RANK METRIC CONVOLUTIONAL CODES WITH APPLICATIONS IN NETWORK STREAMING

by

Rafid Mahmood

A thesis submitted in conformity with the requirements for the degree of Master of Applied Science
Graduate Department of Electrical and Computer Engineering
University of Toronto

© Copyright 2015 by Rafid Mahmood
Abstract

Rank Metric Convolutional Codes with Applications in Network Streaming

Rafid Mahmood
Master of Applied Science
Graduate Department of Electrical and Computer Engineering
University of Toronto
2015

We consider end-to-end error correction codes for real-time streaming over a network, where the source and destination are nodes in a graph. Links between nodes are erasure channels and the source-destination relationship a rank-deficient channel matrix.

In an isolated rank loss model, the channel matrix arbitrarily decreases to a minimum rank within each sliding window of network uses. We prove the column sum rank metric determines the effectiveness of a code and construct a family of codes that achieve the maximum column sum rank.

In a burst rank loss model, a fraction of links are simultaneously erased for consecutive network uses. We derive the capacity and prove achievability with a layered construction using the previously introduced codes. Simulations over statistical models reveal our construction to be superior over baseline codes.
Acknowledgements

First and foremost, I thank God for bringing me to a position where I can write acknowledgements.

My most sincere thanks go to Professor Ashish Khisti, firstly, for offering me the opportunity to work with him, and secondly, for everything that he has done since as an advisor. Under his guidance and support, I have learned much about asking the right questions, formulating clean problems, and conducting good research. The amount of time that he has spent on me has been extraordinary.

I thank Dr. Ahmed Badr, without whom I would not be nearly as productive as I have been. Aside from being my first and best collaborative partner in research, Dr. Badr is an excellent mentor and friend. Working with him has produced some very enjoyable and gratifying experiences, and our regular discussions on careers and life have been a tremendous influence on me.

I have had the opportunity to learn under several world-class professors over these past two years: Professors Ravi Adve, Stark Draper, Frank Kschischang, and Wei Yu. I am especially grateful to Professors Tarek Abdelrahman, Stark Draper, and Frank Kschischang for serving as members of my thesis committee. Their comments and suggestions have been helpful in improving both my research and my writing.

I began graduate studies along with several friends from my undergraduate years: Nadeesha Amarasinghe, Hassan Farooq, Simon Li, Sanket Pandit, Namal Rajatheva, Angelia Tian, and Max Yuan. I deeply cherish our years of companionship and am glad we took this journey together. In graduate school, I have made many new friends: Arvin Ayoughi, Binbin Dai, Farrokh Etezadi, Kaveh Mahdaviani, Siyu Liu, Pratik Patel, Foad Sohrabi, Gokul Sridharan, Soroush Tabatabaei, Louis Tan, Wanyao Zhao, and Yuhan Zhou. I hope that my life over the past two years wasn’t determined exclusively by graduate studies and am also thankful for my many friends outside of the university who have helped to preserve my sanity throughout.

Finally, my heartfelt gratitude goes to my parents, who have given me an unending appetite for learning and provided me the total support to feed it for every step of my life. I cannot imagine a single achievement in my life in which they have not played a significant part. They are, ultimately, the foundation of my being and I only hope for their best wishes in my future endeavours.
Bibliographic Note

Both of Chapters 3 and 4 are joint works with Ahmed Badr and Ashish Khisti. A portion of Chapter 3 was presented at the International Symposium on Information Theory (ISIT) 2015 in Hong Kong [23].
Contents

1 Introduction 1
  1.1 Streaming Communication .............................................. 1
  1.2 Network Communication .................................................. 3
  1.3 Network Streaming ........................................................ 6
  1.4 Outline ........................................................................ 6

2 Preliminaries 8
  2.1 Notation ................................................................. 8
  2.2 Finite Fields ............................................................... 8
  2.3 Super-regular Matrices .................................................... 9
  2.4 $m$-MDS Codes ........................................................... 11
  2.5 Rank Metric Codes ......................................................... 13

3 Maximum Sum Rank Codes 15
  3.1 Introduction ................................................................. 15
  3.2 Problem Setup ............................................................... 15
    3.2.1 Encoder ................................................................ 15
    3.2.2 Network Model ....................................................... 16
    3.2.3 Decoder ................................................................ 16
  3.3 Maximum Sum Rank Codes ................................................ 17
  3.4 Code Construction .......................................................... 21
    3.4.1 Super-regularity Preservation ..................................... 21
    3.4.2 Encoder ................................................................ 25
  3.5 Numerical Results ........................................................... 26

4 Recovery Of Bursts In Networks Codes 30
  4.1 Introduction ................................................................. 30
  4.2 Problem Setup ............................................................... 31
    4.2.1 Encoder ................................................................ 31
    4.2.2 Network Model ....................................................... 31
    4.2.3 Decoder ................................................................ 32
  4.3 Main Result ................................................................. 32
    4.3.1 Streaming Capacity of the Burst Rank Loss Network .......... 32
    4.3.2 Numerical Comparisons ......................................... 33
List of Figures

1.1 Examples of two erasure channels. .......................................................... 2
1.2 Different network coding approaches. ......................................................... 3
1.3 A network with a failing link. .................................................................. 5
1.4 The Encoder and Decoder are connected by \( r \) disjoint paths, determined by the min-cut of the graph. Each path is effectively responsible for a channel packet—column of \( X_t \)—at each time instance. Each received packet is a linear combination of the transmitted packets. 7

3.1 An example of an Isolated Rank Loss Sliding Window Network \( CH_1(10, 14, 3) \). The rank of each channel matrix is labelled inside its associated box and the total rank-deficiency of the channel matrix in any window of \( W = 3 \) shots is at most \( S = 14 \). ........................................ 16
3.2 A hypothetical transition along the state Trellis graph is highlighted. The codeword sequence \( x_{\{0,5\}} \) generated by this path is not a valid sequence for the active column sum rank, but is valid for the column sum rank. ......................................................... 17
3.3 The result of Gaussian elimination applied on the matrix form of a row of \( D_3 \) containing three polynomials. Non-zero entries are denoted by \( X \). The inverse isomorphism of the result generates a vector of polynomials with increasing degrees. .................................................. 23
3.4 Testing 100 randomly generated linear time-invariant convolutional codes \( C[4, 2, 1] \) over \( \mathbb{F}_2^M \) by multiplying the generator with the set of all full-rank block diagonal matrices in \( \mathbb{F}_2^M \) that satisfy the rank conditions in (3.3). The red lines denote the median value with the box encircling the 25-th to 75-th percentiles. The whiskers denote approximately the 1-st and 99-th percentiles. Points beyond this range are considered outliers. ........................................ 28

4.1 An example of a Burst Rank Loss Network \( CH_B(4, 2, 3, 5) \). Every burst consists of at most \( B = 3 \) shots, with gaps lasting at least \( G = 5 \) shots. The rank of each channel matrix is labelled inside its associated box and does not fall below \( r - \rho = 2 \) during a burst. .......... 31
4.2 A comparison of achievable rates using existing codes against the streaming capacity given in Theorem 7. The decoding deadline \( T = 30 \) and gap length \( G = 30 \) are fixed. The burst length is \( B = 10 \) and \( B = 30 \) for the two setups, while the tolerable rank loss per time instance \( p \) is varied. The solid black line marking the capacity is achievable using our proposed code. ................................................................. 35
4.3 A block diagram illustrating the encoding steps of a ROBIN code. The source packet is first split into two sub-packets and a different code is applied to each. The resulting parity-checks are then combined to form the overall parity-check packet. Finally, the parity-check packet and the urgent source packet are concatenated to generate the channel packet. .... 37
4.4 Example of a ROBIN code with memory $T_{\text{eff}} = 6$ recovering from a burst $B = 3$ in a network with $r = 3$. Each column represents a received channel packet along a given path, which is a linear combination of the transmitted packets. The hatched columns are the linearly dependent combinations. ................................................................. 40

4.5 A periodic burst rank loss network, where the channel matrix rank $(\mathbf{A}^*) = r - p$ during bursts and rank $(\mathbf{A}) = r$ during perfect communication. ................................................................. 41

4.6 Simulations over a Gilbert Channel modelling a network with rank $(\mathbf{A}_t) = 6$ in the good-state and 3 in the bad-state. The channel and code parameters are given in Table 4.1. ................................................................. 45

4.7 Simulations over a Gilbert Channel modelling a network with rank $(\mathbf{A}_t) = 16$ in the good-state and $16 - p$ in the bad-state. The channel and code parameters are given in Table 4.1. ................................................................. 47

A.1 The periodic rank loss channel, with the sets $U_i, V_i, W_i$ identified for the $i$-th period. ................................................................. 53
Chapter 1

Introduction

1.1 Streaming Communication

Real-time internet streaming is motivated by two primary factors: causality and delay. Source packets arrive sequentially at a transmitter, which must generate channel packets using only the packets available up to that time. A receiver observes the channel output and must reproduce each source packet within a maximum delay. The most obvious streaming application where such constraints are required is real-time video conferencing. The video frames recorded by one user are packetized and transmitted and must be reproduced sequentially and with minimal perceptible delay at the destination. When a packet has not been recovered within the imposed deadline, it is discarded as a loss and skipped in playback.

The design of good error correcting codes that contain both causal encoders and low-delay decoders often involves the following general model. Let \( k \in \mathbb{N} \) be the code dimension and \( n \geq k \) be the code length. Communication occurs over a finite field \( \mathbb{F}_q \) containing \( q \) elements. At each time instance \( t \geq 0 \), a source packet \( s_t \in \mathbb{F}_q^k \) is randomly generated i.i.d. as a \( k \)-length vector. The transmitter must construct a channel packet \( x_t \in \mathbb{F}_q^n \) using a causal encoding function, i.e.,

\[
    x_t = \gamma_t(s_0, \ldots, s_t).
\]

(1.1)

A receiver observes \( y_t = CH\{x_t\} \). We describe the specific channel models \( CH\{\cdot\} \) in further detail later. The receiver must estimate the source packet \( s_t \) with a delay of at most \( T \) subsequent channel uses via a decoding function giving

\[
    \hat{s}_t = \eta_t(y_0, \ldots, y_{t+T}).
\]

(1.2)

If \( \hat{s}_t = s_t \), then the source packet has been correctly recovered by the deadline; otherwise it is considered lost. An error correcting code \( C[n, k] \) with rate \( R = \frac{k}{n} \) consists of the encoding and decoding functions that protect a \( k \)-length source vector using a \( n \)-length codeword. We are interested in streaming codes defined by (1.1) and (1.2) that consistently guarantee \( \hat{s}_t = s_t \) over a given channel.

**Definition 1.** A feasible streaming code is one where each source packet is correctly recovered within the maximum delay, i.e., \( \hat{s}_t = s_t \) for all \( t \geq 0 \).

If a code is feasible for a given channel, we say that its rate \( R \) is achievable with zero errors.
Definition 2. The streaming capacity of a channel is the supremum of all achievable rates over that channel.

The streaming capacity is dependent on the channel characteristics and the maximum tolerable decoding delay $T$. Although a single code may be feasible over many channels, different channels often require specialized codes to achieve the streaming capacity. For example, consider streaming over a single-link packet erasure channel. At each time instance, the received packet $y_t \in \mathbb{F}_q \cup \{\star\}$ is determined to be

$$y_t = \begin{cases} x_t & \text{correct transmission} \\ \star & \text{erasure} \end{cases}.$$ 

The channel dynamics, or more specifically, how the channel behaves for a sequence of subsequent uses, can significantly affect the streaming capacity. A channel where erasures occur in arbitrary positions possesses a different capacity and thus requires a different code compared to a channel where erasures occur strictly in bursts. Consider the two different channels in Fig. 1.1 which both feature the same number of erasures. In Fig. 1.1b, up to three consecutive packets are erased in a burst. For example, $x_0$ is erased and the receiver observes nothing until $x_3$. Clearly, streaming under a maximum delay $T < 3$ is not feasible. In Fig. 1.1a however, there are no consecutive erasures and the tight delay constraints are acceptable.

Probabilistic channel models may often be difficult to analyze, especially when considering perfect zero-error recovery. Adversarial models are alternatively used to approximate real channels. Many models use a sliding window—any consecutive sequence of transmissions—and enforce a set of permissible loss patterns in each window. Alternatively, a channel can be defined to enforce gaps of perfect communication after a loss pattern in order to reset the decoder. Two examples are given below for the single-link packet erasure channel.

- **Isolated Erasure Sliding Window Channel:** In every sliding window of $W$ consecutive transmissions, the channel can introduce up to $S \leq W$ erasures at any positions [4,19,30]. An example is given in Fig. 1.1a.

- **Burst Erasure Channel:** The channel can introduce at most a single burst of up to $B$ consecutive erasures, but must permit a gap of at least $G \geq B$ perfect transmissions immediately after the burst [3,4,14,19,25]. An example is given in Fig. 1.1b.

Burst erasures versus isolated erasures are the two baseline channel models most often considered for streaming over a single-link erasure channel. Note that the Burst Erasure Channel can be rewritten as a sliding window model simply by defining all windows of length $W > G$ to contain at most one $B$-length
burst [3, 4, 25]. However, we use the classic definition of a burst channel [14], for the intuitiveness it provides in subsequent proofs. More sophisticated sliding window channel models are often conceived as simply extensions or combinations of the two.

An important parameter to consider for streaming over an isolated erasure channel is the column Hamming distance [13, 30]. As a convolutional code extension of the minimum Hamming distance of a block code, the column Hamming distance determines the maximum tolerable number of erasures in a window from which a packet at the beginning of the window is guaranteed to be recoverable by the end. A family of convolutional code extensions to Maximum Distance Separable (MDS) block codes, known as $m$-MDS codes maximize the column Hamming distance and can be shown to achieve the streaming capacity of the Isolated Erasure Sliding Window Channel [2, 8, 11].

Although $m$-MDS codes can recover from bursts, they do not generally achieve the streaming capacity of the Burst Erasure Channel. The capacity there is achievable however, by using a family of specialized burst correcting codes known as Maximally-Short (MS) codes [3, 4]. These codes are constructed by splitting the source packet into two sub-packets and using $m$-MDS codes and repetition codes to independently protect the sub-packets. Selecting the appropriate splitting ratio and code rates permits MS codes to recover every packet in a burst of length $B$ with delay exactly equal to $T$.

1.2 Network Communication

The single-link erasure channel is only a simplified abstraction of more sophisticated communication networks. Rather than a single transmitter and receiver operating in isolation, internet communication is more accurately characterized by a directed graph connecting many nodes, each transmitting and receiving data packets over different links. Each node is directly connected—separated by a single link—to a set of neighbouring nodes. If one node wishes to communicate with another in the network, it transmits data packets to its neighbours, which each transmit to their neighbours, and so on until the destination
node has received the desired packets. We assume that transmission over links occurs instantaneously, i.e., a destination node receives packets at the same time instance that they are transmitted. When using generation-based network coding [5], each packet contains an attached timestamp and all nodes operate in sync. This eliminates the necessity of factoring individual link delays in the model.

A consequence of the graphical multi-node setup is that multicasting is easily permissible, as there may be multiple destination nodes simultaneously intending to recover the source. In multicasting, we assume that all destination nodes possess identical decoding requirements, i.e., they all desire the same source packets within the same tolerable delay. The results in this work are given with respect to a single source and destination node, but with the implication that other destination nodes are simultaneously acting the same.

Links naturally possess a finite capacity, i.e., each link can only transmit one channel packet at each time instance. With multiple paths from a source to a destination node, the source node can transmit multiple channel packets in one transmission. Consequently, the allocation strategy of the outgoing links of each node becomes relevant. A naive method is blind routing, where the source node sends different packets along each outgoing link and each intermediate node randomly picks from its received packets to transmit along its own outgoing links. This method does not achieve the highest throughput—the number of decoded source packets in one network use [1]. As packets are simply mathematical units of information, nodes may alternatively transmit linear combinations of the channel packets. This method, known as linear network coding, does achieve the highest possible throughput [1,10,16]. We provide an example in Fig. 1.2 where a source node transmits the two channel packets $x_0$ and $x_1$ described here as column vectors of a matrix $X$, which $n$ symbols. Naive routing is attempted in Fig. 1.2a, leading to the centre node receiving both channel packets but only transmitting one. Consequently, one destination node always fails to receive both packets. In Fig. 1.2b, a linear network code is implemented and the centre node transmits a linear function of $x_0$ and $x_1$. As a result, the destination node $Y$ receives

$$\begin{pmatrix} y_0 \\ y_1 \end{pmatrix} = \begin{pmatrix} x_0 \\ x_1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}.$$  

Using linear network coding, the destination nodes in a network receive a linear transformation of the transmitted channel packets. In application, the coefficients in the outgoing linear combinations of each node can be marked in header bits of the channel packets, meaning that they are all known at the destination node [10]. We write the transformation or channel matrix as $A \in \mathbb{F}_q^{r \times r}$ and assume that it is known to the decoder. While the matrix may be designed for any dimensions, the network architecture permits a maximum rank $r$ as determined by the min-cut of the graph [1]. Without loss of generality, we assume the matrix to be a square matrix of order $r$. In practice, this value is referred to as the achievable network coding rate.

For simplicity, we further assume that $r \mid n$. Using the channel matrix, the relationship between the source and destination nodes then is written as $Y = XA$, where $X, Y \in \mathbb{F}_q^{2 \times r}$ contain the transmitted and received channel symbols. This collection of packets is referred to as a macro-packet or a generation and denoted as a matrix $X = \left( \begin{array}{c} x_0 \\ \vdots \\ x_{r-1} \end{array} \right)$. A single macro-packet can be transmitted perfectly in one network use or shot. In both linear and random linear network coding, the channel matrix is known to the destination node, either as predetermined coefficients or via header bits of the transmitted packets [10]. When $A$ is full-rank, the destination node inverts the matrix and recovers the channel packets.
Note that the channel packets belong to an *extension field* \( F_{q^M} \), whereas the channel matrix belongs to a *ground field* \( F_q \). The destination node observes linear combinations in \( F_q \) of the channel packets. We assume this setup throughout the work. The value of \( M \geq 0 \) is bounded from above depending on the code.

Extending the single-link packet erasure channel, each link in the network can be considered an erasure channel. When a link transmits an erasure, it is effectively removed from the network for that shot and we say that it is failing or deactivated. Any node receiving an erasure symbol will set the coefficient of that symbol to zero in the outgoing linear combination. Effectively, it is as if the erased link itself is non-existent in the network. The result of a single deactivated link may propagate throughout the network, causing the channel matrix to change to a rank-deficient form. In the example in Fig. 1.3, we show the prior network but with one link failing. Consequently, \( Y \) receives

\[
Y = \begin{pmatrix} y_0 \\ y_1 \end{pmatrix} = \begin{pmatrix} x_0 \\ x_1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix}.
\]

The rank-deficient channel matrix is not invertible and we cannot recover both channel packets at the destination node. From Menger’s Theorem [9, Chapter 4], one deactivated link can damage at most one of the min-cut paths connecting the transmitter and receiver. It follows that the maximum rank of \( A \) is reduced by at most 1 [12]. Some redundancy is now necessary in order to protect the channel packets. One solution is to use an end-to-end code as a precode the channel packets [15,29]. For example, Maximum Rank Distance (MRD) block codes have been shown to be useful for this scenario [7,17,28,29]. In this case, the destination node receives a rank-deficient linear transformation of the codeword symbols and can recover the source when the total rank deficiency does not exceed a maximum tolerable value. Using these codes, we can view network communication as a rank metric extension of communication over the erasure channel.
1.3 Network Streaming

In this work, we study the problem of designing end-to-end codes that guarantee low-delay decoding over networks with deactivated links. The network is modelled as a rank-deficient channel matrix connecting a source and destination node. At each time instance $t$, the source node receives a source packet $s_t \in \mathbb{F}_q^{k_M}$, which is a row vector containing $k$ source symbols over the extension field. The transmitter (of the source node) applies an end-to-end code defined by (1.1) to generate a codeword $x_t \in \mathbb{F}_q^{n_M}$, containing $n$ channel symbols. The codeword is rearranged into a macro-packet of $r$ channel packets, given as column vectors comprising the matrix $X_t \in \mathbb{F}_q^{n_r \times r}$. The destination node receives a linear transformation of the transmitted channel packets $Y_t = X_t A_t$, where $A_t \in \mathbb{F}_q^{r \times r}$ is the channel matrix [15]. Using (1.2) with the received channel packets, the receiver (at the destination node) recovers the source packet within a maximum delay $T$. The network model is summarized in Fig. 1.4.

