Repairing Cartesian Codes with Linear Exact Repair Schemes

Daniel William Valvo

Thesis submitted to the faculty of the Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Mathematics

Gretchen Matthews, Chair
Constantin Mihalcea
Daniel Orr

May 11th, 2020
Blacksburg, Virginia

Keywords: Cartesian code, Reed-Solomon code, Multivariate Polynomials, Finite Fields, Distributed Storage Systems

Repairing Cartesian Codes with Linear Exact Repair Schemes

Daniel William Valvo

Abstract

In this paper, we develop schemes to recover a single erasure when using a Cartesian code, in the context of a distributed storage system. Particularly, we develop schemes with considerations to minimize the associated bandwidth and maximize the associated dimension. The problem of recovering a missing node's data exactly in a distributed storage system is known as the *exact repair problem*. Previous research has studied the exact repair problem for Reed-Solomon codes, and produced significantly reduced bandwidth repair schemes. We focus on Cartesian codes, and show we can enact a recovery using a linear exact repair scheme framework, similar to the one outlined by Guruswami and Wooters in 2017 [1]. Our schemes achieve similar bandwidth results and dimension restrictions to the schemes produced for Reed-Solomon codes.

Repairing Cartesian Codes with Linear Exact Repair Schemes

Daniel William Valvo

General Audience Abstract

Distributed storage systems are systems which store a single data file over multiple storage nodes. Each storage node has a certain storage efficiency, the "space" required to store the information on that node. The value of these systems, is their ability to safely store data for extended periods of time. We want to design distributed storage systems such that if one storage node fails, we can recover it from the data in the remaining nodes. Recovering a node from the data stored in the other nodes requires the nodes to communicate data with each other. Ideally, these systems are designed to minimize the bandwidth, the inter-nodal communication required to recover a lost node, as well as maximize the storage efficiency of each node. A great mathematical framework to build these distributed storage systems on is erasure codes. In this paper, we will specifically develop distributed storage systems that use Cartesian codes. We will show that in the right setting, these systems can have a very similar bandwidth to systems build from Reed-Solomon codes, without much loss in storage efficiency.

Contents

1 Introduction

A distributed storage system is a way of storing a single data file over several nodes. If one node loses it's share of the data, we want to recover it from the data in the available nodes. The problem of recovering a missing node's data exactly is known as the *exact repair problem*. The standard approach is to map a data file onto a codeword in an erasure code. Then, each symbol of the codeword can be stored on its own node in a distributed storage system. With this set-up, recovering a missing node is equivalent to recovering an erasure in a codeword. In this paper, we will specifically be concerned with distributed storage systems that use evaluation codes. Classical algorithms to recover an erasure in an evaluation code require a certain subset of nodes to send all of their data to recover just one erasure, requiring a considerable amount of bandwidth. An alternative solution is to develop an algorithm which requires more nodes to send only *part* of their data. In some cases, this requires less information to be sent overall, therefore decreasing the necessary repair bandwidth. Previous research has studied these alternative algorithms on distributed storage system using Reed-Solomon and Reed-Mueller codes [1][3]. Specifically algorithms that implement *linear exact repair schemes* deliver promising results. Using this foundation, we will develop and analyze low-bandwidth algorithms on distributed storage systems that use more general Cartesian codes.

To start, we introduce the formal definition of code.

Definition 1.1 (Code) *We say that $\mathcal{C}$ is an $[n, \kappa, d]$ code over F to mean that $\mathcal{C}$ is an F-subspace of F^n, $\dim_F(\mathcal{C}) = \kappa$, and $d = \min\{wt(c) \mid c \neq 0\}$. Such a code is able to recover $d-1$ erasures using information from all other coordinates. Note, for any $c \in \mathcal{C}$, the weight, $wt(c)$, is the number of non-zero coordinates of c.*

One particularly useful type of code is a *linear code*. An $[n, \kappa, d]$ code over F is a linear code if it is a linear subspace of the vector space F^n. Every linear code is accompanied by what is known as a *dual code*.

Definition 1.2 (Dual Code) *Given an $[n, \kappa, d]$ linear code $\mathcal{C}$ over F, the dual code is $\mathcal{C}^\perp = \{x \in F^n \mid x \cdot c = 0$ for all $c \in \mathcal{C}\}$. The dual code is itself an $[n, n-\kappa]$ code.*

The dual code has many uses in coding theory, and we will find it is also essential to the implementation of linear exact repair schemes. We will now move on to defining objects important for our specific applications.

2 Preliminaries

Recall, this paper is largely concerned with developing low-bandwidth algorithms for fixing erasures in evaluation codes. Specifically Cartesian codes. We will spend this section introducing definitions and notation we will use throughout this paper when discussing these objects.

Firstly, we should introduce evaluation codes. An evaluation code is simply a code whose codewords come from the evaluation of particular functions on a particular evaluation set. The particular functions allowed, and the evaluation set are the two factors that determine the properties of an evaluation code.

The most widely used evaluation code is the Reed-Solomon code. It specifically works with functions that are polynomials of a certain degree. The formal definition follows.

Definition 2.1 (Reed-Solomon Code) *Let F be a finite field and let $A \subseteq F$. Define $n \coloneqq |A|$, and index the elements of A as follows: $A = \{a^{(1)}, a^{(2)}, \ldots, a^{(n)}\}$. Let k be any positive integer such that $k \leq n$. Then the Reed-Solomon code over F with evaluation set A is*

$$RS_k(F, A) = \{\ (f(a^{(1)}), f(a^{(2)}), \ldots, f(a^{(n)}))\ \mid\ f \in F[x] \text{ with degree less than } k\ \}.$$

Given a vector $v \in (F \setminus \{0\})^n$, the generalized Reed-Solomon code is

$$GRS_k(F, A, \vec{v}) = \{\ (v_1 f(a^{(1)}), v_2 f(a^{(2)}), \ldots, v_n f(a^{(n)}))\ \mid\ f \in F[x] \text{ with degree less than } k\ \}.$$

Remark 2.1 (Reed-Solomon Dual) *$RS_k(F, A)^{\perp} = GRS_{k'}(F, A, \vec{v})$, where $k' = n - k$ and $\vec{v}$ is a particular vector in $(F \setminus \{0\})^n$ with $\vec{v} = (v_1, v_2, \ldots, v_n)$. For our purposes, the particular vector $\vec{v}$ is unimportant, we just care that such a vector exists. However, the referenced Guruswarmi and Wooters paper has more information on how to calculate $\vec{v}$ from A. [1]*

Initially, work on creating low bandwidth exact repair schemes was done for distributed storage systems that use Reed-Solomon codes. In this paper, we adapt those ideas to the more general setting of Cartesian codes. We define Cartesian codes below.

Definition 2.2 (Cartesian Code) *Let F be a finite field. An evaluation set is a set $\mathcal{A} = A_1 \times A_2 \times \cdots \times A_m \subseteq F^m$. Define $N \coloneqq |\mathcal{A}|$ and index the elements of $\mathcal{A}$ as follows: $\mathcal{A} = \{\vec{\alpha}^{(1)}, \vec{\alpha}^{(2)}, \ldots, \vec{\alpha}^{(N)}\}$. For each i in $\{1, \ldots, m\}$, define $n_i \coloneqq |A_i|$. Note that $N = \prod_{i=1}^{m} n_i$. Then the Cartesian code associated with this evaluation set is*

$$C_k(F, \mathcal{A}) = \{\ (f(\vec{\alpha}^{(1)}), f(\vec{\alpha}^{(2)}), \ldots, f(\vec{\alpha}^{(N)}))\ \mid\ f \in F[x_1, ..., x_m] \text{ with degree less than } k\ \}.$$

Given $v \in (F \setminus \{0\})^n$, the generalized Cartesian code is

$$C_k(F, \mathcal{A}, \vec{v}) = \{ \ (v_{\vec{\alpha}^{(1)}} f(\vec{\alpha}^{(1)}), \ldots, v_{\vec{\alpha}^{(N)}} f(\vec{\alpha}^{(N)})) \ \mid \ f \in F[x_1, ..., x_m] \text{ with } \deg(f) < k \ \}.$$

Remark 2.2 *$C_k(F, A)^\perp = C_{k'}(F, \mathcal{A}, \vec{v})$, where $k' = \sum_{i=1}^m (n_i - 1) - k + 1$ and $\vec{v} = (v_{\vec{\alpha}^{(1)}}, v_{\vec{\alpha}^{(2)}}, \ldots, v_{\vec{\alpha}^{(N)}})$ is a particular vector in $(F \setminus \{0\})^N$ dependent on $\mathcal{A}$ and k, given by the 2018 López, Manganiello, Matthews paper [2]. For our purposes, the exact vector is unimportant, we only care that $\vec{v} \in F \setminus \{0\}$.*

As always in coding theory, maximizing the dimension of the code is a high priority to be balanced with other interests. A code's dimension is a measure of the amount of information we can pack into a single codeword. We always want to maximize k, so as to make our code more efficient to store. Since we will be focusing on Cartesian codes, we will explicitly state the dimension of a Cartesian code here, as developed in previous work [2].

Theorem 2.1 (Dimension of Cartesian code) *Given the Cartesian code $C_k(F, \mathcal{A})$, where $\mathcal{A} = A_1 \times \cdots \times A_m$ with $n_j \coloneqq |A_j|$ for all $j \in \{1, \ldots, m\}$, the dimension of $C_k(F, \mathcal{A})$ follows.*

Case (i): $k - l > |\mathcal{A}|$

$$\dim_F(C_k(F, \mathcal{A})) = \sum_{j=1}^{m} (n_j - 1)$$

Case (ii): $k - 1 \leq |\mathcal{A}|$

$$\dim_F(C_k(F, \mathcal{A})) = \sum_{j=0}^{m} (-1)^j \sum_{1 \leq i_1 \leq \cdots \leq i_j \leq m} \binom{m + k - n_{i_1} - \cdots - n_{i_j}}{k - n_{i_1} - \cdots - n_{i_j}}$$

The important thing to notice is that the dimension of a Cartesian code is solely determined by the evaluation set $\mathcal{A}$ if $k - 1 > |\mathcal{A}|$. If $k - 1 \leq |\mathcal{A}|$ the dimension is increasing with m and k. For this paper, we will almost exclusively study cases with smaller k and larger evaluation sets, so we will focus on Case (ii). Generally, when developing schemes to recover erasures with Cartesian codes, we will consider the field F and the evaluation set $\mathcal{A} \in F^m$ to be given and fixed. Then we will intelligently choose k to work well with our lower bandwidth schemes. However, as happens so often in coding theory, there is an inherent trade-off between correcting capabilities and dimension. We will find that a smaller k generally allows us to create lower bandwidth schemes, but the above theorem indicates that a smaller k yields a lower dimensional code. Therefore, any implementation of our schemes needs to choose k to balance dimensional and bandwidth interests.

We now pivot to defining general notation to be used with finite fields.

Throughout this paper, when discussing codes over F, we will often be concerned with a subfield $B \leq F$, and how it relates to F. Any subset of F has a dimension over B, or $\dim_B$, which refers to the number of F elements required to span the set with B-linear combinations.

Lastly, we define the field trace, which will be utilized throughout our work.

Definition 2.3 (Field Trace) *Let F be a finite field of order q^t with a subfield $B \leq F$ of order q. Then the field trace is function from F to B such that for any $a \in F$,*

$$\mathrm{tr}_{F/B}(a) = \sum_{s=0}^{t-1} a^{q^s} = 1 + a^q + a^{q^2} + \cdots + a^{q^{t-1}} .$$

Remark 2.3 *Note that the $\mathrm{tr}_{F/B}$ is a B-linear function. It is known that any linear functional $L(x)$ from F to B is equal to $\mathrm{tr}_{F/B}(ax)$ for some $a \in F$ [1].*

With these definitions in place, we can now formally set up the research problem to be addressed in this paper.

3 Setup

Consider the Cartesian code $C_k(F, \mathcal{A})$. Note, throughout this paper, whenever a general object such as this is introduced, it is implicitly assumed that the parameters meet the requirements of the definition. In this case, it is assumed that F is a finite field with an evaluation set $\mathcal{A} \in F^m$ and k is a positive integer. Define $N \coloneqq |\mathcal{A}|$. Then, we can index $\mathcal{A} = \{\vec{\alpha}^{(1)}, \ldots, \vec{\alpha}^{(N)}\}$.

We can now construct a distributed storage system of N nodes that utilizes this Cartesian code $C_k(F, \mathcal{A})$. Assume we have a way to encode data files to codewords in $C_k(A)$. Given a data file to store, first encode the file to a codeword $(f(\vec{\alpha}^{(1)}), \ldots, f(\vec{\alpha}^{(N)})) \in C_k(F, A)$. Then, for each $\vec{\alpha} \in \mathcal{A}$, store $f(\vec{\alpha})$ at its own node, Node-$\vec{\alpha}$. This effectively stores our data $(f(\vec{\alpha}^{(1)}), \ldots, f(\vec{\alpha}^{(N)}))$ across N nodes, Node-$\vec{\alpha}^{(1)}$, Node-$\vec{\alpha}^{(2)}$, ..., Node-$\vec{\alpha}^{(N)}$. Meaning this setup describes a distributed storage system with N nodes, as desired. Now, onto the *exact repair problem.*

Suppose for some $\vec{\alpha}^* \in \mathcal{A}$, the data at Node-$\vec{\alpha}^*$, $f(\vec{\alpha}^*)$, is lost. Assume every other Node-$\vec{\alpha}$ where $\vec{\alpha} \neq \vec{\alpha}^*$ is available, so we have access to $f(\vec{\alpha})$, the data stored at Node-$\vec{\alpha}$ for all $\vec{\alpha} \in \mathcal{A} \setminus \vec{\alpha}^*$. Notice, since f is a polynomial in m variables over F with $\deg(f) < k$, by definition the evaluation of f on $\mathcal{A}$, $(f(\vec{\alpha}^{(1)}), f(\vec{\alpha}^{(2)}), ..., f(\vec{\alpha}^{(N)}))$, is a codeword in $C_k(F, \mathcal{A})$.

Then, recovering the data at Node-$\vec{\alpha}^*$ given the data at Node-$\vec{\alpha}$ for all $\vec{\alpha} \in \mathcal{A} \setminus \vec{\alpha}^*$ is equivalent to recovering $f(\vec{\alpha}^*)$ given $f(\vec{\alpha})$ for all $\vec{\alpha} \in \mathcal{A} \setminus \vec{\alpha}^*$. In other words, recovering the data at Node-$\vec{\alpha}^*$ is equivalent to fixing a single erasure at coordinate $\vec{\alpha}^*$ from the codeword $(f(\vec{\alpha}^{(1)}), f(\vec{\alpha}^{(2)}), ..., f(\vec{\alpha}^{(N)})) \in C_k(F, \mathcal{A})$. The remainder of this paper is dedicated to developing *exact repair schemes* to fix this erasure with the least amount of inter-nodal communication [3]. We will formalize this idea with the following definitions.

Definition 3.1 (Exact Repair Scheme) *Let $C_k(F, \mathcal{A})$ be a Cartesian code with $|\mathcal{A}| = N$ and index $\mathcal{A} = \{\vec{\alpha}^{(1)}, \ldots, \vec{\alpha}^{(N)}\}$. Construct a distributed storage system of N nodes, Node-$\vec{\alpha}^{(1)}$, ..., Node-$\vec{\alpha}^{(N)}$, that stores a codeword of $C_k(F, \mathcal{A})$ as follows. Given a codeword $c = (f(\vec{\alpha}^{(1)}), \ldots, f(\vec{\alpha}^{(N)})) \in C_k(F, A)$, store c on the distributed storage system by storing $f(\vec{\alpha})$ on Node-$\vec{\alpha}$.*
An algorithm R is a exact repair scheme for $C_k(F, \mathcal{A})$ with respect to $\vec{\alpha}^*$ if it can be broken into the following two steps.
Input a codeword $(f(\vec{\alpha}^{(1)}), \ldots, f(\vec{\alpha}^{(N)})) \in C_k(F, \mathcal{A})$ stored on the distributed storage system.

1. *For all $\vec{\alpha} \in \mathcal{A} \setminus \vec{\alpha}^*$, at Node-$\vec{\alpha}$ calculate some information $U_{\vec{\alpha}}(f(\vec{\alpha}))$ from $f(\vec{\alpha})$. Send $U_{\vec{\alpha}}(f(\vec{\alpha}))$ to Node-$\vec{\alpha}^*$.*

2. At Node-$\vec{\alpha}^*$, use $U_{\vec{\alpha}}$ for all $\vec{\alpha} \in \mathcal{A} \setminus \vec{\alpha}^*$ to calculate $f(\vec{\alpha}^*)$.

Output $f(\vec{\alpha}^*)$.

