$ rows when trying to decode $s_k(t)$ from $\mathbf{H}^{(t_1)}$.*

Proof: We notice that the decodability proof in the end of Appendix C-A shows that we can use the first i_{end} rows of matrix $\mathbf{A} = \mathbf{H}^{(t_1)}$ to decode all the symbols $s_k(t)$ satisfying $k \in [1, K]$ and $t \in (t_1 - \alpha, t_1]$. We then observe that all the non-zero entries of the last $(\sum_{t'=1}^{t_1} C_{t'} - i_{\text{end}})$ rows correspond to the same set of (already decodable) symbols $s_k(t)$, $t \in (t_1 - \alpha, t_1]$, $k \in [1, K]$. Therefore, the last $(\sum_{t'=1}^{t_1} C_{t'} - i_{\text{end}})$ rows do not provide any additional information when compared to the first i_{end} rows. The proof of this lemma is complete. □

The main proof of Proposition 2 will be built upon the lossless simplification Lemmas 17 to 19.

2) Main Proof of Proposition 2: We note that Proposition 2 focuses exclusively on the case that there exists $\tau_j \in (0, t_1]$. In this case, we first apply the Forward Diagonal Labeling (FDL) subroutine in Appendix C-A to obtain the label set S_L and the last row index i_{end} value. Define $\mathbf{A}_{i_{\text{end}}}$ as the submatrix induced by the first i_{end} rows of $\mathbf{A} = \mathbf{H}^{(t_1)}$, i.e., removing the last $(\sum_{t'=1}^{t_1} C_{t'} - i_{\text{end}})$ rows of $\mathbf{A}$. See Fig. 15 for illustration. We also construct another set $\widetilde{S}_L$ from the FDL output set S_L as follows.

$$\widetilde{S}_L = \left\{ (i, j+1) : i \leq \sum_{t'=1}^{\tau_{j^*}} C_{t'}, \text{ and } (i,j) \in S_L \right\} \cup \left\{ (i,j) : i > \sum_{t'=1}^{\tau_{j^*}} C_{t'}, \text{ and } (i,j) \in S_L \right\} \quad (136)$$

That is, $(i, j+1)$ is added to the new set if $(i, j) \in S_L$ and $i \leq \sum_{t'=1}^{\tau_{j^*}} C_{t'}$ while the same (i, j) is inserted in $\widetilde{S}_L$ if $(i, j) \in S_L$ and $i > \sum_{t'=1}^{\tau_{j^*}} C_{t'}$. In our example of Fig. 15, we use the green circles to represent the new set $\widetilde{S}_L$. We then have

Lemma 20. *$A_{i,j} \neq 0$ for all $(i, j) \in \widetilde{S}_L$. Furthermore, all elements in $\widetilde{S}_L$ have distinct row indices and have distinct column indices.*

Per the construction of the FDL routine, $A_{i,j} \neq 0$ for all $(i, j) \in S_L$ and each element of S_L has distinct row and column indices. Note that $\widetilde{S}_L$ is derived from S_L by shifting the upper half of S_L by one location to the right, see Fig. 15. Lemma 20 then shows that the new set $\widetilde{S}_L$ (similar to the original set S_L) still contains diagonal elements (no two j_1

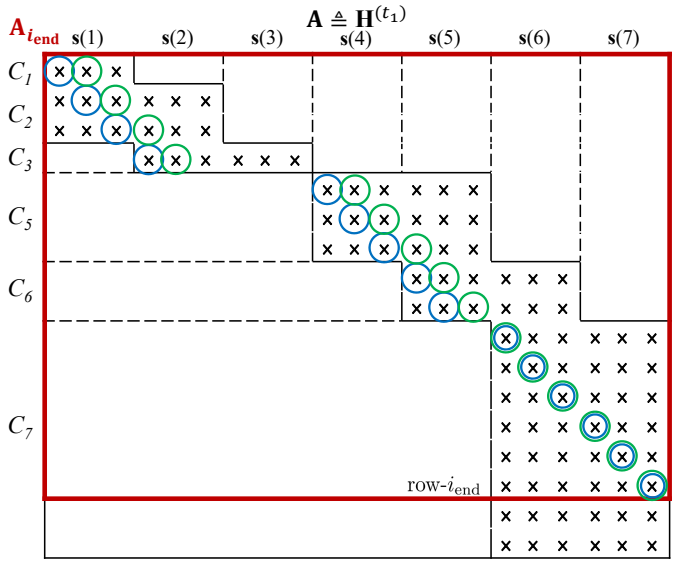

Fig. 15: The same example of Fig. 14. The red box indicates the submatrix $\mathbf{A}_{i_{\text{end}}}$ being considered; the blue circles denote the output set S_L of the FDL subroutine; and the green circles represent the new set $\widetilde{S}_L$.

and j_2 are identical) and all those diagonal elements are non-zero.

The proof of Lemma 20 is technical and is relegated to Appendix F. The intuition is, however, quite straightforward and is explained as follows. Eq. (126) shows that $j_{\mathsf{FDL}}(i) < j_{\text{nz.max}}(i)$. That is, there is at least one more non-zero entry on the immediate right of $(i, j_{\mathsf{FDL}}(i)) \in S_L$. As a result, the shifted version of $(i, j_{\mathsf{FDL}}(i)+1) \in \widetilde{S}_L$ will still be non-zero and form distinct diagonal elements as well.

We are now ready to prove the following lemma, which, when combined with Lemmas 17 to 20, leads to Proposition 2.

Lemma 21. *Suppose there exists a $\tau_j \in (0, t_1)$. Define τ_{j^*} as the largest τ_j within the interval $(0, t_1)$. For any $t \in (0, \tau_{j^*} - \alpha]$ and any $k \in [1, K]$, the location vector of $s_k(t)$ is not in the row space of $\mathbf{A}_{i_{\text{end}}}$.*

Proof: For any fixed $t \in (0, \tau_{j^*} - \alpha] \subset (0, t_1)$ and any $k \in [1, K]$, instead of directly showing that the location vector of $s_k(t)$ is not in the row space of $\mathbf{A}_{i_{\text{end}}}$, we show that for any i_{end}-dimensional observation vector $\mathbf{y}_{i_{\text{end}}}$, the following equation

$$\mathbf{A}_{i_{\text{end}}} \mathbf{s}_1^{t_1} = \mathbf{y}_{i_{\text{end}}} \tag{137}$$

has two root vectors $\widehat{\mathbf{s}}_1^{t_1}$ and $\widetilde{\mathbf{s}}_1^{t_1}$ satisfying $\widehat{s}_k(t) \neq \widetilde{s}_k(t)$, where $\widehat{s}_k(t)$ (resp. $\widetilde{s}_k(t)$) is the value corresponding to the symbol of interest $s_k(t)$ in the vector $\widehat{\mathbf{s}}_1^{t_1}$ (resp. $\widetilde{\mathbf{s}}_1^{t_1}$). In other words, the $s_k(t)$ value cannot be uniquely determined by (137) and the proof of Lemma 21 is complete.

To that end, define $j^\circ = (t-1)K + k$ as the column index of $s_k(t)$ being considered. We can then isolate the impact of the j°-th column and write

$$(\mathbf{A}_{i_{\text{end}}} \backslash \mathbf{a}_{j^\circ}) \cdot (\mathbf{s}_1^{t_1} \backslash s_k(t)) = \mathbf{y}_{i_{\text{end}}} - s_k(t) \cdot \mathbf{a}_{j^\circ} \tag{138}$$

where $(\mathbf{A}_{i_{\text{end}}} \backslash \mathbf{a}_{j^\circ})$ is the matrix after removing the j°-th column from $\mathbf{A}_{i_{\text{end}}}$ and $(\mathbf{s}_1^{t_1} \backslash s_k(t))$ is the cumulative vector after removing the entry corresponding to $s_k(t)$. In the sequel, we prove that $(\mathbf{A}_{i_{\text{end}}} \backslash \mathbf{a}_{j^\circ})$ is always of full row rank regardless how we choose $t \in (0, \tau_{j^*} - \alpha]$ and $k \in [1, K]$, which implies that for any arbitrary $s_k(t)$ value satisfying $t \in (0, \tau_{j^*} - \alpha]$ and $k \in [1, K]$ we can always find a companying $(\mathbf{s}_1^{t_1} \backslash s_k(t))$ that satisfies (138), and hence $s_k(t)$ is not decodable.