Each shot is independent of the others. We consider communication over a window of length $W$, similar to the single-link models. From time $t$ to $t + W - 1$, the transmitter-receiver relationship is described

$$
\begin{pmatrix}
Y_t & \cdots & Y_{t+W-1}
\end{pmatrix} =
\begin{pmatrix}
X_t & \cdots & X_{t+W-1}
\end{pmatrix}
\begin{pmatrix}
A_t & & \\
& \ddots & \\
& & A_{t+W-1}
\end{pmatrix}
$$

(1.3)

using a block diagonal channel matrix [26]. The notation $\text{diag}(A_t, \ldots, A_{t+W-1})$ describes the above channel matrix.

Let $\rho_i = \text{rank}(A_i)$. Due to the block diagonal structure, the total rank of the channel in a window is equal to the sum rank of the individual channels, i.e., $\sum_{i=t}^{t+W-1} \rho_i = \text{rank}(\text{diag}(A_t, \ldots, A_{t+W-1}))$. When operating without any failures, the network yields $\rho_i = r$, but deactivated links may cause a rank-deficient channel matrix. It remains to determine the channel dynamics. In this work, we focus on two different channel characteristics: one featuring isolated rank losses in the channel matrices within each window and one featuring strictly burst loss patterns. We propose adversarial network models that are rank-metric analogues of the Isolated Erasure Sliding Window Channel and the Burst Erasure Channel introduced in Section 1.1. We derive desirable code properties before constructing end-to-end codes that achieve the streaming capacity in both scenarios. Numerical analysis detail both the feasibility of construction in terms of the necessary field sizes and provide simulations over statistical channels.

1.4 Outline

This thesis is outlined as follows. Preliminaries are provided in Chapter 2. We review fundamentals of finite fields and the matrix property of super-regularity, which becomes important in our code constructions. Two important families of codes, $m$-MDS codes and MRD codes, are introduced as progenitors to our codes.

In Chapter 3, we construct the Isolated Rank Loss Sliding Window Network as a generalization of the Isolated Erasure Sliding Window Channel. We define the column sum rank of a code and propose several interesting properties. A family of codes referred to as Maximum Sum Rank (MSR) codes are introduced and shown to achieve the maximum column sum rank. We show that these codes achieve perfect recovery over the Isolated Rank Loss Network and discuss practical constructions.
Figure 1.4: The Encoder and Decoder are connected by $r$ disjoint paths, determined by the min-cut of the graph. Each path is effectively responsible for a channel packet—column of $X_t$—at each time instance. Each received packet is a linear combination of the transmitted packets.

We construct the Burst Rank Loss Network as a generalization of the Burst Erasure Channel in Chapter 4. The streaming capacity is stated and compared to the achievable rates of MSR and MRD codes. We show that these codes are only capacity achieving for special cases. A new family of codes referred to as Recovery Of Bursts In Networks (ROBIN) codes are introduced and constructed by splitting the source packet and applying MSR and MRD codes to the two sub-packets. The decoding steps are detailed along with an example. We prove the converse and conclude with simulations over statistical network models.

The thesis is concluded in Chapter 5, where we summarize our results and discuss several areas for future work.
Chapter 2

Preliminaries

2.1 Notation

The following notation is used throughout this work. We count all indices of a vector, matrix, or sequence starting from the 0-th entry. A vector $\mathbf{x}_t = (x_{t,0}, \ldots, x_{t,n-1})$ is denoted in lower-case bold text with its entries italicized. A matrix $\mathbf{A}_t$ is written in upper-case bold text. The first subscript generally indicates the time index and any subsequent subscript refers to the entry index when necessary.

A sequence of vectors over time is written with bracket subscripts as $\mathbf{x}_{[t,t+j]} = (x_{t}, \ldots, x_{t+j})$. We describe block diagonal (channel) matrices similarly, i.e., $\mathbf{A}_{[t,t+j]} = \text{diag} (\mathbf{A}_t, \ldots, \mathbf{A}_{t+j})$. A sequence of identical matrices are sometimes used to form a block diagonal matrix in this work. We use the notation $\text{diag} (\mathbf{A}_t)^n = \text{diag} (\mathbf{A}_t, \ldots, \mathbf{A}_t)$ for the block diagonal matrix comprised of $n$ copies of $\mathbf{A}_t$.

Vectors are assumed to be row vectors unless otherwise stated. We denote $\mathbf{I}_{j \times j}$ as the $j \times j$ identity matrix. The $j \times k$ all-zero matrix is written $\mathbf{0}_{j \times k}$.

2.2 Finite Fields

Let $q$ be a prime power and $\mathbb{F}_q$ the finite field with $q$ elements. This field is referred to as the ground field. For $M \geq 0$, let $\mathbb{F}_q^M$ be an extension field. A primitive element $\alpha \in \mathbb{F}_q^M$ is one whose consecutive powers can generate all non-zero elements of that field, i.e., $\mathbb{F}_q^M = \{0, \alpha, \alpha^2, \ldots, \alpha^{q^M-1}\}$ [21, Chapter 2].

Let $\mathbb{F}_q[X]$ be a polynomial ring of the ground field. The minimal polynomial of any element $\alpha \in \mathbb{F}_q^M$ is the lowest degree monic polynomial $p_{\alpha}(X) \in \mathbb{F}_q[X]$ for which $\alpha$ is a root. By definition, the minimal polynomial is irreducible. When $\alpha$ is a primitive element, $p_{\alpha}(X)$ possesses degree equal to $M$ and is referred to as a primitive polynomial.

Lemma 1. If $f(\alpha) = 0$ for any $f(X) \in \mathbb{F}_q[X]$, then $p_{\alpha}(X) \mid f(X)$.

Proof. The proof can be found in [22, Chapter 4]. If we divide $f(X) = p_{\alpha}(X)q(X) + r(X)$ to have a remainder term, then $\text{deg} r(X) < \text{deg} p_{\alpha}(X)$. However, $f(\alpha) = 0$ implies that $r(\alpha) = 0$, which contradicts $p_{\alpha}(X)$ being the minimal polynomial. \qed

The extension field $\mathbb{F}_q^M$ is isomorphic to the $M$-dimensional vector space $\mathbb{F}_q^M$ over the ground field. Let $\alpha_0, \ldots, \alpha_{M-1} \in \mathbb{F}_q^M$ map to a basis for the vector space. A basis is defined as being normal when each $\alpha_i = \alpha^q$ for some $\alpha \in \mathbb{F}_q^M$. The generating element $\alpha$ is referred to as a normal element. Note
that a normal element is not necessarily primitive. We use the notation $\alpha[i] = \alpha^{q^i}$ to describe the $i$-th Frobenius power of $\alpha$. Just as any vector in $\mathbb{F}_q^M$ can be written as a linear combination of the basis vectors, every element $f \in \mathbb{F}_q^M$ can be written as a linear combination in $\mathbb{F}_q$ of the basis elements. Using the normal basis, $f$ becomes a linearized polynomial $f(X) = \sum_{i=0}^{M-1} f_i X[i] \in \mathbb{F}_q[X]$ evaluated at the normal element $X = \alpha$. The coefficients of this polynomial can be mapped

$$f(\alpha) = \sum_{i=0}^{M-1} f_i \alpha[i] \mapsto f = (f_0, \ldots, f_{M-1})^T$$

(2.1)

to the entries of a unique vector $f \in \mathbb{F}_q^{M \times 1}$. In a linearized polynomial, every monomial term has a Frobenius power. A linearized polynomial possesses a $q$-degree, which is defined as the largest Frobenius power.

The mapping between the extension field and vector space can be extended to vector spaces over the extension field and matrix spaces over the ground field. A vector of linearized polynomials maps to a matrix whose columns correspond to the coefficients of each polynomial. Using (2.1), we define the mapping between the vector and matrix spaces as

$$\phi_n : \mathbb{F}_q^n \rightarrow \mathbb{F}_q^M \times n$$

$$\left((f_0(\alpha), \ldots, f_{n-1}(\alpha)) \mapsto \left( \begin{array}{c|c} f_0 & \vdots & f_{n-1} \end{array} \right) \right).$$

(2.2)

For every finite extension of a finite field, there exists at least one element that is both a normal and a primitive element [18]. Such an element is referred to as being primitive normal.

### 2.3 Super-regular Matrices

For $b \in \mathbb{N}$, let $\sigma$ be a permutation of the set $\{0, \ldots, b-1\}$. A permutation is comprised of a series of transpositions, which are defined as two entries of the set switching positions. For example, the permutation producing $\{2, 3, 0, 1\}$ from the set with $b = 4$ is constructed using two transpositions. The sign function of a permutation measures its parity, i.e., $\text{sgn} \sigma$ is equal to 1 when $\sigma$ is constructed from an even number of transpositions, and equal to $-1$ otherwise. Let $S_b$ denote the set of all possible permutations. The determinant of a matrix $D = [D_{i,j}]$ of order $b$ can be calculated by summing over all elements of $S_b$ in the Leibniz formula

$$\det D = \sum_{\sigma \in S_b} \text{sgn} \sigma \prod_{i=0}^{b-1} D_{i,\sigma(i)}.$$  

(2.3)

Each product $\prod_{i=0}^{b-1} D_{i,\sigma(i)}$ is referred to as a term in the summation. When every term is equal to 0, the matrix is said to have a trivial determinant. An obvious example of such a matrix is one with an all-zero row or column, but this is not a necessary feature. For example, the determinant of the matrix

$$\begin{pmatrix} 1 & 2 & 3 \\ 0 & 0 & 4 \\ 0 & 0 & 5 \end{pmatrix}$$
is trivially zero. The Leibniz formula definition of trivially zero determinants is further used to define a super-regular matrix.

**Definition 3.** A super-regular matrix is a matrix for which every square sub-matrix with a non-trivial determinant is non-singular.

One of the most well-known class of super-regular matrices are Cauchy matrices [27, Chapter 11]. These matrices are often used as the parity-check blocks $\mathbf{P}$ in the generator matrices $\mathbf{G} = \left( \mathbf{I}_{k \times k} \quad \mathbf{P} \right)$ of systematic MDS codes. In fact, super-regularity can be shown to be a necessary property for any parity-check block of a systematic MDS generator matrix [27, Chapter 11].

In this work, we are interested in super-regular matrices with the block Hankel structure

$$
\mathbf{T} = \begin{pmatrix}
\mathbf{T}_0 & \mathbf{T}_1 & \cdots & \mathbf{T}_m \\
\vdots & \vdots & \ddots & \vdots \\
\mathbf{T}_0 & \cdots & \mathbf{T}_{m-1} & \mathbf{T}_m
\end{pmatrix},
$$

(2.4)

where $\mathbf{T}_j \in \mathbb{F}_{q^m}^{n \times n}$ for $j = 0, \ldots, m$. Although super-regular matrices with a pre-determined structure have not been studied as thoroughly as unrestricted super-regular matrices, some properties and constructions are known [2, 8, 11]. In [8], the authors proved that a Toeplitz super-regular matrix of any finite size exists over every ground field $\mathbb{F}_q$. Several interesting preservative properties were introduced in [11]. A construction of a block Hankel matrix was proposed in [2] as super-regular over a sufficiently large extension field. We outline this construction below.

**Theorem 1** (Almeida et al., [2]). For $n, m \in \mathbb{N}$, let $M = q^{n(m+2)-1}$. Let $\alpha \in \mathbb{F}_{q^m}$ be a primitive element and a root of the minimal polynomial $p_\alpha(X)$. For $j = 0, \ldots, m$, the blocks $\mathbf{T}_j \in \mathbb{F}_{q^m}^{n \times n}$ are defined

$$
\mathbf{T}_j = \begin{pmatrix}
\alpha^{[nj]} & \alpha^{[nj+1]} & \cdots & \alpha^{[n(j+1)-1]} \\
\alpha^{[nj+1]} & \alpha^{[nj+2]} & \cdots & \alpha^{[n(j+1)]} \\
\vdots & \vdots & \ddots & \vdots \\
\alpha^{[n(j+1)-1]} & \alpha^{[n(j+1)]} & \cdots & \alpha^{[n(j+2)-2]}
\end{pmatrix},
$$

(2.5)

The block Hankel matrix $\mathbf{T}$ in (2.4) using the constituent blocks in (2.5) is a super-regular matrix.

The complete proof is available in [2], but we outline the steps below. Each non-zero entry of the matrix is a linearized monomial $\bar{T}_{i,j}(X)$ evaluated at $X = \alpha$. The $q$-degrees of these monomials increase as one moves further down and to the right inside $\mathbf{T}$. We use the following inequalities:

$$
\deg_q \bar{T}_{i',j}(X) < \deg_q \bar{T}_{i,j}(X), \text{ if } i' < i
$$

(2.6a)

$$
\deg_q \bar{T}_{i,j'}(X) < \deg_q \bar{T}_{i,j}(X), \text{ if } j' < j.
$$

(2.6b)

We use the variable $X$ when discussing properties of linearized polynomials and evaluate the polynomials at $X = \alpha$ specifically when calculating the determinant of a matrix. Let $\mathbf{D} = [D_{i,j}(\alpha)]$ be any square sub-matrix of $\mathbf{T}$ with order $b$. All sub-matrices of $\mathbf{T}$ preserve the degree inequalities in (2.6). From Definition 3, if $\mathbf{T}$ is super-regular and $\mathbf{D}$ has a non-trivial determinant, then $\det \mathbf{D} \neq 0$. 

The Leibniz formula solves for the determinant. Each non-zero term \( D_\sigma(\alpha) = \prod_{i=0}^{k-1} D_{i,\sigma(i)}(\alpha) \) is a product of linearized monomials, although these terms themselves are not linearized. The determinant is now a polynomial \( D(\alpha) = \sum_{\sigma \in S_r} \text{sgn} \sigma D_\sigma(\alpha) \). Using (2.6), the degree of this polynomial can be bounded.

**Lemma 2** (Almeida et al., [2]). For \( \bar{T} \) defined in Theorem 1, let \( D \) be any square sub-matrix with a non-trivial determinant. Let \( D(X) \) be the polynomial, which when \( X = \alpha \), evaluates its determinant. The degree of \( D(X) \) is bounded

\[
1 \leq \deg D(X) < q^{n(m+2)-1}.
\]

In [2], the matrices defined in (2.5) contained entries with Frobenius powers \( \alpha^{2^i} \), where \( q = 2 \) is fixed, rather than for a general \( q \). As a result, the original work used the upper bound bound \( \deg D(X) < 2^{n(m+2)-1} \). A direct extension reveals that using Frobenius powers with arbitrary \( q \) for (2.5) results in a new upper bound that is less tight than in the original work. As a consequence, Theorem 1 as provided above requires a larger field size than the original introduction in [2]. It is not immediately obvious why the general construction which introduces slackness on the bound on the field size is tolerated. We show in later chapters however, that with an arbitrary \( q \) and an appropriate choice of the primitive element \( \alpha \), the super-regular blocks \( T_j \) may be constructed to behave as Gabidulin code generator matrices [7]. This property becomes relevant in our code constructions.

For completeness, we prove our upper bound in Lemma 2 in Appendix A.1. The lower bound does not change in our extension, so we refer the reader to [2] for the full proof. The lower bound implies that \( D(X) \) is not the zero polynomial and is derived by algorithmically finding a unique permutation \( \bar{\sigma} = \arg \max_\sigma \deg D_\sigma(X) \), which generates the highest degree monomial term in the Leibniz formula. Because \( \bar{\sigma} \) is unique, there is no other term which can negate this monomial. Then, \( \deg D(X) = \deg D_{\bar{\sigma}}(X) \), meaning \( D(X) \) is not the zero polynomial.

The upper bound gives \( \deg D(X) < \deg p_\alpha(X) \), and consequently, \( p_\alpha(X) \nmid D(X) \). By Lemma 1, \( \alpha \) is not a root of \( D(X) \). The lower and upper bounds cover both potential cases to ensure \( D(\alpha) \neq 0 \). Therefore, any \( D \) with a non-trivial determinant is also non-singular and consequently, \( \bar{T} \) is a super-regular matrix.

### 2.4 \( m \)-MDS Codes

Let \( C[n,k,m] \) be a linear time-invariant convolutional code with memory \( m \). For a source sequence \( s_{[0,j]} = (s_0, \ldots, s_j) \in \mathbb{F}_q^{k(j+1)} \), the codeword sequence \( x_{[0,j]} = s_{[0,j]} G_{j}^{\text{EX}} \) is determined using the extended form generator matrix

\[
G_{j}^{\text{EX}} = \begin{pmatrix}
G_0 & G_1 & \ldots & G_j \\
G_0 & \ldots & G_{j-1} \\
\vdots & & \ddots & \ddots \\
G_0 & & & \ddots & \ddots \\
& & & \ddots & \ddots \\
& & & & \ddots & \ddots \\
& & & & & \ddots & \ddots \\
& & & & & & \ddots & \ddots \\
& & & & & & & \ddots & \ddots \\
& & & & & & & & \cdots & \cdots \\
& & & & & & & & & \cdots & \cdots \\
& & & & & & & & & & \cdots & \cdots \\
& & & & & & & & & & & \cdots & \cdots \\
& & & & & & & & & & & & \cdots & \cdots \\
& & & & & & & & & & & & & \cdots & \cdots \\
& & & & & & & & & & & & & & \cdots & \cdots \\
& & & & & & & & & & & & & & & \cdots & \cdots \\
& & & & & & & & & & & & & & & & \cdots & \cdots \\
& & & & & & & & & & & & & & & & & \cdots & \cdots \\
& & & & & & & & & & & & & & & & & & \cdots & \cdots \\
\end{pmatrix}
\tag{2.7}
\]

where \( G_j \in \mathbb{F}_q^{k \times n} \) and \( G_j = 0 \) for \( j > m \) [13, Chapter 1]. We assume from here on that \( G_0 \) always has full row rank. This guarantees that \( G_j^{\text{EX}} \) also possesses full row rank, which is a property used in
subsequent results.

The Hamming weight of a vector \( wt_H(x_t) \) measures the number of non-zero entries. The Hamming weight of the codeword sequence \( x_{[0,j]} \) is a sum of the Hamming weight of each codeword, i.e.,

\[
wt_H(x_{[0,j]}) = \sum_{t=0}^{j} wt_H(x_t).
\]

A useful property of convolutional codes is the \( j \)-th column distance of a code

\[
d_H(j, C) = \min_{x_{[0,j]} \in C, s_0 \neq 0} wt_H(x_{[0,j]}),
\]

which is defined as the minimum Hamming weight amongst codewords whose initial source \( s_0 \) is non-zero [8]. The notation is simplified to \( d_H(j) \) when \( C \) is obvious. Note that because \( G_0 \) is full-rank, \( s_0 \neq 0 \) immediately implies that \( x_0 \neq 0 \) as well.