Notice, by the above definition, any exact repair scheme for $C_k(F, \mathcal{A})$ with respect to $\vec{\alpha}^*$ is an algorithm that fixes an erasure at coordinate $\vec{\alpha}^*$ for any codeword in $C_k(F, \mathcal{A})$. The above definition can easily be generalized to work with any code, but since we will focus on Cartesian codes, we decided to define exact repair schemes accordingly.

As stated above, our goal is to recover our missing data using the least amount of information from each node as possible. Specifically, we want to minimize the amount of information sent between nodes in an exact repair scheme. We call the amount of information sent between nodes the repair bit-bandwidth, which we formally define below.

Definition 3.2 (Repair Bit-Bandwidth) *Let $C_k(F, \mathcal{A})$ be a Cartesian code with $|\mathcal{A}| = N$ and index $\mathcal{A} = \{\vec{\alpha}^{(1)}, \ldots, \vec{\alpha}^{(N)}\}$. Let R be exact repair scheme for $C_k(F, \mathcal{A})$ with respect to $\vec{\alpha}^*$. Construct a distributed storage system with nodes Node-$\vec{\alpha}^{(1)}$, ..., Node-$\vec{\alpha}^{(N)}$ as in the above Definition 3.1. Given any $(f(\vec{\alpha}^{(1)}), \ldots, f(\vec{\alpha}^{(N)})) \in C_k(F, \mathcal{A})$ stored on the distributed storage system, by definition R sends some information $U_{\vec{\alpha}}(f(\vec{\alpha}))$ to Node-$\vec{\alpha}^*$ to Node-$\vec{\alpha}^*$ for every $\vec{\alpha} \in \mathcal{A} \setminus \vec{\alpha}^*$*

The repair bandwidth of R is the maximum amount of bits sent to Node-$\vec{\alpha}^*$ in algorithm R for any codeword $(f(\vec{\alpha}^{(1)}), \ldots, f(\vec{\alpha}^{(N)})) \in C_k(F, \mathcal{A})$.

Note, in the literature the term "repair bandwidth" usually refers to the number of certain field elements sent between Nodes in an exact repair scheme. However, we choose to look at the number of bits communicated instead, because we want to consider more general algorithms that might not communicate elements of the same field to Node-$\vec{\alpha}^*$. We are interested in finding schemes that have minimal repair bit-bandwidths, as these schemes will require less bandwidth to execute in an actual implementation.

With these definitions in mind, the formal goal of this paper is to develop exact repair schemes for $C_k(F, \mathcal{A})$ with respect to $\vec{\alpha}^*$ that minimize the repair bit-bandwidth and maximize k. We work to develop these schemes in the coming sections.

4 Reed-Solomon Codes and Repair

We will start by looking at exact repair schemes for Reed-Solomon codes. Recall, Reed-Solomon codes are a special class of a Cartesian codes that use univariate polynomials. The traditional scheme for recovering erasures with Reed-Solomon codes involves interpolating these polynomials. We outline this approach below.

4.1 Direct Approach

Consider a codeword $(f(\vec{\alpha}^{(1)}), \ldots, f(\vec{\alpha}^{(N)}))$ in the Cartesian code $C_k(F, \mathcal{A})$. Let m be the integer such that $\mathcal{A} \in F^m$. Note, by the definition of Cartesian codes, $f \in F[x_1, \ldots, x_m]$ with $\deg(f) < k$. Suppose the symbol at coordinate $\vec{\alpha}^* \in \mathcal{A}$, in this case $f(\vec{\alpha}^*)$, is lost. The most straight forward way of recovering $f(\vec{\alpha}^*)$ is to simply interpolate the points $(\vec{\alpha}, f(\vec{\alpha}))$ to recreate the polynomial f. Then, we can evaluate f at $\vec{\alpha}^*$ and we have recovered the missing data. If f is a univariate polynomial with $\deg(f) < k$, we know we can recreate f with at most k points.

However, interpolating multivariate polynomials is *not* that straight forward. Currently, there is no known method of interpolating a general multivariate polynomial given a general evaluation set [4]. We will avoid this problem by restricting our Cartesian codes to just the univariate case, Reed-Solomon codes. Note, as the elements of $\mathcal{A}$ are no longer ordered-tuples, we will forgo the vector arrow over these elements for the rest of this section. For example $\vec{\alpha}^*$ will be referred to as α^*, $\vec{\alpha}^{(j)}$ will be referred to as $\alpha^{(j)}$, etc. In this context, the problem of recovering the lost data $f(\alpha^*)$ is equivalent to fixing a single erasure at coordinate α^* in the codeword $(f(\alpha^{(1)}), \ldots f(\alpha^{(N)}))$. Interpolating polynomials in one variable is a well-studied problem, and it has been used to develop the classic scheme below [1].

Direct Scheme[$\mathbf{RS}_k(F, \mathcal{A}), \alpha^*$]
Let F be a finite field with an evaluation set $\mathcal{A} \subseteq F$ and let k be a positive integer. Assume $|\mathcal{A}| > k$. Construct a distributed storage system using $RS_k(F, \mathcal{A})$ as in Definition 3.1. Pick $\alpha^* \in \mathcal{A}$.

Input a codeword $((f(\vec{\alpha}^{(1)}), \ldots, f(\vec{\alpha}^{(N)})) \in \mathrm{RS}_k(F, \mathcal{A})$ stored on the distribution storage system.

Randomly choose a subset $\tilde{\mathcal{A}} \subseteq \mathcal{A} \setminus \alpha^*$ of size k.

1. For every $\alpha \in \tilde{\mathcal{A}} \setminus \alpha^*$ send $f(\alpha)$ stored at Node-$\vec{\alpha}$ to Node-α^*.
2. At Node-α^*, use the sent values of $f(\alpha)$ for all $\alpha \in \tilde{\mathcal{A}}$ to recreate the polynomial f. Then calculate $f(\alpha^*)$.

Output $f(\alpha^*)$

Remark 4.1 *For the rest of this paper,* ***Direct Scheme*** $[\boldsymbol{RS_k(F, \mathcal{A}), \alpha^*}]$ *refers to the direct scheme algorithm with those parameters. If previously undefined it is assumed that the parameters are general objects of that form that meet the conditions of the algorithm.*

By the reasoning above, **Direct Scheme** $[\mathbf{RS_k(F, \mathcal{A}), \alpha^*}]$ will fix an erasure at α^* in a code word $(f(\vec{\alpha}^{(1)}), \ldots, f(\vec{\alpha}^{(m)})) \in \mathrm{RS}_k(\mathcal{A})$. Further, notice the algorithm can be split into the two parts discussed in Definition 3.1. Thus, **Direct Scheme** $[\mathbf{RS_k(F, \mathcal{A}), \alpha^*}]$ is an exact repair scheme for $\mathrm{RS}_k(F, \mathcal{A})$ with respect to α^*. Even though this is a very elegant algorithm, the **Direct Scheme** has inherent issues. Mainly there are bandwidth concerns, as we will discuss in detail below.

Theorem 4.1 (Repair Bit-Bandwidth for the Direct Scheme) *The repair bit-bandwidth of* ***Direct Scheme*** $[\boldsymbol{RS_k(F, \mathcal{A}), \alpha^*}]$ *is* $k \log_2(|F|)$.

Proof. Notice, for any f, step 1 of **Direct Scheme** $[\mathbf{RS_k(F, \mathcal{A}), \alpha^*}]$ sends $f(\alpha)$ from Node-α to Node-α^* for every α in the k sized subset $\tilde{\mathcal{A}}$. Meaning, overall it sends k elements of F. Thus, since the amount of information contained in an element of F is $\log_2(|F|)$, the repair bandwidth of this scheme is $k \log_2(|F|)$, as desired.

□

We should note, this bandwidth seems ludicrously high. The amount of information contained in $f(\alpha^*)$, a single element of F, is $\log_2(|F|)$. Compare that to the bit-bandwidth of the scheme, $k \log_2(|F|)$, and we see that this scheme transmits $k \log_2(|F|)$ bits of data to recover just $\log_2(|F|)$ bits of data. This means we could end up sending k times the bits of data we are recovering, a huge inefficiency. Especially as k gets large, the **Direct Scheme** becomes unusuable from a bandwidth perspective. Further, in order to meet the conditions of the **Direct Scheme** $[\mathbf{RS_k(F, \mathcal{A}), \alpha^*}]$, the dimension of our Reed-Solomon code $\mathrm{RS}_k(F, \mathcal{A})$ is limited by the constraint $k < |\mathcal{A}|$. Although for very large $|\mathcal{A}|$ this is less of a concern.

The high bandwidth of implementing the natural exact repair scheme for a Reed-Solomon code is one reason why Reed-Solomon codes were traditionally overlooked as a candidate for the base code of a distributed storage system. However, for any code there are many possible repair schemes. We can construct a scheme for a Reed-Solomon code with a greatly improved bandwidth by using a *linear exact repair scheme*.

4.2 Linear Exact repair for Reed-Solomon Codes

The rest of this section will be dedicated to discussing previous research in using linear exact repair schemes to recover erasures for Reed-Solomon codes. Recall, as we are in the Reed-Solomon case, our goal is to recover a lost node in a distributed storage system based on the Reed-Solomon code $\mathrm{RS}(F, \mathcal{A})$. In this context, the problem of recovering the lost data $f(\alpha^*)$ is equivalent to fixing a single erasure at coordinate α^* in the codeword $(f(\alpha^{(1)}), \dots f(\alpha^{(N)}))$. We will draw most of our inspiration from the aforementioned influential Guruswarmi and Wooters papers [1].

The first pivotal idea one encounters when studying linear exact repair codes is in the following observation [1].

Observation 4.1 (Property of Dual Bases) *Let F be a finite field with $B \leq F$ such that $[F : B] = t$. Further let $Z = \{z_1, \dots, z_t\}$ be a basis for F over B with an associated dual basis $Y = \{\lambda_1, \dots, \lambda_t\}$. Then, for any $c \in F$ it is known that*

$$c = \sum_{i \in \{1,\dots,t\}} \mathrm{tr}_{F/B}(z_i c)\lambda_i$$

Now to apply the above observation to our missing data, $f(\alpha^*)$, let us first choose a basis Z for F over B. Note $[F : B] = t$, so we can index the Z elements $Z = \{z_1, \dots, z_t\}$. Next, let Y be the dual basis associated with Z and index the Y elements $Y = \{\lambda_1, \dots, \lambda_t\}$, where λ_i is the associated dual basis element to z_i for all $i \in \{1, \dots, t\}$. Then, by the observation above,

$$f(\alpha^*) = \sum_{i \in \{1,\dots,t\}} \mathrm{tr}_{F/B}(z_i f(\alpha^*)) \ .$$

This may at first feel like an observation of little value. However, as shown in the Guruswarmi and Wooters paper [1], there is a very nice and subtle way to compute $\mathrm{tr}_{F/B}(z_i f(\alpha^*))$ for each $i \in \{1, \dots, t\}$.

The process of calculating these traces leads us to our first linear exact repair scheme. The execution of every linear exact repair scheme we discuss in this paper starts with precomputing some very particular coefficients. We discuss how to find these coefficients and if they even exist following the algorithm. For now, define the following to be repair coefficients. Note, we will define repair coefficients for a general Cartesian code, but in this section we will only use them with Reed-Solomon codes.

Definition 4.1 (Repair Coefficients) *Consider the Cartesian code $C_k(F, \mathcal{A})$. Let m be the positive integer such that $\mathcal{A} \subseteq F^m$.*

*The elements $\mu_{i,\vec{\alpha}} \in F$ defined for all $i \in \{1, \ldots, t\}$ and $\vec{\alpha} \in \mathcal{A} \setminus \vec{\alpha}^*$ are called **repair coefficients** of $C_k(F, A)$ over B with respect to $\vec{\alpha}^* \in \mathcal{A}$ if they satisfy the following property.*

Given any $f \in F[x_1, \ldots, x_m]$ with $\deg(f) < k$, for all $i \in \{1, \ldots, t\}$

$$z_i f(\vec{\alpha}^*) = \sum_{\vec{\alpha} \in \mathcal{A} \setminus \vec{\alpha}^*} \mu_{i,\vec{\alpha}} f(\vec{\alpha}) \,, \tag{4.1}$$

where the set $Z = \{z_1, \ldots, z_t\}$ forms a basis for F over B. We will refer to Z as the basis associated with the repair coefficients.

*Given a set of repair coefficients, define the **$\vec{\alpha}$-dimension** of our coefficients to be $d_{\vec{\alpha}} \coloneqq \dim_B\{\mu_{i,\vec{\alpha}} : i \in \{1, \ldots, t\}\}$. Note $\dim_B$ refers to the minimum number of F elements needed to span $\{\mu_{i,\vec{\alpha}} : i \in \{1, \ldots, t\}\}$ as a vector space over B.*

*The **total** dimension of our repair coefficients is $\sum_{\vec{\alpha} \in \mathcal{A} \setminus \vec{\alpha}^*} d_{\vec{\alpha}}$.*

*Then, for each $i \in \{1, \ldots, t\}$ and $\alpha \in \mathcal{A} \setminus \alpha^*$ there exist d_α **spanning elements** $\omega_{\alpha,1}, \ldots, \omega_{\alpha,d_\alpha} \in F$ with associated **span constants** $\beta_{i,\alpha,1}, \ldots, \beta_{i,\alpha,d_\alpha} \in B$ such that*

$$\mu_{i,\alpha} = \beta_{i,\alpha,1}\omega_{\alpha,1} + \cdots + \beta_{i,\alpha,d_\alpha}\omega_{\alpha,d_\alpha}. \tag{4.2}$$

With repair coefficients defined, we have everything we need to go through a linear exact repair algorithm. The following algorithm is the algorithm presented in the 2017 Guruswarmi and Wooters paper [1], and will be stated without proof. However, in section 5 we present and prove a more general algorithm that admits this algorithm as a special case.

Algorithm : **LERS**$[\mathbf{RS}_k(\boldsymbol{F}, \boldsymbol{\mathcal{A}}), \boldsymbol{B}, \boldsymbol{M}, \boldsymbol{\alpha^*}]$ A Linear exact repair scheme to recover a single $f(\alpha^*)$

Parameters:

- Reed-Solomon code $\mathrm{RS}_k(F, \mathcal{A})$ defined by the finite field F with an evaluation set $\mathcal{A} \in F$ and a positive integer k.
- A subfield $B \leq F$ such that $t \coloneqq [F : B]$.
- The element $\alpha^* \in \mathcal{A}$
- M, a set of repair coefficients $\mu_{i,\alpha}$ for $\mathrm{RS}_k(F, \mathcal{A})$ with respect to α^* defined for all $i \in \{1, \ldots, t\}$ and $\alpha \in \mathcal{A} \setminus \alpha^*$
  - The associated basis, $Z = \{z_1, \ldots, z_t\}$
  - The associated spanning elements $\omega_{\alpha,1}, \ldots, \omega_{\alpha,d_\alpha}$ for all $\alpha \in \mathcal{A} \setminus \alpha^* \in F$ stored at Node-α
  - The associated span constants $\beta_{i,\alpha,1}, \ldots, \beta_{i,\alpha,d_\alpha} \in B$ for all $i \in \{1, \ldots, t\}$ and $\alpha \in \mathcal{A} \setminus \alpha^*$ stored at Node-α^*

Input: A codeword $(f(\alpha^{(1)}), \ldots, f(\alpha^{(N)}) \in \mathrm{RS}_k(F, \mathcal{A})$ stored on a distributive storage system with $f(\alpha)$ successively stored at Node-α for all $\alpha \in \mathcal{A} \setminus \alpha^*$.

Output: $f(\alpha^*)$

1: **for all** $\alpha \in \mathcal{A} \setminus \alpha^*$ **do**
2: At Node-α compute $u_{\alpha,j} \coloneqq \mathrm{tr}_{F/B}(\omega_{\alpha,j} f(\alpha))$ for each $j \in \{1, \ldots, d_\alpha\}$ and send to Node-α^*.
Note $u_{\alpha,j} \in B$ for all $j \in \{1, \ldots, d_\alpha\}$.
3: **end for**

4: **for all** $i \in \{1, \ldots, t\}$ **do**
5: At Node-α^*, use the newly collected traces above to calculate

$$s_i \coloneqq \sum_{\alpha \in \mathcal{A} \setminus \alpha^*} \beta_{i,\alpha,1} u_{\alpha,1} + \cdots + \beta_{i,\alpha,d_\alpha} u_{\alpha,d\alpha} \, .$$

Note $s_i = \mathrm{tr}_{F/B}(z_i f(\alpha))$.
6: **end for**
7: Let $Y = \{\lambda_1, \ldots, \lambda_t\}$ be the dual basis associated with the basis Z. At Node-α^* compute

$$f(\alpha^*) = \sum_{i \in \{1, \ldots, t\}} s_i \lambda_i$$

Remark 4.2 $\boldsymbol{LERS[RS_k(F,\mathcal{A}),B,M,\alpha^*]}$ *exactly recovers* $f(\alpha^*)$ *for any* $f \in F[x_1,...,x_m]$ *using only information gathered from nodes* $\{\text{Node-}\alpha \mid \alpha \in \mathcal{A} \setminus \alpha^*\}$. *Thus* $\boldsymbol{LERS[RS_k(F,\mathcal{A}),B,M,\alpha^*]}$ *is an exact repair scheme for* $RS_k(F,\mathcal{A})$ *with respect to* α^*.