Consider two cases that depend on the value of $j^\circ = (t-1)K + k$. Case 1: Suppose there is no i such that $(i, j^\circ) \in S_L$. In this case, it is straightforward that $(\mathbf{A}_{i_{\text{end}}} \backslash \mathbf{a}_{j^\circ})$ is of full row rank since all the (i, j) entries of S_L are still in the matrix $(\mathbf{A}_{i_{\text{end}}} \backslash \mathbf{a}_{j^\circ})$ and the matrix is of full row rank due to **GMDS**.

Case 2: Suppose there exists an i° such that $(i^\circ, j^\circ) \in S_L$. We now argue that the corresponding i° must satisfy $i^\circ \leq \sum_{t'=1}^{\tau_{j^*}} C_{t'}$. Suppose not, the i°-th row must corresponding to an observation $\mathbf{y}(t')$ satisfying $t' > \tau_{j^*}$. Because we only have finite memory length α, the j°-th position, which is non-zero per our construction, must satisfy $j^\circ \geq (\tau_{j^*} - \alpha)K + 1$. However, this contradicts the construction $j^\circ = (t-1)K + k \leq (\tau_{j^*} - \alpha)K$ since we focus on $t \in (0, \tau_{j^*} - \alpha]$ and $k \in [1, K]$. By contradiction, we must have $i^\circ \leq \sum_{t'=1}^{\tau_{j^*}} C_{t'}$.

We now "stitch" together the original set S_L and the shifted set $\widetilde{S}_L$ in (136) in the following way. Specifically, we define $\widehat{S} = \{(i,j) \in S_L \text{ and } j < j^\circ\} \cup \left\{(i,j) \in \widetilde{S}_L \text{ and } j > j^\circ\right\}$. By this definition, we quickly have (i) the set $\widehat{S}$ does not contain any entry that is in the j°-th column of $\mathbf{A}_{i_{\text{end}}}$; (ii) all the entries of $\mathbf{A}_{i_{\text{end}}}$ that correspond to $\widehat{S}$ are non-zero since all the entries of $\mathbf{A}_{i_{\text{end}}}$ corresponding to S_L (resp. $\widetilde{S}_L$) are non-zero; and (iii) all the row and column indices of $\widehat{S}$ are unique due to Lemma 20.

Finally, we also have (iv) $\left|\widehat{S}\right| = \text{rows}(\mathbf{A}_{i_{\text{end}}}) = i_{\text{end}}$. That is, every row of $\mathbf{A}_{i_{\text{end}}}$ has exactly one entry in $\widehat{S}$. The reason is that in Case 2 we assume there exists an i° such that $(i^\circ, j^\circ) \in S_L$ and $i^\circ \leq \sum_{t'=1}^{\tau_{j^*}} C_{t'}$. Since $\widetilde{S}_L$ consists of the elements on the right of S_L for any row $i^\circ \leq \sum_{t'=1}^{\tau_{j^*}} C_{t'}$, we have $(i^\circ, j^\circ) \notin \widetilde{S}_L$. As a result, by stitching half of S_L and half of $\widetilde{S}_L$, we ensure that the new set $\widehat{S}$ has the same number of rows as both S_L and $\widetilde{S}_L$, which is $\text{rows}(\mathbf{A}_{i_{\text{end}}}) = i_{\text{end}}$.

By properties (ii) to (iv) and **GMDS**, the submatrix of $\mathbf{A}_{i_{\text{end}}}$ induced by $\widehat{S}$ is of full row rank. Therefore the$(\mathbf{A}_{i_{\text{end}}} \backslash \mathbf{a}_{j^\circ})$ must of full row rank due to property (i) of $\widehat{S}$. Hence the value of $s_k(t)$ cannot be uniquely determined by (138). This implies that $s_k(t)$ cannot be determined uniquely by the observation $\mathbf{y}_{i_{\text{end}}}$. □

Note that Lemma 21 immediately implies Proposition 2 and Lemma 3. The proof is complete.

Appendix D
Proof of Lemma 15

The proof of Lemma 15 involves careful analysis of the results of FDL, thus see Fig. 14 for reference. While the intuition is relatively straightforward as discussed right after Lemma 15, the proof involves very technical application of mathematical induction that can only be rigorously stated via detailed/unambiguous definitions and case studies. An interested reader is encouraged to construct additional examples similar to Fig. 14 and use them to examine the statements of

Lemma 15 and gain further intuition before delving into the detailed proofs in this appendix.

First, we prove (123) by induction. Substituting (17) in (16), $\widehat{I}_d(t)$ can be expressed as

$$\widehat{I}_d(t) = \Big(K - C_t + \min\Big(\widehat{I}_d(t-1), \alpha K\Big)\Big)^+. \tag{139}$$

We now prove (123) for the case of $t = 1 < t_1$. Note that the condition $1 < t_1$ implies $\widehat{I}_d(1) > 0$ since if $\widehat{I}_d(1) = 0$, we would have $I_d(1) = 0$ and the first 0-hitting time becomes $t_1 = 1$. Recall that in every time slot $t \geq 1$, the number of incoming source symbols is K, and C_t coded symbols are successfully received at the destination. This implies that the cumulative receiver matrix $\mathbf{H}^{(t)}$ gains K columns and C_t rows when compared to $\mathbf{H}^{(t-1)}$. For $t = 1$, $\mathbf{H}^{(1)}$ is a C_1-by-K matrix and $\widehat{I}_d(t-1) = \widehat{I}_d(0) = 0$. By (139) and because $t = 1 < t_1$ implies $\widehat{I}_d(1) > 0$, we have

$$\widehat{I}_d(1) = (K - C_1)^+ > 0 \iff \widehat{I}_d(1) = K - C_1 > 0.$$

When $t = 1$, the right hand side of (123) becomes

$$1 \cdot K - \max\left(-\alpha K,\ j_{\mathsf{FDL}}(C_1)\right) = K - C_1. \tag{140}$$

where the equality follows from $j_{\mathsf{FDL}}(C_1) = C_1 \geq 0$ since the cumulative receiver matrix $\mathbf{H}^{(1)}$ at time 1 is a fat matrix due to $K > C_1$ and the FDL subroutine would just return the uninterrupted diagonal line starting from $(1, 1)$ without any shift operation. Hence, (123) holds for $t = 1 < t_1$.

Next, we assume that (123) holds for all $t' \in [1, t-1]$ for some $t < t_1$. Note that the condition $t < t_1$ implies $\widehat{I}_d(t') > 0$ for all $t' \in [1, t]$ because $I_d(t')$ (and $\widehat{I}_d(t')$) must not hit 0 before t_1. The inequality part of (123) is thus proved. We then prove that the formula in (123) indeed describes the $\widehat{I}_d(t)$ value.

Since (123) holds for $t-1$, we have

$$\widehat{I}_d(t-1) = (t-1)K - \max\left((t-\alpha-2)K, j_{\mathsf{FDL}}(\sum_{t'=1}^{t-1} C_{t'})\right) > 0. \tag{141}$$

We split the proof into two cases. Case 1: $C_t = 0$ and Case 2: $C_t > 0$. In Case 1: $C_t = 0$, we clearly have $j_{\mathsf{FDL}}(\sum_{t'=1}^{t-1} C_{t'}) = j_{\mathsf{FDL}}(\sum_{t'=1}^{t} C_{t'})$. See $t = 4$ in Fig. 14 for example. By plugging (141) into (139), we have

$$\begin{aligned}
&\widehat{I}_d(t) \\
&= K - 0 + \min\Bigg((t-1)K \\
&\quad - \max\left((t-\alpha-2)K, j_{\mathsf{FDL}}(\sum_{t'=1}^{t-1} C_{t'})\right), \alpha K\Bigg) &(142)\\
&= K + \min\left((\alpha+1)K, (t-1)K - j_{\mathsf{FDL}}(\sum_{t'=1}^{t} C_{t'}), \alpha K\right) &(143)\\
&= tK - \max\left((t-\alpha-1)K, j_{\mathsf{FDL}}(\sum_{t'=1}^{t} C_{t'})\right) &(144)
\end{aligned}$$

where (143) follows from basic algebra and $C_t = 0$; and (144) follows from basic algebra and the observation that $(\alpha+1)K \geq \alpha K$. By (144), we have proved (123) for Case 1.

Case 2: $C_t > 0$. Again by plugging (141) into (139), we have

$$\widehat{I}_d(t) = K - C_t + \min\left(\alpha K, (t-1)K - j_{\mathsf{FDL}}(\sum_{t'=1}^{t-1} C_{t'})\right) \tag{145}$$

$$= tK - \max\left((t-\alpha-1)K, j_{\mathsf{FDL}}(\sum_{t'=1}^{t-1} C_{t'})\right) - C_t > 0 \tag{146}$$

following similar simplification steps as in (142) to (144).