Several properties of the column Hamming distance were treated in [8, 30]. We summarize three relevant ones below. In Chapter 3, we prove rank metric analogous properties to these.

**Property 1 (Tomas et al., [30]).** Consider an erasure channel where the prior source sequence \( s_{[0,t-1]} \) is known to the decoder by time \( t+j \). If there are at most \( d_H(j) - 1 \) symbol erasures in the window \([t, t+j]\), then \( s_t \) is recoverable by time \( t+j \). Conversely, there is at least one source sequence for which \( s_t \) is not recoverable by time \( t+j \) if there are \( d_H(j) \) erasures in the window \([t, t+j] \).

Property 1 states that a convolutional code with a given column Hamming distance \( d_H(W-1) \) is feasible of guaranteeing perfect decoding with delay \( W-1 \) over any Isolated Erasure Sliding Window Channel with at most \( S < d_H(W-1) \) erasures in every length \( W \) window.

**Property 2 (Gluessing-Luerssen et al., [8]).** The \( j \)-th column Hamming distance of a code is upper bounded \( d_H(j) \leq (n-k)(j+1) + 1 \). If \( d_H(j) \) meets this bound with equality, then \( d_H(i) \) meets its respective bound for all \( i \leq j \).

This property asserts a convolutional code extension of the Singleton bound. Property 1 states the desirability of maximizing the column Hamming distance. In addition, a convolutional code may possess a profile of maximum column Hamming distances. Achieving the Singleton bound for the \( j \)-th column Hamming distance implies that all prior column distances also attain their respective upper bounds. This permits low-delay recovery for all \( i \leq j \).

**Property 3 (Gluessing-Luerssen et al., [8]).** Consider the set of \( k(j+1) \times k(j+1) \) full-sized minors of \( G_j^{\text{EX}} \) formed from the columns indexed \( 0 \leq t_0 < \cdots < t_{k(j+1)-1} \), where \( t_s k > s n - 1 \) for \( s = \{0, \ldots, j \} \). Then, \( d_H(j) = (n-k)(j+1) + 1 \) if and only if every such minor is non-zero.

Property 3 provides a generator matrix property of convolutional codes that attain the upper bound for \( d_H(j) \). It is a convolutional extension of the MDS block code generator property that every \( k \times k \) full-sized minor must be non-zero [27, Chapter 11]. The lower bound on the indices in Property 3 is meant to prevent constructing trivial minors of \( G_j^{\text{EX}} \), which invariably exist due to the block triangular structure of the generator matrix. If there are any zeros in the main diagonal of any sub-matrix, then the determinant can be shown to be trivially zero [8]. Note however that the super-regularity of \( G_j^{\text{EX}} \) implies that the full-size minors are non-zero. By Definition 3 then, if \( G_j^{\text{EX}} \) is super-regular, Property 3 is implied and \( d_H(j) \) achieves its Singleton bound.
There exist several families of convolutional codes that attain the maximum $d_H(j)$ at some point. An interesting family known as $m$-MDS codes achieve $d_H(m) = (n - k)(m + 1) + 1$, where $m$ is the code memory. We will use a super-regular $G_{m}^{\text{EX}}$ to generate the code.

The classical means of decoding convolutional codes is through the Viterbi algorithm. Using the State Trellis graph of the convolutional code, the Viterbi algorithm considers a sequence of received packets $y_{[0,t+j]}$ and estimates the most likely sequence of states through which the encoder may have traversed. Consequently, the complexity of recovering a sequence $s_{[0,t]}$ grows linearly with the number of edges in the Trellis graph, i.e., $O(tq^{M(n+k)})$ operations over the extension field. Given however that we primarily consider erasure channels, the decoding problem is alternatively reducible to solving the linear system of equations of the received symbols [30]. Using Property 1, the objective is recovering the $d_H(j) - 1$ erased symbols in the window $[t,t+j]$. The system is a matrix of order $d_H(j) - 1$, which is invertible in $O(d_H(j)^3)$ field operations. We refer the reader to [30] for additional optimizations that can be performed for $m$-MDS convolutional codes over an erasure channel.

2.5 Rank Metric Codes

Consider a vector $x \in \mathbb{F}_{q^M}^n$ over the extension field. The rank of the vector $x$ is defined as the rank of its associated matrix $\phi_n(x)$. For any two vectors $x, \hat{x} \in \mathbb{F}_{q^M}^n$, the rank distance is

$$d_R(x, \hat{x}) = \text{rank} (\phi_n(x) - \phi_n(\hat{x})).$$

It is well known that the rank distance is a metric [7], and is upper bounded by the Hamming distance. For any linear block code $C[n,k]$, the minimum rank distance

$$d_R(C) = \min_{x \in C, x \neq 0} \text{rank} (\phi_n(x))$$

is the smallest rank amongst all non-zero codewords. The minimum rank distance is an analogue of the minimum Hamming distance of a code and possesses similar properties. For example, the minimum rank distance of a code must satisfy a Singleton-like bound $d_R \leq \min \{1, \frac{M}{n} \} (n - k) + 1$ [7]. We assume that $M \geq n$ from here on; $d_R$ is now bounded exactly by the Singleton bound. Any code that meets this bound with equality is referred to as a Maximum Rank Distance (MRD) code. Such codes possess the following property.

**Theorem 2** (Gabidulin, [7]). Let $G \in \mathbb{F}_{q^M}^{k \times n}$ be the generator matrix of an MRD code. The product of $G$ with any full-rank matrix $A \in \mathbb{F}_q^{n \times k}$ achieves rank $GA = k$.

*Proof.* Suppose that there exists a full-rank $A \in \mathbb{F}_q^{n \times k}$ for which rank $(GA) < k$. Then there exists a non-zero source vector $s \in \mathbb{F}_{q^M}^k$ such that $sGA = xA = 0$. This however implies that rank $(\phi_n(x)) \leq n - k$, which is a contradiction. \qed

The original theorem in [7] showed that the parity-check matrix $H \in \mathbb{F}_{q^M}^{(n-k) \times n}$ multiplied with a full-rank $A \in \mathbb{F}_q^{n \times (n-k)}$ always yields a product whose rank is $n - k$. However, we prefer the complementary theorem for the generator matrix in this work.

When $M \geq n$, every MRD code is also an MDS code. Gabidulin codes are an important family of codes that are MRD and hence, also MDS. To construct a Gabidulin code, let $g_0, \ldots, g_{n-1} \in \mathbb{F}_{q^M}$ be a
set of elements that are linearly independent over $\mathbb{F}_q$. The generator matrix for a Gabidulin code $C[n, k]$ is

$$G = \begin{pmatrix} g_0 & g_1 & \cdots & g_{n-1} \\ g_0^{[1]} & g_1^{[1]} & \cdots & g_{n-1}^{[1]} \\ \vdots & \vdots & \ddots & \vdots \\ g_0^{[k-1]} & g_1^{[k-1]} & \cdots & g_{n-1}^{[k-1]} \end{pmatrix}.$$ 

In practice, $g_i$ can be drawn from a subset of a normal basis, i.e., $g_i = g_i^{[i]}$ for some normal element $g \in \mathbb{F}_{q^M}$. Gabidulin codes with this construction are known as $q$-cyclic Gabidulin codes and possess several additional properties compared to the general construction [7].

We use a systematic encoder for Gabidulin codes when feasible. A systematic generator matrix $G = \begin{pmatrix} I_{k \times k} & P \end{pmatrix}$ can be constructed from any non-systematic Gabidulin code by means of Gaussian elimination. Codewords of a systematic code are written as $x_t = (s_t, p_t)$, where $s_t \in \mathbb{F}_{q^M}^k$ are the message symbols and $p_t \in \mathbb{F}_{q^M}^{n-k}$ are the parity-check symbols.
Chapter 3

Maximum Sum Rank Codes

3.1 Introduction

In this chapter, we address network streaming under isolated losses. For the single-link Isolated Erasure Sliding Window Channel, the defining parameter is the maximum number of erasures that can be tolerated in a window [4, 30]. In the network setting, we count instead the total rank deficiency in any window. Just as the column Hamming distance is the relevant metric in single-link streaming, we introduce a rank metric analogue: the column sum rank. A new family of convolutional codes known as Maximum Sum Rank (MSR) codes are proposed. MSR codes aim to maximize the column sum rank similarly to how $m$-MDS codes maximize the column Hamming distance.

An alternative family of rank metric convolutional codes was proposed in [31]. The works there focused on a slightly different code metric: the active column sum rank. Upon expanding on the difference between the column sum rank and the active column sum rank, we show that our metric, unlike the active variant, is both sufficient and necessary to evaluate whether a code is feasible for the network extension to the Isolated Erasure Sliding Window Channel.

This chapter is outlined as follows. The network streaming problem is introduced in Section 3.2. In Section 3.3, we define and derive several properties for the column sum rank. We introduce MSR codes and provide a construction in Section 3.4. Here, a new class of matrices over the extension field that preserve super-regularity after multiplication with block diagonal matrices in the ground field are introduced. We conclude the chapter with a discussion on practical constructions, examples, and the necessary field size.

3.2 Problem Setup

The streaming problem is defined in three steps: the encoder, network model, and decoder. The general model described in Sections 1.1 and 1.3 is expanded below.

3.2.1 Encoder

At each time instance $t \geq 0$, a source packet $s_t \in \mathbb{F}_q^n$ arrives at a source node in the network. A codeword $x_t \in \mathbb{F}_q^n$ is generated using (1.1). We consider the class of linear time-invariant encoders
for our scenario and use convolutional codes with the structure in (2.7). A rate $R = \frac{k}{n}$ encoder with memory $m$ generates the codeword

$$x_t = \sum_{i=0}^{m} s_{t-i} G_i.$$  \hspace{1cm} (3.1)

For this chapter, we assume without loss of generality that $n = r$, where $r$ is the achievable network coding rate derived from the min-cut of the network. A codeword consequently consists of $n$ channel packets, each of which is a symbol rather than a vector (as initially defined). The extension to vector channel packets is trivial and briefly addressed in Chapter 4. Note that $x_t = X_t$ without any reshaping. Consequently, we use the vector notation throughout this chapter.

### 3.2.2 Network Model

The codeword is instantaneously transmitted over the network, facing zero delay over the links. The destination node sequentially observes $y_t = x_t A_t$, where $A_t \in F_{q}^{r \times r}$ is the channel matrix at time $t$. Communication over a window $[t, t+W-1]$ of $W$ shots is given using a block diagonal channel matrix as in (1.3).

Let $\rho_t = \text{rank} (A_t)$. The sum rank of the individual channels is equal to the total rank of the channel in the window, i.e., $\sum_{i=t}^{t+W-1} \rho_i = \text{rank} (A_{[t,t+W-1]})$. A perfectly operating network yields $\rho_t = r$, but unreliable connections may result in a rank-deficient channel matrix. To facilitate the extension from the single-link streaming problem, we introduce a network variation of the Isolated Erasure Sliding Window Channel by using rank deficiencies in place of symbol erasures. In Fig. 3.1, an example is provided of the network defined below.

**Definition 4.** Consider a network where for all $t \geq 0$, the receiver observes $y_t = x_t A_t$, with rank $(A_t) = \rho_t \leq r$. The **Isolated Rank Loss Sliding Window Network** $\mathcal{CH}_I(r, S, W)$ has the property that in any sliding window of length $W$, the rank of the block diagonal channel matrix decreases by no more than $S$, i.e., $\sum_{i=t}^{t+W-1} \rho_i \geq rW - S$.

In analysis, we disregard the linearly dependent columns of $A_t$ and the associated received symbols. At each time instance, the receiver effectively observes $y_t^* = x_t A_t^*$, where $A_t^* \in F_{q}^{r \times \rho_t}$ is the reduced channel matrix containing only the linearly independent columns.

### 3.2.3 Decoder

Let $T$ be the maximum delay permissible by the receiver node. A packet received at time $t$ must be recovered by time $t + T$ using the delay-constrained decoder in (1.2). If $\hat{s}_t = s_t$, then the source packet is perfectly recovered by the deadline; otherwise, it is declared lost. Using the linear model, we reduce the problem to inverting the transfer matrix between the source and received channel packets. From
Chapter 3. Maximum Sum Rank Codes

17

Figure 3.2: A hypothetical transition along the state Trellis graph is highlighted. The codeword sequence \(x_{[0,5]}\) generated by this path is not a valid sequence for the active column sum rank, but is valid for the column sum rank.

Definition 1, a linear code \(C\) is feasible over \(\mathcal{CH}_1(r, S, W)\) if there exists an encoder and decoder pair, which completely recovers every source packet with delay \(T\). We assume that \(T = W - 1\) for the majority of this chapter and design convolutional codes with memory \(m = T\). Coding for channels where \(T \neq W - 1\) is briefly addressed near the end. Our codes maximize the column sum rank, which is introduced in the next section.

3.3 Maximum Sum Rank Codes

Using the bijection \(\phi_n(\cdot)\) in (2.2) to map vectors in the extension field to matrices in the ground field, the sum rank distance between codeword sequences \(x_{[0,j]}\) and \(\hat{x}_{[0,j]}\) is defined as the sum of the rank distance between each \(x_t\) and \(\hat{x}_t\). We introduce the \(j\)-th column sum rank of a code

\[
d_R(j, C) = \min_{x_{[0,j]} \in C, x_0 \neq 0} \sum_{t=0}^{j} \text{rank} (\phi_n(x_t))
\]

(3.2)
as an analogue of the column Hamming distance.

The column sum rank can alternatively be seen as the minimum sum rank over all codeword sequences that are generated upon exiting the zero state of the State Trellis graph at time 0. We remark that this is not the only rank metric studied for convolutional codes. Using an alternative Trellis graph approach, the active column sum rank was proposed in a prior work [31]. The \(j\)-th active column sum rank is the minimum sum rank amongst codeword sequences that are generated by exiting the zero state at time 0 and not re-entering it between time 1 and \(j - 1\). Naturally, this is a more restrictive set of codeword sequences than that considered in (3.2), which includes codeword sequences that return back to the zero state before time \(j\). An example is provided in Fig. 3.2, where a potential path along the State Trellis graph is highlighted. The path in question generates a codeword sequence considered for the column sum rank but not the active sum rank. We show in the following theorem that the column sum rank, rather than its active variant is both necessary and sufficient for network streaming. The codes proposed in this Chapter consequently aim to maximize the column sum rank.

Theorem 3. Let \(C[n, k, m]\) be a convolutional code used over the window \([0, W - 1]\). For \(t = 0, \ldots, W - 1\), let \(A_t^* \in \mathbb{F}_q^{r \times p}\) be full-rank matrices and \(A_{[0,W-1]}^* = \text{diag} (A_0^*, \ldots, A_{W-1}^*)\) be a channel matrix.
1. If \( d_R(W - 1) > nW - \sum_{t=0}^{W-1} \rho_t \), then \( s_0 \) is always recoverable by time \( W - 1 \).

2. If \( d_R(W - 1) \leq nW - \sum_{t=0}^{W-1} \rho_t \), then there exists at least one codeword sequence and channel matrix for which \( s_0 \) is not recoverable by time \( W - 1 \).

**Proof.** Consider two source sequences \( s_{[0,W-1]} = (s_0, \ldots, s_{W-1}) \) and \( \hat{s}_{[0,W-1]} = (\hat{s}_0, \ldots, \hat{s}_{W-1}) \), where \( s_0 \neq \hat{s}_0 \). Suppose they respectively generate the codeword sequences \( x_{[0,W-1]} \) and \( \hat{x}_{[0,W-1]} \), where \( x_{[0,W-1]}A^*_t [0,W-1] = \hat{x}_{[0,W-1]}A^*_t [0,W-1] \). Then, \( (x_{[0,W-1]} - \hat{x}_{[0,W-1]})A^*_t [0,W-1] = 0 \). The difference sequence \( x_{[0,W-1]} - \hat{x}_{[0,W-1]} \) is a hypothetical codeword sequence whose sum rank is at least \( d_R(W - 1) \). However, \( (x_t - \hat{x}_t)A^*_t = 0 \) implies \( \text{rank}(\phi_n(x_t) - \phi_n(\hat{x}_t)) \leq n - \text{rank}(A^*_t) \) for \( t \leq W - 1 \). By summing each of the inequalities, we arrive at the following contradiction on the sum rank of the codeword sequence:

\[
\sum_{t=0}^{W-1} \text{rank}(\phi_n(x_t) - \phi_n(\hat{x}_t)) \leq nW - \sum_{t=0}^{W-1} \rho_t < d_R(W - 1).
\]

For the converse, let \( s_{[0,W-1]} = (s_0, \ldots, s_{W-1}) \), where \( s_0 \neq 0 \), generate \( x_{[0,W-1]} \) with sum rank equal to \( d_R(W - 1) \). For \( t = 0, \ldots, W - 1 \), let \( \rho_t = n - \text{rank}(\phi_n(x_t)) \). There exist matrices \( A^*_t \in F_q^{r \times \rho_t} \) such that each \( x_tA^*_t = 0 \). By summing all \( \rho_t \), the matrix \( A^*_{[0,W-1]} = \text{diag}(A^*_0, \ldots, A^*_W) \) must have rank equal to \( nW - d_R(W - 1) \) and \( s_{[0,W-1]} \) is indistinguishable from the all-zero source sequence over this channel. \qed

When using time-invariant encoders, Theorem 3 guarantees that all source packets are recovered with delay less than \( W \) over a sliding window channel. Assuming all prior packets have been decoded, we recover each \( s_t \) using the window \([t, t+W-1]\). The contributions of \( s_{[0,t-1]} \) are negated from the received packet sequence used for decoding. Theorem 3 is effectively a rank metric analogue to Proposition 1, which describes how column Hamming distance bounds the number of tolerable erasures in single-link streaming [30].

**Remark 1.** Aside from rank-deficient channel matrices, adversarial errors can also be considered using rank metric codes. Consider a single-link single-shot system where the receiver observes \( y = x + e \), with \( e \in F_q^M \) being an additive error vector. The decoder for an MRD code can recover the source if \( \text{rank}(e) \leq \frac{d_R - 1}{2} \) [7]. MRD codes reflect a rank analogue of the error correcting capability of MDS codes. The column sum rank operates in similar fashion to guarantee low-delay recovery over the error channel. We leave this discussion to be continued in Appendix A.2.

In the following lemma, we propose an analogue to Proposition 2. The column sum rank can be bounded \( d_R(j) \leq (n-k)(j+1) + 1 \). The sum rank of a codeword sequence cannot exceed its Hamming weight, meaning that the upper bound on column Hamming distance is inherited by the rank metric. Furthermore, if the \( j \)-th column sum rank achieves the Singleton bound, all prior column sum ranks also attain their respective bounds.

**Lemma 3.** If \( d_R(j) = (n-k)(j+1) + 1 \), then \( d_R(i) = (n-k)(i+1) + 1 \) for all \( i \leq j \).

**Proof.** It suffices to prove for \( i = j - 1 \). Let \( C[n,k,m] \) be a code for which \( d_R(j-1) \) is at most \( (n-k)j \), but \( d_R(j) \) achieves the maximum. Consider the source sequence \( s_{[0,j-1]} \) which generates \( x_{[0,j-1]} \) with
sum rank equal to $d_R(j-1)$. We next determine $s_j$ to obtain $x_j = \sum_{t=0}^{j-1} s_t G_{j-t} + s_j G_0$. The summation up to $j - 1$ produces a vector whose Hamming weight is at most $n$. Because $\text{rank}(G_0) = k$, $s_j$ can be selected specifically in order to negate up to $k$ non-zero entries of the first summation. This implies that $\text{wt}_H(x_j) \leq n - k$ and consequently, $\text{rank}(x_j) \leq n - k$. Therefore, the sum rank of $x_{[0,j]}$ is upper bounded by $d_R(j-1) + n - k \leq (n - k)(j + 1)$, which is a contradiction. \hfill \Box

The above proof depends on the condition that $G_0$ is full-rank. This assumption is also required in the Hamming metric analogous proof for Proposition 2 [8].