This remark was proven by Guruswarmi and Wooters in 2017 [1].

Note the above $\mathbf{LERS[RS_k(F,\mathcal{A}),B,M,\alpha^*]}$ is only defined when $\mathcal{A} \in F^m$ for $m = 1$. In the next sections we will define a more general algorithm for cases where m is greater than 1.

Taking this algorithm as given, we have just shown a method for recovering $f(\alpha^*)$ from the data in the other nodes. We can now begin analyzing the bit-bandwidth efficiency of $\mathbf{LERS[RS_k(F,\mathcal{A}),B,M,\alpha^*]}$. Notice, the only elements sent between nodes by this algorithm are the elements $u_{\alpha,j}$. The Guruswarmi and Wooters paper notes there are

$$\sum_{\alpha \in \mathcal{A} \setminus \alpha^*} \dim_B(\{\mu_{i,\alpha} \mid i \in \{1,\ldots,t\}\}) = \sum_{\alpha \in \mathcal{A} \setminus \alpha^*} d_\alpha$$

elements of the form $u_{\alpha,j}$ that need to be sent. It is also noted that $u_{\alpha,j} \in B$ for all $\alpha \in \mathcal{A}$ and $j \in \{1,\ldots,d_\alpha\}$. Note, the information contained in a single B element is $\log_2(|B|)$. Therefore, since this algorithm requires $\sum_{\alpha \in \mathcal{A} \setminus \alpha^*} d_\alpha$ elements of B to be sent between nodes, the bit-bandwidth of this scheme is $\log_2(|B|) \sum_{\alpha \in \mathcal{A} \setminus \alpha^*} d_\alpha$.

Further, since the coefficients $\mu_{i,\alpha} \in M$ are said to be independent of our evaluation polynomial f, we notice that the bit-bandwidth of our scheme is $\log_2(|B|) \sum_{\alpha \in \mathcal{A} \setminus \alpha^*} d_\alpha$ no matter the f.

Therefore, the repair bit-bandwidth of $\mathbf{LERS[RS_k(F,\mathcal{A}),B,M,\alpha^*]}$ with respect to α^* is $\log_2(|B|) \sum_{\alpha \in \mathcal{A} \setminus \alpha^*} d_\alpha = D \log_2(|B|)$, where D is the total dimension of our repair coefficients.

Theorem 4.2 *If there exists a set of repair coefficients for* $RS_k(F,\mathcal{A})$ *over* $B \leq F$ *with respect to* $\alpha^* \in \mathcal{A}$ *with total dimension* D, *then there is an exact repair scheme for* $RS_k(F,\mathcal{A})$ *with respect to* α^* *with repair bit-bandwidth* $D \log_2(|B|)$, *where* D *is the total dimension of our repair coefficients.*

Proof. Let M be a set of repair coefficients for $\mathrm{RS}_k(F,\mathcal{A})$ over $B \leq F$ with respect to $\alpha^* \in \mathcal{A}$ with total dimension D. Then $\mathbf{LERS[RS_k(F,\mathcal{A}),B,M,\alpha^*]}$ is an exact repair scheme for $\mathrm{RS}_k(F,\mathcal{A})$ with respect to α^* with repair bit-bandwidth $D \log_2(|B|)$.

□

Comparing to the repair bit-bandwidth of the direct scheme from Theorem 4.1, we see the following. The repair bit-bandwidth for **Direct Scheme**$[\mathbf{RS}_{\boldsymbol{k}}(\boldsymbol{F},\boldsymbol{\mathcal{A}}),\boldsymbol{\alpha}^*]$ is $k\log_2(|F|) = k\log_2(|B|^t) = kt\log_2(|B|)$. Therefore, $\mathbf{LERS}[\mathbf{RS}_{\boldsymbol{k}}(\boldsymbol{F},\boldsymbol{\mathcal{A}}),\boldsymbol{B},\boldsymbol{M},\boldsymbol{\alpha}^*]$ having a smaller repair bit-bandwidth than **Direct Scheme**$[\mathbf{RS}_{\boldsymbol{k}}(\boldsymbol{F},\boldsymbol{\mathcal{A}}),\boldsymbol{\alpha}^*]$ is equivalent to $D\log_2(|B|) < kt\log_2(|B|)$. Thus, if $D < kt \implies D\log_2(|B|) < kt\log_2(|B|)$ and therefore $\mathbf{LERS}[\mathbf{RS}_{\boldsymbol{k}}(\boldsymbol{F},\boldsymbol{\mathcal{A}}),\boldsymbol{B},\boldsymbol{M},\boldsymbol{\alpha}^*]$ using repair coefficients $\mu_{i,\alpha}$ has a smaller repair bit-bandwidth than the **Direct Scheme**$[\mathbf{RS}_{\boldsymbol{k}}(\boldsymbol{F},\boldsymbol{\mathcal{A}}),\boldsymbol{\alpha}^*]$.

Remark 4.3 $\boldsymbol{LERS}[\boldsymbol{RS}_{\boldsymbol{k}}(\boldsymbol{F},\boldsymbol{\mathcal{A}}),\boldsymbol{B},\boldsymbol{M},\boldsymbol{\alpha}^*]$ *will have a smaller repair bit-bandwidth than* ***Direct Scheme***$[\boldsymbol{RS}_{\boldsymbol{k}}(\boldsymbol{F},\boldsymbol{\mathcal{A}}),\boldsymbol{\alpha}^*]$ *if and only if*

$$\sum_{\alpha\in\mathcal{A}\setminus\alpha^*} d_\alpha < kt \ .$$

In terms of our distributed storage system, this means if $\sum_{\alpha\in\mathcal{A}\setminus\alpha^*} d_\alpha < kt$, then recovering the failed Node-α^* with $\mathbf{LERS}[\mathbf{RS}_{\boldsymbol{k}}(\boldsymbol{F},\boldsymbol{\mathcal{A}}),\boldsymbol{B},\boldsymbol{M},\boldsymbol{\alpha}^*]$ requires less inter-nodal communication than recovering the failed Node-α^* with **Direct Scheme**$[\mathbf{RS}_{\boldsymbol{k}}(\boldsymbol{F},\boldsymbol{\mathcal{A}}),\boldsymbol{\alpha}^*]$. It now seems that determining the bandwidth of $\mathbf{LERS}[\mathbf{RS}_{\boldsymbol{k}}(\boldsymbol{F},\boldsymbol{\mathcal{A}}),\boldsymbol{B},\boldsymbol{M},\boldsymbol{\alpha}^*]$ requires more insight into the mysterious $\mu_{1,\alpha} \in M$ coefficients described above. We will now discuss how to find these coefficients.

4.2.1 Constructing μ-Coefficients

Now we will finally construct the $\mu_{i,\alpha}$ coefficients used in $\mathbf{LERS}[\mathbf{RS}_k(\boldsymbol{F},\boldsymbol{\mathcal{A}}),\boldsymbol{B},\boldsymbol{M},\boldsymbol{\alpha}^*]$. Again, we find foundational ideas in the Guruswarmi and Wooters paper [1]. Recall, in the univariate case our code is a Reed-Solomon code $\mathrm{RS}_k(F,\mathcal{A})$. For any $f \in F[x]$, we have $(f(\alpha^{(1)}),\ldots,f(\alpha^{(N)}))$ is a codeword in $\mathrm{RS}_k(F,\mathcal{A})$. The key idea for finding these μ-coefficients is to look at codewords in the dual code. By Definition 2.1, the dual to this Reed-Solomon code is the generalized Reed Solomon code $\mathrm{GRS}_{N-k}(F,\mathcal{A},\vec{v})$ where $N \coloneqq |\mathcal{A}|$ and $\vec{v} = (v_{\alpha^{(1)}},\ldots,v_{\alpha^{(N)}})$ is a particular element in $(F\setminus\{0\})^N$. The exact $\vec{v}$ is unimportant for our purposes, we only care that it is in $(F\setminus\{0\})^N$. Notice, if p is a polynomial in $F[x]$ with $\deg(p) < N-k$ then $(v_{\alpha^{(1)}}p(\alpha^{(1)}),\ldots,v_{\alpha^{(N)}}p(\alpha^{(N)})) \in \mathrm{GRS}_{N-k}(F,\mathcal{A})$. Thus, by definition of a dual code,

$$0 = (v_{\alpha^{(1)}}p(\alpha^{(1)}),\ldots,v_{\alpha^{(N)}}p(\alpha^{(N)}))\cdot(f(\alpha^{(1)}),\ldots,f(\alpha^{(N)})) = \sum_{\alpha\in\mathcal{A}} v_\alpha p(\alpha)f(\alpha)\ .$$

Then, subtracting $v_{\alpha^*}p(\alpha^*)f(\alpha^*)$ from both sides we achieve

$$-v_{\alpha^*}p(\alpha^*)f(\alpha^*) = \sum_{\alpha\in\mathcal{A}\setminus\alpha^*} v_\alpha p(\alpha)f(\alpha)\ .$$

This is a key insight found in the Guruswarmi and Wooters paper [1] that led to the following theorem. Before we define the theorem, let us define this useful definition for *repair polynomials*. Again, we will define these polynomials for general Cartesian codes, but in this section we will only use them with Reed-Solomon codes.

Definition 4.2 (Repair Polynomials) *Consider a Cartesian code $C_k(F,\mathcal{A})$ with evaluation set $\mathcal{A} = A_1\times\cdots\times A_m \subseteq F^m$ for some positive integer m and a subfield $B \le F$ with $t \coloneqq [F:B]$. Define $n_j \coloneqq |A_j|$ for all $j\in\{1,\ldots,m\}$. Then we have the following.*

The polynomials $p_i \in F[x_1,\ldots,x_m]$ defined for all $i\in\{1,\ldots,t\}$ are called a set of ***repair polynomials*** *for $C_k(F,\mathcal{A})$ over B with respect to $\vec{\alpha}^*$ if the following properties are satisfied.*

(i) $\deg(p_i) < k' = \sum_{j=1}^m (n_j-1) - k + 1$ *for all* $i\in\{1,\ldots,t\}$.

(ii) The set $Z = \{z_1,\ldots,z_t\}$, where $z_i \coloneqq p_i(\vec{\alpha}^*)$ for all i, is a basis for F over B.

Define the ***$\vec{\alpha}$-dimension*** *of these repair polynomials to be*
$\delta_{\vec{\alpha}} = \dim_B\{p_i(\vec{\alpha}) \mid i\in\{1,\ldots,t\}\}$.

The ***total dimension*** *of this set of repair polynomials is*

$$\sum_{\vec{\alpha}\in\mathcal{A}\setminus\vec{\alpha}^*} \delta_{\vec{\alpha}}\ .$$

Now to the theorem.

Theorem 4.3 (Coefficient to Dual Polynomial Equivalence) *Consider a Cartesian code $C_k(F, \mathcal{A})$ with an evaluation set $\mathcal{A} = A_1 \times \cdots \times A_m \subseteq F^m$ for some positive integer m and a subfield $B \leq F$ with $t \coloneqq [F : B]$. Define $n_j \coloneqq |A_j|$ for all $j \in \{1, \ldots, m\}$. Then the following are equivalent.*

(1) There is a set of repair coefficients for $C_k(F, A)$ over B with respect to $\vec{\alpha}^*$ with total dimension D.

(2) There is a set of repair polynomials for $C_k(F, \mathcal{A})$ over B with respect to $\vec{\alpha}^*$ with total dimension D.

The Guruswarmi and Wooters paper proved Theorem 4.3 for the case when $m = 1$, the Reed-Solomon case [1, Observation 8]. We will prove the theorem for all $m \geq 1$ in the next section. For now, we will continue on in the case where $m = 1$.

With this new theorem, we can immediately show a relationship between the existence of repair polynomials and the repair bit-bandwidth of $\mathbf{LERS}[\mathbf{RS_k}(\boldsymbol{F}, \boldsymbol{\mathcal{A}}), \boldsymbol{B}, \boldsymbol{M}, \boldsymbol{\alpha^*}]$.

Corollary 4.1 (Repair Scheme from Repair Polynomials) *Consider a Reed-Solomon code $RS_k(F, \mathcal{A})$, a subfield $B \leq F$, and an element $\alpha^* \in \mathcal{A}$.*

If there exists a set of repair polynomials for $RS_k(F, \mathcal{A})$ over B with respect to α^* with total dimension D, then there exists an exact repair scheme for $RS_k(F, \mathcal{A})$ with respect to α^* with repair bit-bandwidth $D \log_2(|B|)$.

Proof. Let P be a set of repair polynomials for $\mathrm{RS}_k(F, \mathcal{A})$ over B with respect to α^*. Then, by Remark 4.3 there exists a set of repair polynomials M for $\mathrm{RS}_k(F, \mathcal{A})$ over B with respect to α^*. Then, by Theorem 4.2 there is a repair scheme $\mathbf{LERS}[\mathbf{RS_k}(\boldsymbol{F}, \boldsymbol{\mathcal{A}}), \boldsymbol{B}, \boldsymbol{M}, \boldsymbol{\alpha^*}]$, for $RS_k(F, \mathcal{A})$ with respect to α^* with repair bit-bandwidth $D \log_2(|B|)$, as desired.

□

Remark 4.4 *For any set of repair polynomials, P we will refer to $\boldsymbol{LERS}[\boldsymbol{RS_k}(\boldsymbol{F}, \boldsymbol{\mathcal{A}}), \boldsymbol{B}, \boldsymbol{P}, \boldsymbol{\alpha^*}]$ to mean the algorithm $\boldsymbol{LERS}[\boldsymbol{RS_k}(\boldsymbol{F}, \boldsymbol{\mathcal{A}}), \boldsymbol{B}, \boldsymbol{M}, \boldsymbol{\alpha^*}]$ where M are the repair coefficients constructed from P.*

Now, let us review some particular repair polynomials discovered in the Guruswarmi and Wooters paper [1].

Observation 4.2 (Low Bandwidth Repair Polynomials) *Given a Reed-Solomon code $RS_k(F, \mathcal{A})$, a subfield $B \leq F$, and an element $\alpha^* \in \mathcal{A}$ such that $k \leq N - \frac{|F|}{|B|}$, then there are repair polynomials, P, for $RS_k(F, \mathcal{A})$ over B with respect to α^* with total dimension $N-1$. Additionally $\boldsymbol{LERS[RS_k(F, \mathcal{A}), B, P, \alpha^*]}$ is an exact repair scheme for $RS_k(F, \mathcal{A})$ with respect to α^* with repair bit-bandwidth $(N-1)\log_2(|B|)$.*

Proof. A proof of this observation was one of the main results of the Guruswarmi and Wooters paper [1]. They achieved this result by constructing the following polynomials. Let $Z = \{z_1, \ldots, z_t\}$ be any basis for F over B. Then, for each i, let

$$p_i(x) = \frac{\mathrm{tr}_{F/B}(z_i(x - \alpha^*))}{x - \alpha^*} \ .$$

It was shown in their paper that these polynomials satisfy the conditions to be repair polynomials for $\mathrm{RS}_k(F, \mathcal{A})$ over B with respect to $\vec{\alpha}^*$ with total dimension $N-1$. I will briefly present their reasoning here.

Firstly, we will show $\deg(p_i(x)) < N - k$ for all $i \in \{1, \ldots, t\}$. Note for any i,

$$p_i(x) = \frac{\mathrm{tr}_{F/B}(z_i(x-\alpha^*))}{x-\alpha^*} = \frac{\sum_{s=0}^{t-1} z_i^{q^s}(x-\alpha^*)^{q^s}}{x-\alpha^*} = \sum_{s=0}^{t-1} z_i^{q^s}(x-\alpha^*)^{q^s-1} = z_i + \sum_{s=1}^{t-1} z_i^{q^s}(x-\alpha^*)^{q^s-1} \ .$$

Then, clearly $\deg(p_i(x)) = q^{t-1} - 1$ for all i. Now $q^{t-1} - 1 = \frac{q^t}{q} - 1 = \frac{|F|}{|B|} - 1 < \frac{|F|}{|B|}$. Note we are assuming $k \leq N - \frac{|F|}{|B|}$, so $\frac{|F|}{|B|} \leq N - k$. Hence, $\deg(p_i(x)) < \frac{|F|}{|B|} \leq N - k$, so $\deg(p_i(x)) < N - k$, as desired.