We now analyze the value of $j_{\mathsf{FDL}}(\sum_{t'=1}^{t} C_{t'})$ in (123) by tracing the steps of the FDL subroutine. Specifically, we have $j_{\mathsf{FDL}}(\sum_{t'=1}^{t-1} C_{t'}) < (t-1)K$ by the "> 0" inequality in the induction assumption (141). Recall that if the FDL ends before reaching row $\sum_{t'=1}^{t-1} C_{t'}$, we will have the corresponding $j_{\mathsf{FDL}}(\sum_{t'=1}^{t-1} C_{t'}) = \infty$. As a result, at row $\sum_{t'=1}^{t-1} C_{t'}$, the FDL subroutine is still continuing. See Lines 5 to 7 of the FDL pseudocode. Therefore, at the next row $i = \sum_{t'=1}^{t-1} C_{t'} + 1$, we must have

$$\begin{aligned}
&j_{\mathsf{FDL}}((\sum_{t'=1}^{t-1} C_{t'}) + 1) = \\
&\max\left((t-\alpha-1)K + 1,\ j_{\mathsf{FDL}}(\sum_{t'=1}^{t-1} C_{t'}) + 1\right).
\end{aligned} \tag{147}$$

The reason is as follows. At row $i = \sum_{t'=1}^{t-1} C_{t'}$, the FDL outputs $j_{\mathsf{FDL}}(\sum_{t'=1}^{t-1} C_{t'})$. At the next row $i = \sum_{t'=1}^{t-1} C_{t'} + 1$, the FDL will first check whether the location $j = j_{\mathsf{FDL}}(\sum_{t'=1}^{t-1} C_{t'}) + 1$ is non-zero. If so, the FDL would output $j_{\mathsf{FDL}}(\sum_{t'=1}^{t-1} C_{t'} + 1) = j_{\mathsf{FDL}}(\sum_{t'=1}^{t-1} C_{t'}) + 1$. If not, the FDL would output the first non-zero element on the right of $j_{\mathsf{FDL}}(\sum_{t'=1}^{t-1} C_{t'}) + 1$. Since the first non-zero element of row $i = \sum_{t'=1}^{t-1} C_{t'} + 1$ is at the location $(t-\alpha-1)K+1$, by comparing the values of $j_{\mathsf{FDL}}(\sum_{t'=1}^{t-1} C_{t'})+1$ and $(t-\alpha-1)K+1$, we have (147).

We now argue that at row $\sum_{t'=1}^{t} C_{t'}$, the FDL subroutine has not ended yet. To that end, we notice that by Lines 8 and 9 of the FDL pseudocode, the FDL subroutine ends if there exists an $i \in (\sum_{t'=1}^{t-1} C_{t'}, \sum_{t'=1}^{t} C_{t'}]$ such that $j_{\mathsf{FDL}}(i) = j_{\text{nz.max}}(i)$. That is, the diagonal line $j_{\mathsf{FDL}}(i)$ hits the rightmost non-zero element $j_{\text{nz.max}}(i)$ at or before the $(\sum_{t'=1}^{t} C_{t'})$-th row. We then make the following two observations. Firstly, for all $i \in (\sum_{t'=1}^{t-1} C_{t'}, \sum_{t'=1}^{t} C_{t'}]$ we have $j_{\text{nz.max}}(i) = tK$ by (120) and (121). Secondly, when $i = (\sum_{t'=1}^{t-1} C_{t'}) + 1$ we have (147). As a result, there exists an $i \in (\sum_{t'=1}^{t-1} C_{t'}, \sum_{t'=1}^{t} C_{t'}]$ such that $j_{\mathsf{FDL}}(i) = j_{\text{nz.max}}(i)$ if and only if

$$\max\left((t-\alpha-1)K,\ j_{\mathsf{FDL}}(\sum_{t'=1}^{t-1} C_{t'})\right) + C_t \geq tK \tag{148}$$

where the left-hand side describes the location of the diagonal line until row $\sum_{t'=1}^{t} C_{t'}$ and the right-hand side describes the location of the rightmost non-zero element. However, (146) implies NOT (148). As a result, at row $\sum_{t'=1}^{t} C_{t'}$, the FDL

subroutine has not ended yet. Because the FDL subroutine has not ended at row $\sum_{t'=1}^{t} C_{t'}$, we have

$$\begin{aligned} & j_{\mathsf{FDL}}(\sum_{t'=1}^{t} C_{t'}) \\ & = \max\left((t-\alpha-1)K,\ j_{\mathsf{FDL}}(\sum_{t'=1}^{t-1} C_{t'})\right) + C_t < \infty. \end{aligned} \quad (149)$$

Finally, the inequality $j_{\mathsf{FDL}}(\sum_{t'=1}^{t} C_{t'}) < \infty$ implies that the corresponding element is non-zero. Since at row $\sum_{t'=1}^{t} C_{t'}$, the leftmost non-zero element is at location $(t-\alpha-1)K+1$. We thus also have

$$j_{\mathsf{FDL}}(\sum_{t'=1}^{t} C_{t'}) \geq (t-\alpha-1)K+1. \quad (150)$$

By plugging (149) into (123) and using (150) to remove the $\max$ operation in (123), the expression of (123) is identical to (146). We have thus completed the proof of Case 2. By mathematical induction, we have thus proved (123) for all $t < t_1$.

As for (124), since $\widehat{I}_d(t_1-1) > 0$ and $\widehat{I}_d(t_1) = 0$, by (139) we have

$$C_{t_1} \geq K + \min\left(\widehat{I}_d(t_1-1), \alpha K\right) > 0. \quad (151)$$

This implies that we have $C_{t_1} \geq 1$ additional rows beneath row $\sum_{t'=1}^{t_1-1} C_{t'}$. Note that when proving (123), we have already shown that the FDL subroutine will not end at row $i \leq \sum_{t'=1}^{t_1-1} C_{t'}$ nor before. Since we have $C_{t_1} \geq 1$ additional rows beneath row $\sum_{t'=1}^{t_1-1} C_{t'}$, we only need to prove that we can still find at least one non-zero entry in the next new row $i = \sum_{t'=1}^{t_1-1} C_{t'} + 1$, which will imply that FDL can proceed to the next row and thus $\sum_{t'=1}^{t_1-1} C_{t'} < i_{\text{end}}$. The inequality $i_{\text{end}} \leq \sum_{t'=1}^{t_1} C_{t'}$ is trivially true because the right-hand side is the total number of rows.

From (123) when $t = t_1 - 1$,

$$\begin{aligned} \widehat{I}_d(t_1-1) &= (t_1-1)K \\ &\quad - \max\left((t_1-\alpha-1)K, j_{\mathsf{FDL}}(\sum_{t'=1}^{t_1-1} C_{t'})\right) > 0, \end{aligned} \quad (152)$$

which implies $(t_1-1)K > j_{\mathsf{FDL}}(\sum_{t'=1}^{t_1-1} C_{t'})$. Since the location of the rightmost non-zero entry for the next row $i = \sum_{t'=1}^{t_1-1} C_{t'}+1$ is $j_{\text{nz.max}}(i) = t_1K > j_{\mathsf{FDL}}(\sum_{t'=1}^{t_1-1} C_{t'})$, the FDL subroutine can successfully proceed from row $\sum_{t'=1}^{t_1-1} C_{t'}$ to the next row $i = \sum_{t'=1}^{t_1-1} C_{t'} + 1$. Eq. (124) is thus proved.

To prove (125), we first note that by (124), $\sum_{t'=1}^{t_1-1} C_{t'} < i_{\text{end}} \leq \sum_{t'=1}^{t_1} C_{t'}$ and hence $j_{\text{nz.max}}(i_{\text{end}}) = t_1K$. Next, we note that by Lines 3, 8 and 9 of the FDL pseudocode, the FDL subroutine ends only if either condition (i) $i_{\text{end}} = \sum_{t'=1}^{t_1} C_{t'}$ and $j_{\mathsf{FDL}}(i_{\text{end}}) < \text{cols}(\mathbf{A}) = t_1K$, or condition (ii) $j_{\mathsf{FDL}}(i_{\text{end}}) = j_{\text{nz.max}}(i_{\text{end}}) = t_1K$ holds.