If the $j$-th column sum rank is maximized, then the code possesses a profile of maximum column sum ranks up to $j$. The implication in conjunction with Theorem 3 is that decoding occurs opportunistically. When decoding $s_t$, the receiver can check whether $d_R(i) > n(i+1) - \text{rank}(A_{[t,t+i]})$ for each $i \leq W - 1$.

Once the condition is met, $s_t$ is recoverable at time $t + i$ instead of waiting until time $t + W - 1$. The profile renders the code robust to the introduction of rank-deficient channel matrices at the end of the window that would otherwise prevent decoding. It is possible that $\text{rank}(A_{[t,t+i]})$ is sufficiently large but $\text{rank}(A_{[t,t+W-1]})$ is not. In this case, attempting to decode at time $t + W - 1$ is not feasible using Theorem 3, but $s_t$ can be recovered at time $t + i$, before the end of the window. This observation is interesting for showing an opportunistic decoding ability, but ultimately serves as a special case.

Codes achieving the Singleton bound for $d_R(m)$ are referred to as Maximum Sum Rank (MSR) codes. They directly parallel $m$-MDS codes, which maximize the $m$-th column Hamming distance [8]. In fact, since $d_R(j) \leq d_R(j)$, MSR codes automatically maximize the column Hamming distance and can be seen as a special case of $m$-MDS codes.

By Theorem 3, an MSR code with memory $T = W - 1$ recovers source packets with delay $T$, when the rank of the channel matrix is at least $k(T + 1)$ in each sliding window of length $W$. We prove the existence of MSR codes in the next section, but first discuss a matrix multiplication property for the generator matrix. The following theorem serves as an extension of Theorem 2 to convolutional codes transmitted over independent network uses.

**Theorem 4.** For $t = 0, \ldots, j$, let $\rho_t \leq r$ satisfy

\[
\sum_{i=0}^{t} \rho_i \leq k(t + 1)
\]

for all $t \leq j$ and with equality for $t = j$. The following are equivalent for any convolutional code.

1. $d_R(j) = (n - k)(j + 1) + 1$

2. For all full-rank $A_{[0,j]}^* = \text{diag}(A_0^*, \ldots, A_j^*)$ with blocks $A_t^* \in \mathbb{F}_q^{r \times \rho_t}$, the product $G_j^{\text{EX}} A_{[0,j]}^*$ is non-singular.

**Proof.** We first prove $1 \Rightarrow 2$. Consider a code where $d_R(j) = (n - k)(j + 1) + 1$ and suppose there exists an $A_{[0,j]}^*$ satisfying (3.3), for which $G_j^{\text{EX}} A_{[0,j]}^*$ is singular. Then there exists a codeword sequence $x_{[0,j]}$, where $x_{[0,j]} A_{[0,j]}^* = \mathbf{0}$. The first codeword $x_0$ is not necessarily non-zero, meaning the sum rank of $x_{[0,j]}$ is not constrained by $d_R(j)$. We let $l = \arg \min_x x_l \neq \mathbf{0}$ and consider the codeword sequence $x_{[l,j]}$, whose sum rank is at least $d_R(l - j)$. Because $x_t A_t^* = \mathbf{0}$ for $t = l, \ldots, j$, we bound $\text{rank}(\phi_n(x_t)) \leq n - \rho_t$ in
this window. The sum rank of $x_{[i,j]}$ is bounded:

$$\sum_{i=l}^{j} \text{rank} (\phi_n(x_i)) \leq n(j-l+1) - \sum_{i=l}^{j} \rho_i \leq (n-k)(j-l+1).$$

The second line follows from $\sum_{t=1}^{j} \rho_t \geq k(j-l+1)$, which can be derived when (3.3) is met with equality for $t = j$. Due to Lemma 3, the column sum rank achieves $d_R(j-l) = (n-k)(j-l+1) + 1$. The sum rank of $x_{[i,j]}$ is less than $d_R(j-l)$, which is a contradiction.

We prove $2 \Rightarrow 1$ by using a code with $d_R(j) \leq (n-k)(j+1)$ and constructing an $A_{[0,j]}^*$ for which $G^{EX}_{[0,j]}$ is singular. Let $m = \arg \min_i d_R(i) \leq (n-k)(i+1)$ be the first instance where the column sum rank fails to attain maximum and consider the sequence $x_{[0,m]}$ with the minimum column sum rank. We show that there exist matrices $A_i^* \in \mathbb{F}_q^{r \times \rho_i}$ satisfying both (3.3) and $x_i A_i^* = 0$ for $t = 0, \ldots, m$. In addition, we aim to construct $A_{[0,m]}^*$ to have dimension $(m+1)r \times (m+1)k$. This is relevant later in the proof.

When $m = 0$, the column rank of $x_0$ cannot exceed $n-k$. For every $\rho_0 \leq n - \text{rank} (\phi_n(x_0))$, there exists an $A_0^*$ for which $x_0 A_0 = 0$. We choose an $A_0$ with rank $\rho_0 = k$.

When $m > 0$, the sum rank of $x_{[0,m-1]}$ is equal to $(n-k)m + 1 + k_1$ for some $k_1 \geq 0$. Let $\rho_t = r - \text{rank} (\phi_n(x_t))$ for $t = 0, \ldots, m-1$ and choose the appropriate $A_t^*$ for which $x_t A_t^* = 0$. The summation of all $\rho_t$ is bounded

$$\sum_{i=0}^{t} \rho_i = r(t+1) - \sum_{i=0}^{t} \text{rank} (\phi_n(x_i)) \leq k(t+1) - 1,$$

confirming that (3.3) is satisfied for $t \leq m-1$. The second line follows from the fact that the sum rank of $x_{[0,t]}$ is at least $(n-k)(t+1) + 1$. For $t = m-1$, there is an exact solution to $\sum_{i=0}^{m-1} \rho_i = mk - 1 - k_1$. It remains to choose an appropriate $\rho_m$. The rank of our $x_m$ is at most $n - k - 1 - k_1$, so for every $\rho_m \leq k + 1 + k_1$, there exists an $A_m^*$ satisfying $x_m A_m^* = 0$. However, we must also ensure that $A_m^*$ has full column rank, i.e., $\rho_m \leq r$. The sum rank of $x_{[0,m-1]}$ cannot exceed the sum rank of $x_{[0,m]}$, which is strictly less than $(n-k)(m+1)+1$. This implies that $k_1 \leq r-k-1$. We choose $\rho_m = k+1+k_1$ exactly, as this value cannot exceed $r$. Our choice further guarantees that $A_{[0,m]}^*$ has dimension $(m+1)n \times (m+1)k$.

The remaining $A_{m+1}^*, \ldots, A_j^*$ can be any full-rank $r \times k$ matrices, thus satisfying (3.3) for all $t \leq j$. The product $G^{EX}_{[0,j]} A_{[0,j]}^*$ is given as

$$\begin{pmatrix} G^{EX}_m & X \\ Y & A_{[0,m]}^* \end{pmatrix} \begin{pmatrix} A_{[0,m]}^* \\ A_{[m+1,j]}^* \end{pmatrix} = \begin{pmatrix} G^{EX}_m A_{[0,m]}^* & X A_{[m+1,j]}^* \\ Y A_{[m+1,j]}^* \end{pmatrix},$$

where $X$ and $Y$ denote the remaining blocks that comprise $G^{EX}_{[0,j]}$. The block $G^{EX}_m A_{[0,m]}^*$ is a square matrix with a zero determinant. Therefore, $\det G^{EX}_{[0,j]} A_{[0,j]}^*$ is also zero.

By Theorem 3, if $\sum_{i=0}^{t} \rho_i < k(t+1)$ for $t \leq j$, then the decoder does not yet possess sufficient information to guarantee $s_0$ is recoverable by time $t$. Once the bound is achieved for some $t$, we can invert $G^{EX}_{[0,t]}$ and recover $s_0$ with that delay. Thus, (3.3) ensures that $[0,j]$ is a feasible decoding
Theorem 4 provides a tighter set of conditions that encapsulate Proposition 3 for \(m\)-MDS codes. The proposition only considers full-size minors of \(G_j^{\text{EX}}\), which can be interpreted as forcing each \(A_t\) to be comprised solely from columns of \(I_{n \times n}\). MSR codes require a general set of full-rank channel matrices for which a full-rank product is guaranteed. In the next section, we construct an extended generator matrix. Theorem 4 is used afterwards to verify that the generator produces an MSR code.

### 3.4 Code Construction

In this section, we first introduce a new family of matrices that preserve super-regularity after multiplication with block diagonal matrices in the ground field. MSR codes can be constructed using such matrices similarly to how \(m\)-MDS codes are constructed using conventional super-regular matrices.

#### 3.4.1 Super-regularity Preservation

Because MSR codes are also \(m\)-MDS, it is natural to assume that their generators can be constructed using super-regular matrices. However, recall that MRD block codes possess an additional matrix multiplication property given in Theorem 2, that is not generally shared by MDS block codes. Theorem 4 extends this to convolutional codes. In this section, we connect the matrix multiplication property in Theorem 4 to super-regularity.

Consider the case when the element \(\alpha\) generating the matrices in (2.5) is primitive normal. The block diagonal \(\bar{T}\) from Theorem 1 remains super-regular, but now each of the \(T_j\) resemble generator matrices for rate \(R = 1\) Gabidulin codes. Suppose that we let \(A_t \in \mathbb{F}_q^{r \times r}\) be a non-singular matrix in the ground field and construct \(A_{[0,m]} = \text{diag}(A_t; m + 1)\). The product \(F = \bar{T}A_{[0,m]}\) is a block Hankel matrix

\[
\begin{pmatrix}
F_0 \\
F_0 & F_1 \\
\vdots & \vdots & \vdots \\
F_0 & \cdots & F_{m-1} & F_m
\end{pmatrix},
\]

where \(F_j = T_jA_t\) for \(j = 0, \ldots, m\). Each block \(\begin{pmatrix} F_0 \\ \vdots \\ F_j \end{pmatrix}\) is referred to as a column block of \(F\). Furthermore, it can be shown that each \(F_j\) has the structure

\[
F_j = \begin{pmatrix}
f_0^{[nj]} & f_1^{[nj]} & \cdots & f_{n-1}^{[nj]} \\
f_0^{[nj+1]} & f_1^{[nj+1]} & \cdots & f_{n-1}^{[nj+1]} \\
\vdots & \vdots & \vdots & \vdots \\
f_0^{[n(j+1)-1]} & f_1^{[n(j+1)-1]} & \cdots & f_{n-1}^{[n(j+1)-1]}
\end{pmatrix},
\]  

where \((f_0, \ldots, f_{n-1}) = (\alpha^0, \ldots, \alpha^{n-1})A_t\) are linearly independent elements over \(\mathbb{F}_q\). \([7]\). Furthermore, each element \(f_i\) is a linearized polynomial \(f_i(X)\) evaluated at \(X = \alpha\), with coefficients from \(A_t = [A_{k,i}]\).
All non-zero entries of $F$ are described as linearized polynomials

$$f_i^{[j]} = \sum_{k=0}^{n-1} A_{k,i} \alpha^{[k+j]}.$$  \hspace{1cm} (3.5)

The polynomials on any given column have the same set of coefficients, but the degree of the monomial terms in the polynomials increase as one moves downwards along $F$, i.e., $f_i^{[j]}(X) = f_i(X^{[j]})$. The $q$-degree of each linearized polynomial is bounded

$$j \leq \deg_{q} f_i^{[j]}(X) \leq n - 1 + j.$$  \hspace{1cm} (3.6)

These bounds depend primarily on the Frobenius power $j$ of the entry in the matrix. Consequently, the polynomial entries on any fixed row of the block $F_i$ all share the same bound.

Although $A_{[0,m]}$ was constructed above by repeating a single $A_t$, similar results are reached when letting $A_{[0,m]} = \text{diag}(A_0, \ldots, A_m)$ be constructed from different matrices. Each column block of $F$ is now generated using a different $A_t$ and (3.5) must be modified to reflect the appropriate polynomial coefficients. Consequently, there are a different set of linearly independent $f_0, \ldots, f_{n-1}$ for different column blocks. The non-zero entries however, remain linearized polynomials with the same bounds and (3.6) is preserved. Regardless of the choice of $A_{[0,m]}$, the structure of $F$ is similar to $T$, but with polynomials of varying degrees rather than monomials with fixed degrees. We can propose an analogue of (2.6) for $F$.

**Lemma 4.** For $t = 0, \ldots, m$, let $A_t \in \mathbb{F}_q^{n \times n}$ be non-singular matrices and $A_{[0,m]} = \text{diag}(A_0, \ldots, A_m)$. Let $\bar{T}$ be be the super-regular matrix in (2.4). The product $F = \bar{T}A$ satisfies the following.

1. $\deg_{q} F_{i',j}(X) < \deg_{q} F_{i,j}(X)$, if $i' < i$.
2. $\deg_{q} F_{i,j'}(X) < \deg_{q} F_{i,j}(X)$, if $F_{i,j'}$ is an entry of a different column block to the left of that from which $F_{i,j}$ is drawn.

Statement 1 in Lemma 4 is identical to (2.6a). Statement 2 is a weakened variation of (2.6b) that only holds when the two entries are drawn from different column blocks of $F$. Both statements follow directly from (3.6). The above lemma is needed to show that $F$ is also a super-regular matrix when $\alpha$ is primitive normal.

**Theorem 5.** For $t = 0, \ldots, m$, let $A_t \in \mathbb{F}_q^{n \times n}$ be any non-singular matrices. We then construct $A_{[0,m]} = \text{diag}(A_0, \ldots, A_m)$. Let $\bar{T} \in \mathbb{F}_q^{n \times n}$ be the super-regular matrix in (2.4). If $M \geq q^{n(m+2)} - 1$ and $\alpha$ is primitive normal, then $F = \bar{T}A_{[0,m]}$ is super-regular.

**Proof.** We show that $F$ is super-regular by considering three increasingly general scenarios. The problem in subsequent cases for $F$ can be converted to the first, which we will prove super-regular. Furthermore, we assume without loss of generality that $A_0 = A_1 = \cdots = A_m$. This simplifies notation and allows us to freely use the previous polynomial structures and bounds. Justification is provided at the end of the proof.

**Case 1:** Consider when the degrees of the polynomials $f_i(X)$ are strictly increasing with $i$, i.e. $\deg f_0(X) < \cdots < \deg f_{n-1}(X)$. Because the polynomials are linearized and bounded, the degrees must take the following values:

$$\deg f_0(X) = 1, \deg f_1(X) = q, \ldots, \deg f_{n-1}(X) = q^{n-1}.$$  \hspace{1cm} (3.7)
In this case, $\mathbf{F}$ fully satisfies (2.6), as opposed to only Lemma 4. Every polynomial entry in $\mathbf{F}$ has the same degree as the monomial at the same position in $\mathbf{T}$. We let $\mathbf{D}$ be a sub-matrix of $\mathbf{F}$, and construct $\mathbf{D}$ from $\mathbf{T}$ using the same row and column indices. The monomial entries of $\mathbf{D}$ share the same degrees as the polynomials in the corresponding positions of $\mathbf{D}$. The Leibniz formula for the determinant yields $\text{deg } D(X) = \text{deg } \bar{D}(X)$. Lemma 2 holds for $D(X)$ and it follows that $\mathbf{D}$ is non-singular. Therefore, $\mathbf{F}$ is super-regular.

It is clear that all of the sub-matrices of $\mathbf{F}$ need only fully satisfy (2.6) in order for the determinant of each sub-matrix to be bounded by Lemma 2. Once the lemma is proven to hold, we apply the argument in Section 2.3 verbatim to reveal that any given sub-matrix is non-singular. In the remaining cases, we will modify the matrices to return to Case 1, thus satisfying (2.6).

**Case 2:** Consider a more general scenario where the degrees of $f_i(X)$ are different but not necessarily in increasing order. As a result, (2.6b) does not always hold but Lemma 4 preserves (2.6a). Permuting the columns of $\mathbf{F}$ allows us to re-arrange each column block to produce a matrix $\hat{\mathbf{F}}$ that satisfies (3.7). The set of sub-matrices of $\mathbf{F}$ and $\hat{\mathbf{F}}$ are identical up to column permutations and therefore, every sub-matrix of $\mathbf{F}$ is non-singular.

**Case 3:** We now consider the scenario with no restrictions on the degrees of $f_i(X)$. Naturally, there may exist multiple polynomials sharing the same degree. Column permutations alone cannot transform $\mathbf{F}$ to satisfy (3.7). As a result, we show how by using elementary column operations, each sub-matrix of $\mathbf{F}$ with a non-trivial determinant can be transformed to one with an increasing degree distribution. The matrix is then interpreted as a sub-matrix of a super-regular $\hat{\mathbf{F}}$ that satisfies (3.7) and therefore (2.6).

Let $\mathbf{D}$ be a sub-matrix of $\mathbf{F}$ with a non-trivial determinant. Lemma 4 is clearly preserved. The authors in [2] revealed that $\mathbf{D}$ can be written as

$$
\mathbf{D} = \begin{pmatrix}
\mathbf{O}_1 & \mathbf{D}_1 \\
\mathbf{O}_2 & \mathbf{D}_1 & \\
\vdots & & \\
\mathbf{O}_h & & \\
\mathbf{D}_h & & \mathbf{D}_0
\end{pmatrix},
$$

(3.8)

where $h \geq 0$. Each $\mathbf{O}_i$ is a zero matrix and each $\mathbf{D}_i$ is a matrix containing non-zero polynomials drawn
from a single column block of $F$. Let $k_i$ be the number of columns in each $D_i$. The polynomials on each row of $D_i$ share the same bounds on degrees and are linearly independent amongst themselves.

We apply elementary column operations on each $D_i$ separately in order to ensure that $D$ satisfies (2.6b). Using the bijection $\phi_i(\cdot)$ from (2.2), each row of $D_i$ maps to a matrix in $F_q^{m \times k_i}$. Consider a particular row of $D_i$, which consists of polynomials whose degrees are bounded between $j$ and $n - 1 + j$ due to (3.6). The isomorphic matrix of this row vector may possess non-zero entries only between the $j$-th to $n - 1 + j$-th rows. Because the polynomials are linearly independent, the matrix has full column rank. Using Gaussian elimination, we transform it to reduced column echelon form, i.e., the structure in Fig. 3.3. Applying $\phi_i^{-1}(\cdot)$ on the result matrix gives a vector of polynomials with strictly increasing degrees.

By (3.5), the column operations to modify one row also modifies the polynomials in all other rows to share the same differences amongst degrees. Effectively, there exists a matrix $M_i \in F_q^{n \times k_i}$ that ensures each $D_i, M_i$ satisfies the conditions of (2.6b). By constructing $M = \text{diag}(M_0, \ldots, M_m)$, we produce $\hat{D} = DM$, for which (2.6) is completely satisfied. $\hat{D}$ can be treated as a sub-matrix of a super-regular matrix achieving (3.7). Then $\det \hat{D} = \det D \det M$ implies that $D$ is non-singular and $F$ is super-regular.