Now to show $\{p_i(\alpha^*) | i \in \{1, \ldots, t\}\}$ is a basis for F over B. By the above expansion,

$$p_i(\alpha^*) = z_i + \sum_{s=1}^{t-1} z_i^{q^s}(\alpha^* - \alpha^*)^{q^s-1} = z_i \ .$$

Thus, $\{p_i(\alpha^*) \mid i \in \{1, \ldots, t\}\} = \{z_i \mid i \in \{1, \ldots, t\}\} = Z$, a basis for F over B. Therefore, by Definition 4.2, the p_i's are repair polynomials for $\mathrm{RS}_k(F, \mathcal{A})$ over B with respect to α^*

Finally, let us show the total dimension of these polynomials is $N-1$. Notice, by definition $\mathrm{tr}_{F/K}$ is a function that sends elements in F to B. Thus, fixing an $\alpha \in \mathcal{A} \setminus \alpha^*$, we see that $\{p_i(\alpha) \mid i \in \{1, \ldots, t\}\} = \{\frac{\mathrm{tr}_{F/B}(z_i(\alpha-\alpha^*))}{\alpha-\alpha^*} \mid \ i \in \{1, \ldots, t\}\} = \{\frac{b_i}{\alpha-\alpha^*} \mid \text{for some } b_i \in B \text{ and } i \in \{1, \ldots, t\}\}$ which is spanned by a B-linear combination of $\{\frac{1}{\alpha-\alpha^*}\}$. Hence, $d_\alpha \coloneqq \dim_B\{p_i(\alpha) \mid i \in \{1, \ldots, t\}\} = 1$. Since the α fixed was arbitrary, $d_\alpha = 1$ for all $\alpha \in \mathcal{A} \setminus \alpha^*$. Therefore, the total dimension of our repair polynomials is

$\sum_{\alpha \in \mathcal{A} \setminus \alpha^*} d_\alpha = \sum_{\alpha \in \mathcal{A} \setminus \alpha^*} 1 = |\mathcal{A}| - 1 = N - 1.$

□

This immediately leads to the following remark.

Remark 4.5 *By Corollary 4.1, using the repair polynomials defined above, P,* $\boldsymbol{LERS[RS_k(F, \mathcal{A}), B, P, \alpha^*]}$ *is an exact repair scheme for $RS_k(F, \mathcal{A})$ with repair bit-bandwidth $(N-1)\log_2(|B|)$.*

Let's discuss the efficiency of using these polynomials in the repair scheme $\mathbf{LERS[RS_k(F, \mathcal{A}), B, P, \alpha^*]}$. Firstly, notice if $N - 1 < kt$, then the repair bit-bandwidth of $\mathbf{LERS[RS_k(F, \mathcal{A}), B, M, \alpha^*]}$ executed with these polynomials is $(N-1)\log_2(|B|)$. Less than $kt\log_2(|B|)$, the repair bit-bandwidth of the direct scheme. Smaller N lead to schemes with smaller repair bit-bandwidths. However, recall it is required for $k \leq N - \frac{|F|}{|B|}$ in this scheme, implying a smaller N necessitates a smaller k. When implementing a scheme, the user must always choose their own balance between maximizing k, and minimizing the repair bit-bandwidth

Now that we have reviewed the previous research and methods, we have the necessary background knowledge to discuss recovery schemes for Cartesian codes.

5 Linear Exact Repair for Cartesian codes

The previous work with Reed Solomon codes is groundbreaking, but it only accounts for evaluation codes that use univariate polynomials. Just one particular subset of all possible Cartesian codes. We will now finally discuss how to execute a recovery using a linear exact repair scheme for a Cartesian code with an arbitrary m. The algorithm presented below is highly inspired by the 2017 Guruswarmi and Wooters paper [1], with some key adaptations for our setting. Recall, in the context of the **Set-up**, we are attempting to develop a scheme to recover a lost piece of data $f(\vec{\alpha}^*)$, given the available data set $\{f(\vec{\alpha}) \mid \vec{\alpha} \in \mathcal{A} \setminus \vec{\alpha}^*\}$. Further, we want our recovery scheme to allow the highest possible degree from our multivariate polynomial f, while minimizing the repair bit-bandwidth.

For generality sake, we will consider the following additional objects to be used in the linear exact repair scheme. Let $\tilde{m}$ be a positive integer such that $\tilde{m} \leq m$. Further let $\tilde{f} \in F[x_1, \ldots, x_{\tilde{m}}]$ be any polynomial in $\tilde{m}$ variables over F and define an evaluation set $\mathcal{T} \subseteq F^{\tilde{m}}$. Lastly, let $\boldsymbol{\gamma}^*$ be a particular element in $\mathcal{T}$. In some cases, f and $\tilde{f}$ will line-up, and in some $\mathcal{T}$ and $\mathcal{A}$ will line-up. It all depends on what we are trying to achieve. However, this general set-up allows us to formally discuss all of these different cases at once.

Now consider a distributed storage system capable of storing all of our evaluations. For each $\boldsymbol{\gamma} \in \mathcal{T}$, including $\boldsymbol{\gamma}^*$, store $\tilde{f}(\boldsymbol{\gamma})$ on a node of the distributed storage system. Call that node Node-$\boldsymbol{\gamma}$. Store the evaluations in such a way that $\tilde{f}(\boldsymbol{\gamma}^*)$ is stored on a distinct node from all other $\tilde{f}(\boldsymbol{\gamma})$. Note that for $\boldsymbol{\gamma}^{(1)} \neq \boldsymbol{\gamma}^{(2)} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*$, Node-$\boldsymbol{\gamma}^{(1)}$ and Node-$\boldsymbol{\gamma}^{(2)}$ may or may not be distinct. As above we are going to assume the $\tilde{f}(\boldsymbol{\gamma}^*)$ data on Node-$\boldsymbol{\gamma}^*$ is lost. The goal of this section is to develop an algorithm to exactly recover $\tilde{f}(\boldsymbol{\gamma}^*)$ from the $\tilde{f}(\boldsymbol{\gamma})$ data stored on the rest of the nodes.

We may now discuss the linear exact repair scheme algorithm. For readability sake, every step of the algorithm is labeled, and an explanation of that step is provided below.

Algorithm : **LERS2**$[\boldsymbol{C_k(F,\mathcal{T}),B,M,\gamma^*}]$ A linear exact repair scheme to recover a specific $\tilde{f}(\boldsymbol{\gamma}^*)$

Parameters:

- Cartesian code $C_k(F,\mathcal{T})$ defined by the finite field F with an evaluation set $\mathcal{T} \in F^m$ for a positive integer m and a positive integer k.
- A subfield $B \leq F$ such that $t \coloneqq [F:B]$.
- The element $\boldsymbol{\gamma}^* \in \mathcal{T}$
- Repair coefficients $\mu_{i,\boldsymbol{\gamma}}$ for $C_k(F,\mathcal{T})$ over B with respect to $\boldsymbol{\gamma}^*$ defined for all $i \in \{1,\ldots,t\}$ and $\alpha \in \mathcal{T} \setminus \boldsymbol{\gamma}^*$
  - The associated basis, $Z = \{z_1,\ldots,z_t\}$
  - The associated spanning elements $\omega_{\boldsymbol{\gamma},1},\ldots,\omega_{\boldsymbol{\gamma},d_{\boldsymbol{\gamma}}}$ for all $\boldsymbol{\gamma} \in \mathcal{T} \setminus \alpha^* \in F$ stored at Node-$\boldsymbol{\gamma}$
  - The associated span constants $\beta_{i,\boldsymbol{\gamma},1},\ldots,\beta_{i,\boldsymbol{\gamma},d_{\boldsymbol{\gamma}}} \in B$ for all $i \in \{1,\ldots,t\}$ and $\boldsymbol{\gamma} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*$ stored at Node-$\boldsymbol{\gamma}^*$

Input: A codeword $(\tilde{f}(\boldsymbol{\gamma}^{(1)}),\ldots,\tilde{f}(\boldsymbol{\gamma}^{(N)})) \in C_k(F,\mathcal{T})$ stored on a distributive storage system with $\tilde{f}(\boldsymbol{\gamma})$ successively stored at Node-$\boldsymbol{\gamma}$ for all $\boldsymbol{\gamma} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*$.

Output: $\tilde{f}(\boldsymbol{\gamma}^*)$

1: **for all** $\boldsymbol{\gamma} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*$ **do**
2: At Node-$\boldsymbol{\gamma}$ compute $u_{\boldsymbol{\gamma},j} \coloneqq \mathrm{tr}_{F/B}(\omega_{\boldsymbol{\gamma},j}\tilde{f}(\boldsymbol{\gamma}))$ for each $j \in \{1,\ldots,d_{\boldsymbol{\gamma}}\}$ and send to Node-$\boldsymbol{\gamma}^*$.
Note $u_{\boldsymbol{\gamma},j} \in B$ for all $j \in \{1,\ldots,d_{\boldsymbol{\gamma}}\}$.
3: **end for**

4: **for all** $i \in \{1,\ldots,t\}$ **do**
5: At Node-$\boldsymbol{\gamma}^*$, use the newly collected traces above to calculate

$$s_i \coloneqq \sum_{\boldsymbol{\gamma} in \mathcal{T} \setminus \boldsymbol{\gamma}^*} \beta_{i,\boldsymbol{\gamma},1}u_{\boldsymbol{\gamma},1} + \cdots + \beta_{i,\boldsymbol{\gamma},d_{\boldsymbol{\gamma}}}u_{\boldsymbol{\gamma},d\boldsymbol{\gamma}} \ .$$

Note $s_i = \mathrm{tr}_{F/B}(z_i\tilde{f}(\boldsymbol{\gamma}))$.
6: **end for**
7: Let $Y = \{\lambda_1,\ldots,\lambda_t\}$ be the dual basis associated with the basis Z. At Node-$\boldsymbol{\gamma}^*$ compute

$$\tilde{f}(\boldsymbol{\gamma}^*) = \sum_{i \in \{1,\ldots,t\}} s_i\lambda_i$$

Theorem 5.1 ***LERS2*$[\boldsymbol{C_k(F,\mathcal{T}),B,M,\gamma^*}]$** *is an exact repair scheme for $C_k(F,\mathcal{T})$ with respect to coordinate $\boldsymbol{\gamma}^*$.*

Proof. To prove **LERS2**$[\boldsymbol{C_k(F,\mathcal{T}),B,M,\gamma^*}]$ calculates $\tilde{f}(\boldsymbol{\gamma}^*)$, we will go through the algorithm line by line.

(1)-(3): In these lines, for every $\boldsymbol{\gamma} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*$ we calculate $u_{\boldsymbol{\gamma},j} \coloneqq \mathrm{tr}_{F/B}(\omega_{\boldsymbol{\gamma},j}\tilde{f}(\boldsymbol{\gamma}))$ for every $j \in \{1,\ldots,d_{\boldsymbol{\gamma}}\}$. These calculations are carried out at Node-$\boldsymbol{\gamma}$. Note $\tilde{f}(\boldsymbol{\gamma})$ is stored at Node-$\boldsymbol{\gamma}$, and for every $j \in \{1,\ldots,d_{\boldsymbol{\gamma}}\}$, $\omega_{\boldsymbol{\gamma},j}$ is stored at Node-$\boldsymbol{\gamma}$. Thus Node-$\boldsymbol{\gamma}$ has everything it needs to carry out these calculations.

The algorithm also notes that $u_{\boldsymbol{\gamma},j} \in B$ for all $\boldsymbol{\gamma} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*$ and $j \in \{1,\ldots,d_{\boldsymbol{\gamma}}\}$. This is because by definition $u_{\boldsymbol{\gamma},j} = \mathrm{tr}_{F/B}(\omega_{\boldsymbol{\gamma},j}\tilde{f}(\boldsymbol{\gamma}))$ and the range of $\mathrm{tr}_{F/B}$ is in B.

Then, for each $\boldsymbol{\gamma}$ and for all $j \in \{1,\ldots,d_{\boldsymbol{\gamma}}\}$, the algorithm sends these elements of B, $u_{\boldsymbol{\gamma},j}$, to Node-$\boldsymbol{\gamma}^*$ to be used in the next lines.

(4)-(6) Here, the algorithm first calculates s_i in terms of $\beta_{i,\boldsymbol{\gamma},j}$ and $u_{\boldsymbol{\gamma},j}$. Note, for all $i \in \{1,\ldots,t\}$, $\boldsymbol{\gamma} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*$, and $j \in \{1,\ldots,d_{\boldsymbol{\gamma}}\}$ we know $\beta_{i,\boldsymbol{\gamma},j}$ is stored at Node-$\boldsymbol{\gamma}^*$ and $u_{\boldsymbol{\gamma},j}$ was sent to Node-$\boldsymbol{\gamma}^*$. Hence, Node-$\boldsymbol{\gamma}^*$ has everything it needs to execute these calculations.

Then, the algorithm claims $s_i = \mathrm{tr}_{F/B}(z_i\tilde{f}(\boldsymbol{\gamma}^*))$ for every $i \in \{1,\ldots,t\}$. To prove this, fix an $i \in \{1,\ldots,t\}$. Then, by definition,

$$s_i \coloneqq \sum_{\boldsymbol{\gamma}\in\mathcal{T}\setminus\boldsymbol{\gamma}^*} \beta_{i,\boldsymbol{\gamma},1}u_{\boldsymbol{\gamma},1} + \cdots + \beta_{i,\boldsymbol{\gamma},d_{\boldsymbol{\gamma}}}u_{\boldsymbol{\gamma},d\boldsymbol{\gamma}} \ .$$

We will now manipulate this equation algebraically.

By definition of $u_{\boldsymbol{\gamma},j}$,

$$s_i = \sum_{\boldsymbol{\gamma}\in\mathcal{T}\setminus\boldsymbol{\gamma}^*} \beta_{i,\boldsymbol{\gamma},1}\,\mathrm{tr}_{F/B}((\omega_{\boldsymbol{\gamma},1})\tilde{f}(\boldsymbol{\gamma})) + \cdots + \beta_{i,\boldsymbol{\gamma},d_{\boldsymbol{\gamma}}}\,\mathrm{tr}_{F/B}((\omega_{\boldsymbol{\gamma},d_{\boldsymbol{\gamma}}})\tilde{f}(\boldsymbol{\gamma})) \ .$$

By B-linearity of $\mathrm{tr}_{F/B}$,

$$s_i = \sum_{\boldsymbol{\gamma}\in\mathcal{T}\setminus\boldsymbol{\gamma}^*} \mathrm{tr}_{F/B}(\beta_{i,\boldsymbol{\gamma},1}(\omega_{\boldsymbol{\gamma},1})\tilde{f}(\boldsymbol{\gamma})) + \cdots + \mathrm{tr}_{F/B}(\beta_{i,\boldsymbol{\gamma},d_{\boldsymbol{\gamma}}}(\omega_{\boldsymbol{\gamma},d_{\boldsymbol{\gamma}}})\tilde{f}(\boldsymbol{\gamma})) \ .$$

By B-linearity of $\mathrm{tr}_{F/B}$,

$$s_i = \sum_{\boldsymbol{\gamma}\in\mathcal{T}\setminus\boldsymbol{\gamma}^*} \mathrm{tr}_{F/B}(\beta_{i,\boldsymbol{\gamma},1}(\omega_{\boldsymbol{\gamma},1})\tilde{f}(\boldsymbol{\gamma}) + \cdots + \beta_{i,\boldsymbol{\gamma},d_{\boldsymbol{\gamma}}}(\omega_{\boldsymbol{\gamma},d_{\boldsymbol{\gamma}}})\tilde{f}(\boldsymbol{\gamma})) \ .$$

Factoring out $\tilde{f}(\boldsymbol{\gamma})$,

$$s_i = \sum_{\boldsymbol{\gamma} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*} \mathrm{tr}_{F/B}((\beta_{i,\boldsymbol{\gamma},1}(\omega_{\boldsymbol{\gamma},1}) + \cdots + \beta_{i,\boldsymbol{\gamma},d_{\boldsymbol{\gamma}}}(\omega_{\boldsymbol{\gamma},d_{\boldsymbol{\gamma}}}))\tilde{f}(\boldsymbol{\gamma})) \ .$$

By definition of $\omega_{\boldsymbol{\gamma},j}$ and $\beta_{i,\boldsymbol{\gamma},j}$,

$$s_i = \sum_{\boldsymbol{\gamma} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*} \mathrm{tr}_{F/B}(\mu_{i,\boldsymbol{\gamma}} \tilde{f}(\boldsymbol{\gamma})) \ .$$

By B-linearity of $\mathrm{tr}_{F/B}$,

$$s_i = \mathrm{tr}_{F/B} \left(\sum_{\boldsymbol{\gamma} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*} \mu_{i,\boldsymbol{\gamma}} \tilde{f}(\boldsymbol{\gamma}) \right) \ .$$

Thus,

$$s_i = \mathrm{tr}_{F/B} \left(\sum_{\boldsymbol{\gamma} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*} \mu_{i,\boldsymbol{\gamma}} \tilde{f}(\boldsymbol{\gamma}) \right) \ .$$

However, notice by the definition of $\mu_{i,\boldsymbol{\gamma}}$ in equation 4.1,

$$z_i \tilde{f}(\boldsymbol{\gamma}^*) = \sum_{\boldsymbol{\gamma} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*} \mu_{i,\boldsymbol{\gamma}} \tilde{f}(\boldsymbol{\gamma}).$$

Then taking the trace of both sides yields,

$$\mathrm{tr}_{F/B}(z_i \tilde{f}(\boldsymbol{\gamma}^*)) = \mathrm{tr}_{F/B} \left(\sum_{\boldsymbol{\gamma} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*} \mu_{i,\boldsymbol{\gamma}} \tilde{f}(\boldsymbol{\gamma}) \right) \ .$$

Therefore,

$$s_i = \mathrm{tr}_{F/B} \left(\sum_{\boldsymbol{\gamma} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*} \mu_{i,\boldsymbol{\gamma}} \tilde{f}(\boldsymbol{\gamma}) \right) = \mathrm{tr}_{F/B}(z_i \tilde{f}(\boldsymbol{\gamma}^*)) \ , \tag{5.1}$$

as claimed.