We then prove it by contradiction that condition (i) can never be true. Assume that condition (i) holds. Extending from (147), we have

$$j_{\mathsf{FDL}}(i_{\text{end}}) = j_{\mathsf{FDL}}(\sum_{t'=1}^{t_1} C_{t'}) = j_{\mathsf{FDL}}(\sum_{t'=1}^{t_1-1} C_{t'} + C_{t_1}) \quad (153)$$

$$= \max\left((t_1-\alpha-1)K,\ j_{\mathsf{FDL}}(\sum_{t'=1}^{t_1-1} C_{t'})\right) + C_{t_1} < t_1K. \quad (154)$$

By basic algebra, (154) can be rewritten as

$$C_{t_1} < \min\left((\alpha+1)K,\ t_1K - j_{\mathsf{FDL}}(\sum_{t'=1}^{t_1-1} C_{t'})\right). \quad (155)$$

However, by plugging (152) into (151), we have

$$\begin{aligned} & C_{t_1} \\ & \geq K + \min\left(\min\left(\alpha K, (t_1-1)K - j_{\mathsf{FDL}}(\sum_{t'=1}^{t_1-1} C_{t'})\right), \alpha K\right) \end{aligned} \quad (156)$$

$$= \min\left((\alpha+1)K,\ t_1K - j_{\mathsf{FDL}}(\sum_{t'=1}^{t_1-1} C_{t'})\right), \quad (157)$$

which contradicts (155). Hence, (125) always holds.

Finally, to prove (126), we split our discussion into two groups: $i \in [1, \sum_{t'=1}^{t_1-1} C_{t'}]$ and $i \in (\sum_{t'=1}^{t_1-1} C_{t'}, i_{\text{end}})$. For any $i \in [1, \sum_{t'=1}^{t_1-1} C_{t'}]$, there exists a unique $t < t_1$ that satisfies $C_t > 0$ and (121). We thus have

$$\begin{aligned} & j_{\text{nz.max}}(i) - j_{\mathsf{FDL}}(i) = tK - j_{\mathsf{FDL}}(i) \\ & \geq tK - j_{\mathsf{FDL}}(\sum_{t'=1}^{t} C_{t'}) && (158) \\ & \geq tK - \max\left((t-\alpha-1)K, j_{\mathsf{FDL}}(\sum_{t'=1}^{t} C_{t'})\right) && (159) \\ & = \widehat{I}_d(t) > 0 && (160) \end{aligned}$$

where (158) follows from $j_{\mathsf{FDL}}(\cdot)$ being an increasing function; (159) follows by adding a $\max$ operation into the formula; and (160) follows from (123) and from the fact that $\widehat{I}_d(t') > 0$ for all $t' < t_1$. Ineq. (126) is thus proved for those $i \in [1, \sum_{t'=1}^{t_1-1} C_{t'}]$.

For $i \in (\sum_{t'=1}^{t_1-1} C_{t'}, i_{\text{end}})$, the rows being considered are corresponding to time t_1. As a result, $j_{\text{nz.max}}(i) = t_1K$. Because $j_{\mathsf{FDL}}(\cdot)$ is a strictly increasing function, we also have $j_{\mathsf{FDL}}(i) < j_{\mathsf{FDL}}(i_{\text{end}})$ for those $i < i_{\text{end}}$. Using these two observations, one can easily derive (126) as a simple corollary of (125). The proof of (126) for those $i \in (\sum_{t'=1}^{t_1-1} C_{t'}, i_{\text{end}})$ is complete.

Appendix E
Proof of Lemma 16

We note that the FDL labeling between row i and row $(i+1)$ is "interrupted" if and only if the matrix entry at location $(i+1, j_{\mathsf{FDL}}(i)+1)$ is zero, which means that Line 6 of the FDL subroutine has to choose a new "interrupted" location

$j_{\mathsf{FDL}}(i+1) > j_{\mathsf{FDL}}(i)+1$ for the $(i+1)$-th row, rather than the "consecutive" location $j_{\mathsf{FDL}}(i+1) = j_{\mathsf{FDL}}(i)+1$.

We first consider any two consecutive rows i and $(i+1)$ of the same observation $\mathbf{y}(t)$. Recall that i_{end} is the very last row in S_L, we thus have $(i+1) \le i_{\text{end}}$. Note that $(i, j_{\mathsf{FDL}}(i)) \in S_L$ is a non-zero entry since FDL only outputs non-zero entries. Since both rows belong to the same observation $\mathbf{y}(t)$, we must have $(i+1, j_{\mathsf{FDL}}(i))$ is also a non-zero entry because how we encode during each time slot, see (3). Since the non-zeros of each row (in particular, row $(i+1)$) must be consecutive and since $j_{\text{nz.max}}(i+1) \ge j_{\text{nz.max}}(i) > j_{\mathsf{FDL}}(i)$ due to (126), the entry $(i+1, j_{\mathsf{FDL}}(i)+1)$ must be non-zero. This thus implies that there must not be any interruption between rows i and $(i+1)$ if both rows belong to the same $\mathbf{y}(t)$.

We now consider the case that rows i and $(i+1)$ belong to two different slots $\mathbf{y}(t)$ and $\mathbf{y}(t')$, respectively. An equivalent statement is that

$$\exists\ \widetilde{t} < t_1 \text{ such that} \tag{161}$$

$$i = \sum_{t'=1}^{\widetilde{t}} C_{t'} < i+1 \le \sum_{t'=1}^{\widetilde{t}+1} C_{t'}. \tag{162}$$

Suppose the condition of the first half of Lemma 16 holds, i.e., there is no ζ-hitting time τ_j in $(0, t_1]$. In this case, we have $0 < I_d(t) \le \alpha K$, or equivalently $0 < \widehat{I}_d(t) \le \alpha K$ for all $t \in (0, t_1)$. By (123), we have

$$\begin{aligned}\widehat{I}_d(t) &= tK - \max\big((t-\alpha-1)K, j_{\mathsf{FDL}}(\sum_{t'=1}^{t} C_{t'})\big) \\ &= \min\big((\alpha+1)K,\ tK - j_{\mathsf{FDL}}(\sum_{t'=1}^{t} C_{t'})\big) \le \alpha K\end{aligned} \tag{163}$$

which implies that $j_{\mathsf{FDL}}(\sum_{t'=1}^{t} C_{t'}) \ge (t-\alpha)K$ for all $t \in (0, t_1)$. Since our $\widetilde{t}$ satisfies $\widetilde{t} < t_1$, we have

$$j_{\mathsf{FDL}}(i)+1 = j_{\mathsf{FDL}}(\sum_{t'=1}^{\widetilde{t}} C_{t'})+1 \ge (\widetilde{t}-\alpha)K+1. \tag{164}$$

Since row $i+1$ belongs to the observation $\mathbf{y}(\widetilde{t}+1)$ at time $\widetilde{t}+1$, see (162), the leftmost non-zero entry of row $(i+1)$ is at location $(\widetilde{t}-\alpha)K+1$. Eq. (164) then implies the entry $(i+1, j_{\mathsf{FDL}}(i)+1)$ is non-zero. As a result, there must be no interruption between rows i and $(i+1)$. The first half of Lemma 16 is proved.

Consider the second half of Lemma 16. There exists at least one $\tau_j \in (0, t_1)$. Define τ_{j^*} as the one with the largest j. This implies that $\widehat{I}_d(\tau_{j^*}) > \alpha K$ and $\widehat{I}_d(t) \le \alpha K$ for all $t \in (\tau_{j^*}, t_1]$. By identical arguments as in the first half of proof, the diagonal line segment from row $i_{\text{start}} = \sum_{t'=1}^{\tau_{j^*}} C_{t'} + 1$ to $i_{\text{end}} = \max\{i : (i,j) \in S_L\}$ must be consecutive (or equivalently "uninterrupted"). Since we always have $j_{\text{end}} = j_{\mathsf{FDL}}(i_{\text{end}}) = t_1 K$ by (125), the remaining step is to show that $j_{\text{start}} = (\tau_{j^*} - \alpha)K + 1$.