For this proof, we had let $A_0 = A_1 = \cdots = A_n$. Without the assumption, the product $F$ would be comprised of a different set of $f_i(\alpha)$ for each column block. Because each $D_i$ is a sub-matrix from a single column block and column operations are performed for each $D_i$ independently, the polynomial degrees can always be transformed in order to satisfy (3.7). \hfill \Box

The key technique in this proof is the usage of an elementary column operation matrix $M$ to permute the polynomial entries of $F$. An example of the method to transform $F$ to $\hat{F}$ is given below.

**Example 1.** Let $n = 4, m = 1$ and $\hat{T}$ be the super-regular matrix of Theorem 5 generated from a primitive and normal element $\alpha \in F_{2^{2048}}$. Let $A_0$ and $A_1$ be the following two non-singular square matrices:

$$A_0 = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, A_1 = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix}.$$

We use $A_{[0,1]} = \text{diag}(A_0, A_1)$ as the block-diagonal matrix and generate the product

$$TA_{[0,1]} = \begin{pmatrix} \alpha^0 + \alpha^1 & \alpha^1 + \alpha^2 + \alpha^3 & \alpha^1 & \alpha^0 + \alpha^2 \\ \alpha^1 + \alpha^2 & \alpha^2 + \alpha^3 + \alpha^4 & \alpha^2 & \alpha^1 + \alpha^3 \\ \alpha^2 + \alpha^3 & \alpha^3 + \alpha^4 + \alpha^5 & \alpha^3 & \alpha^2 + \alpha^4 \\ \alpha^3 + \alpha^4 & \alpha^4 + \alpha^5 + \alpha^6 & \alpha^4 & \alpha^3 + \alpha^5 \\ \alpha^4 + \alpha^5 & \alpha^5 + \alpha^6 + \alpha^7 & \alpha^5 & \alpha^4 + \alpha^6 \\ \alpha^5 + \alpha^6 & \alpha^6 + \alpha^7 + \alpha^8 & \alpha^6 & \alpha^5 + \alpha^7 \\ \alpha^6 + \alpha^7 & \alpha^7 + \alpha^8 + \alpha^9 & \alpha^7 & \alpha^6 + \alpha^8 \\ \alpha^7 + \alpha^8 & \alpha^8 + \alpha^9 + \alpha^{10} & \alpha^8 & \alpha^7 + \alpha^9 \end{pmatrix}.$$

The entries generating the left column block $\alpha^1 + \alpha^2, \alpha^0, \alpha^0 + \alpha^2, \alpha^3$ are linearly independent polynomials. Similarly, the entries generating the right column block $\alpha^0 + \alpha^1, \alpha^1 + \alpha^2 + \alpha^3, \alpha^1, \alpha^0 + \alpha^1, \alpha^2, \alpha^3, \alpha^4, \alpha^5, \alpha^6, \alpha^7, \alpha^8, \alpha^9, \alpha^{10}$.
\(\alpha^{[2]}\) are also respectively linearly independent amongst each other. Note that the \(q\)-degree of these polynomials are upper and lower bounded between 3 and 0. Furthermore, for any fixed row of \(\mathbf{T}\mathbf{A}_{[0,1]}\), the polynomials on the left column block always have a lower degree than those from the right column block.

Now consider the sub-matrix formed from rows \(R_i, i \in \{1, 2, 4, 5\}\) and columns \(C_j, j \in \{3, 4, 5, 6\}\). This matrix

\[
\mathbf{D} = \begin{pmatrix}
\alpha^{[1]} + \alpha^{[2]} & \alpha^{[2]} + \alpha^{[4]} & \alpha^{[4]} & \alpha^{[2]} \\
\alpha^{[2]} + \alpha^{[3]} & \alpha^{[3]} + \alpha^{[4]} & \alpha^{[4]} & \alpha^{[3]} \\
\alpha^{[3]} + \alpha^{[5]} & \alpha^{[5]} + \alpha^{[6]} + \alpha^{[7]} & \alpha^{[5]} \\
\alpha^{[4]} + \alpha^{[5]} & \alpha^{[6]} + \alpha^{[7]} + \alpha^{[8]} & \alpha^{[6]}
\end{pmatrix}
\]

does not satisfy (2.6b). The degree of the polynomials in \(C_4\) and \(C_6\) are the same and lower than the degree of the polynomials in \(C_5\). We apply the following elementary column operations on \(\mathbf{D}\).

1. \(C_5 \leftrightarrow C_6\)
2. \(C_4 - C_5 \rightarrow C_4\)

The transformed matrix

\[
\mathbf{\hat{D}} = \begin{pmatrix}
\alpha^{[1]} & \alpha^{[2]} & \alpha^{[3]} + \alpha^{[4]} & \alpha^{[4]} \\
\alpha^{[2]} & \alpha^{[3]} & \alpha^{[4]} + \alpha^{[5]} \\
\alpha^{[3]} & \alpha^{[4]} & \alpha^{[5]} + \alpha^{[6]} + \alpha^{[7]} \\
\alpha^{[4]} & \alpha^{[5]} & \alpha^{[6]} + \alpha^{[7]} + \alpha^{[8]}
\end{pmatrix}
\]

completely satisfies (2.6). It follows that this matrix and therefore, \(\mathbf{D}\) are non-singular.

**Remark 2.** It is interesting to note that \(\mathbf{F}\) can easily be modified using elementary column operations from the onset to a super-regular matrix whose polynomials achieve (3.7). However, it is not obvious whether this transformation of \(\mathbf{F}\) preserves super-regularity. While column permutations do not change the set of sub-matrices, column addition operations on \(\mathbf{F}\) will result in a matrix with an entirely different set of sub-matrices. As applying elementary column operations on the super-regular matrix is equivalent to multiplication with a square full-rank matrix, there clearly exist sub-matrices of the product that cannot be constructed from elementary column operations on a sub-matrix of the original super-regular matrix. Consequently, our proof opts to directly show that each sub-matrix is indeed non-singular.

### 3.4.2 Encoder

The rows of \(\mathbf{T}\) are permuted to the structure of an extended generator matrix

\[
\mathbf{T} = \begin{pmatrix}
\mathbf{T}_0 & \mathbf{T}_1 & \cdots & \mathbf{T}_m \\
\mathbf{T}_0 & \cdots & \mathbf{T}_{m-1} \\
\vdots & \vdots \\
\mathbf{T}_0
\end{pmatrix}
\]

Since every sub-matrix of \(\mathbf{T}\) has a counterpart in \(\mathbf{\hat{T}}\) identical up to row permutations, this block Toeplitz matrix is also super-regular. \(\mathbf{G}_{m,m}^{\text{EX}}\) is then constructed as a sub-matrix of \(k(m + 1)\) rows from \(\mathbf{T}\). This process parallels the construction of \(m\)-MDS generator matrices [8].
Theorem 6. Let $T$ be the super-regular matrix in (3.9) generated using a primitive normal $\alpha \in \mathbb{F}_{q^M}$, where $M = q^{\lfloor (m+2)/2 \rfloor}$. Let $0 \leq i_1 < \cdots < i_k < n$ and construct a $(m+1)k \times (m+1)n$ sub-matrix $G_m^{\text{EX}}$ of $T$ from rows indexed $jn+i_1,\ldots,jn+i_k$ for $j = 0,\ldots,m$. This matrix is the extended generator of an MSR convolutional code $C[n,k,m]$.

Proof. We show that $G_m^{\text{EX}}$ satisfies Theorem 4. Assume without loss of generality that $i_1 = 0,\ldots,i_k = k - 1$. Each $T_i$ is divided into $\begin{pmatrix} G_i \\ T'_i \end{pmatrix}$, where $G_i \in \mathbb{F}_{q^M}$ are the blocks of the extended generator matrix. For $t = 0,\ldots,m$, let $A_t \in \mathbb{F}_{q^r}$ be non-singular matrices. We similarly divide $A_t = \begin{pmatrix} A_t^* \\ A'_t \end{pmatrix}$, where the two blocks $A_t^* \in \mathbb{F}_{q^{r \times p_t}}$ and $A'_t \in \mathbb{F}_{q^{r \times (n-p_t)}}$ represent the reduced channel matrix and some remaining matrix respectively. Let $A_{[0,m]} = \text{diag}(A_0,\ldots,A_m)$. The product can be written as

$$TA_{[0,m]} = \begin{pmatrix} T_0 A_0 & T_1 A_1 & \cdots & T_m A_m \\ T_0 A_1 & T_1 A_2 & \cdots & T_m A_{m-1} \\ \vdots & \vdots & \ddots & \vdots \\ T_0 A_m \end{pmatrix},$$

where

$$T_i A_j = \begin{pmatrix} G_i A_j^* \\ T'_j A_j^* \\ \vdots \\ T_j A_j^* \end{pmatrix}.$$

The sub-matrix of $TA_{[0,m]}$ containing only the rows and columns involving $G_i A_j^*$ is equal to the product $G_m^{\text{EX}} A_{[0,m]}^*$. Because $TA_{[0,m]}$ is super-regular, the determinant of $G_m^{\text{EX}} A_{[0,m]}^*$ is either trivially zero or non-trivial and therefore non-zero. The conditions for the product to have a non-trivial determinant are equivalent to the rank conditions in (3.3); this is further explained in Appendix A.3. Ultimately for all $A_{[0,m]}^*$ satisfying (3.3), the product has a non-zero determinant. Thus, $G_m^{\text{EX}}$ satisfies Theorem 4 and $C[n,k,m]$ achieves $d_R(m) = (n-k)(m+1)+1$. 

The above technique allows for constructing MSR codes of any parameter. We can now directly use these codes for network streaming over sliding window networks. an MSR code $C[n,k,T]$ feasibly recovers every packet over $CH_1(r,S,W)$ with delay $T = W - 1$ if $S < d_R(W - 1)$.

In application, the decoding deadline $T$ is not always exactly equal to $W - 1$. If the decoder relaxes the delay constraint, i.e., $T \geq W$, the same code $C[n,k,T]$ achieves perfect recovery over every Isolated Rank Loss Network that has $S < d_R(W - 1)$. However if $T < W - 1$, then the maximum achievable $S$ is $d_R(T) - 1$.

### 3.5 Numerical Results

In this section, we provide several examples of MSR codes that exist in extension fields over $\mathbb{F}_2$. As every extension field can be written as a polynomial field modulo a primitive polynomial $\mathbb{F}_q[X]/(\rho_n(X))$, every element in the field is described as a polynomial. In this section, we use an octal notation to represent polynomials. For extension fields over $\mathbb{F}_2$, the polynomial coefficients are either one or zero and thus a polynomial can be written as a binary vector of length $M$ with the highest degree term on the left. Naturally, the binary vector can be represented in octal notation. For example, consider the polynomial
Chapter 3. Maximum Sum Rank Codes

### Table 3.1: Achievable field sizes under which codes constructed using Theorem 6. The bound required by the theorem for each set of code parameters is provided in the middle column.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Field Bound</th>
<th>Achievable Field</th>
<th>$\alpha$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$[4, 2, 1]$</td>
<td>$F_{2^{2048}}$</td>
<td>$F_{211} = F_2 / \langle [4005]_8 \rangle$</td>
<td>$[3]_8$</td>
</tr>
<tr>
<td>$[3, 2, 2]$</td>
<td>$F_{2^{2048}}$</td>
<td>$F_{211} = F_2 / \langle [4005]_8 \rangle$</td>
<td>$[3]_8$</td>
</tr>
<tr>
<td>$[3, 1, 2]$</td>
<td>$F_{2^{2048}}$</td>
<td>$F_{211} = F_2 / \langle [4005]_8 \rangle$</td>
<td>$[3]_8$</td>
</tr>
<tr>
<td>$[2, 1, 2]$</td>
<td>$F_{2^{128}}$</td>
<td>$F_{27} = F_2 / \langle [211]_8 \rangle$</td>
<td>$[11]_8$</td>
</tr>
<tr>
<td>$[2, 1, 1]$</td>
<td>$F_{2^{64}}$</td>
<td>$F_{2^5} = F_2 / \langle [45]_8 \rangle$</td>
<td>$[3]_8$</td>
</tr>
</tbody>
</table>

Theorem 6 guarantees the construction if $M = 2^{11}$, i.e., $\alpha$ is a primitive normal element of $F_{2^{2048}}$.

The construction in Theorem 6 can generate MSR codes over much smaller field sizes than those given by the bound of $M \geq q^{n(m+2)} - 1$. Table 3.1 provides a list of code parameters and the field sizes on which they satisfy the matrix multiplication property of Theorem 4.

Random linear constructions of MSR codes are useful as an alternative to the structured constructions. Here we randomly generate blocks $G_j$ for $j \leq m$ and test whether the extended generator matrix satisfies Theorem 4. If the generator matrix satisfies the theorem, then the code naturally achieves the maximum $m$-th column sum rank. Similar to the structured constructions, Random Linear MSR codes exist for significantly smaller fields than what the sufficiency statement requires. We provide several examples of these random linear codes below.

**Example 2.** Let $\alpha = [9]_8$ be a primitive normal element in $F_{2^7} = F_2 / \langle [211]_8 \rangle$. The following extended generator matrix

$$G_2^{\text{EX}} = \begin{pmatrix}
\alpha & \alpha^{[1]} & \alpha^{[2]} & \alpha^{[3]} & \alpha^{[4]} & \alpha^{[5]} & \alpha^{[6]} & \alpha^{[7]} & \alpha^{[8]}
\alpha^{[2]} & \alpha^{[3]} & \alpha^{[4]} & \alpha^{[5]} & \alpha^{[6]} & \alpha^{[7]} & \alpha^{[8]} & \alpha^{[9]} & \alpha^{[10]}
\alpha & \alpha^{[1]} & \alpha^{[2]} & \alpha^{[3]} & \alpha^{[4]} & \alpha^{[5]}
\alpha^{[2]} & \alpha^{[3]} & \alpha^{[4]} & \alpha^{[5]} & \alpha^{[6]} & \alpha^{[7]}
\alpha & \alpha^{[1]} & \alpha^{[2]}
\alpha^{[2]} & \alpha^{[3]} & \alpha^{[4]}
\end{pmatrix}$$

satisfies Theorem 4, making it the generator for an MSR code $C[3, 2, 2]$. Theorem 6 guarantees the construction if $M = 2^{11}$, i.e., $\alpha$ is a primitive normal element of $F_{2^{2048}}$.

**Example 3.** Consider the matrix

$$G_1^{\text{EX}} = \begin{pmatrix}
[222]_8 & [63]_8 & [313]_8 & [363]_8 & [321]_8 & [115]_8 & [333]_8 & [345]_8 \\
\end{pmatrix} \in F_{2^8} = F_2 / \langle [435]_8 \rangle,$$
Figure 3.4: Testing 100 randomly generated linear time-invariant convolutional codes $C[4, 2, 1]$ over $F_{2^M}$ by multiplying the generator with the set of all full-rank block diagonal matrices in $F_2$ that satisfy the rank conditions in (3.3). The red lines denote the median value with the box encircling the 25-th to 75-th percentiles. The whiskers denote approximately the 1-st and 99-th percentiles. Points beyond this range are considered outliers.

whose blocks $G_0$ and $G_1$ were randomly generated. This matrix satisfies Theorem 4, making it a valid generator for a Random Linear MSR code $C[4, 2, 1]$. Theorem 6 suggests that $M = 2^{11}$.

Example 4. Consider the matrix

$$G_{2}^{\text{EX}} = \begin{pmatrix} 62_{8} & 52_{8} & 66_{8} & 53_{8} & 57_{8} & 52_{8} & 55_{8} & 22_{8} & 7_{8} \\ 75_{8} & 3_{8} & 73_{8} & 60_{8} & 31_{8} & 13_{8} & 3_{8} & 3_{8} & 64_{8} \\ 62_{8} & 52_{8} & 66_{8} & 53_{8} & 57_{8} & 52_{8} \\ 75_{8} & 3_{8} & 73_{8} & 60_{8} & 31_{8} & 13_{8} \\ 62_{8} & 52_{8} & 66_{8} \\ 75_{8} & 3_{8} & 73_{8} \end{pmatrix} \in F_{2^6 \times 9} = F_2/\langle 103 \rangle_8,$$

whose blocks $G_0$, $G_1$, and $G_2$ were randomly generated. This matrix satisfies Theorem 4, making it a valid generator for a Random Linear MSR code $C[3, 2, 2]$. Theorem 6 suggests that $M = 2^{11}$.

In order to better understand the use of Random Linear MSR codes as substitutes for the construction in Section 3.4, we randomly generate 100 $G_{1}^{\text{EX}}$ for codes $C[4, 2, 1]$ over $F_{2^M}$ for various values of $M$. A box plot is given in Fig. 3.4, detailing the performance of these randomly constructed extended generator matrices in terms of satisfying Theorem 4. Each extended generator matrix is multiplied with every potential channel matrix, i.e., all $A_{[0,1]} = \text{diag}(A_0, A_1) \in F_2^{8 \times 4}$ that satisfy (3.3). If the product is full-rank, then the code guarantees protection against that potential channel matrix. A rank-deficient product implies that the source packet cannot be recovered for this channel. The number of rank-deficient products are counted and the fraction of rank-deficient products versus all full-rank $A_{[0,1]}$ is calculated for each generator matrix. Random Linear MSR codes exist even for $F_{2^5}$, in contrast to the sufficiency constraint requiring $F_{2^{2048}}$. Most generator matrices for $M = 7$ onwards can be described
as *nearly* MSR, producing full-rank products for almost all channel matrices. From $M = 8$ onwards, a randomly generated $G_{1}^{EX}$ has a high probability of satisfying Theorem 4. This test ultimately serves to show that despite the large field size required in Theorem 6, MSR codes can be feasibly constructed over practical fields. Furthermore, as the field size increases, the probability of a randomly constructed extended generator matrix being an MSR generator increases. Although not capable of perfect recovery over the adversarial model, codes that produce full-rank products for almost every channel matrix can be used in practical scenarios with a small error.
Chapter 4

Recovery Of Bursts In Networks Codes

4.1 Introduction

In this chapter, we address network streaming under burst losses. This problem is defined differently from the Isolated Rank Loss Sliding Window Network, which focused only on the rank deficiency in a window. The isolated loss network in the previous chapter was modelled using a sliding window setup, whereas we instead use a network extension of the classic burst erasure channel defined in Section 1.1. The maximum tolerable duration of a burst becomes the relevant parameter for both the single-link and network scenarios. The key difference in the network extension is that the burst does not erase the entire packet, but only erases up to a maximum number of links, forcing the channel matrix to become rank-deficient. A comparative single-link scenario would be if the burst erases only a fraction of the symbols. Subsequently, the network model requires an additional parameter: the maximum rank deficiency occurring per shot. Note that the problem can be redefined into a sliding window setting, but we avoid this notation to preserve the intuitive definition originally proposed in [14].

We design an end-to-end code with a layered structure and prove that it achieves zero-error low-delay recovery of every packet in this network. Our approach generalizes prior results on the multiple-link channel in [24, Chapter 8]. The model in the previous work simplified to multiple parallel links between the transmitter and receiver with one acting as a burst erasure channel. In contrast, we permit a general channel matrix along with only multiple links simultaneously deactivating. The code constructions in the prior work involved layering and diagonally interleaving MDS block codes. This technique seems specific to the parallel link case and does not easily generalize to a matrix channel. Our code construction layers both MSR codes and Gabidulin codes similarly to single-link burst correcting codes [3, 4].

This chapter is outlined as follows. The network streaming problem is introduced in Section 4.2. The streaming capacity is introduced in Section 4.3, where we also address how neither Gabidulin codes nor MSR codes generally achieve the streaming capacity. We introduce a new family of codes in Section 4.4. The encoding and decoding steps are detailed and we identify that this code indeed achieves the maximum rate. A converse proof to the capacity is given in Section 4.5. Finally, simulation results are provided in Section 4.6, where we evaluate the performances of the various codes over statistical network models.
Chapter 4. Recovery Of Bursts In Networks Codes

4.2 Problem Setup

The problem is defined in three steps: the encoder, network model, and decoder. We expand the general model described in Sections 1.1 and 1.3 below.