(7) We start line 7 by referencing the dual basis Y associated with the basis Z. Then the algorithm claims that

$$\tilde{f}(\boldsymbol{\gamma}^*) = \sum_{i \in \{1,\ldots,t\}} s_i \lambda_i \ .$$

To prove this, first consider the right side of the equation. By equation 5.1,

$$\sum_{i\in\{1,\dots,t\}} s_i\lambda_i = \sum_{i\in\{1,\dots,t\}} \mathrm{tr}_{F/B}(z_i\tilde{f}(\boldsymbol{\gamma}^*))\lambda_i \ .$$

Then by properties of dual bases,

$$\sum_{i\in\{1,\dots,t\}} \mathrm{tr}_{F/B}(z_i\tilde{f}(\boldsymbol{\gamma}^*))\lambda_i = f(\tilde{\boldsymbol{\gamma}}^*) \ .$$

Hence,

$$\tilde{f}(\boldsymbol{\gamma}^*) = \sum_{i\in\{1,\dots,t\}} s_i\lambda_i$$

as claimed.

□

Now let us do some preliminary analysis on the bandwidth of the exact repair scheme $\mathbf{LERS2}[\boldsymbol{C_k(F,\mathcal{T}),B,M,\gamma^*}]$. Notice, the only data being sent between nodes in the whole scheme are the $u_{\boldsymbol{\gamma},j}$ elements. As noted above, for all $\boldsymbol{\gamma}\in\mathcal{T}$ and for all $j\in\{1,\dots,d_{\boldsymbol{\gamma}}\}$, $u_{\boldsymbol{\gamma},j}\in B$. The amount of information in any single element of B is $\log_2(|B|)$. Further, for each $\boldsymbol{\gamma}\in\mathcal{T}\setminus\boldsymbol{\gamma}^*$ there are $d_{\boldsymbol{\gamma}}$ elements of the form $u_{\boldsymbol{\gamma},j}$. Therefore there are $\sum_{\boldsymbol{\gamma}\in\mathcal{T}\setminus\boldsymbol{\gamma}^*} d_{\boldsymbol{\gamma}}$ elements of B to be to sent. Hence, in total this scheme sends $\left(\sum_{\boldsymbol{\gamma}\in\mathcal{T}\setminus\boldsymbol{\gamma}^*} d_{\boldsymbol{\gamma}}\right)\log_2(|B|)$ bits of information. Meaning the repair bit-bandwidth of $\mathbf{LERS2}[\boldsymbol{C_k(F,\mathcal{T}),B,M,\gamma^*}]$ using repair coefficients $\mu_{i,\boldsymbol{\gamma}}$ to recover $\tilde{f}(\boldsymbol{\gamma}^*)$ is $\left(\sum_{\boldsymbol{\gamma}\in\mathcal{T}\setminus\boldsymbol{\gamma}^*} d_{\boldsymbol{\gamma}}\right)\log_2(|B|) = D\log_2(|B|)$. Where D is the total dimension of our repair coefficients.

Remark 5.1 (Bit-Bandwidth of $\mathbf{LERS2}[\boldsymbol{C_k(F,\mathcal{T}),B,M,\gamma^*}]$) *The repair bit-bandwidth of $\boldsymbol{LERS2}[\boldsymbol{C_k(F,\mathcal{T}),B,M,\gamma^*}]$ is $D\log_2(|B|)$, where D is the total dimension of our repair coefficients.*

Again, we find the most consistent way to find repair coefficients, is to use repair polynomials. Recall Theorem 4.3.

Theorem (Coefficient to Dual Polynomial Equivalence) *Given a finite field F with evaluation set $\mathcal{T}=A_1\times\cdots\times A_m\subseteq F^m$ for some positive integer m, a subfield $B\leq F$ with $t\coloneqq[F:B]$, and an integer $k\geq 0$. Define $n_j\coloneqq|A_j|$ for all $j\in\{1,\dots,m\}$, and index $\mathcal{T}=\{\boldsymbol{\gamma}^{(1)},\dots,\boldsymbol{\gamma}^{(N)}\}$. Let $\boldsymbol{\gamma}^*\in\mathcal{T}$. Then the following are equivalent.*

(1) There is a set of repair coefficients for $C_k(F, \mathcal{T})$ over B with respect to $\boldsymbol{\gamma}^*$ with total dimension D.

(2) There is a set of repair polynomials for for $C_k(F, \mathcal{T})$ over B with respect to $\boldsymbol{\gamma}^*$ with total dimension D.

In the above section we noted this theorem was proven for the special case of $m = 1$ by Guruswarmi and Wooters in 2017 [1]. We will now prove this fact for an arbitrary positive integer m.

Proof. (1) $\implies$ (2): Let $M = \{\mu_{i,\boldsymbol{\gamma}} \mid i \in \{1, \ldots, t\} \text{ and } \boldsymbol{\gamma} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*\}$ be a set of repair coefficients for $C_k(F, \mathcal{T})$ over B with respect to $\boldsymbol{\gamma}^*$ with total dimension D. Then, by Definition 4.1, there is an associated basis, $Z = \{z_1, \ldots, z_t\}$, such that for any $\tilde{f} \in F[x_1, \ldots, x_m]$ with $\deg(\tilde{f}) < k$,

$$z_i \tilde{f}(\boldsymbol{\gamma}^*) = \sum_{\boldsymbol{\gamma} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*} \mu_{i,\boldsymbol{\gamma}} \tilde{f}(\boldsymbol{\gamma})$$

for all $i \in \{1, \ldots, t\}$.

Then, for all $i \in \{1, \ldots, t\}$, subtracting $z_i \tilde{f}(\boldsymbol{\gamma}^*)$ from both sides yields,

$$0 = \sum_{\boldsymbol{\gamma} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*} \mu_{i,\boldsymbol{\gamma}} \tilde{f}(\boldsymbol{\gamma}) - z_i \tilde{f}(\boldsymbol{\gamma}^*) \, .$$

Let $\mu_{i,\boldsymbol{\gamma}^*} \coloneqq -z_i$ for all i. Then we have that,

$$0 = \sum_{\boldsymbol{\gamma} \in \mathcal{T}} \mu_{i,\boldsymbol{\gamma}} \tilde{f}(\boldsymbol{\gamma}) \, .$$

Thus, for each i, we have that $(\mu_{i,\boldsymbol{\gamma}^{(1)}} \tilde{f}(\boldsymbol{\gamma}^{(1)}), \ldots, \mu_{i,\boldsymbol{\gamma}^{(N)}} \tilde{f}(\boldsymbol{\gamma}^{(N)}))$ is a codeword in the dual code $C_k(\mathcal{T})^\perp$. Recall, $C_k(\mathcal{T})^\perp = C_{k'}(\mathcal{T}, \vec{v})$, where $k' = \sum_{i=1}^{m} (n_i - 1) - k + 1$ and $\vec{v} = (v_{\boldsymbol{\gamma}^{(1)}}, v_{\boldsymbol{\gamma}^{(2)}}, \ldots, v_{\boldsymbol{\gamma}^{(N)}})$ is a particular vector in $(F \setminus \{0\})^N$. Hence, for each i we know $(\mu_{i,\boldsymbol{\gamma}^{(1)}} \tilde{f}(\boldsymbol{\gamma}^{(1)}), \ldots, \mu_{i,\boldsymbol{\gamma}^{(N)}} \tilde{f}(\boldsymbol{\gamma}^{(N)})) \in C_{k'}(\mathcal{T}, \vec{v})$ which implies there exists a polynomial $p_i \in F[x_1, \ldots, x_m]$ with $\deg(p) < k'$ such that,

$$(\mu_{i,\boldsymbol{\gamma}^{(1)}} \tilde{f}(\boldsymbol{\gamma}^{(1)}), \ldots, \mu_{i,\boldsymbol{\gamma}^{(N)}} \tilde{f}(\boldsymbol{\gamma}^{(N)})) = (v_{\boldsymbol{\gamma}^{(1)}} p_i(\boldsymbol{\gamma}^{(1)}), \ldots, v_{\boldsymbol{\gamma}^{(N)}} p_i(\boldsymbol{\gamma}^{(N)})) \, .$$

Therefore, for all $i \in \{1, \ldots, t\}$ and $\boldsymbol{\gamma} \in \mathcal{T}$, $\mu_{i,\boldsymbol{\gamma}} = v_{\boldsymbol{\gamma}} p_i(\boldsymbol{\gamma}) \implies p_i(\boldsymbol{\gamma}) = v_{\boldsymbol{\gamma}}^{-1} \mu_{i,\boldsymbol{\gamma}}$.
Let $P = \{p_i \mid i \in \{1, \ldots, t\}\}$ be the set of these polynomials.

By Definition 4.2, in order for P to be a set of repair polynomials for $C_k(F, \mathcal{T})$ over B with respect to $\boldsymbol{\gamma}^*$, it must satisfy the following conditions.

(i) $\deg(p_i) < k' = \sum_{j=1}^{m}(n_j - 1) - k + 1$ for all $i \in \{1, \ldots, t\}$

(ii) The set $\{p_i(\boldsymbol{\gamma}^*) \mid i \in \{1, \ldots, t\}\}$ must form a basis for F over B

For (i), since p_i are elements of the dual polynomial, we have already shown $\deg(p_i) < k'$ for all $i \in \{1, \ldots, t\}$.

Then, for (ii), consider
$\{p_i(\boldsymbol{\gamma}^*) \mid i \in \{1, \ldots, t\}\} = \{v_{\boldsymbol{\gamma}^*}^{-1}\mu_{i,\boldsymbol{\gamma}^*} \mid i \in \{1, \ldots, t\}\} = \{v_{\boldsymbol{\gamma}^*}^{-1} z_i \mid i \in \{1, \ldots, t\}\} = v_{\boldsymbol{\gamma}^*}^{-1} Z$.
Since Z is a basis for F over B, $v_{\boldsymbol{\gamma}^*}^{-1} Z$ is also a basis for F over B.

Thus, by definition, $P = \{p_i \mid i \in \{1, \ldots, t\}\}$ is a set of repair polynomials for $C_k(F, \mathcal{T})$ over B with respect to $\boldsymbol{\gamma}^*$.

Further, consider $\delta_{\boldsymbol{\gamma}} = \dim_B\{p_i(\boldsymbol{\gamma}) \mid i \in \{1, \ldots, t\}\}$ for all $\boldsymbol{\gamma} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*$. For any $\boldsymbol{\gamma} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*$, $\delta_{\boldsymbol{\gamma}} = \dim_B\{p_i(\boldsymbol{\gamma}) \mid i \in \{1, \ldots, t\}\} = \dim_B\{v_{\boldsymbol{\gamma}}^{-1}\mu_{i,\boldsymbol{\gamma}} \mid i \in \{1, \ldots, t\}\} = \dim_B v_{\boldsymbol{\gamma}}^{-1}\{\mu_{i,\boldsymbol{\gamma}} \mid i \in \{1, \ldots, t\}\}$. Since multiplication by a constant vector does not change the dimension of a set, $\dim_B v_{\boldsymbol{\gamma}}^{-1}\{\mu_{i,\boldsymbol{\gamma}} \mid i \in \{1, \ldots, t\}\} = \dim_B\{\mu_{i,\boldsymbol{\gamma}} \mid i \in \{1, \ldots, t\}\}$.
Notice, for any $\boldsymbol{\gamma}$, we have that $\dim_B\{\mu_{i,\boldsymbol{\gamma}} \mid i \in \{1, \ldots, t\}\} = d_{\boldsymbol{\gamma}}$, the $\boldsymbol{\gamma}$-dimension of our repair coefficients M. Then the total dimension of our repair polynomials, P, is

$$\sum_{\boldsymbol{\gamma} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*} \delta_{\boldsymbol{\gamma}} = \sum_{\boldsymbol{\gamma} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*} d_{\boldsymbol{\gamma}} = D \, .$$

Hence, P is a set or repair coefficients with total dimension D, as desired.

(2) $\implies$ (1): Let $P = \{p_1, \ldots, p_t\}$ be a set of repair polynomials for $C_k(F, \mathcal{T})$ over B with respect to $\boldsymbol{\gamma}^*$ with total dimension D. Then, by Definition 4.2,
$\deg(p_i) < k' = \sum_{j=1}^{m}(n_j - 1) - k + 1$ for all $i \in \{1, \ldots, t\}$ and $Z' = \{z'_1, \ldots, z'_t\}$, where $z_i = p_i(\boldsymbol{\gamma}^*)$ for all $i \in \{1, \ldots, t\}$, forms a basis for F over B.

Because $\deg(p_i) < k'$, we know p_i is an allowable polynomial to be used in the dual code, $C_k(F, \mathcal{A})^{\perp}$. Let $\vec{v} = (v_{\boldsymbol{\gamma}^{(1)}}, \ldots, v_{\boldsymbol{\gamma}^{(N)}}) \in (F \setminus \{0\})^N$ be the vector such that
$C_k(F, \mathcal{T})^{\perp} = C_{k'}(F, \mathcal{T}, \vec{v})$. Hence,

$$(v_{\boldsymbol{\gamma}^{(1)}} p_i(\boldsymbol{\gamma}^{(1)}), \ldots, v_{\boldsymbol{\gamma}^{(N)}} p_i(\boldsymbol{\gamma}^{(N)})) \in C_{k'}(F, \mathcal{T}, \vec{v}) = C_k(F, \mathcal{T})^{\perp} \, .$$

Let $\mu_{i,\boldsymbol{\gamma}} = v_{\boldsymbol{\gamma}} p_i(\boldsymbol{\gamma})$ for all $i \in \{1, \ldots, t\}$ and $\boldsymbol{\gamma} \in \mathcal{T}$. Define M to be the set of these $\mu_{i,\boldsymbol{\gamma}}$. Thus,

$$(\mu_{i,\boldsymbol{\gamma}^{(1)}}, \ldots, \mu_{i,\boldsymbol{\gamma}^{(N)}}) \in C_k(F, \mathcal{T})^{\perp} \, ,$$

for all $i \in \{1, \ldots, t\}$.

Note, by Definition 4.1, M is a set of repair coefficients for $C_k(F, \mathcal{T})$ over B with respect to $\boldsymbol{\gamma}^*$ if for every $\tilde{f} \in F[x_1, \ldots, x_m]$ with $\deg(\tilde{f}) < k$ we have,

$$z_i \tilde{f}(\boldsymbol{\gamma}^*) = \sum_{\boldsymbol{\gamma} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*} \mu_{i,\boldsymbol{\gamma}} \tilde{f}(\boldsymbol{\gamma}) \ ,$$

and $Z = \{z_1, \ldots, z_t\}$ forms a basis for F over B.