To that end, we notice that we must have $C_{\tau_{j^*}+1} > 0$. The reason is that if $C_{\tau_{j^*}+1} = 0$, we will have $\widehat{I}_d(\tau_{j^*}+1) > \alpha K$, which contradicts that τ_{j^*} is the last ζ-hitting time. We also note that $\widehat{I}_d(\tau_{j^*}) > \alpha K$ and (123) together imply

$$\begin{aligned}\widehat{I}_d(\tau_{j^*}) &= \tau_{j^*}K - \max\big((\tau_{j^*}-\alpha-1)K, j_{\mathsf{FDL}}(\sum_{t'=1}^{\tau_{j^*}} C_{t'})\big) \\ &= \min\big((\alpha+1)K,\ \tau_{j^*}K - j_{\mathsf{FDL}}(\sum_{t'=1}^{\tau_{j^*}} C_{t'})\big) > \alpha K\end{aligned} \tag{165}$$

which in turn implies that

$$j_{\mathsf{FDL}}(\sum_{t'=1}^{\tau_{j^*}} C_{t'}) < (\tau_{j^*} - \alpha)K. \tag{166}$$

Our goal is to find the value of $j_{\text{start}} = j_{\mathsf{FDL}}(\sum_{t'=1}^{\tau_{j^*}} C_{t'} + 1)$. By Lines 5 and 6 of the FDL pseudocode, we must have

$$\begin{aligned}&j_{\mathsf{FDL}}(\sum_{t'=1}^{\tau_{j^*}} C_{t'} + 1) \\ &= \max\big(j_{\mathsf{FDL}}(\sum_{t'=1}^{\tau_{j^*}} C_{t'})+1, ((\tau_{j^*}+1)-\alpha-1)K+1\big)\end{aligned} \tag{167}$$

where the first term of the max operation, $j_{\mathsf{FDL}}(\sum_{t'=1}^{\tau_{j^*}} C_{t'})+1$, represents that the next diagonal element following $(\sum_{t'=1}^{\tau_{j^*}} C_{t'}, j_{\mathsf{FDL}}(\sum_{t'=1}^{\tau_{j^*}} C_{t'}))$, and the second term of the max operation represents the first non-zero element of row $\sum_{t'=1}^{\tau_{j^*}} C_{t'} + 1$ since $C_{\tau_{j^*}+1} > 0$ implies that for the rows corresponding to time $\tau_{j^*}+1$, the leftmost non-zero element is at the $(((\tau_{j^*}+1)-\alpha-1)K+1)$-th column. The max operation in (167) essentially checks whether the next diagonal element is non-zero. If so, $j_{\mathsf{FDL}}(\sum_{t'=1}^{\tau_{j^*}} C_{t'}+1) = j_{\mathsf{FDL}}(\sum_{t'=1}^{\tau_{j^*}} C_{t'})+1$. If not, the FDL routine labels the first non-zero element at the $(((\tau_{j^*}+1)-\alpha-1)K+1)$-th column. By (166), we thus have $j_{\text{start}} = j_{\mathsf{FDL}}(\sum_{t'=1}^{\tau_{j^*}} C_{t'} + 1) = (\tau_{j^*} - \alpha)K + 1$. The proof of the second half of Lemma 16 is complete.

Appendix F
Proof of Lemma 20

The construction of $\widetilde{S}_L$ "shifts the elements to the right" when and only when $i \le \sum_{t'=1}^{\tau_{j^*}} C_{t'}$. Consider any such i. By (126), we have $j_{\mathsf{FDL}}(i) < j_{\text{nz.max}}(i)$ with strict inequality. Note that by the construction of the RLSC encoder, the non-zero elements in each row are always consecutive. As a result, the "right-neighbor element" $A_{i, j_{\mathsf{FDL}}(i)+1} \ne 0$ and we have proved the first half of Lemma 20.

For the second half of the lemma, because S_L has distinct row indices, per our construction (136) $\widetilde{S}_L$ also contains distinct row indices. As for the column indices, we first note that for any (i_1, j_1), (i_2, j_2) in S_L, we always have $i_1 < i_2 \Leftrightarrow j_1 < j_2$. If we focus only on those $i \le \sum_{t'=1}^{\tau_{j^*}} C_{t'}$, the corresponding j (the column indices) are all distinct since $\widetilde{S}_L$ contains the next-right neighbor of the (i,j) in S_L for all $i \le \sum_{t'=1}^{\tau_{j^*}} C_{t'}$. Similarly, if we focus only on those $i > \sum_{t'=1}^{\tau_{j^*}} C_{t'}$, the corresponding j (column indices) are all distinct since $\widetilde{S}_L$ retains the (i,j) pairs in S_L for all $i > \sum_{t'=1}^{\tau_{j^*}} C_{t'}$. As a result, we only need to prove that for the new set $\widetilde{S}_L$, the column index j_1 when $i_1 = \sum_{t'=1}^{\tau_{j^*}} C_{t'}$ is strictly

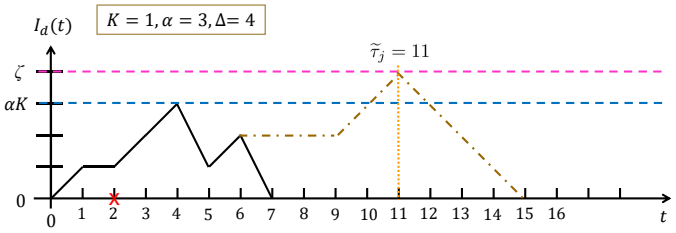

Fig. 16: An example for Case 1 in the proof of Proposition 3 when focusing at $t=2$. The new $\widetilde{I}_d(t')$ (the brown dash-dotted curve) has the same realization as the original $I_d(t')$ (the black curve) for $t' \le t+\Delta = 6$. During $t' \in [t+\Delta+1, t+\Delta+\alpha] = [7,9]$, $\widetilde{I}_d(t')$ stays flat. For $t' \ge t+\Delta+\alpha = 9$, $\widetilde{I}_d(t')$ first increments until hitting ζ and then decrements until hitting 0.

smaller than the column index j_2 when $i_2 = \sum_{t'=1}^{\tau_{j^*}} C_{t'} + 1$. Also see Fig. 15 in which we desire the last shifted green circle to be strictly on the left of the first non-shifted green circle.

It turns out that we have already established this fact when proving the second half of Lemma 16. Specifically, because j_1 in $\widetilde{S}_L$ is obtained by the shifted position in S_L, we have

$$j_1 = j_{\mathsf{FDL}}(\sum_{t'=1}^{\tau_{j^*}} C_{t'}) + 1. \tag{168}$$

Also, because j_2 in $\widetilde{S}_L$ is obtained by the original position in S_L, we have

$$j_2 = j_{\mathsf{FDL}}(\sum_{t'=1}^{\tau_{j^*}} C_{t'} + 1). \tag{169}$$

By (166) and (167), we have $j_1 < j_2$. This shows that all column indices of $\widetilde{S}_L$ are distinct.

Appendix G
Proof of Proposition 3

It is easy to verify that when $\Delta \to \infty$, Proposition 3 implies Proposition 2. As a result, Proposition 3 can be viewed as a stronger version of Proposition 2. In the sequel, we prove Proposition 3 by bootstrapping the results of Proposition 2.

We prove it by contradiction. We first consider Case 1: there exists no $\tau_j \in (t_{i_0}, t_{i_0+1})$. Given any arbitrary deadline constraint $\Delta < \infty$ and arbitrary memory length $\alpha < \infty$, assume that $\mathbf{s}(t)$ is Δ-decodable, i.e., decodable at time $t+\Delta$, for some $t \in (t_{i_0}, t_{i_0+1} - \Delta)$. This implies that regardless of the realization of $\{I_d(\cdot)\}$ after time $t+\Delta$, slot $\mathbf{s}(t)$ will remain decodable after time $t+\Delta$ since it is already decodable at time $t+\Delta$.

Note that $t+\Delta < t_{i_0+1}$ because Proposition 3 in this case focuses on $t \in (t_{i_0}, t_{i_0+1} - \Delta)$. At time $t+\Delta$, the random process $I_d(t)$ thus has not hit 0 since time $t+\Delta < t_{i_0+1}$. As a result, we can always replace the realization[10] of $\{I_d(t') : t' \in [t+\Delta+1, \infty)\}$ by a new realization $\{\widetilde{I}_d(t') : t' \in [t+\Delta+1, \infty)\}$ satisfying simultaneously (i) $\widetilde{I}_d(t') = I_d(t+\Delta)$ for all $t' \in [t+\Delta+1, t+\Delta+\alpha]$; (ii) $\widetilde{I}_d(t') = \widetilde{I}_d(t'-1)+1$ for all $t' \ge t+\Delta+\alpha+1$ until $\widetilde{I}_d(t') = \zeta$; and (iii) after the τ' satisfying $\widetilde{I}_d(\tau') = \zeta$, i.e., we have a new ζ-hitting time $\widetilde{\tau}_j = \tau'$, we let $\widetilde{I}_d(t') = K \cdot (\alpha + 1 - t' + \widetilde{\tau}_j)$ for all $t' \in (\widetilde{\tau}_j, \widetilde{\tau}_j + \alpha + 1]$. In other words, the new realization of $\widetilde{I}_d(t')$ will remain the same for α slots, then increment continuously until hitting ζ, and then decrement until hitting 0. See Fig. 16 for illustration. One can see that with the new realization $\widetilde{I}_d(t')$, the new hitting times $\widetilde{t}_{i_0+1}$ and $\widetilde{\tau}_{j^*}$ (there is only one $\widetilde{\tau}_j \in (t_{i_0}, \widetilde{t}_{i_0+1})$) always satisfy $t \le \widetilde{\tau}_{j^*} - \alpha$. By Proposition 2, $\mathbf{s}(t)$ is not decodable under the new realization $\widetilde{I}_d(t')$ no matter how large we set the decoding time T, which contradicts our initial assumption that $\mathbf{s}(t)$ is Δ-decodable under $I_d(t')$ and thus under $\widetilde{I}_d(t')$. The proof of Case 1 is complete.