4.2.1 Encoder

At each time $t \geq 0$, a source packet $s_t \in \mathbb{F}_q^k$ arrives at a transmitter node. Each source packet is assumed to be uniformly sampled and drawn independent of all other packets. A codeword $x_t \in \mathbb{F}_q^n$ is generated using (1.1). We consider the class of linear time invariant encoders and focus on convolutional constructions. A rate $R = \frac{k}{n}$ encoder with memory $m$ generates $x_t$ using a set of generator matrices $G_i \in \mathbb{F}_q^k \times \mathbb{F}_q^n$ for $i \leq m$ as in (3.1).

Recall that the network cannot transmit more than $r$ channel packets in one time instance. In the previous chapter, we had assumed that $n = r$, but we now let $n = r\nu$ for $\nu \in \mathbb{N}$. The codeword $x_t$ becomes a macro-packet $X_t = (x_t, \ldots, x_{t, r-1}) \in \mathbb{F}_q^{\nu \times r}$. Throughout this chapter, we interchange between matrix and vector notation when one is preferable over the other in analysis. The $i$-th column of $X_t$ is a channel packet, now described as a column vector $x_{t, i}$ of length $\nu$.

4.2.2 Network Model

At each time instance, a macro-packet is transmitted over the network. There is zero delay between links and the channel transfer matrix $A_t \in \mathbb{F}_q^{r \times r}$ is determined by a random linear network code. For $t \geq 0$, the receiver observes $Y_t = X_t A_t$. All channel packets are transmitted simultaneously and the receiver observes $r$ different linear combinations of the transmitted packets. We can alternatively write this relationship in the vector notation as $y_t = x_t \text{diag} (A_t; \nu)$, where $x_t$ is the codeword in (3.1) and $y_t \in \mathbb{F}_q^{r \times r}$ is the received vector. The effective channel matrix $\text{diag} (A_t; \nu)$ is a block diagonal matrix consisting of $\nu$ copies of $A_t$.

Let $\rho_t = \text{rank} (A_t)$. The rank of the channel in a window is equal to the sum rank of the blocks, i.e., $\sum_{i=t}^{t+j} \rho_i = \text{rank} (A_{t:t+j})$. A perfectly operating network yields $\rho_t = r$, in which case the decoder simply inverts the channel matrix to retrieve the macro-packet. Unreliable links can modify the channel matrix to a rank-deficient form. We define the Burst Rank Loss Network, generalizing the Burst Erasure Channel.

Definition 5. Consider a network where for all $t \geq 0$, the receiver observes $Y_t = X_t A_t$, where $\text{rank} (A_t) = \rho_t$. In the Burst Rank Loss Network $\mathcal{CH}_B (r, p, B, G)$, the channel matrix satisfies $r > \rho_t \geq r - p$ for a burst of at most $B$ consecutive shots before observing $\rho_t = r$ for a gap of at least $G \geq B$ consecutive shots.

An example of a Burst Rank Loss Network is provided in Fig. 4.1. Analysis of streaming video traffic shows that internet streaming is susceptible primarily to burst losses rather than isolated erasures,
making this model one of practical significance [6]. Furthermore, bursts often occur separated from each
other by a significant margin, motivating the practical constraint $G \geq B$. We perform simulations of
our codes over Markov channel models in Section 4.6 and show that the Burst Rank Loss Network
satisfactorily approximates Markov channel models designed to characterize real networks.

In analysis, the linearly dependent columns of the transfer matrix are discarded along with the
associated columns of the received macro-packet. The receiver effectively observes
\[ Y_t^* = X_t A_t^* \]
where $A_t^* \in \mathbb{F}_q^{r \times p_t}$ is the reduced channel matrix, containing only the linearly independent columns.

**Remark 3.** Recent works in streaming over burst erasure channels use the sliding window model for
analysis. The network introduces at most one burst of length $B$ in any interval of length $W > B$. When
there are delay constraints, it is easy to translate between the two models simply by letting $G = W - 1$. Thus, $G \geq B$ is preserved and our results in this chapter remain the same for either model.

### 4.2.3 Decoder

Let $T$ be the maximum delay permissible by the decoder. A packet received at time $t$ must be recovered
by $t + T$ using (1.2). If $s_t = s_u$, then the source packet is perfectly recovered by the deadline; otherwise,
it is declared lost. From Definition 1, a code is feasible over $CH_B(r, p, B, G)$ if there exists an encoder
and decoder pair that are capable of recovering every source packet with delay $T$. A rate $R = \frac{k}{n}$ is
achievable for delay $T$ if there exists a feasible code with that rate. The supremum of achievable rates
is the streaming capacity.

### 4.3 Main Result

In this section, we state the streaming capacity of the Burst Rank Loss Network. The achievability given
in Section 4.4 introduces a new family of structured codes based on rank metric codes. We refer to these
codes as Recovery Of Bursts In Networks (ROBIN) codes. The converse is proven in Section 4.5.

#### 4.3.1 Streaming Capacity of the Burst Rank Loss Network

The streaming capacity of the single-link burst erasure channel was derived in [3, 4]. We propose below
the streaming capacity for the Burst Rank Loss Network.

**Theorem 7.** The streaming capacity for the Burst Rank Loss Network $CH_B(r, p, B, G)$ with $G \geq B$ is
given by

\[
C = \begin{cases} 
\frac{T_{\text{eff}}r + B(r - p)}{(T_{\text{eff}} + B)r} & T \geq B \\
\frac{r - B}{r} & T < B 
\end{cases}
\]

where $T_{\text{eff}} = \min(T, G)$ is the effective delay.

We prove the achievability and converse for this theorem over the subsequent sections. The upper
bound above does not change when $G < B$. However, we do not address the achievability of this region
in this work. The precursory works on single-link burst erasure streaming do not include this scenario
either, although several simple coding schemes are known for specific channel parameters [3, 4]. One
consideration is that for large $T$, the $G < B$ region introduces the problem of low-delay recovery under
multiple bursts rather than a single burst. We refer the reader to [20], where some specific constructions were provided for the multiple-burst problem.

Setting \( r = p \) in (4.1) recovers the streaming capacity for the Burst Erasure Channel [4]. The problem simplifies to all \( r \) paths simultaneously disappearing, which is effectively an erasure of the entire macro-packet. In contrast to [4], or alternatively [24, Chapter 8], we permit variable \( p \) paths to fail.

### 4.3.2 Numerical Comparisons

We review the performance of existing rank metric codes and compare their achievable rates with the streaming capacity. Gabidulin codes can be directly used as end-to-end codes in network streaming, by being treated as convolutional codes with zero memory. Each source packet is encoded and decoded independently of the others and consequently, neither the burst length, gap length, nor the decoding deadline affect the code performance. The decoder successfully recovers the packet immediately if rank \( (A_t) \) is sufficiently large; otherwise the packet is discarded entirely. Thus, the only parameters of interest are \( r \) and \( p \). In general, a Gabidulin code achieves zero-error recovery of every packet in \( \mathcal{CH}_B(r, p, B, G) \) for the following rates:

\[
R \leq \frac{r - p}{r}.
\]  

(4.2)

Comparing with (4.1), the streaming capacity of \( \mathcal{CH}_B(r, p, B, G) \) is achievable with a Gabidulin code when \( T < B \). However, for \( T \geq B \), these codes provide sub-optimal rates.

MSR codes can also be used directly for network streaming. Using Theorem 3 in Section 3.3, we can produce a low-delay streaming guarantee for this network. Recall that unlike in the previous chapter, we have \( n = r \nu \) where \( \nu > 0 \). In the following lemma, we provide the relevant decoding ability of MSR codes.

**Lemma 5.** For \( m \geq T \), let \( C[n, k, m] \) be an MSR code used to stream over the window \([0, T]\). Let \( A_{[0,T]} \) be the channel matrix.

1. If rank \( (A_{[0,j]}) \geq R(j + 1)r \), then the first source packet \( s_0 \) is recoverable by time \( j \leq T \).
2. If rank \( (A_{[0,T]}) \geq R(T + 1)r \) with the additional constraint that

\[
\text{rank} (A_i) = \begin{cases} 
  r - p & i \in [0, B - 1] \\
  r & i \in [B, T]
\end{cases},
\]

(4.3)

then every source packet in the window \( s_{[0,T]} \) is entirely recoverable by time \( T \).

**Proof.** Statement 1 is a translation of Theorem 3 for \( \nu > 1 \). The extension is obvious when considering the channel in the vector notation. For completeness, we provide the proof in Appendix A.4.

For Statement 2, the rank constraint on \( A_{[0,T]} \) in conjunction with (4.3) implies

\[
Bp \leq (1 - R)(T + 1)r.
\]

(4.4)

The left hand side is equal to the rank deficiency of \( A_{[0,T]} \). The proof for Statement 2 is divided into two cases for when \( p \leq (1 - R)r \) and when \( p > (1 - R)r \).
Case 1: If \( p \leq (1 - R)r \), each source packet is recovered immediately after the channel macro-packet is received. The condition on \( p \) implies that \( \text{rank}(A_0) = r - p \geq Rr \). Using Statement 1 for \( j = 0 \), \( s_0 \) is guaranteed recoverable at time 0. We repeat the argument for each subsequent packet.

Case 2: If \( p > (1 - R)r \), each source packet is recovered at time \( t + T \). Because \( \text{rank}(A_{0,T}) \geq R(T + 1)r \), \( s_0 \) is recoverable at time \( T \) using Statement 1 for \( j = T \). In order to recover \( s_1 \), we shrink the interval of interest to \([1, T]\). Due to the block diagonal channel matrix, the rank of the channel matrix is bounded

\[
\text{rank}(A_{[1,T]}) = \text{rank}(A_{[0,T]}) - \text{rank}(A_0) \\
\geq R(T + 1)r - r + p \\
> RTr.
\]

The third line follows from the inequality condition on \( p \). Using Statement 1 for \( j = T - 1 \) ensures that \( s_1 \) is recoverable at time \( T \). We then repeat the argument for the subsequent source packets \( s_i \) by showing that \( \text{rank}(A_{[i,T]}) > R(T + 1 - i)r \) for all \( i = 0, \ldots, T \) and applying Statement 1 for \( j = T - i \).

Lemma 5 considers two different channel loss patterns, the first being arbitrary rank losses in a window and the second being a burst. The total tolerable rank deficiency of the channel matrix is the same for both cases and the decoder does not perform better for any specific pattern. An MSR code with memory \( m = T_{\text{eff}} \) is feasible over \( CH_B(r, p, B, G) \) for the following rates:

\[
R \leq \max \left( \frac{(T_{\text{eff}} + 1)r - Bp}{(T_{\text{eff}} + 1)r}, \frac{r - p}{r} \right) \tag{4.5}
\]

The two rates mark the different approaches an MSR code can take. The left term is drawn from rearranging (4.4) and is dominant when \( T_{\text{eff}} \geq B \). If the burst length exceeds the code memory, i.e., \( T_{\text{eff}} < B \), then the code must recover some of the packets before the burst ends. Given that \( G_0 \) acts as an MRD code, simply adopting the MRD strategy and recovering according to Theorem 2 yields a higher rate. Thus, \( T_{\text{eff}} < B \) directly corresponds to the capacity when \( T < B \). The streaming capacity of \( CH_B(r, p, B, G) \) is not achieved for \( T \geq B \) in general. It is achievable with an MSR code only for \( T < B \) and \( B = 1 \) when \( T \geq B \).

Figure 4.2 compares achievable rates for MRD and MSR codes in (4.2) and (4.5) respectively with the channel capacity in (4.1). Baseline streaming codes used in the single-link scenario are also addressed. The \( m \)-MDS code acts as a single-link version of the MSR code, while the MS code achieves the streaming capacity for the single-link Burst Erasure Channel. Both of these codes treat rank-deficient channel matrices as erasure channels where the entire channel macro-packet is discarded. When \( \text{rank}(A_t) = r \), the decoders invert the channel matrix and apply the respective single-link decoding technique. As a result, neither of these single-link codes vary their rate for variable \( p \).

Two different streaming setups are considered, comparing a smaller \( B = 10 \) versus larger \( B = 30 \) for a fixed \( r = 10, T = 30, \) and \( G = 30 \). The tolerable rank deficiency \( p \) is varied from 1 to \( r \). When \( p = r \), the channel is effectively an erasure channel where \( A_t \) is either a full-rank matrix or simply \( 0_{r \times r} \). At this point, the single-link codes achieve their best rates relative to the capacity. This is also the area of highest disparity between the achievable rates of the MRD and MSR codes and the channel capacity, as the codes are designed to take advantage of the partial information available in a rank-deficient channel. While MSR codes excel in comparison to the other codes for small \( p \), they are effectively equivalent to
Figure 4.2: A comparison of achievable rates using existing codes against the streaming capacity given in Theorem 7. The decoding deadline $T = 30$ and gap length $G = 30$ are fixed. The burst length is $B = 10$ and $B = 30$ for the two setups, while the tolerable rank loss per time instance $p$ is varied. The solid black line marking the capacity is achievable using our proposed code.
Chapter 4. Recovery Of Bursts In Networks Codes

$m$-MDS codes when $p = r$. Moreover, the MRD codes fails absolutely at this point, as the receiver observes no useful linear combinations of the transmitted channel packets.

4.4 Recovery Of Bursts In Networks Codes

We prove the achievability of Theorem 7 with a new family of codes designed to recover from a burst of rank-deficient channel matrices. These codes are referred to as ROBIN codes and achieve the streaming capacity over $\mathcal{CH}_B(r, p, B, G)$.

4.4.1 Encoder

Recall the prior assumption for the codeword length $n = \nu r$. For our construction, we further assume that $k = \nu r$ for some $\nu > 0$. Consequently, the rate simplifies to $R = \frac{\nu}{\nu}$ and the encoder is designed for any pair of $\nu$ and $\nu$. When introducing the decoder, the value of $\nu$ and $\nu$ are fixed as functions of the channel parameters and decoding delay. The encoding steps are provided below and summarized in Fig. 4.3.

1. **Split source:** Let $\kappa_u > 0$ and $\kappa_v = \nu - \kappa_u$ be two code parameters to which we assign values later. Split the source packet

   $s_t = (s_t,0,\ldots,s_t,\kappa_u r - 1; s_t,\kappa_u r,\ldots,s_t,\kappa_u r + \kappa_v r - 1)$

   into two groups of $\kappa_u r$ urgent $(s^u_t)$ and $\kappa_v r$ non-urgent $(s^v_t)$ source symbols.

   The terminology of urgent and non-urgent symbols is inherited from prior layered constructions of streaming codes [4, 25]. When decoding, all non-urgent source symbols affected in a burst are recovered simultaneously once the decoder receives sufficient information. In contrast, the urgent source symbols are recovered immediately when the associated parity-check symbols arrive.

2. **MSR code:** Let $\kappa_p > 0$ be a code parameter to which we assign a value later. Apply an MSR code $\mathcal{C}[(\kappa_v + \kappa_p)r, \kappa_v r, T_{\text{eff}}]$ over $s^v_t$ in order to generate the non-urgent codeword

   $x^v_t = \sum_{i=0}^{T_{\text{eff}}} s^v_{t-i} G_i$.

3. **MRD code:** Apply a systematic MRD code $\mathcal{C}[(\kappa_u + \kappa_p)r, \kappa_u r]$ over $s^u_t$ in order to generate the urgent codeword

   $(s^u_t, p^u_t) = s^u_t G$.

4. **Shift and zero-pad:** Shift $p^u_t$ in time with delay $T_{\text{eff}}$ for later use. At time $t$, we receive the the delayed parity-check vector $p^u_{t-T_{\text{eff}}}$. Zero-pad this with $\kappa_v r$ symbols in order to construct $(p^u_{t-T_{\text{eff}}}, 0)$. The zero-padded vector now has the same length as $x^v_t$. 

Figure 4.3: A block diagram illustrating the encoding steps of a ROBIN code. The source packet is first split into two sub-packets and a different code is applied to each. The resulting parity-checks are then combined to form the overall parity-check packet. Finally, the parity-check packet and the urgent source packet are concatenated to generate the channel packet.

5. **Concatenate:** Concatenate the urgent source symbols with the summation of the non-urgent codeword and parity-check symbols to construct the ROBIN codeword

\[
x_t = \left( s_t^u, \ldots, s_t^{(\kappa_u-1)r}, x_t^v, \ldots, x_t^{(\kappa_p-1)r} \right).
\]

The codeword is effectively divided into three partitions: the urgent source symbols \( s_t^u \), the non-urgent symbols with entries from \( x_t^v \), and the overlapped parity-check symbols from the summation of \( x_t^v \) and \( p_t^{uT_{T_{eff}}} \). The three partitions contain \( \kappa_u r \), \( \kappa_v r \), and \( \kappa_p r \) symbols respectively. The channel macro-packet is formed by sequencing the codeword into vectors of length \( r \). There are naturally \( \kappa_u \), \( \kappa_v \), and \( \kappa_p \) vectors comprised solely of urgent, non-urgent, and overlapped parity-check symbols respectively.
A single column of $X_t$ contains $\kappa_u$, $\kappa_v$, and $\kappa_p$ urgent, non-urgent, and overlapped symbols respectively. The columns are each channel packets. The rank of $A_t$ decreasing by one implies that $\kappa_u + \kappa_v + \kappa_p$ linear combinations of symbols become redundant.

### 4.4.2 Decoder

We set the code parameters to the following values:

$$\kappa = T_{\text{eff}} + B(1 - \frac{p}{r}), \kappa_u = B, \kappa_v = T_{\text{eff}} - B\frac{p}{r}, \kappa_p = B\frac{p}{r}. \quad (4.9)$$

It is assumed that (4.9) yields integer valued results, but this may not always be the case in practice. When a parameter is not an integer, we multiply all code parameters by a normalizing factor. As $\nu$ is simply a function of the parameters, the normalizing factor affects it as well, leaving the code rate unchanged.

The urgent source symbols are protected by an MRD code $C[B(r + p), Br]$ and the non-urgent source symbols are protected by an MSR code $C[T_{\text{eff}}, T_{\text{eff}} - Bp, T_{\text{eff}}]$. We show below that the decoder with the parameters in (4.9) can completely recover all packets from a burst of length $B$ when there is a sufficiently large gap $G \geq B$. The procedure is in two steps. The non-urgent source symbols erased in the burst are first recovered simultaneously. Then the urgent source symbols are iteratively recovered with delay $T_{\text{eff}}$.

Suppose that there is a burst beginning at time $t$, i.e., $\text{rank}(A_i) = r - p$ for $i = t, \ldots, t + B - 1$. All prior source packets are known to the decoder and communication is perfect afterwards in the window $[t + B, t + B + T_{\text{eff}} - 1]$. Recall that $T_{\text{eff}} \leq G$. We show below that all source packets $s_{[t,t+1]}$ are recoverable within their respective decoding deadlines and detail the steps to recover $s_t$ by time $t + T_{\text{eff}}$ below. The remaining source packets affected by the burst are decoded using the same steps.

**Step 1:** The decoder recovers $s_t^{[v]}$ by time $t + T_{\text{eff}}$. This step is divided into two actions. To begin, $s_{[t,t+T_{\text{eff}}]}$ are recovered simultaneously at time $t + T_{\text{eff}} - 1$. At time $t$, the decoder then recovers $s_{t+T_{\text{eff}}}$.