Let $\tilde{f} \in F[x_1, \ldots, x_m]$ with $\deg(\tilde{f}) < k$. Then, for every $i \in \{1, \ldots, t\}$,

$$(\mu_{i,\boldsymbol{\gamma}^{(1)}}, \ldots, \mu_{i,\boldsymbol{\gamma}^{(N)}}) \in C_k(F, \mathcal{T})^\perp \implies 0 = (\mu_{i,\boldsymbol{\gamma}^{(1)}}, \ldots, \mu_{i,\boldsymbol{\gamma}^{(N)}}) \cdot (\tilde{f}(\boldsymbol{\gamma}^{(1)}), \ldots, \tilde{f}(\boldsymbol{\gamma}^{(N)})) \ .$$

This implies,

$$0 = \sum_{\boldsymbol{\gamma} \in \mathcal{T}} \mu_{i,\boldsymbol{\gamma}} \tilde{f}(\boldsymbol{\gamma}) \ .$$

Then, subtracting $\mu_{i,\boldsymbol{\gamma}^*} \tilde{f}(\boldsymbol{\gamma}^*)$ from both sides yields,

$$-\mu_{i,\boldsymbol{\gamma}^*} \tilde{f}(\boldsymbol{\gamma}^*) = \sum_{\boldsymbol{\gamma} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*} \mu_{i,\boldsymbol{\gamma}} \tilde{f}(\boldsymbol{\gamma}) \ .$$

Let $z_i = -\mu_{i,\boldsymbol{\gamma}}$. Recall, $\mu_{i,\boldsymbol{\gamma}^*} = v_{\boldsymbol{\gamma}^*} p_i(\boldsymbol{\gamma}^*) = v_{\boldsymbol{\gamma}^*} z'_i$ for all $i \in \{1, \ldots, t\}$. Thus,

$$Z = \{z_1, \ldots, z_t\} = \{-\mu_{i,\boldsymbol{\gamma}^*} \mid i \in \{1, \ldots, t\}\} = \{-v_{\boldsymbol{\gamma}^*} z'_i \mid i \in \{1, \ldots, t\}\} \ .$$

However, notice,

$$-v_{\boldsymbol{\gamma}^*}\{z'_1, \ldots, z'_t\} = -v_{\boldsymbol{\gamma}^*} Z' \ .$$

Since Z' is a basis for F over B, $-v_{\boldsymbol{\gamma}^*} Z'$ is a basis for F over B.

Therefore Z is a basis for F over B, which implies M is a set of repair coefficients, as desired.

Further, for any $\boldsymbol{\gamma} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*$, the $\boldsymbol{\gamma}$-dimension of M is $d_{\boldsymbol{\gamma}} = \dim_B\{\mu_{i,\boldsymbol{\gamma}} \mid i \in \{1, \ldots, t\}\} = \dim_B\{v_{\boldsymbol{\gamma}} p_i(\boldsymbol{\gamma}) \mid i \in \{1, \ldots, t\}\} = \dim_B v_{\boldsymbol{\gamma}}\{p_i(\boldsymbol{\gamma}) \mid i \in \{1, \ldots, t\}\}$. Again, notice multiplying by a constant, does not change the dimension of a set. Thus,
$\dim_B v_{\boldsymbol{\gamma}}\{p_i(\boldsymbol{\gamma}) \mid i \in \{1, \ldots, t\}\} = \dim_B\{p_i(\boldsymbol{\gamma}) \mid i \in \{1, \ldots, t\}\} = \delta_{\boldsymbol{\gamma}}$, the $\boldsymbol{\gamma}$-dimension of P.

Thus, the total dimension of M is,

$$\sum_{\boldsymbol{\gamma} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*} d_{\boldsymbol{\gamma}} = \sum_{\boldsymbol{\gamma} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*} \delta_{\boldsymbol{\gamma}} = D \ .$$

Hence, M is a set of repair coefficients for $C_k(F, \mathcal{A})$ over B with respect to $\boldsymbol{\gamma}^*$ with total dimension D, as was claimed.

□

The utility of this theorem, is that just by finding repair polynomials, we can find repair coefficients, and create a recovery scheme.

Remark 5.2 *We will refer to algorithm* $\boldsymbol{LERS2[C_k(F,\mathcal{T}), B, P, \gamma^*]}$ *to mean* $\boldsymbol{LERS2[C_k(F,\mathcal{T}), B, M, \gamma^*]}$ *run with repair coefficients* M *derived from repair polynomials* P*, as outlined in the above proof.*

Like in the Reed-Solomon case, we can determine the repair bit-bandwidth of a $\mathbf{LERS2[C_k(F,\mathcal{T}), B, P, \gamma^*]}$ just from the total dimension of the repair polynomials, P.

Corollary 5.1 *If there exists a set of repair polynomials,* P*, for* $C_k(F, \mathcal{T})$ *over* B *with respect to* $\boldsymbol{\gamma}^*$ *with total dimension* D*, then there is an exact repair scheme for* $C_k(F, \mathcal{T})$ *with respect to* $\boldsymbol{\gamma}^*$ *with an repair bit-bandwidth of* $D \log_2(|B|)$*,* $\boldsymbol{LERS2[C_k(F,\mathcal{T}), B, P, \gamma^*]}$ *.*

Proof. By the theorem above, if there exists repair polynomials, P, for $C_k(F, \mathcal{T})$ over B with respect to $\boldsymbol{\gamma}^*$ with total dimension D, then there exists repair coefficients M for $C_k(F, \mathcal{T})$ over B with respect to $\boldsymbol{\gamma}^*$ with total dimension D. Hence, by Theorem 5.1, $\mathbf{LERS2[C_k(F,\mathcal{T}), B, P, \gamma^*]}$, which is equivalent to $\mathbf{LERS2[C_k(F,\mathcal{T}), B, M, \gamma^*]}$, is an exact repair scheme for $C_k(F, \mathcal{T})$ with respect to $\boldsymbol{\gamma}^*$ with repair bit-bandwidth D.

□

We have now boiled the problem of recovering a lost symbol from $C_k(F, \mathcal{T})$ down to finding these special repair polynomials. Constructing these polynomials is the sole topic of the next two subsections. The methods we provide give multiple ways to find repair polynomials, and therefore construct recovery schemes for Cartesian codes.

5.1 Extracting a Reed-Solomon code

Our first approach to creating a recovery scheme for a Cartesian code is to simply avoid the Cartesian code. We do this by looking at a Reed-Solomon code that maps onto our Cartesian code in a particularly nice way. The main idea is to construct univariate polynomials that evaluate to a subset of the codewords in a Cartesian code. Then, we can use the repair properties of a Reed-Solomon code derived from those univariate polynomials to repair erasures in the Cartesian code. We start with a remark concerning the notation surrounding particularly useful polynomials.

Remark 5.3 *Throughout this section, a polynomial $h(x)$ from F to F^m refers to a function $h(x) = (h_1(x), \ldots, h_m(x))$, where each $h_j(x)$ is a polynomial in $F[x]$ for every $j \in \{1, \ldots, m\}$. We refer to the* $\deg(h(x))$ *to mean,*

$$\max_{j \in \{1,\ldots,m\}} (\deg(h(x))) \ .$$

It is known, if f is a polynomial in $F[x_1, \ldots, x_m]$, then $\deg(f(h(x)) \leq \deg(f)\deg(h)$.

With the following remark in mind, we can now construct the discussed repair scheme.

Theorem 5.2 *Let F be a finite field with a subfield $B \leq F$ such that $t := [F : B]$ and an evaluation set $\mathcal{A} = A_1 \times \cdots \times A_m \subseteq F^m$ for some positive integer m. Define $n_j = |A_j|$ for all $j \in \{1, \ldots, m\}$ and $N = |\mathcal{A}|$. Pick an $\vec{\alpha}^* \in \mathcal{A}$ and consider the Cartesian code $C_k(F, \mathcal{A})$.*

Suppose the following exist.

(i) *A finite field extension $\tilde{F}$ over B such that $s = [\tilde{F} : B]$.*

(ii) *A polynomial $h : \tilde{F} \to F^m$ such that $h(x) = \vec{0}$ has exactly one solution, $x = 0$. Define $\mathcal{T} = h^{-1}(\{\vec{\alpha} - \vec{\alpha}^* \mid \vec{\alpha} \in \mathcal{A}\})$.*

(iii) *Repair polynomials $P = \{p_1, \ldots, p_T\}$ for $RS_{\tilde{k}}(\tilde{F}, \mathcal{T})$ with respect to the 0 element, such that*

$$k \leq \frac{\tilde{k}}{\deg(h)} \ .$$

Let D be the total dimension of P.

Then there is an exact repair scheme for $C_k(F, \mathcal{A})$ with respect to $\vec{\alpha}^*$.

Proof. Let $f \in F[x_1, \ldots, x_m]$ such that $\deg(f) < k$. Then note $(f(\vec{\alpha}^{(1)}), \ldots, f(\vec{\alpha}^{(N)}))$ is a code word $C_k(F, \mathcal{A})$, where $\mathcal{A} = \{\vec{\alpha}^{(1)}, \ldots, \vec{\alpha}^{(N)}\}$. By definition of an exact repair scheme, we first need to consider $(f(\vec{\alpha}^{(1)}), \ldots, f(\vec{\alpha}^{(N)}))$ stored on a distributed storage system with

each piece of data $f(\vec{\alpha}^{(1)}), \ldots, f(\vec{\alpha}^{(N)})$ stored at distinct nodes Node-$\vec{\alpha}^{(1)}$ $\ldots$, Node-$\vec{\alpha}^{(N)}$ respectively. Then an exact repair scheme will be a process to recover $f(\vec{\alpha}^*)$ from the information in the other nodes.

Consider the function $\tilde{f} : \tilde{F} \to F$ by

$$\tilde{f} = f(h(x) + \vec{\alpha}^*) \ .$$

In our distributed storage system, the data $\tilde{f}(\boldsymbol{\gamma}) = f(h(\boldsymbol{\gamma}) + \vec{\alpha}^*)$ is stored at Node-$h(\boldsymbol{\gamma}) + \vec{\alpha}^*$ for all $\boldsymbol{\gamma} \in \mathcal{T}$.

Note,

$$\deg(\tilde{f}) \leq \deg(f(h(x) + \vec{\alpha}^*)) = \deg(f(h(x)) = \deg(f)\deg(h) \ .$$

However, by definition $\deg(f) < k$, so,

$$\deg(f)\deg(h) < k\deg(h)$$

Then, by (iii) in the setup,

$$k \leq \frac{\tilde{k}}{\deg(h)} \ ,$$

so,

$$\deg(\tilde{f}) < k\deg(h) \leq \frac{\tilde{k}}{\deg(h)}\deg(h) = \tilde{k} \ .$$

Define $\tilde{N} = |\mathcal{T}|$, and index by $\mathcal{T} = \{\boldsymbol{\gamma}^{(1)}, \ldots \boldsymbol{\gamma}^{(\tilde{N})}\}$

Thus, $\tilde{f} \in \tilde{F}[x]$ has degree less than $\tilde{k}$, so $(\tilde{f}(\boldsymbol{\gamma}^{(1)}), \ldots, \tilde{f}(\boldsymbol{\gamma}^{(\tilde{N})})) \in \mathrm{RS}_{\tilde{k}}(\tilde{F}, \mathcal{T})$.

Recall $\mathcal{T} = h^{-1}(\{\vec{\alpha} - \vec{\alpha}^* \mid \vec{\alpha} \in \mathcal{A}\})$. Hence, $\vec{\alpha}^* \in \mathcal{A}$, so $\mathcal{T}$ contains the element $h^{-1}(\vec{\alpha}^* - \vec{\alpha}^*) = h^{-1}(0) = 0$ since h was defined such that 0 is the only solution to $h(x) = \vec{0}$.

Thus it makes sense to discuss exact repair scheme of $\mathrm{RS}_{\tilde{F},\tilde{k}}(\mathcal{T})$ with respect to 0. Specifically, note that P is a set of repair polynomials for $\mathrm{RS}_{\tilde{F},\tilde{k}}(\mathcal{T})$ with respect to 0.

Therefore, executing $\mathbf{LERS[RS_{\tilde{k}}(\tilde{F}, \mathcal{T}), P, 0]}$ with the function $\tilde{f}$ will recover $\tilde{f}(0) = f(h(0) + \vec{\alpha}^*) = f(\vec{\alpha}^*)$, the data we needed to recover. Further, by Corollary 5.1 it will recover this lost data calling on $D\log_2(|B|)$ bits of information, where D is the total dimension of P. However, it is not obvious if any of these $D\log_2(|B|)$ bits come from the

lost node, Node-$\vec{\alpha}^*$. If some did, the distributed storage system would not have access to this data, and this repair scheme would fail. We need to show all of this information is stored in nodes other that Node-$\vec{\alpha}^*$.

We know the scheme enacts the repair just using information from $\tilde{f}(\boldsymbol{\gamma})$ for $\boldsymbol{\gamma} \in \mathcal{T} \setminus \{0\}$. Note for any $\boldsymbol{\gamma} \in \mathcal{T} \setminus \{0\}$, the data $\tilde{f}(\boldsymbol{\gamma})$ is stored at Node-$(h(\boldsymbol{\gamma}) + \vec{\alpha}^*)$. Also note, the data $\tilde{f}(0)$ is stored at Node-$(h(0) + \vec{\alpha}^*)$ = Node-$(\vec{\alpha}^*)$. However, if $\vec{\alpha}^* = h(\boldsymbol{\gamma}) + \vec{\alpha}^*$ that would imply $h(\boldsymbol{\gamma}) = 0$ for a $\boldsymbol{\gamma}$ that is not 0. Which contradicts the definition of h. Therefore $h(\boldsymbol{\gamma}) + \vec{\alpha}^* \neq \vec{\alpha}^*$, and since the Nodes are distinct for all $\vec{\alpha} \in \mathcal{A}$, Node-$\vec{\alpha}^*$ is distinct from Node-$(h(\boldsymbol{\gamma}) + \vec{\alpha}^*)$ for all $\boldsymbol{\gamma} \in \mathcal{T} \setminus \{0\}$.

Therefore $f(\vec{\alpha}^*)$ was recovered using $D \log_2(|B|)$ bits of data only from nodes other than Node-$\vec{\alpha}^*$. Because f was arbitrary, this describes an exact repair scheme for $C_k(F, \mathcal{A})$ with respect to $\vec{\alpha}^*$ with repair bit-bandwidth $D \log_2(|B|)$, as desired.

□

The above theorem is applicable in any setting where we can map a Reed-Solomon code onto a Cartesian code. A very simple application of this is when the evaluation set, $\mathcal{A} = A_1 \times \cdots \times A_m$ has one factor $A_j = F$. We discuss this case in the following corollary.

Corollary 5.2 *Consider the Cartesian code $C_k(F, \mathcal{A})$, where $\mathcal{A} = A_1 \times \cdots \times A_m$ with $A_{j^*} = F$ for some $j^* \in \{1, \ldots, m\}$. Let $h(x)$ be a polynomial from F to F^m such that the only solution to $h(x) = \vec{0}$ is 0. Assume $k \leq |F| \left(1 - \frac{1}{|B|}\right)$. Choose an $\vec{\alpha}^* \in \mathcal{A}$.*

Then there is an exact repair scheme for $C_k(F, \mathcal{A})$ with respect to $\vec{\alpha}^*$ with repair bit-bandwidth $(|F| - 1) \log_2(|B|)$.

Proof. We will prove this corollary using Theorem 5.2. In order to use this theorem, we need the following.

(i) A finite field extension $\tilde{F}$ over B such that $s = [\tilde{F} : B]$.

(ii) A polynomial $h : \tilde{F} \to F^m$ such that $h(x) = \vec{0}$ has exactly one solution, $x = 0$. Define $\mathcal{T} = h^{-1}(\{\vec{\alpha} - \vec{\alpha}^* \mid \vec{\alpha} \in \mathcal{A}\})$.

(iii) Repair polynomials $P = \{p_1, \ldots, p_T\}$ for $\mathrm{RS}_{\tilde{k}}(\tilde{F}, \mathcal{T})$ with respect to the 0 element, such that

$$k \leq \frac{\tilde{k}}{\deg(h)} .$$

Firstly, let $\tilde{F} = F$. Thus $[\tilde{F} : B] = [F : B] = t$.