Case 2: There exists a $\tau_{j^*} \in (t_{i_0}, t_{i_0+1})$. By Corollary 2, $\mathbf{s}(t)$ is not Δ-decodable for all $t \in (t_{i_0}, \tau_{j^*} - \alpha + 1)$. As a result, to prove (24) we only need to prove that $\mathbf{s}(t)$ is not Δ-decodable for all $t \in (t_{i_0}, t_{i_0+1} - \Delta)$. Suppose $\mathbf{s}(t)$ is Δ-decodable for some $t \in (t_{i_0}, \max(\tau_{j^*} - \alpha + 1, t_{i_0+1} - \Delta))$. This implies that regardless of the realization of $\{I_d(\cdot)\}$ after time $t+\Delta$, slot $\mathbf{s}(t)$ will remain decodable after time $t+\Delta$ since it is already decodable at time $t+\Delta$. In an almost identical fashion as in Case 1, we replace the realization of $I_d(t')$ including and after time $t+\Delta+1$ such that the new realization $\widetilde{I}_d(t')$ will have a new ζ-hitting time $\widetilde{\tau}_{j^*}$ and a new 0-hitting time $\widetilde{t}_{i_0+1}$ satisfying $t+\alpha \le \widetilde{\tau}_{j^*}$. By Proposition 2, $\mathbf{s}(t)$ is not decodable under $\widetilde{I}_d(t')$ no matter how large we set the decoding time T, which contradicts the initial assumption that $\mathbf{s}(t)$ is Δ-decodable under $I_d(t')$ and $\widetilde{I}_d(t')$. The proof of Case 2 is complete.

Appendix H
Simplication of (36)

To prove (33) and (36) are identical, we first note that for any 2×2 block matrix

$$\mathbf{B} = \begin{bmatrix} \mathbf{B}_1 & \mathbf{B}_2 \\ \mathbf{0} & \mathbf{B}_3 \end{bmatrix}, \tag{170}$$

if $\mathbf{B}_1$ and $\mathbf{B}_3$ are invertible,

$$\mathbf{B}^{-1} = \begin{bmatrix} \mathbf{B}_1^{-1} & -\mathbf{B}_1^{-1}\mathbf{B}_2\mathbf{B}_3^{-1} \\ \mathbf{0} & \mathbf{B}_3^{-1} \end{bmatrix}. \tag{171}$$

Again, we denote

$$\mathbf{Q} = \begin{bmatrix} \Gamma_{\phi,\phi} & \Gamma_{\phi,\zeta} \\ \Gamma_{\zeta,\phi} & \Gamma_{\zeta,\zeta} \end{bmatrix}.$$

From (33),

$$\mathbb{E}\{t_{i_0+1} - t_{i_0}\}$$
$$= \vec{\delta}_1^{\top} \left(\begin{bmatrix} 1 & -\Gamma_{0,\phi} & -\Gamma_{0,\zeta} \\ \mathbf{0} & \mathbf{I}_\zeta - \mathbf{Q} \end{bmatrix} \right)^{-1} \vec{\mathbf{1}} \tag{172}$$
$$= \vec{\delta}_1^{\top} \begin{bmatrix} 1 & \begin{bmatrix} \Gamma_{0,\phi} & \Gamma_{0,\zeta} \end{bmatrix} (\mathbf{I}_\zeta - \mathbf{Q})^{-1} \\ \mathbf{0} & (\mathbf{I}_\zeta - \mathbf{Q})^{-1} \end{bmatrix} \vec{\mathbf{1}} \tag{173}$$
$$= 1 + \begin{bmatrix} \Gamma_{0,\phi} & \Gamma_{0,\zeta} \end{bmatrix} (\mathbf{I}_\zeta - \mathbf{Q})^{-1} \vec{\mathbf{1}}. \tag{174}$$

[10] Since $I_d(t)$ is determined by the channel realization C_t, we can alter the realization of $I_d(t)$ by altering the realization of C_t. In our construction, the realization of $\{I_d(\cdot)\}$ before $t+\Delta$ remains intact, i.e., $I_d(t') = \widetilde{I}_d(t')$ for all $t' \in [0, t+\Delta]$.

Because Γ is a probability transition matrix, the sum of each row is always one. We thus have

$$\Gamma_{0,0} = 1 - \left[\begin{array}{cc} \Gamma_{0,\boldsymbol{\phi}} & \Gamma_{0,\zeta} \end{array}\right] \vec{\mathbf{1}} \tag{175}$$

$$\left[\begin{array}{cc} \Gamma_{\boldsymbol{\phi},0}^{\top} & \Gamma_{\zeta,0}^{\top} \end{array}\right]^{\top} = (\vec{\mathbf{1}} - \mathbf{Q}\vec{\mathbf{1}}). \tag{176}$$

Substituting (175) and (176) into (36), we have

$$\begin{aligned}
&\mathbb{E}\left\{t_{i_0+1} - t_{i_0}\right\} \\
&= \left(1 - \left[\begin{array}{cc} \Gamma_{0,\boldsymbol{\phi}} & \Gamma_{0,\zeta} \end{array}\right] \vec{\mathbf{1}}\right) \\
&\quad + \sum_{k=2}^{\infty} k \left[\begin{array}{cc} \Gamma_{0,\boldsymbol{\phi}} & \Gamma_{0,\zeta} \end{array}\right] \mathbf{Q}^{k-2} \left(\vec{\mathbf{1}} - \mathbf{Q}\vec{\mathbf{1}}\right) && (177) \\
&= 1 + \left[\begin{array}{cc} \Gamma_{0,\boldsymbol{\phi}} & \Gamma_{0,\zeta} \end{array}\right] \sum_{k=1}^{\infty} \mathbf{Q}^{k-1} \vec{\mathbf{1}} && (178) \\
&= 1 + \left[\begin{array}{cc} \Gamma_{0,\boldsymbol{\phi}} & \Gamma_{0,\zeta} \end{array}\right] \left(\mathbf{I}_{\zeta} - \mathbf{Q}\right)^{-1} \vec{\mathbf{1}} && (179)
\end{aligned}$$

where (178) follows from basic rearrangements of the terms in the summation. Jointly, (174) and (179) prove that (36) can be simplified to the expression in (33).

Appendix I
Proof of $\mathsf{term}_4 = \mathsf{term}_5$

To formalize the intuitive idea in the proof of Lemma 8 and especially to reconcile that (51) is defined for a fixed t and (52) is defined for a fixed i_0, we first use the strong Markov property to substitute the t in term_4 by the j_0-th ζ-hitting time τ_{j_0}, for any arbitrarily given constant $j_0 \in [1, \infty)$. We then have

$$\begin{aligned}
\mathsf{term}_4 = \mathbb{E}\{&\max(-\alpha, H_{\tau_{j_0}}(0) - \Delta - 1) \\
&\mid \tau_{j_0} < \infty, H_{\tau_{j_0}}(0) < H_{\tau_{j_0}}(\zeta)\}.
\end{aligned} \tag{180}$$

For the given $j_0 \in [1, \infty)$, we also define the event

$$\mathcal{A}_{j_0} \triangleq \{\tau_{j_0} < \infty \text{ and } \forall t \in (\tau_1, \tau_{j_0}), I_d(t) \neq 0\}. \tag{181}$$

Note that $\mathcal{A}_{j_0}$ is measurable with respect to the filtration $\mathcal{F}_{\tau_{j_0}}$ induced by the stopping time τ_{j_0}, i.e., at time τ_{j_0} we can unambiguously determine whether we are in the event $\mathcal{A}_{j_0}$ or not. Since $\mathcal{A}_{j_0} \in \mathcal{F}_{\tau_{j_0}}$, by the strong Markov property, knowing what has happened until time τ_{j_0} will not affect the future after τ_{j_0}. We thus have

$$\begin{aligned}
\mathsf{term}_4 = \mathbb{E}\{&\max(-\alpha, H_{\tau_{j_0}}(0) - \Delta - 1) \\
&\mid H_{\tau_{j_0}}(0) < H_{\tau_{j_0}}(\zeta), \mathcal{A}_{j_0}\}.
\end{aligned} \tag{182}$$

Define the following random set:

$$\mathcal{J} \triangleq \{j \geq 1 : \tau_j < \infty, H_{\tau_j}(0) < H_{\tau_j}(\zeta)\} \tag{183}$$

which contains all the j values such that after ζ-hitting time τ_j, the Markov chain $I_d(t)$ hits 0 before hitting ζ again. We define $\overline{J} \triangleq \min \mathcal{J}$ as the first such j. Here we again use the convention that $\min \emptyset = \infty$. For example, $\{\overline{J} < \infty\}$ is thus the event that there exists at least one $\tau_j < \infty$ such that after τ_j the process $I_d(t)$ hits 0 before hitting ζ again.