All source packets before time $t$ are known to the decoder, meaning $p_t^{[u]}$ can be computed and negated from the associated overlapped symbols in the interval $[t, t + T_{\text{eff}} - 1]$. The remaining symbols in the non-urgent and overlapped sections are all linear combinations of only $X_t^{[v]}$. The MSR code protecting the non-urgent source symbols is now decodable. From Statement 2 of Lemma 5, all non-urgent source symbols can be recovered at time $t + T_{\text{eff}} - 1$ if

$$Bp \leq \frac{\kappa_p}{\kappa_v + \kappa_p} T_{\text{eff}} r.$$ 

Substituting the code parameters in (4.9) to the above reveals that the condition is met with equality. Then $s_t^{[v]}$ are all recoverable by time $t + T_{\text{eff}} - 1$.

$s_{t+T_{\text{eff}}}$ is recovered at time $t + T_{\text{eff}}$ by considering a window of length 1. Once the effects of the prior source packets are negated, the receiver effectively observes

$$s_{t+T_{\text{eff}}} G_{0} \text{diag}(A_{t+T_{\text{eff}}};\kappa_p + \kappa_v) + (p_t^{[u]}, 0_{\kappa_v}) \text{diag}(A_{t+T_{\text{eff}}};\kappa_p + \kappa_v)$$

in the vector notation. The second term is assumed to have caused a rank deficiency in the first $\kappa_p$ columns of the channel matrix and we consequently discard those columns and the associated received
symbols. Due to \( G_0 \) possessing the properties of an MRD generator matrix, using Theorem 2 reveals \( s^u_{t+T_{eff}} \) to be recoverable.

**Step 2:** The decoder recovers \( s^u_t \) at time \( T_{eff} \). Because \( s^u_{t,t+T_{eff}} \) are all known to the decoder, \( x^u_{t+T_{eff}} \) is computable and can be negated from the overlapped symbols at time \( t + T_{eff} \). Thus, \( p^u_t \) is available to the decoder at time \( t + T_{eff} \). The MRD code protecting the urgent source symbols is now decodable. In the vector notation, the receiver observes the product of \( (s^u_t, p^u_t) \) with a concatenation of the channel matrices at time \( t \) and \( t + T_{eff} \). Using (4.7), we write this as

\[
\begin{bmatrix}
 s^u_t G \\
 \end{bmatrix}
 = \begin{bmatrix}
 \text{diag}(A_t^s; \kappa_u) \\
 \text{diag}(A_{t+T_{eff}}^s; \kappa_p)
 \end{bmatrix}.
\]

The rank of the channel matrix above is \( \kappa_u (r - p) + \kappa_p r \), being the summation of the ranks of \( A_t^s \) and \( A_{t+T_{eff}}^s \). By Theorem 2, the source can be completely recovered if

\[
\kappa_u (r - p) + \kappa_p r \geq \kappa_u r.
\]

Substituting the values of (4.9) to the above reveals the condition is met with equality. Thus, \( s^u_t \) is recovered at time \( t + T_{eff} \).

The next source packet \( s_{t+1} \) must be recovered by time \( t + T_{eff} + 1 \). All prior non-urgent source symbols are known to the decoder, so \( s^u_{t+T_{eff}+1} \) can be recovered using Step 1. Once \( x^u_{t+T_{eff}+1} \) is computed, Step 2 allows \( s^u_{t+1} \) to be recovered. We repeat this technique for the subsequent packets in the burst by first recovering the non-urgent symbols at each time instance. Each urgent source vector is thus sequentially recovered iteratively with delay \( T_{eff} \).

### 4.4.3 Decoding Example

In Fig. 4.4, we provide an example of the encoding and decoding steps. Consider a channel \( CH_{3B}(3,2,3,6) \), which experiences a burst affecting two paths in the interval \([0,2]\). For simplicity, we assume that when full rank, \( A = I_{r \times r} \) is an Identity matrix, and otherwise \( A^s \) contains only the third column of \( I_{3 \times 3} \).

The receiver then observes all transmitted channel packets when there are no losses and only one of the packets in each shot during the burst. The rank deficiency in the channel matrix is then equivalent to causing channel packet erasures. The network in the interval \([0,6]\) is given in Fig. 4.4a, where the erased links are shaded.

We use a ROBIN code with memory \( T_{eff} = 6 \). Using (4.9), the source packet \( s_t \) is split into \( s^u_t \in F^9_{q^M} \) and \( s^v_t \in F^{12}_{q^M} \). The non-urgent source sub-packet is protected with an MSR code \( C[18,12,6] \), to generate \( x^v_t \), whereas the urgent source sub-packet is protected with an MRD code \( C[15,9] \), to generate \( (s^u_t, p^u_{t-T_{eff}}) \). In each channel packet, the encoder transmits 3 urgent source symbols, 2 overlapped symbols, and 4 non-urgent symbols.

The decoding steps to recover \( s_0 \) at time \( t = 6 \) are as follows. We count the number of erased symbols versus the number of received symbols to determine recoverability. In Fig. 4.4b, the decoder first negates \( p^u_{[6-1]} \) from the other overlapped symbols. This leaves \( x^v_{[0,5]} \) at the decoder. There are 36 erased symbols of the MSR code in the burst out of the 108 transmitted, leaving a total of 72 correctly received symbols. This is sufficient to recover the 72 non-urgent source symbols comprising \( s^v_6 \). The decoder then recovers \( s^u_6 \) at time \( t = 6 \) using the 12 non-urgent received symbols of \( x^v_6 \).

In Fig. 4.4c, the decoder recovers \( s^u_6 \) at time \( t = 6 \). Given \( s^v_{[0,6]} \), the decoder reconstructs \( x^v_6 \) and
Chapter 4. Recovery Of Bursts In Networks Codes

Figure 4.4: Example of a ROBIN code with memory $T_{\text{eff}} = 6$ recovering from a burst $B = 3$ in a network with $r = 3$. Each column represents a received channel packet along a given path, which is a linear combination of the transmitted packets. The hatched columns are the linearly dependent combinations.

(a) A slot consists of 3 channel packets—columns—each of which includes 3 urgent source, 2 overlapped, and 4 non-urgent symbols. Failing links in the interval $[0, 2]$ force several columns (shown as shaded) of the respective macro-packets redundant.

(b) $p_{t-6, 1}$ is negated from the overlapped symbols. The overlapped and non-urgent symbols form an MSR code with 12 rank losses per shot in the first three shots, allowing the recovery of $s_{[0, 5]}$.

(c) After recovering and negating $s_6$, the overlapped section at $t = 6$ and urgent section at $t = 0$ form an MRD code with 6 rank losses.

Figure 4.4: Example of a ROBIN code with memory $T_{\text{eff}} = 6$ recovering from a burst $B = 3$ in a network with $r = 3$. Each column represents a received channel packet along a given path, which is a linear combination of the transmitted packets. The hatched columns are the linearly dependent combinations.
Figure 4.5: A periodic burst rank loss network, where the channel matrix rank \((A^*) = r - p\) during bursts and rank \((A) = r\) during perfect communication.

A Gabidulin codeword \((s^0_u, p^0_u)\) remains with 6 erasures. The 3 correctly received symbols of \(s^0_u\) plus 6 received symbols of \(p^0_u\) are sufficient to recover the 9 symbols comprising \(s^u\) within the deadline.

### 4.5 Upper Bound on the Burst Loss Network

In this section, we prove the converse to Theorem 7. The proof for an upper bound on achievable rates for \(CH_B(r, p, B, G)\) follows similar to prior proofs establishing the streaming capacity of the Burst Erasure Channel [4, 25].

#### 4.5.1 Upper Bound for \(T \geq B\)

We first address the case of \(T \geq B\). A network with \(r\) paths, out of which \(p\) paths periodically deactivate in a burst, is described in Fig. 4.5. This network has a period of \(B + G\), where the ranks of the channel matrices \(A_i\) in the \(i\)-th period is given by

\[
\text{rank}(A_i) = \begin{cases} 
  r - p & t \in [i(B + G), i(B + G) + B - 1] \\
  r & t \in [i(B + G) + B, (i + 1)(B + G) - 1]
\end{cases}
\]

for \(i \in \{0, \ldots, L\}\). Here, \(L\) is the total number of periods over which communication occurs. When the channel matrix is full-rank, we denote it \(A_i = A\), whereas the rank-deficient channel matrix is \(A_i = A^*\). Every period consists of the network experiencing a burst of length \(B\) followed by a gap of full-rank channel matrices for \(G\) shots, meaning we can describe it as \(CH_B(r, p, B, G)\). We use a counting argument for the number of linearly independent received combinations of symbols over the first period \([0, B + G - 1]\) to show

\[
(B + G)nR \leq B(n - np) + Gn \\
R \leq \frac{Gr + B(r - p)}{(B + G)r}.
\]  \hspace{1cm} (4.10)

When \(T \geq G\), we substitute \(G = T_{\text{eff}}\) to (4.10), returning the streaming capacity. This result can also be obtained by directly solving for the Shannon capacity of the entire periodic network.

This counting argument does not lead to the capacity when \(T < G\). By considering the decoding delay constraint for a streaming code, the upper bound in (4.10) can be tightened. Consider a new periodic network with period \(B + T\). The channel matrices in the \(i\)-th period has ranks described as...
follows:
\[
\text{rank}(A_t) = \begin{cases} 
    r - p & t \in [i(B + T), i(B + T) + B - 1] \\
    r & t \in [i(B + T) + B, (i + 1)(B + T) - 1]
\end{cases}
\]  
(4.11)

For this network, every burst of length \(B\) is followed by a gap of only \(T\) shots. The channel matrices remain \(A^*\) and \(A\) respectively as described above. This channel cannot be described by \(CH_B(r, p, B, G)\) as the gaps are insufficiently short. Nonetheless, recall that any code that is feasible over \(CH_B(r, p, B, G)\) must recover every source packet with delay \(T\). We argue the following claim.

**Claim 1.** Any code feasible over \(CH_B(r, p, B, G)\) is also feasible over the network defined in (4.11).

A simplified argument is presented here, whereas a rigorous information theoretic proof is given in Appendix A.5. The key insight is that when the decoder is concerned with recovering \(s_t\), the channel is only relevant in the interval \([0, t + T]\). In analysis, the periodic network can be replaced by a hypothetical effective network \(\text{diag}(A_0, \ldots, A_{t+T}, A, \ldots)\), comprised of the original channel matrices for \([0, t + T]\) with all subsequent shots assumed to be transmitted perfectly. We prove feasibility in two steps for each period of the network, beginning with the first period \([0, B + T - 1]\).

**Step 1:** First consider the packets affected by the burst, i.e., \(t \in [0, B - 1]\). The maximum tolerable delay is \(T\), but we relax the constraints to require that every packet affected by the burst is completely recovered by time \(B + T - 1\) before the second burst begins. From the perspective of these packets, the network is equivalent to a hypothetical effective network where the channel matrix is rank-deficient in the window \([0, B - 1]\) and full-rank for all subsequent shots. A feasible streaming code guarantees that every packet in the burst is recoverable within the decoding constraint for this network.

**Step 2:** Next, we recover the source packets for \(t \in [B, B + T - 1]\) remaining in the period. Similar to the previous packets, we relax the requirement to every packet in this interval being completely recovered by time \(2B + 2T - 1\) at the end of the second period. At time \(B + T - 1\), the packets affected by the first burst have all been recovered and their effect can be negated for the current packets of interest. We permit the network to start a new burst for \(B\) shots. The network is then equivalent to a hypothetical effective network where the channel matrices follow (4.11) for \([0, 2B + T - 1]\), before becoming full-rank again for all subsequent shots. A feasible streaming code guarantees that every packet in the interval \([B, B + T - 1]\) is recoverable within the decoding constraint for this network.

The packets affected by the second burst at \([B + T, 2B + T - 1]\) are addressed next. The effect of all packets in the first period is negated by time \(2B + 2T - 1\), so we reuse the argument for the packets affected by the first burst, i.e., all source packets in the window \([B + T, 2B + T - 1]\) are completely recovered by time \(2B + 2T - 1\). The first burst has been recovered, so the effective network contains only a single burst of length \(B\) and a feasible code guarantees recoverability. The remaining packets in the second period are then recovered in the same way as in the above by time \(3B + 3T - 1\).

A feasible decoder effectively completes decoding from one period of the network before pursuing the next. Suppose that this periodic channel continues for \(L\) periods. Furthermore, we permit \(L + 1\)-th grace period that ensures enough transmissions for the packets in the \(L\)-th period to be recovered. Reusing the counting argument reveals

\[
L(B + T)nR \leq (L + 1)B(n - \nu p) + (L + 1)Tn.
\]
By substituting $T = T_{\text{eff}}$ and letting $L$ grow asymptotically, we re-arrange this equation to recover (4.1). Thus, all achievable rates are bounded by the capacity for both cases. This completes the converse.

### 4.5.2 Upper Bound for $T < B$

When $T < B$, the first packet in a burst of length $B$ must be recovered before the burst ends. In fact, assuming that all prior source packets are known, any packet $s_t$ is expected to be recoverable by the end of an interval $[t, t + T]$ experiencing only bursts. Consequently, the inter-burst gap is irrelevant and any feasible code over $CH_B(r, p, B, G)$ is also feasible over a perpetually rank-deficient network, i.e., $\text{rank}(A_t) = r - p$ for all $t \geq 0$. Decoding is performed similar to the previous scenario where $T < G$, but by considering each source packet individually. Once $s_t$ is recovered at time $t + T$, we move to the next time instance and negate the effect of the recovered packet. Then $s_{t+1}$ is recovered in the same manner. The previous counting argument yields

$$nR \leq n - vp,$$

which can be rearranged to the capacity. An information theoretic argument similar to the one given in Appendix A.5 can be made to show the same results.

### 4.6 Simulation Results

We use simulations to evaluate the performance of ROBIN codes over statistical networks. Single-link burst erasure streams have been modelled in previous works using a Gilbert channel [3,4]. We appropriate this model to characterize a network with bursts of rank-deficient matrices. The Gilbert channel is a Markov model with two states: a good-state where the channel matrix $A_t = A$ is full-rank, i.e., $\rho_t = r$, and a bad-state where the channel matrix $A_t = A^*$ is rank-deficient, i.e., $\rho_t = r - p$. The bad-state represents a burst loss event, i.e., $p$ paths deactivate forcing the receiver to observe $p$ linearly dependent combinations of the channel packets for each shot while the channel remains in the bad-state. The transition probability from good-state to bad-state is given by the parameter $\alpha$ whereas the transition probability from bad-state to good-state is equal to $\beta$. The length of a burst is a geometric random

<table>
<thead>
<tr>
<th></th>
<th>Fig. 4.6</th>
<th>Fig. 4.7</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\alpha$</td>
<td>$5 \times 10^{-4}$</td>
<td>$5 \times 10^{-4}$</td>
</tr>
<tr>
<td>$\beta$</td>
<td>(0.1, ..., 0.8)</td>
<td>0.3</td>
</tr>
<tr>
<td>$r$</td>
<td>6</td>
<td>16</td>
</tr>
<tr>
<td>$p$</td>
<td>3</td>
<td>(2, ..., 16)</td>
</tr>
<tr>
<td>Channel Length</td>
<td>$10^8$</td>
<td>$10^8$</td>
</tr>
<tr>
<td>Rate $R$</td>
<td>$\frac{10}{13}$</td>
<td>(0.94, ..., 0.52)</td>
</tr>
<tr>
<td>Delay $T$</td>
<td>7</td>
<td>17</td>
</tr>
</tbody>
</table>

Table 4.1: Parameters used in simulations.
variable with mean $\frac{1}{\beta}$, whereas the length of the gaps between bursts is geometric with mean $\frac{1}{\alpha}$. The adversarial Burst Rank Loss Network approximates the Gilbert channel, with $B$ and $G$ chosen to emulate $\frac{1}{\beta}$ and $\frac{1}{\alpha}$ respectively.

Decoders are not explicitly implemented in the experiments. The analysis consists of calculating whether a given packet is recoverable by counting the number of available linear combinations. To generate easily computable simulations, we assume that the MSR and $m$-MDS codes have infinite memory for themselves and as constituents. Consequently, source packets can be recovered after the deadline but are not considered successfully decoded. We measure the packet loss rate, given by the frequency of source packets that are not completely recoverable within their respective deadlines. The burst length parameter $\beta$ is varied along the $x$-axis in the first experiment, whereas in the second, we vary the rank-deficiency $p$. We plot loss probability on the $y$-axis.

### 4.6.1 Variable Burst Length Parameter

For this experiment, the bad-state transition probability $\beta$ is varied from 0.1 to 0.8. The good-state transition probability $\alpha$ is equal to $5 \times 10^{-4}$. Both $r = 6$ and $p = 3$ are fixed. Fig. 4.6b shows the burst length distribution for $\beta = 0.5$. The code rate is set at $R = \frac{20}{26} \approx 0.77$ and the decoder permits a maximum tolerable delay $T = 7$. These channel and code parameters are summarized in Table 4.1.

In Fig. 4.6a, the packet loss rate of the ROBIN code is measured and compared to the loss rate of the MSR code and their single-link counterparts: the MS and $m$-MDS codes.

- **$m$-MDS code:** The $m$-MDS code loss rate is the dashed cyan line with '△'. Having been designed for single-link channels, the $m$-MDS code discards the macro-packet if the channel matrix is rank-deficient. The maximum burst length for perfect recovery with delay $T = 6$ is $B = 1$, meaning if the channel is in the bad-state for two consecutive time instances, the first packet is not recoverable within the deadline. However, this code is capable of recovering part of a burst within the delay constraint. For example, if a burst of length 2 occurs at time $t$, both source packets are recovered simultaneously at time $t + 7$. Consequently, the second source packet meets its deadline even though the first packet fails. This ability to recover a fraction of packets affected by a burst is referred to as 'partial recovery'.

- **MS code:** The dashed purple line with '∇' represents the MS code loss rate. Having been designed for single-link streaming, this code also considers a rank-deficient channel as an erasure of the entire macro-packet. The MS code affords a larger $B = 3$ than the $m$-MDS code, as marked in Fig. 4.6b. As a result, it achieves a lower loss rate in comparison to the $m$-MDS code.

Both the MS and the $m$-MDS codes perform slightly better than the rank metric codes for $\beta = 0.1$. This region is characterized by long bursts, which are not recoverable by any of the codes. The $m$-MDS code (and by extension, the MS code) are systematic constructions, meaning that packets after a burst are immediately recovered simply by reading the source symbols. In contrast, the MSR and ROBIN codes are non-systematic. This forces even the packets after a burst to not be recoverable until all prior packets are recovered. Using a systematic code gives a technical improvement over the non-systematic codes, but the effect is not visible except in the extreme case of long bursts.

- **MSR code:** The loss rate of the MSR code is given by the solid red line marked '○'. This code
Figure 4.6: Simulations over a Gilbert Channel modelling a network with rank \((A_t) = 6\) in the good-state and 3 in the bad-state. The channel and code parameters are given in Table 4.1.
does not discard the entire macro-packet for a rank-deficient channel. Furthermore, this code is capable of partial recovery for bursts exceeding the maximum tolerable length. The MSR code then has a significantly lower loss probability in comparison to the single-link codes. If a burst of length 4 occurs at time $t$, the last three source packets in the burst are recoverable within their respective deadlines using an MSR code. In contrast, all four source packets are lost when using a MS code. The MSR code shows that rank metric codes designed for streaming over a network provide significant gains in comparison to single-link codes. However, the maximum tolerable burst length $B = 3$ is significantly smaller than that of the ROBIN code, as marked in Fig. 4.6b.