Next, let $h(x) = (h_1(x), \ldots, h_m(x))$, where $h_{j^*}(x) = x$ and $h_j(x) = 0$ for all $j \neq j^*$. Note, if $h(x) = \vec{0}$ that implies $h_j(x) = 0$, which implies $x = 0$. Hence, the only solution to $h(x) = \vec{0}$ is $x = 0$. Then, let $\mathcal{T} = h^{-1}(\{\vec{\alpha} - \vec{\alpha}^* \mid \vec{\alpha} \in \mathcal{A}\})$. Pick $\vec{a}^{(\boldsymbol{\gamma})} \in \mathcal{A}$ such that in the jth coordinate, $\vec{a}^{(\boldsymbol{\gamma})}_{j^*} = \boldsymbol{\gamma}$ for all $\boldsymbol{\gamma} \in F$. Note, for each $\boldsymbol{\gamma} \in F$, we know there exists an $\vec{a}^{(\boldsymbol{\gamma})} \in \mathcal{A}$, since $\mathcal{A} = A_1 \times \cdots \times A_m$ with $A_{j^*} = F$. Then, consider $\{\vec{a}^{(\boldsymbol{\gamma})} - \vec{\alpha}^* \mid \boldsymbol{\gamma} \in F\}$. For any $\boldsymbol{\gamma}_1 \neq \boldsymbol{\gamma}_2$, we have $\vec{a}^{(\boldsymbol{\gamma}_1)}_{j^*} = \boldsymbol{\gamma}_1 \neq \boldsymbol{\gamma}_2 = \vec{a}^{(\boldsymbol{\gamma}_2)}_{j^*}$, which implies $\vec{a}^{(\boldsymbol{\gamma}_1)} \neq \vec{a}^{(\boldsymbol{\gamma}_2)}$, in turn implying $\vec{a}^{(\boldsymbol{\gamma}_1)} - \vec{\alpha}^* \neq \vec{a}^{(\boldsymbol{\gamma}_2)} - \vec{\alpha}^*$. Thus $|\{\vec{a}^{(\boldsymbol{\gamma})} - \vec{\alpha}^* \mid \boldsymbol{\gamma} \in F\}| = |F|$. Note, $\{\vec{a}^{(\boldsymbol{\gamma})} - \vec{\alpha}^* \mid \boldsymbol{\gamma} \in F\} \subseteq \{\vec{\alpha} - \vec{\alpha}^* \mid \vec{\alpha} \in \mathcal{A}\}$, which implies $|\{\vec{\alpha} - \vec{\alpha}^* \mid \vec{\alpha} \in \mathcal{A}\}| \geq |F|$. Therefore $|\mathcal{T}| = |h^{-1}(\{\vec{\alpha} - \vec{\alpha}^* \mid \vec{\alpha} \in \mathcal{A}\})| \geq |F|$. However, $\mathcal{T} \in F$, so $\mathcal{T} = F$.

Then, for an integer $\tilde{k}$, we have $\mathrm{RS}_{\tilde{k}}(\tilde{F}, \mathcal{T}) = \mathrm{RS}_{\tilde{k}}(F, F)$. Let $\tilde{k} = |F|\left(1 - \frac{1}{|B|}\right)$. Notice, by Theorem 4.2, since $\tilde{k} \leq |F|\left(1 - \frac{1}{|B|}\right)$, there are repair polynomials for $\mathrm{RS}_{\tilde{k}}(F, F)$ over B with respect to any $\boldsymbol{\gamma}^* \in F$ with total dimension $(|F| - 1)$. Specifically there are repair polynomials P with respect to 0. Note, by the definition of $h(x)$ above, $\deg(h(x)) = 1$. Thus,

$$k \leq \frac{|F|}{1}\left(1 - \frac{1}{|B|}\right) = \frac{|F|}{\deg(h)}\left(1 - \frac{1}{|B|}\right) = \frac{|F|\left(1 - \frac{1}{|B|}\right)}{\deg(h)} = \frac{\tilde{k}}{\deg(h)}\,.$$

Therefore, by Theorem 5.2, there is an exact repair scheme for $C_k(F, \mathcal{A})$ with respect to $\vec{\alpha}^*$ with repair bit-bandwidth $(|F| - 1)\log_2(|B|)$, as desired.

□

Specifically, for any $\vec{\alpha}^*$ in the finite field F and any subfield $B \leq F$, the above corollary implies if $k \leq |F|\left(1 - \frac{1}{|B|}\right)$ there is an exact repair scheme for $C_k(F, F^m)$ over B with respect to $\vec{\alpha}^*$ with repair bit-bandwidth $(|F| - 1)\log_2(|B|)$.

Notice, this means with the same restriction on k, there is an exact repair scheme for $C_k(F, F^m)$ with the exact same repair bit-bandwidth as $\mathrm{RS}_k(F, F)$, mainly $(|F| - 1)\log_2(|F|)$. Thus, no matter how large we make a Cartesian code, we can repair an erasure with this low bandwidth. Further, in a certain sense, the dimension of these allowable Cartesian codes is just as reasonable as the dimension of these allowable Reed-Solomon codes as they have the same restriction on k.

We will now discuss another method for repairing Cartesian codes that constructs higher bandwidth schemes with generally larger dimensions.

5.2 Pure Multivariate Approach

In this approach, we will construct repair polynomials for Cartesian codes by modifying repair polynomials for Reed-Solomon codes. The main idea is to take univariate repair polynomials for a Reed-Solomon code, and feed them a multivariate function to construct multivariate repair polynomials for a Cartesian code. We start with the following remark and lemma.

Remark 5.4 *Throughout this section, the notation $g(\mathcal{T} - \boldsymbol{\gamma}^*)$ refers to the set $\{g(\vec{x} - \boldsymbol{\gamma}^*) \mid \vec{x} \in \mathcal{T}\}$.*

Lemma 5.1 *Let F be a finite field with a subfield $B \leq F$ such that $t := [F : B]$. Further let $\mathcal{T}$ be an evaluation set containing $\vec{0}$ such that $\mathcal{T} = T_1 \times \cdots \times T_m \subseteq F^m$ for some positive integer m. Define $n_j = |T_j|$ for all $j \in \{1, \ldots, m\}$ and $N = |\mathcal{T}|$. Let k be a positive integer and pick an $\boldsymbol{\gamma}^* = (\boldsymbol{\gamma}_1^*, \ldots \boldsymbol{\gamma}_m^*) \in \mathcal{T}$.*

Assume there exists the following.

(i) *A polynomial $g : F^m \to F$.*

(ii) *An integer $k < |g(\mathcal{T} - \boldsymbol{\gamma}^*)|$ and a polynomial $p \in F[x]$ with $\deg(p) < |g(\mathcal{T} - \boldsymbol{\gamma}^*)| - k$. Further let Γ^* be an element of $\mathcal{T}$.*

Let $\tilde{k}$ be a positive integer such that $\tilde{k} \leq \sum_{i=1}^{m}(n_j - 1) + 1 - \deg(g)(|g(\mathcal{T} - \boldsymbol{\gamma}^*)| - k)$.

Then, the polynomial $\tilde{p}(\vec{x}) = p(g(\vec{x} - \boldsymbol{\gamma}^*) + \Gamma^*)$ has degree less than $\tilde{k}' = \sum_{i=1}^{m}(n_j - 1) - \tilde{k} + 1$.

Proof. To start, consider $\deg(\tilde{p})$.

$$\deg(\tilde{p}(\vec{x})) = \deg(p(g(\vec{x} - \boldsymbol{\gamma}^*) + \Gamma^*)) = \deg(p_i)\deg(g) \ .$$

Note, $\deg(p) < |g(\mathcal{T} - \boldsymbol{\gamma}^*)| - k$, so

$$\deg(p)\deg(g) < (|g(\mathcal{T} - \boldsymbol{\gamma}^*)| - k)\deg(g) \ .$$

Then, we recall the statement in the lemma,
$\tilde{k} \leq \sum_{i=1}^{m}(n_j - 1) + 1 - \deg(g)(|g(\mathcal{T} - \boldsymbol{\gamma}^*)| - k)$. Hence, subtracting from both sides yields,

$$\deg(g)(|g(\mathcal{T} - \boldsymbol{\gamma}^*)| - k) \leq \sum_{i=1}^{m}(n_j - 1) + 1 - \tilde{k} \ .$$

Therefore,

$$\deg(\tilde{p}(\vec{x})) < \sum_{i=1}^{m}(n_j - 1) + 1 - \tilde{k} \ .$$

as desired. □

The above lemma shows that if we compose repair polynomials for Reed-Solomon codes with a special polynomial g, the result will be a new set of polynomials with manageable degrees. These degrees will need to be carefully considered in order for the constructed polynomials to meet the definition of repair polynomials for Cartesian codes.

We next show that these constructed polynomials can form a basis for F over B when evaluated at a particular point. Along with having appropriate degrees, this observation is key to proving the constructed polynomials are repair polynomials.

Lemma 5.2 *Let F be a finite field with a subfield $B \leq F$ such that $t := [F : B]$. Further let $\mathcal{T}$ be an evaluation set containing $\vec{0}$ such that $\mathcal{T} = T_1 \times \cdots \times T_m \subseteq F^m$ for some positive integer m. Define $n_j = |T_j|$ for all $j \in \{1, \ldots, m\}$ and $N = |\mathcal{T}|$. Let k be a positive integer and pick an $\boldsymbol{\gamma}^* = (\boldsymbol{\gamma}_1^*, \ldots \boldsymbol{\gamma}_m^*) \in \mathcal{T}$.*

Assume there exists the following.

(i) *A polynomial $g : F^m \to F$ such that the only solution to $g(\vec{x}) = 0$ is $\vec{0}$.*

(ii) *An element $\Gamma^* \in F$ and a set of polynomials $P = \{p_1, \ldots, p_t\} \subseteq F[x]$.*

Then, the set of polynomials $\tilde{P} = \{\tilde{p}_1, \ldots, \tilde{p}_t\} \subseteq F[x_1, \ldots, x_m]$ with $\tilde{p}_i(\vec{x}) = p_i(g(\vec{x} - \boldsymbol{\gamma}^*) + \Gamma^*)$ for all $i \in \{1, \ldots, t\}$ are such that $\tilde{p}_i(\gamma^*) = p_i(\Gamma^*)$ for all $i \in \{1, \ldots, t\}$.

Proof. Firstly, for any $i \in \{1, \ldots, t\}$ consider $\tilde{p}_i(\boldsymbol{\gamma}^*)$.

$$\tilde{p}_i(\boldsymbol{\gamma}^*) = p_i(g(\boldsymbol{\gamma}^* - \boldsymbol{\gamma}^*) + \Gamma^*) = p_i(g(\vec{0}) + \Gamma^*) \ .$$

Since $\vec{0}$ is a solution to $g(\vec{x}) = 0$, we know $g(\vec{0}) = 0$. Thus,

$$p_i(g(\vec{0}) + \Gamma^*) = p_i(\Gamma^*) \ .$$

Thus,

$$\tilde{p}_i(\boldsymbol{\gamma}^*) = p_i(\Gamma^*) \ ,$$

as desired. □

The above two lemmas will be enough to show the constructed polynomials are repair polynomials. We next look into the total dimension of these polynomials, which is important for the repair bit-bandwidth of any scheme derived from them.

Lemma 5.3 *Let F be a finite field with a subfield $B \leq F$ such that $t := [F : B]$. Further let $\mathcal{T}$ be an evaluation set containing $\vec{0}$ such that $\mathcal{T} = T_1 \times \cdots \times T_m \subseteq F^m$ for some positive integer m. Define $n_j = |T_j|$ for all $j \in \{1, \ldots, m\}$ and $N = |\mathcal{T}|$. Let k be a positive integer and pick an $\boldsymbol{\gamma}^* = (\boldsymbol{\gamma}_1^*, \ldots \boldsymbol{\gamma}_m^*) \in \mathcal{T}$.*

Assume there exists the following.

(i) *A polynomial $g : F^m \to F$ such that the only solution to $g(\vec{x}) = 0$ is $\vec{0}$.*

(ii) *An element $\Gamma^* \in F$, a set of polynomials $P = \{p_1, \ldots, p_t\} \subseteq F[x]$, and a set $\mathcal{G} = \{g(\vec{x} - \boldsymbol{\gamma}^*) + \Gamma^* \mid \vec{x} \in \mathcal{T}\} \subseteq F$.*

Define a set of polynomials $\tilde{P} = \{\tilde{p}_1, \ldots, \tilde{p}_t\} \subseteq F^m[x_1, \ldots, x_m]$ such that $\tilde{p}_i(\vec{x}) = p_i(g(\vec{x} - \boldsymbol{\gamma}^*) + \Gamma^*)$.

Then,

$$\sum_{\boldsymbol{\gamma} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*} \dim_B \{\tilde{p}_i(\boldsymbol{\gamma}) \mid i \in \{1, \ldots, t\}\} \leq (N-1)d \ ,$$

where,

$$d \coloneqq \max_{\Gamma \in \mathcal{G} \setminus \Gamma^*} \dim_B \{p_i(\Gamma) \mid i \in \{1, \ldots, t\}\} \ .$$

Proof. Firstly, for all $\boldsymbol{\gamma} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*$ we will show $g(\boldsymbol{\gamma} - \boldsymbol{\gamma}^*) + \Gamma^* \neq \Gamma^*$. Fix a $\boldsymbol{\gamma} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*$. Let

$$\Gamma = g(\boldsymbol{\gamma} - \boldsymbol{\gamma}^*) + \Gamma^* \ .$$

Suppose for sake of contradiction, $\Gamma^* = \Gamma$. Then,

$$\Gamma^* = g(\boldsymbol{\gamma} - \boldsymbol{\gamma}^*) + \Gamma^* \ .$$

That implies,

$$0 = g(\boldsymbol{\gamma} - \boldsymbol{\gamma}^*) \ .$$

Because $\vec{0}$ is the only solution to $g(\vec{x}) = 0$, that would imply $\boldsymbol{\gamma} = \boldsymbol{\gamma}^*$. However we specifically chose $\boldsymbol{\gamma} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*$, so $\boldsymbol{\gamma} \neq \boldsymbol{\gamma}^*$. Therefore,

$$\Gamma \neq \Gamma^* \ .$$

Also, notice $\Gamma \in \mathcal{G}$, since,

$$\Gamma = g(\boldsymbol{\gamma} - \boldsymbol{\gamma}^*) + \Gamma^* \in \{g(\vec{x} - \boldsymbol{\gamma}^*) + \Gamma^* \mid \vec{x} \in \mathcal{T}\} = \mathcal{G} \ .$$

Then, consider $\{\tilde{p}_i(\boldsymbol{\gamma}) \mid \text{ for all } i \in \{1,\dots,t\}\}$. Note,

$$\begin{aligned}&\{\tilde{p}_i(\boldsymbol{\gamma}) \mid \text{ for all } i \in \{1,\dots,t\}\}\\ &= \{p_i(g(\boldsymbol{\gamma}-\boldsymbol{\gamma}^*)+\Gamma^*) \mid \text{ for all } i \in \{1,\dots,t\}\}\\ &= \{p_i(\Gamma) \mid \text{ for all } i \in \{1,\dots,t\}\}\ .\end{aligned}$$

Thus,

$$\dim_B\{\tilde{p}_i(\boldsymbol{\gamma}) \mid \text{ for all } i \in \{1,\dots,t\}\} = \dim_B\{p_i(\Gamma) \mid \text{ for all } i \in \{1,\dots,t\}\}\ .$$

Certainly, since $\Gamma \in \mathcal{G} \setminus \Gamma^*$,

$$\dim_B\{p_i(\Gamma) \mid \text{ for all } i \in \{1,\dots,t\}\} \leq d$$

as defined above. Therefore,

$$\dim_B\{\tilde{p}_i(\boldsymbol{\gamma}) \mid \text{ for all } i \in \{1,\dots,t\}\} \leq d\ .$$

Since, $\boldsymbol{\gamma}$ was an arbitrary element of $\mathcal{T} \setminus \boldsymbol{\gamma}^*$,

$$\dim_B\{\tilde{p}_i(\boldsymbol{\gamma}) \mid \text{ for all } i \in \{1,\dots,t\}\} \leq d$$

for all $\boldsymbol{\gamma} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*$. This implies,

$$\sum_{\boldsymbol{\gamma}\in\mathcal{T}\setminus\boldsymbol{\gamma}^*} \dim_B\{\tilde{p}_i(\boldsymbol{\gamma}) \mid \text{ for all } i \in \{1,\dots,t\}\} \leq \sum_{\boldsymbol{\gamma}\in\mathcal{T}\setminus\boldsymbol{\gamma}^*} d = (N-1)d\ ,$$

as desired.

□

Finally, putting it all together we have the following theorem.

Theorem 5.3 *Let F be a finite field with a subfield $B \leq F$ such that $t := [F : B]$. Further let $\mathcal{T}$ be an evaluation set containing $\vec{0}$ such that $\mathcal{T} = T_1 \times \cdots \times T_m \subseteq F^m$ for some positive integer m. Define $n_j = |T_j|$ for all $j \in \{1,\dots,m\}$ and $N = |\mathcal{T}|$. Let k be a positive integer and pick an $\boldsymbol{\gamma}^* = (\boldsymbol{\gamma}_1^*, \dots \boldsymbol{\gamma}_m^*) \in \mathcal{T}$.*

Assume there exists the following.

(i) *A polynomial $g : F^m \to F$ such that the only solution to $g(\vec{x}) = 0$ is $\vec{0}$.*

(ii) *An element $\Gamma^* \in F$ and a set $\mathcal{G} = \{g(\vec{x} - \boldsymbol{\gamma}^*) + \Gamma^* \mid \vec{x} \in \mathcal{T}\}$ with a set of repair polynomials $P = \{p_1, \ldots, p_t\}$ for $RS_k(F, \mathcal{G})$ with respect to Γ^*, where $k < |\mathcal{G}|$*

Let $\tilde{k}$ be a positive integer such that $\tilde{k} \leq \sum_{i=1}^{m}(n_j - 1) + 1 - \deg(g)(|g(\mathcal{T} - \boldsymbol{\gamma}^*)| - k)$.