Using this definition, we can rewrite the conditioning event in (182) as

$$\{H_{\tau_{j_0}}(0) < H_{\tau_{j_0}}(\zeta)\} \cap \mathcal{A}_{j_0} = \{\overline{J} = j_0\} \tag{184}$$

by the definition of $\mathcal{A}_{j_0}$ in (181). As a result, we have

$$\begin{aligned}
\mathsf{term}_4 &= \mathbb{E}\{\max(-\alpha, H_{\tau_{\overline{J}}}(0) - \Delta - 1) \mid \overline{J} = j_0\} && (185) \\
&= \mathbb{E}\{\max(-\alpha, H_{\tau_{\overline{J}}}(0) - \Delta - 1) \mid \overline{J} < \infty\} && (186)
\end{aligned}$$

where (185) follows from (182) by rewriting the conditional event by its equivalent event in (184); and (186) follows from the fact that the value of (185) is the same for any fixed finite $j_0 \in [1, \infty)$ value.

We now simplify term_5 in a similar way. Specifically, for any fixed $i_0 \in [0, \infty)$ we define the event

$$\mathcal{B}_{i_0} \triangleq \{\text{no } \tau_j \in (0, t_{i_0})\}. \tag{187}$$

Note that $\mathcal{B}_{i_0}$ is measurable with respect to the filtration $\mathcal{F}_{t_{i_0}}$ induced by the stopping time t_{i_0}. By the strong Markov property and similar reasons as when proving (182) we have

$$\begin{aligned}
\mathsf{term}_5 = \mathbb{E}\{&\max(-\alpha, t_{i_0+1} - \tau_{j^*(i_0)} - \Delta - 1) \\
&\mid t_{i_0+1} < \infty, \exists \tau_j \in (t_{i_0}, t_{i_0+1}), \mathcal{B}_{i_0}\}.
\end{aligned} \tag{188}$$

Define the following random set:

$$\mathcal{I} = \{i \geq 0 : t_{i+1} < \infty, \exists \tau_j \in (t_i, t_{i+1})\} \tag{189}$$

which contains all the i values such that the i-th round is a *bad round*, i.e., $I_d(t)$ hits ζ during two consecutive 0-hitting time t_i and t_{i+1}. We define $\overline{I} \triangleq \min \mathcal{I}$ as the first bad round i. Here we again use the convention that $\min \emptyset = \infty$. Using this definition, we can rewrite the conditioning event in (188) as

$$\{t_{i_0+1} < \infty, \exists \tau_j \in (t_{i_0}, t_{i_0+1})\} \cap \mathcal{B}_{i_0} = \{\overline{I} = i_0\}. \tag{190}$$

As a result, we have

$$\begin{aligned}
\mathsf{term}_5 &= \mathbb{E}\{\max(-\alpha, t_{\overline{I}+1} - \tau_{j^*(\overline{I})} - \Delta - 1) \mid \overline{I} = i_0\} && (191) \\
&= \mathbb{E}\{\max(-\alpha, t_{\overline{I}+1} - \tau_{j^*(\overline{I})} - \Delta - 1) \mid \overline{I} < \infty\} && (192)
\end{aligned}$$

where (191) follows from using the equivalent conditional event in (190), and (192) follows from the fact that the value of (191) is the same for any fixed finite $i_0 \in [0, \infty)$ values.

The final step is to note that the conditional events in (186) and (192) are actually the same event. To prove that we will show that $\overline{J} < \infty$ implies $\overline{I} < \infty$ and vice versa. For the forward direction, if $\overline{J} < \infty$, there is at least one $\tau_j < \infty$ such that after time τ_j the $I_d(t)$ hits 0 before hitting ζ again. This implies that there exists an $i \in [0, \infty)$ value such that $t_i < \infty$ and $\tau_j \in (0, t_i)$, which in turn implies that there exists an $i' < \infty$ such that $t_{i'+1} < \infty$ and $\tau_j \in (t_{i'}, t_{i'+1})$ and thus $\overline{I} < \infty$. For the backward direction, if $\overline{I} < \infty$, there is at least an $i \in [0, \infty)$ value such that $t_{i+1} < \infty$ and there exists $\tau_j \in (t_i, t_{i+1})$. For the largest such τ_j within (t_i, t_{i+1}), after time τ_j, the process $I_d(t)$ must hit 0 before hitting ζ again, which implies $\overline{J} < \infty$.

Under the conditional event $\{\overline{J} < \infty\} = \{\overline{I} < \infty\}$, the two random variables $H_{\tau_{\overline{J}}}(0)$ and $(t_{\overline{I}+1} - \tau_{j^*(\overline{I})})$ describe the same (random) phenomenon of the Markov chain $I_d(t)$. That is, both of them consider the event "$I_d(t)$ hits ζ and then hits 0 before hitting ζ for another time"; focus only on the first such

occurrence since time 0; and measure how many slots it takes between hitting ζ and hitting 0 in that particular occurrence. As a result, (186) and (192) must be identical, and we have $\mathsf{term}_4 = \mathsf{term}_5$.

Appendix J
Proofs of term_1 to term_3 Computation

To compute term_1 in (48), we notice that for any arbitrary $k \in [1, \infty)$, we have

$$\begin{aligned}&\Pr(H_t(0) > H_t(\zeta) = k \mid I_d(t) = 0)\\ &= \begin{cases}\Gamma_{0,\zeta} & \text{if } k = 1,\\ \Gamma_{0,\boldsymbol{\phi}} \left(\Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}}\right)^{k-2} \Gamma_{\boldsymbol{\phi},\zeta} & \text{if } k \geq 2.\end{cases}\end{aligned} \tag{193}$$

The value of term_1 is the sum of the probability of all events with $k \in [1, \infty)$. Hence we have

$$\mathsf{term}_1 = \Gamma_{0,\zeta} + \sum_{k=2}^{\infty} \left(\Gamma_{0,\boldsymbol{\phi}} \left(\Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}}\right)^{k-2} \Gamma_{\boldsymbol{\phi},\zeta}\right). \tag{194}$$

Simplifying the summation using (39) leads to (59).

To compute term_2 in (49), we notice that for any arbitrary $k \in [1, \infty)$, we have

$$\begin{aligned}&\Pr(H_t(0) = k < H_t(\zeta) \mid I_d(t) = \zeta)\\ &= \begin{cases}\Gamma_{\zeta,0} & \text{if } k = 1,\\ \Gamma_{\zeta,\boldsymbol{\phi}} \left(\Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}}\right)^{k-2} \Gamma_{\boldsymbol{\phi},0} & \text{if } k \geq 2.\end{cases}\end{aligned} \tag{195}$$

By summing over all events with $k \in [1, \infty)$, we can compute the value of term_2 by

$$\begin{aligned}&\mathsf{term}_2\\ &= -\min(\Delta, \alpha) \cdot \Gamma_{\zeta,0} + \sum_{k=2}^{\Delta-\alpha} \left(-\alpha \Gamma_{\zeta,\boldsymbol{\phi}} \left(\Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}}\right)^{k-2} \Gamma_{\boldsymbol{\phi},0}\right)\\ &\quad + \sum_{k=\max(\Delta-\alpha+1,2)}^{\infty} \left((k - \Delta - 1) \Gamma_{\zeta,\boldsymbol{\phi}} \left(\Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}}\right)^{k-2} \Gamma_{\boldsymbol{\phi},0}\right)\end{aligned} \tag{196}$$

where the first term considers the event $k = 1$; the first summation handles the events when the value of $\max(-\alpha, k - \Delta - 1) = -\alpha$ in (49); and the second summation is for the case of $\max(-\alpha, k - \Delta - 1) = k - \Delta - 1$. Simplifying the above equation using (39) gives us (60).