- **ROBIN code**: The solid blue line marked '□' shows the loss rate of the ROBIN code. Similar to the MSR code, this code is capable of decoding using the linearly independent packets in rank-deficient channel matrices. In addition, the ROBIN code guarantees the largest tolerable burst length $B = 6$. As a result, this code easily outperforms all of the other codes for nearly all values of $\beta$.

The only region in these simulations where ROBIN codes do not significantly outperform the others is when $\beta < 0.2$. This region is indicative of long bursts, with the mean burst length being 10 when $\beta = 0.2$. Note that the mean burst length is already larger than the memory of the codes. All codes fail to recover any packets affected by the bursts and therefore all achieve similarly high loss rates. The $m$-MDS and MS codes slightly outperform the ROBIN codes for $\beta = 0.1$ as previously discussed. This is due to the non-systematic construction, which makes the ROBIN and MSR codes incapable of recovering packets after a long burst with low delay.

### 4.6.2 Variable Rank-deficiency

For this experiment, we fix the Gilbert channel parameters $\alpha = 5 \times 10^{-4}$ and $\beta = 0.3$. The decoder delay $T = 17$ and network rate $r = 16$ are also fixed. We vary the bad-state rank-deficiency $p$ to compare the relative performances to the achievable rates shown in Fig. 4.2. For every tested value of $p$, we change the rate $R$ of the codes, while keeping the ratio $\frac{1-R}{p}$ constant. This ensures that for every pair $(p,R)$, the maximum tolerable burst lengths $B$ of the ROBIN code and the MSR code remain fixed. Naturally, the single-link codes are not dependent on $p$ and therefore their $B$ values decrease as $R$ increases for fixed $\frac{1-R}{p}$. The channel and code parameters are summarized in Table 4.1. The packet loss rates of the codes are displayed in Fig. 4.7b. Comparing with Fig. 4.2, the simulations match the predicted results, justifying the adversarial approach.

- **$m$-MDS code**: The $m$-MDS code loss rate is the dashed cyan line with '△'. Predictably, this code reveals the worst performance throughout the experiment. Matching the achievable rates in Fig. 4.2, the $m$-MDS code and MSR code have the highest disparity for small $p$ and achieve the same performance when $p$ approaches $r$. By forcing constant $\frac{1-R}{p}$, the code rate decreases as $p$ increases, permitting larger bursts to be recoverable.

- **MS code**: The dashed purple line with '∇' represents the MS code loss rate. The performance is comparable to the $m$-MDS code when $p$ is small, but converges to the ROBIN code as $p$ increases. Because the network for $p = r$ effectively behaves as an erasure channel, the MS code and $m$-MDS code match single-link streaming simulations in [4]. MS codes use a similar layering technique to
Figure 4.7: Simulations over a Gilbert Channel modelling a network with rank \((\mathbf{A}_t) = 16\) in the good-state and \(16 - p\) in the bad-state. The channel and code parameters are given in Table 4.1.
ROBIN codes. At $p = r$, their code parameters converge to the same values, and thus, they are effectively equivalent.

- **MSR code**: The loss rate of the MSR code is given by the solid red line marked '○'. This code has a constant tolerable burst length $B$ for every pair $(p, R)$. As a result, the performance remains constant throughout the experiment. The MSR and $m$-MDS code performances naturally become the same when $p = r$, similar to the comparison in Fig. 4.2 of achievable rates. When $p = r$, burst recoverability condition in (4.4) for MSR codes is equivalent to that for $m$-MDS codes [8].

- **ROBIN code**: The solid blue line marked '□' shows the loss rate of the ROBIN code. By keeping the ratio $\frac{1-R}{p}$ constant, we fix $B$ of the ROBIN code. Thus, the loss rate does not change as $p$ increases. When $p = r$, the channel behaves as an erasure channel and the MS and ROBIN code effectively become the same. Once again, this result validates the predictions from Fig. 4.2. The key difference between MS and ROBIN codes is the use of a non-systematic encoder to protect the non-urgent symbols by the ROBIN code. However, this does not yield a significant difference in the loss rates for the above simulations.
Chapter 5

Conclusion

Streaming over erasure channels has recently grown to become an area with considerable impact in applications. This thesis aims to extend and generalize several results in single-link streaming to a multi-node network environment where every link behaves as an erasure channel. Rather than applying a streaming code for every link, we consider a pre-existing linear network code over the network and design an end-to-end streaming code for the source and destination nodes. The erasures in individual links translate to rank deficiency in the channel matrix and we consequently develop a rank metric model for streaming.

Two scenarios are addressed in this thesis. In Chapter 3, we consider streaming over a network extension of the Isolated Erasure Sliding Window Channel. The channel matrix rank decreases arbitrarily down to a minimum value within any sliding window of network uses. The column sum rank is introduced and shown to possess several analogous properties to the column Hamming distance. MS codes are proposed and constructed. They achieve the maximum $m$-th column sum rank, where $m$ is the code memory, and are rank metric analogues of $m$-MDS codes. However, the construction provided for MSR codes are only guaranteed for significantly large extension fields. Numerically, we can show that the property exists for some constructions over smaller fields and can also be found in randomly constructed block diagonal matrices.

In Chapter 4, we develop the network extension to the classic Burst Erasure Channel. The channel matrix becomes rank-deficient for a burst of consecutive network uses before returning to full-rank for a gap period. Just as layered constructions using $m$-MDS codes have been shown to achieve the streaming capacity for the Burst Erasure Channel, we use a layered construction of Gabidulin and MSR codes. The ROBIN codes introduced in this chapter are proven to be capacity achieving. A converse to the streaming capacity is proven and simulations are performed comparing ROBIN codes, MSR codes, and single-link streaming codes.

This thesis reveals many new questions on both the theory of rank metric convolutional codes and network streaming. We summarize several below.

- Although $m$-MDS codes attain the maximum column Hamming distance up to the code memory, they are not the only ones to do so. In fact, there exist several variants of convolutional codes with strong Hamming distance properties, i.e. Strongly-MDS codes and Maximum Distance Profile (MDP) codes both attain the maximum column Hamming distance for points larger than $m$, and MDS convolutional codes possess the maximum free Hamming distance. It would be interesting
to see whether rank metric analogues of these codes also exist.

- The MSR encoder given in Subsection 3.4.2 is non-systematic, whereas systematic $n$-MDS encoders can be constructed using super-regular matrices. There is no general theory available yet for systematic MSR encoders. A hypothetical systematic encoder can significantly impact practical constructions, particularly for layered code constructions.

- The streaming capacity is incomplete for $B > G$ over both single-link burst channels and burst networks. Codes exist for some special cases, but there is no known general solution. Although this is not an area of practical significance, achievability proofs this region would complete all regions for both problems.

- There exist many sophisticated erasure channels formed from combining the isolated and burst channels; the Robust Burst Erasure Channel for example, permits there to be either a burst or several isolated erasures in any sliding window. More specialized layered codes are known to outperform existing baseline codes on the single-link channels and the layering technique can easily be extended to network versions of these codes. However, it can be shown that the network versions are not always as dominant over networks as the original layered codes are over erasure channels. This suggests that layering may not always be the best approach in these scenarios and alternative coding approaches may provide better results for the more sophisticated network models.
Appendix A

Omitted Arguments

A.1 Upper Bound on \( \text{deg} \, D(X) \) in Lemma 2

The bounds were proven in [2] for \( q = 2 \) in order to show that the matrix is super-regular. We will prove the upper bound for more general \( q \). We refer the reader to the original work for the lower bound, as the technique is identical.

The degree of \( D(X) \) is bounded by considering the degree of the polynomial terms. When constructing each \( D_\sigma(X) = \prod_{i=0}^{b-1} D_{i,\sigma(i)}(X) \), up to two entries from the last row and column can be used. Choosing \( D_{b-1,b-1} \) prevents selecting a second entry, but due to (2.6), \( D_{b-1,b-1}(X) \) has a greater degree than the sum of any other two entries. Note that we can bound \( \text{deg} \, D_{b-1,b-1}(X) \leq q^{n(m+2)-2} \). We next consider the second last row and column. However, the degrees of entries here are bounded \( \text{deg} \, D_{b-2,b-2}(X) \leq q^{n(m+2)-4} \). This argument is used recursively to bound the degree of \( D_\sigma(X) \):

\[
\sum_{i=0}^{b-1} \text{deg} \, D_{i,\sigma(i)}(X) \leq \sum_{k=0}^{b-1} q^{n(m+2)-2-2k} = q^{n(m+2)-2} \sum_{k=0}^{b-1} q^{-2k} < q^{n(m+2)-2} \frac{1}{1-q^{-2}} < q^{n(m+2)-1}.
\]

A.2 The Column Sum Rank in Error Correction

The column Hamming distance of a code determines low-delay error correction capabilities along with erasure correction. The column sum rank behaves similarly. An analogous argument to Theorem 3 is presented below.

**Proposition 1.** Let \( C[n,k,m] \) be a convolutional code used to stream over the window \( [0,W-1] \). Suppose the receiver observes \( y_{[0,W-1]} = x_{[0,W-1]} + e_{[0,W-1]} \), where \( e_{[0,W-1]} \in F_q^{n(j+1)} \) is an error sequence with \( \sum_{t=0}^{W-1} \text{rank} \, (\phi_n(e_t)) = E \).

1. If \( d_R(W-1) > 2E \), then \( s_0 \) is recoverable by time \( W-1 \).
2. If \( d_R(W - 1) \leq 2E \), then there exists at least one codeword and error sequence for which \( s_0 \) is not recoverable by time \( W - 1 \).

**Proof.** Consider two source sequences \( s_{[0,W-1]} \) and \( \hat{s}_{[0,W-1]} \), which generate the codeword sequences \( x_{[0,W-1]} \) and \( \hat{x}_{[0,W-1]} \). Suppose there exist two error sequences \( e_{[0,W-1]} \) and \( \hat{e}_{[0,W-1]} \), for which \( x_{[0,W-1]} + e_{[0,W-1]} = \hat{x}_{[0,W-1]} + \hat{e}_{[0,W-1]} = y_{[0,W-1]} \), where both \( \sum_{t=0}^{W-1} \text{rank}(\phi_n(e_t)) \) and \( \sum_{t=0}^{W-1} \text{rank}(\phi_n(\hat{e}_t)) \) are at most \( \frac{2d_R(W-1) - 2}{2} \). Then, we arrive at the following contradiction on the column sum rank of the code:

\[
d_R(j) \leq \sum_{t=0}^{W-1} \text{rank}(\phi_n(x_t) - \phi_n(\hat{x}_t)) \\
\leq \sum_{t=0}^{W-1} \left( \text{rank}(\phi_n(x_t) - \phi_n(y_t)) + \text{rank}(\phi_n(y_t) - \phi_n(\hat{x}_t)) \right) \\
\leq \sum_{t=0}^{W-1} \left( \text{rank}(\phi_n(e_t)) + \text{rank}(\phi_n(\hat{e}_t)) \right).
\]

For the converse, consider the source sequences \( s_{[0,W-1]} \) and \( \hat{s}_{[0,W-1]} \) that generate codeword sequences \( x_{[0,W-1]} \) and \( \hat{x}_{[0,W-1]} \) for which \( \sum_{t=0}^{W-1} \left( \text{rank}(\phi_n(x_t) - \phi_n(\hat{x}_t)) \right) = d_R(W - 1) \). Assume without loss of generality that the linearly independent elements of each \( x_t - \hat{x}_t \) all lie in the first \( d_R(j) \) positions. Then we partition \( x_{[0,W-1]} - \hat{x}_{[0,W-1]} = (a, b, c) \), where \( a, b \in \mathbb{F}_{q^{M}}^{d_R(j)} \) are two sets containing the linearly independent elements of each \( x, y \) and \( c \in \mathbb{F}_{q^{M}}^{W - d_R(W-1)} \) is a vector where every element can be written as a linear combination of the elements of \( a \) and \( b \). Furthermore, we write \( c = c_a + c_b \), where the two terms are the contributions of the elements of \( a \) and \( b \) respectively.

Consider the following error vectors: \( e_{[0,W-1]} = (-a, 0, -c_a) \) and \( \hat{e}_{[0,W-1]} = (0, b, c_b) \). Both of these errors have sum rank equal to \( \frac{d_R(W-1)}{2} \). Moreover, \( x_{[0,W-1]} + e_{[0,W-1]} = \hat{x}_{[0,W-1]} + \hat{e}_{[0,W-1]} \), meaning that a decoder cannot distinguish between the two codeword sequences.  

\[\square\]

### A.3 Conditions on \( G_m^{\text{EX}} A_{[0,m]}^* \) having a Non-trivial Determinant in Theorem 6

Recall that all sub-matrices of \( T \) follow the structure of \( D \) in (3.8). For a given sub-matrix, let \( l_i \) be the number of rows of each \( D_i \) block. Similarly, let \( m_i \) be the number of columns of each \( O_i \) block. In [2], it was revealed that \( D \) has a non-trivial determinant only when \( l_i \geq m_i \) for all \( i = 1, \ldots, h \).

The product

\[
G_m^{\text{EX}} A_{[0,m]}^* = \begin{pmatrix}
G_0 A_0^* & G_1 A_1^* & \cdots & G_m A_m^* \\
G_0 A_1^* & G_1 A_1^* & \cdots & \vdots \\
\vdots & \vdots & \ddots & \vdots \\
G_0 A_m^* & \vdots & \cdots & G_m A_m^*
\end{pmatrix}
\]

possesses a reversed structure to (3.8). Each column block has \( k(j + 1) \) rows and \( \rho_j \) columns for \( j = 0, \ldots, m \). We treat the non-zero blocks analogously to the \( D_i \) blocks of \( D \). By the argument in [2], if \( k(j + 1) \geq \sum_{t=0}^{j} \rho_t \) for \( j \leq m \), then \( G_m^{\text{EX}} A_{[0,m]}^* \) has a non-trivial determinant. However, these conditions
A.4 Proof of Statement 1 in Lemma 5

Recall the assumption in Section 4.2 on the code length $n = r\nu$. The argument is identical to that for Theorem 3, except for the key difference being that $\nu = 1$ was fixed in the prior work.

Consider two source sequences $s_{[t,t+T]} = (s_t, \ldots, s_{t+T})$ and $\hat{s}_{[t,t+T]} = (\hat{s}_t, \ldots, \hat{s}_{t+T})$, where $s_t \neq \hat{s}_t$. Suppose they respectively generate two different macro-packet sequences $X_{[t,t+T]}^{*}$ and $\hat{X}_{[t,t+T]}^{*}$, where $X_{[t,t+T]}^{*} = \hat{X}_{[t,t+T]}^{*}$. In vector notation, $X_{[t,t+T]}^{*}$ expands to

$$(x_t, \ldots, x_{t+T}) \begin{pmatrix} \text{diag} (A_t^*; \nu) & \cdots & \text{diag} (A_{t+T}^*; \nu) \end{pmatrix}.$$ 

Note that $\text{rank} (\text{diag} (A_t^*; \nu)) = n \text{rank} (A_t)$. The difference $x_{[t,t+T]} - \hat{x}_{[t,t+T]}$ is a hypothetical codeword sequence whose sum rank is at least $(n - k)(T + 1) + 1$. However, $(x_i - \hat{x}_i) \text{diag} (A_i^*; \nu) = 0$ implies $\text{rank} (\phi(x_i) - \phi(\hat{x}_i)) \leq n - \nu \text{rank} (A_i^*)$ for $i \in [t, t+T]$. By summing each of the inequalities, we arrive at the following contradiction on the sum rank of this codeword sequence:

$$\sum_{i=t}^{t+T} \text{rank} (\phi(x_i) - \phi(\hat{x}_i)) \leq n(T + 1) - \nu \text{rank} (A_{[t,t+T]}^*) \leq n(T + 1) - \nu R(T + 1)r \leq (n - k)(T + 1).$$

A.5 Converse Proof of Theorem 7 for $T < G$

This proof follows similarly to the one in [3]. Consider the periodic network in Fig. A.1 and suppose the network is in use for $L$ periods. An $L + 1$-th period provides sufficient information to recover the packets in the $L$-th period, and the packets in the grace period do not need to be recovered. We use the following notation to facilitate the proof:

$$U_i^1 = \{s_{(T+B)i}, \ldots, s_{(T+B)i+B-1}\} \quad U_i^2 = \{s_{(T+B)i+B}, \ldots, s_{(T+B)(i+1)-1}\}.$$
A full rank channel matrix $\mathbf{W}$

Both relationships arise from the channel transfer matrix. The packets in $\mathbf{W}$ refer to the packets in the period that are affected by the burst, whereas the superscript 2 refers to macro-packets, and received macro-packets respectively in one period of transmission. The superscript 1 refers to the packets in the period that are affected by the burst. The entropy of $W_i$ is bounded

$$H(W_i^1) \leq H(V_i^1) \frac{r - p}{r} \quad \text{(A.1a)}$$

$$H(W_i^2) = H(V_i^2). \quad \text{(A.1b)}$$

Both relationships arise from the channel transfer matrix. The packets in $W_i^2$ and $V_i^2$ are related by a full rank channel matrix $\mathbf{A}$ and thus, the linear transformation does not change the entropy. However, $W_i^1$ and $V_i^1$ are related by a rank-deficient $\mathbf{A}^*$. As the received packets formed from linearly dependent combinations of the channel packets can simply be re-written using the remaining linearly independent channel packets, they decrease the entropy accordingly.

Before proving the converse, we assert two remaining relevant properties of a streaming code:

$$H(U_i^1 \mid V_{[0,i-1]}^1, W_i) = 0 \quad \text{(A.2a)}$$

$$H(U_i^2 \mid V_{[0,i-1]}^1, V_i^1, W_{i+1}^2) = 0. \quad \text{(A.2b)}$$

Both properties are due to the fact that for a feasible code, each source packet must be recovered with delay $T$. Now, let $\mathcal{S}$ denote the alphabet of source symbols and $\mathcal{X}$ the alphabet of channel symbols. The source entropy can then be bounded

$$L(B + T)H(\mathcal{S}) = H(U_{[0,L-1]})$$

$$\leq H(U_{[0,L-1]}, W_{[0,L]})$$

$$= H(W_{[0,L]}) + \sum_{i=0}^{L-1} H(U_i \mid U_{[0,i-1]}, W_{[0,L]})$$

$$= H(W_{[0,L]}) + \sum_{i=0}^{L-1} \left( H(U_i^1 \mid U_{[0,i-1]}, W_{[0,L]}) + H(U_i^2 \mid U_{[0,i-1]}, W_{[0,L]}) \right)$$

$$= H(W_{[0,L]}) + \sum_{i=0}^{L-1} \left( H(U_i^1 \mid V_{[0,i-1]}, W_{[0,L]}) + H(U_i^2 \mid V_i^1, V_{[0,i-1]}, W_{[0,L]}) \right) \quad \text{(A.3)}$$

$$= H(W_{[0,L]}) \quad \text{(A.4)}$$

$$\leq (L + 1)(H(W_i^1) + H(W_i^2))$$

$$\leq (L + 1)(B \frac{r - p}{r} + T) \log |\mathcal{X}|. \quad \text{(A.5)}$$

In the above, (A.3) is due to the channel macro-packets being causally generated from source packets. Using (A.2) gives (A.4). We arrive at (A.5) due to (A.1). The inequality can be re-arranged to bound...
the rate

\[ R = \frac{K}{N} \leq \frac{H(S)}{\log |X|} = \left( \frac{L + 1}{L} \right) \frac{T_{r} + B(r - p)}{(T + B)r} \]

\[ \quad \quad \quad \quad \quad \quad \quad \lim_{L \to \infty} \frac{T_{r} + B(r - p)}{(T + B)r}. \]

Replacing \( T_{\text{eff}} = T \) returns the capacity in (4.1).
Bibliography