For term_3 in (50), we can evaluate the value by partitioning the events based on (195). We thus have

$$\mathsf{term}_3 = \Gamma_{\zeta,0} + \sum_{k=2}^{\infty} \left(\Gamma_{\zeta,\boldsymbol{\phi}} \left(\Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}}\right)^{k-2} \Gamma_{\boldsymbol{\phi},0}\right). \tag{197}$$

Simplifying the summation using (39) leads to (61).

Appendix K
Proof of Corollary 3

By definition, the expectation of (27) is a monotonically decreasing function with respect to Δ. Since $\Pr(t_{i_0+1} - t_{i_0} = \infty) = 0$, the limit when Δ goes to infinity must be zero. Hence, we have (64). The result implies that no error occurs in a good round under no decoding deadline assumption, which is consistent with the propositions derived in Section III-B.

For $\lim_{\Delta\to\infty} \mathbb{E}\{L_{B_2}\}$, recall that $\mathbb{E}\{L_{B_2}\} = \mathsf{term}_1 \cdot \mathsf{term}_2/\mathsf{term}_3$. Only term_2 involves Δ. When $\Delta \to \infty$, term_2 can be written as

$$\begin{aligned}\lim_{\Delta\to\infty} \mathsf{term}_2 &= -\alpha\Gamma_{\zeta,0} - \alpha\Gamma_{\zeta,\boldsymbol{\phi}}\mathbf{A}\Gamma_{\boldsymbol{\phi},0}\\ &\quad + \Gamma_{\zeta,\boldsymbol{\phi}} \left((\mathbf{A})^2 - \mathbf{A}\right) \left(\lim_{\Delta\to\infty} \left(\Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}}\right)^{\psi}\right) \Gamma_{\boldsymbol{\phi},0}.\end{aligned} \tag{198}$$

By (39), the value of $\lim_{\Delta\to\infty} \left(\Gamma_{\boldsymbol{\phi},\boldsymbol{\phi}}\right)^{\psi}$ is zero. We thus have

$$\lim_{\Delta\to\infty} \mathsf{term}_2 = -\alpha \left(\Gamma_{\zeta,0} + \Gamma_{\zeta,\boldsymbol{\phi}}\mathbf{A}\Gamma_{\boldsymbol{\phi},0}\right) = -\alpha \cdot \mathsf{term}_3. \tag{199}$$

Hence, we have

$$\lim_{\Delta\to\infty} \mathbb{E}\{L_{B_2}\} = \lim_{\Delta\to\infty} \frac{\mathsf{term}_1 \cdot \mathsf{term}_2}{\mathsf{term}_3} = -\alpha \cdot \mathsf{term}_1. \tag{200}$$

References

[1] 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.

[2] ——, "Random linear streaming codes in the finite memory length and decoding deadline regime," in *2021 IEEE International Symposium on Information Theory (ISIT)*, 2021, pp. 730–735.

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

[4] H. Ji, S. Park, J. Yeo, Y. Kim, J. Lee, and B. Shim, "Ultra-reliable and low-latency communications in 5G downlink: Physical layer aspects," *IEEE Wireless Communications*, vol. 25, no. 3, pp. 124–130, Jun. 2018.

[5] 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.

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

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

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

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

[11] 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.

[12] 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.

[13] 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.

[14] 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.

[15] M. N. Krishnan and P. V. Kumar, "Rate-optimal streaming codes for channels with burst and isolated erasures," in *2018 IEEE International Symposium on Information Theory (ISIT)*, Jun. 2018, pp. 1809–1813.

[16] 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.

[17] ——, "Online versus offline rate in streaming codes for variable-size messages," in *2020 IEEE International Symposium on Information Theory (ISIT)*, Jun. 2020.

[18] 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.
[19] 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.
[20] E. Martinian, "Dynamic information and constraints in source and channel coding," Ph.D. dissertation, Massachusetts Institute of Technology, 2004.
[21] M. Mitzenmacher, "A survey of results for deletion channels and related synchronization channels," *Probability Surveys*, vol. 6, pp. 1–33, 2009.
[22] 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.
[23] P. Piret and T. Krol, "MDS convolutional codes," *IEEE Transactions on Information Theory*, vol. 29, no. 2, pp. 224–232, 1983.
[24] H. Gluesing-Luerssen, J. Rosenthal, and R. Smarandache, "Strongly-MDS convolutional codes," *IEEE Transactions on Information Theory*, vol. 52, no. 2, pp. 584–598, 2006.
[25] R. Durrett, *Essentials of Stochastic Processes*, 3rd ed. Springer, 2016.

Pin-Wen Su received the B.S. degree in Electrical and Computer Engineering from National Chiao Tung University, Hsinchu, Taiwan, in 2014, and the M.S. degree in Communication Engineering from National Taiwan University, Taipei, Taiwan, in 2016. She is currently pursuing the Ph.D. degree with the Elmore Family School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN, USA. Her research interests include information and coding theory, communication theory, and signal processing for communications.

Yu-Chih Huang (M'14) received the Ph.D. degree in electrical and computer engineering from Texas A&M University (TAMU) in 2013. From 2013 to 2015, he was a Postdoctoral Research Associate with TAMU. In 2015, he joined the Department of Communication Engineering, National Taipei University, Taiwan, as an Assistant Professor and was promoted to an Associate Professor in 2018. In 2020, he joined the Institute of Communications Engineering, National Chiao Tung University (NCTU), Taiwan. He is currently an Associate Professor at National Yang Ming Chiao Tung University (the merger of National Yang Ming University and NCTU in 2021). His research interests are in information theory, coding theory, wireless communications, and statistical signal processing. He received the 2018 IEEE Information Theory Society Taipei Chapter and IEEE Communications Society Taipei/Tainan Chapter's Best Paper Award for Young Scholars and was a recipient of the MOST Young Scholar Fellowship 2020. He is currently serving as an Associate Editor for IEEE Communications Letters.

Shih-Chun Lin (Senior Member, IEEE) received the B.S. and Ph.D. degrees in electrical engineering from the National Taiwan University, Taipei, Taiwan, in 2000 and 2007, respectively. He was a Visiting Student with The Ohio State University, Columbus, OH, USA, in 2007. From 2011 to 2012, he was with the National Taipei University of Technology. From 2012 to 2021, he was with the National Taiwan University of Science and Technology, Taipei. In August 2021, he joined the National Taiwan University as an Associate Professor. His research interests include information theory, communications, and cyber-physical security. He was a TPC Member of the IEEE ICC from 2018 to 2022. He received the Best Paper Award for Young Authors from the IEEE IT/COM Society Taipei/Tainan Chapter in 2015 and twice the Project for Excellent Junior Research Investigators from the Ministry of Science and Technology, Taiwan, in 2015 and 2018. He serves as the TPC Co-Chair for IEEE ICC Workshop on B5G-URLLC 2019 and the Publication Chair for the IEEE GLOBECOM 2020.

I-Hsiang Wang received the B.Sc. degree in electrical engineering from National Taiwan University, Taiwan, in 2006. He received a Ph.D. degree in electrical engineering and computer sciences from the University of California at Berkeley, USA, in 2011. From 2011 to 2013, he was a postdoctoral researcher at École Polytechnique Fèdèrale de Lausanne, Switzerland. Since 2013, he has been at the Department of Electrical Engineering in National Taiwan University, where he is now an associate professor. His research interests include network information theory, networked data analysis, and statistical learning. He was a finalist of the Best Student Paper Award of IEEE International Symposium on Information Theory, 2011. He received the 2017 IEEE Information Theory Society Taipei Chapter and IEEE Communications Society Taipei/Tainan Chapters Best Paper Award for Young Scholars.

Chih-Chun Wang (M'06–SM'15) received the B.E. degree in EE from the National Taiwan University, Taipei, Taiwan, in 1999, and the M.S. and Ph.D. degrees in EE from Princeton University in 2002 and 2005, respectively.

He worked with Comtrend Corporation, Taipei, as a Design Engineer in 2000 and spent the summer of 2004 with Flarion Technologies, NJ, USA. In 2005, he was a Post-Doctoral Researcher with the Department of Electrical Engineering, Princeton University. He joined Purdue University in 2006 and became a Professor in 2017. He is a Professor with the Elmore Family School of Electrical and Computer Engineering, Purdue University. His current research interests are in the low latency 5G wireless networks and the corresponding protocol design, information theory, networks coding, and cyber-physical systems. Other research interests of his fall in the general areas of networking, optimal control, information theory, detection theory, and coding theory. He received the National Science Foundation Faculty Early Career Development (CAREER) Award in 2009. He served as the Technical Co-Chair for the 2017 IEEE Information Theory Workshop. He served as an Associate Editor for IEEE Transcations on Information Theory from 2014 to 2017.