Then, the set of polynomials $\tilde{P} = \{\tilde{p}_1, \ldots, \tilde{p}_t\} \subseteq F^m[x_1, \ldots, x_m]$ such that $\tilde{p}_i(\vec{x}) = p_i(g(\vec{x} - \boldsymbol{\gamma}^*) + \Gamma^*)$ are repair polynomials for $C_{\tilde{k}}(F, \mathcal{T})$ with total dimension less than $(N-1)d$, where

$$d \coloneqq \max_{\Gamma \in \mathcal{G} \setminus \Gamma *} \dim_B\{p_i(\Gamma) \mid i \in \{1, \ldots, t\}\} \ .$$

Proof. By Definition 4.2, in order to prove $\{\tilde{p}_1, \ldots, \tilde{p}_t\}$ are repair polynomials for $C_{\tilde{k}}(F, \mathcal{T})$ we need to show (i) $\deg(\tilde{p}_i) < \tilde{k}' = \sum_{j=1}^{m}(n_j - 1) - \tilde{k} + 1$ and (ii) $\{\tilde{p}_1(\boldsymbol{\gamma}^*), \ldots, \tilde{p}_t(\boldsymbol{\gamma}^*)\}$ forms a basis for F over B.

We will first prove (i). Fix $i \in \{1, \ldots, t\}$. Note, since p_i is in a set of repair polynomials for $\mathrm{RS}_k(F, \mathcal{G})$, we know $\deg(p_i) < |\mathcal{G}| - k$ .However,

$$\begin{aligned} |\mathcal{G}| &= |\{g(\vec{x} - \boldsymbol{\gamma}^*) + \Gamma^* \mid \vec{x} \in \mathcal{T}\}| \\ &= |\{g(\vec{x} - \boldsymbol{\gamma}^*) \mid \vec{x} \in \mathcal{T}\}| \\ &= |g(\mathcal{T} - \boldsymbol{\gamma}^*)| \ , \end{aligned}$$

so,

$$\deg(p_i) < |g(\mathcal{T} - \boldsymbol{\gamma}^*)| - k \ .$$

Then, by Lemma 5.1, since $g \in F[x_1, \ldots, x_m]$ and $\deg(p_i) < |g(\mathcal{T} - \boldsymbol{\gamma}^*)| - k$, we can conclude,

$$\deg(\tilde{p}_i(\vec{x})) < \sum_{i=1}^{m}(n_j - 1) + 1 - \tilde{k} \ ,$$

as desired.

Now to prove (ii). For any $i \in \{1, \ldots, m\}$ consider $\tilde{p}_i(\boldsymbol{\gamma}^*)$.
By Lemma 5.2, since $g \in F[x_1, \ldots, x_m]$ with $g(\vec{0}) = 0$, and $P \subseteq F[x]$,

$$\tilde{p}_i(\boldsymbol{\gamma}^*) = p_i(\Gamma^*) \ .$$

Therefore,

$$\{\tilde{p}_1(\boldsymbol{\gamma}^*), \ldots, \tilde{p}_t(\boldsymbol{\gamma}^*)\} = \{p_1(\Gamma^*), \ldots, p_t(\Gamma^*)\} \ .$$

Then, because P are repair polynomials for $\mathrm{RS}_k(F, \mathcal{G})$ over B with respect to Γ^*, by definition $\{p_1(\Gamma^*), \ldots, p_t(\Gamma^*)\}$ forms a basis for F over B. Thus,

$$\{\tilde{p}_1(\boldsymbol{\gamma}^*), \ldots, \tilde{p}_t(\boldsymbol{\gamma}^*)\}$$

forms a basis for F over B, as desired.

With (i) and (ii) proved we have shown that $\tilde{P} = \{\tilde{p}_1, \ldots, \tilde{p}_t\}$ are repair polynomials for $C_{\tilde{k}}(F, \mathcal{T})$ with respect to $\boldsymbol{\gamma}^*$.

Last, we need to show the total dimension of $\{\tilde{p}_1, \ldots, \tilde{p}_t\}$ is less than $N(d-1)$. Recall that the total dimension of the repair polynomials $\{\tilde{p}_1, \ldots, \tilde{p}_t\}$ is $\sum_{\boldsymbol{\gamma} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*} \tilde{\delta}_{\boldsymbol{\gamma}}$, where

$$\tilde{\delta}_{\boldsymbol{\gamma}} = \dim_B\{\tilde{p}_i(\boldsymbol{\gamma}) \mid \text{ for all } i \in \{1, \ldots, t\}\} .$$

With P, $\tilde{P}$, and d, defined as they are above, by Lemma 5.3,

$$\sum_{\boldsymbol{\gamma} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*} \dim_B\{\tilde{p}_i(\boldsymbol{\gamma}) \mid \text{ for all } i \in \{1, \ldots, t\}\} \leq \sum_{\boldsymbol{\gamma} \in \mathcal{T} \setminus \boldsymbol{\gamma}^*} d = (N-1)d$$

as desired.

In total, $\{\tilde{p}_1, \ldots, \tilde{p}_t\}$ are a set of repair polynomials for $C_{\tilde{k}}(F, \mathcal{T})$ with total dimension less than $(N-1)d$.

□

Of course, we care about constructing repair polynomials, so we can use them in a repair scheme.

Corollary 5.3 *Let F be a finite field with a subfield $B \leq F$ such that $t := [F : B]$. Further let $\mathcal{T}$ be an evaluation set containing $\vec{0}$ such that $\mathcal{T} = T_1 \times \cdots \times T_m \subseteq F^m$ for some positive integer m. Define $n_j = |T_j|$ for all $j \in \{1, \ldots, m\}$ and $N = |\mathcal{T}|$. Let k be a positive integer and pick an $\boldsymbol{\gamma}^* = (\boldsymbol{\gamma}_1^*, \ldots \boldsymbol{\gamma}_m^*) \in \mathcal{T}$.*

Assume there exists the following.

(i) *A polynomial $g : F^m \to F$ such that the only solution to $g(\vec{x}) = 0$ is $\vec{0}$.*

(ii) *An element $\Gamma^* \in F$ and a set $\mathcal{G} = \{g(\vec{x} - \boldsymbol{\gamma}^*) + \Gamma^* \mid \vec{x} \in \mathcal{T}\}$ with a set of repair polynomials $P = \{p_1, \ldots, p_t\}$ for $RS_k(F, \mathcal{G})$ with respect to Γ^*, where $k < |\mathcal{G}|$*

Let $\tilde{k}$ be a positive integer such that $\tilde{k} \leq \sum_{i=1}^{m}(n_j - 1) + 1 - \deg(g)(|g(\mathcal{T} - \boldsymbol{\gamma}^*)| - k)$.

Then, there is a repair scheme for $C_{\tilde{k}}(F, \mathcal{T})$ over B with respect to $\boldsymbol{\gamma}^*$ with upper-bit bandwidth $(N-1)d\log_2(|B|)$, where,

$$d \coloneqq \max_{\Gamma \in \mathcal{G} \setminus \Gamma *} \dim_B\{p_i(\Gamma) \mid i \in \{1, \ldots, t\}\} .$$

Proof. For every $i \in \{1, \ldots, t\}$ define $\tilde{p}_i(\vec{x}) = p_i(\vec{x} - \boldsymbol{\gamma}^*) + \Gamma^*$. Then, by Theorem 5.3, $\tilde{P} = \{\tilde{p}_1, \ldots, \tilde{p}_t\}$ is a set of repair polynomials for $C_{\tilde{k}}(F, \mathcal{T})$ over B with respect to $\boldsymbol{\gamma}^*$ with total dimension less than $(N-1)d$. Then, by Corollary 5.1, there is a repair scheme for $C_{\tilde{k}}(F, \mathcal{T}\}$ with respect to $\boldsymbol{\gamma}^*$ with repair bit-bandwidth $(N-1)d\log_2(|B|)$.

□

We will now look at particular applications of the above corollary, to show the power of this method. Firstly, let's consider a Cartesian code where the evaluation set is F^m.

Corollary 5.4 *Let F be a finite field with a subfield $B \leq F$ such that $t \coloneqq [F : B]$. Consider the Cartesian code $C_{\tilde{k}}(F, F^m)$ such that $\tilde{k} \leq |F|(m - \frac{\deg(g)}{|B|}) - m + 1$. Define $N \coloneqq |F|^m$ and assume $g(\vec{x})$ is a polynomial in $F[x_1, \ldots, x_m]$ such that the only solution to $g(\vec{x}) = 0$ is $\vec{0}$. Then, for any $\boldsymbol{\gamma}^* \in F^m$ there is an exact repair scheme for $C_{\tilde{k}}(F, F^m)$ over B with respect to $\boldsymbol{\gamma}^*$ with repair bit-bandwidth $(N-1)\log_2(|B|)$.*

Proof. We will use Corollary 5.3 to show the desired repair scheme exists. Notice, for the Cartesian code $C_{\tilde{k}}(F, F^m)$ the evaluation set is $\mathcal{T} = F^m$. Thus, $n_j = |F|$ for all $j \in \{1, \ldots, m\}$.

First, fix an element $\boldsymbol{\gamma}^* \in \mathcal{T}$. To use Corollary 5.3, we need to show the following exist.

(i) A polynomial $g : F^m \to F$ such that the only solution to $g(\vec{x}) = 0$ is $\vec{0}$.

(ii) An element $\Gamma^* \in F$ and a set $\mathcal{G} = \{g(\vec{x} - \boldsymbol{\gamma}^*) + \Gamma^* \mid \vec{x} \in \mathcal{T}\}$ with a set of repair polynomials $P = \{p_1, \ldots, p_t\}$ for $\mathrm{RS}_k(F, \mathcal{G})$ with respect to Γ^*, where $k < |\mathcal{G}|$.

Notice, the polynomial $g(\vec{x})$ given in the corollary statement satisfies (i).

Now onto (ii). Fix an arbitrary $\Gamma^* \in F$. Then let $\mathcal{G} = \{g(\vec{x} - \boldsymbol{\gamma}^*) + \Gamma^* \mid \vec{x} \in F^m\} \subseteq F$. By Observation 4.2, since $k \coloneqq |\mathcal{G}| - \frac{|F|}{|B|} \leq |\mathcal{G}| - \frac{|F|}{|B|}$ there are repair polynomials $P = \{p_1, \ldots, p_t\}$ for $\mathrm{RS}_k(F, \mathcal{G})$ over B with respect to Γ^*.
Also from Observation 4.2, it is known for any $\Gamma \in \mathcal{G}$, the Γ-dimension of P is $\delta_\Gamma = \dim_B\{p_i(\gamma) \mid \text{for all } i \in \{1, \ldots, t\}\} = 1$. Hence,

$$d = \max_{\Gamma \in \mathcal{G} \setminus \Gamma^*} \dim_B\{p_i(\gamma) \mid \text{for all } i \in \{1, \ldots, t\}\} = 1 \ .$$

Further, recall $\tilde{k} \leq |F|(m - \frac{\deg(g)}{|B|}) - m + 1$. Then,

$$\begin{aligned}
\tilde{k} &\leq |F| \left(m - \frac{\deg(g)}{|B|} \right) - m + 1 \\
&= m(|F| - 1) + 1 - \deg(g) \frac{|F|}{|B|} \\
&= \sum_{j=1}^{m} (|F| - 1) + 1 - \deg(g) \left(|\mathcal{G}| - |\mathcal{G}| + \frac{|F|}{|B|} \right) \\
&= \sum_{j=1}^{m} (n_j - 1) + 1 - \deg(g) \left(|\mathcal{G}| - \left(|\mathcal{G}| - \frac{|F|}{|B|} \right) \right) \\
&= \sum_{j=1}^{m} (n_j - 1) + 1 - \deg(g)(|\mathcal{G}| - k) \ .
\end{aligned}$$

Thus, by Corollary 5.3, there is a repair scheme for $C_{\tilde{k}}(F, F^m)$ with respect to $\boldsymbol{\gamma}^*$ with repair bit-bandwidth $(N-1)d \log_2(|B|) = (N-1) \log_2(|B|)$.

□

Notice, this above exact repair scheme for $C_k(F, F^m)$ over B has a repair bit-bandwidth of $(|F|^m - 1) \log_2(|B|)$ and requires $\tilde{k} \leq |F| \left(m - \frac{\deg(g)}{|B|} \right) - m + 1$. Compare this to the Reed-Solomon case, where the exact repair scheme for $\mathrm{RS}_k(F, F)$ in Observation 4.2 has a repair bit-bandwidth of $(|F| - 1) \log_2(|B|)$ and requires $k \leq |F| \left(1 - \frac{1}{|B|} \right)$. This new scheme has the exact same repair bit-bandwidth as the Reed-Solomon repair scheme, relative to the size of the evaluation set. Further, it appears $\tilde{k}$ can be approximately m times larger k. However, notice the size of the evaluation set in the Cartesian case is $|F|^m$, much larger than $|F|$, the size of the evaluation set in the Reed-Solomon case. Meaning, the repair bit-bandwidth in the Cartesian case is approximately the repair bit-bandwidth in the Reed-Solomon case raised to the *mth* power. A significant decrease in the bandwidth efficiency.

We can also compare this method to the method of "Extracting Reed-Solomon Codes" presented above. Recall, using the "Extracting" method, given an integer $k \leq |F| \left(1 - \frac{1}{|B|} \right)$, we can construct an exact repair scheme for $C_k(F, F^m)$ with respect to $\boldsymbol{\gamma}^*$ with repair bit-bandwidth $(|F| - 1) \log_2(|B|)$. This method creates schemes with very low repair bit-bandwidths, the same repair bit-bandwidth as the Reed-Solomon case in fact. However, it restricts k by $|F| \left(1 - \frac{1}{|B|} \right)$, compared to the method above which restricts $\tilde{k}$ by $|F| \left(m - \frac{\deg(g)}{|B|} \right) - m + 1$. Hence, the "Pure Multivariate" method has a much higher repair-bit bandwidth, being about the repair bit-bandwidth of the "Extracting

method" raised to the mth power. However, it also admits much larger degrees than the "Extracting" method, by about a factor of m. In practice, any implementer needs to use the method that gives them the right balance between bandwidth and dimension for their system.

Let us end our discussion by presenting a real example. In order to actually apply any of these corollaries, we need to find polynomials $g(\vec{x}) \in F[x_1, \ldots, x_m]$ which have exactly one solution, $\vec{0}$. From the literature we have reviewed, it does not appear that finding these polynomials is a well-studied problem. Although, after using brute force techniques to discover these polynomials, it seems that they are abundant. At the moment, we have constrained our search to polynomials in two variables. We are confident enough in our finding to make the following conjecture.

Conjecture 5.1 *Let F be a finite field. Then, there exists a polynomial $g(\vec{x}) \in F[x_1, x_2]$ with $\deg(g) = 2$ such that the only solution to $g(\vec{x}) = 0$ is $\vec{0}$.*

We have currently found many specific g polynomials for finite fields F with size $|F| = 2^2, 2^3, \ldots, 2^{10}, 3^2, 3^3, 5^2, 5^3, 5^4$, and more. we will record a the specific polynomial for $\mathrm{GF}(5^3)$ below.

Observation 5.1 *Let F be a finite field of order $5^3 = 125$. Then, the degree 2 polynomial $g(x_1, x_2) = x_1^2 + x_1x_2 + x_2^2$ is computationally shown to have exactly one solution, $\vec{0}$.*

This observation leads us to the following result.

Example 5.1 *Let F be a finite field of order 125. Let $\tilde{k} \leq 199$. Then, for any $\boldsymbol{\gamma}^* \in F^2$, there is an exact repair scheme for $C_{\tilde{k}}$ with respect to $\boldsymbol{\gamma}^*$ with a repair bit-bandwidth of $15624 \log_2(5)$ bits.*

Proof. We will show the above example is true using Corollary 5.4. To start, we have F is a finite field of order $125 = 5^3$. Let $B \leq F$ be a subfield of F with order 5. Consider the Cartesian code $C_{\tilde{k}}(F, F^2)$. Note, here $m = 2$ and $N = |F|^2 = 125^2 = 15625$.

Also note, by the above observation, the polynomial $g(x_1, x_2) = x_1^2 + x_1x_2 + x_2^2$ in $F[x_1, x_2]$ has exactly one solution, $\vec{0}$.

Recall $\tilde{k} \leq 199$. Note,

$$\begin{aligned}
|F|\left(m - \frac{\deg(g)}{|B|}\right) - m + 1 &= |F|\left(2 - \frac{2}{|B|}\right) - 2 + 1 \\
&= 2|F|\left(1 - \frac{1}{|B|}\right) - 1 \\
&= 2(125)\left(1 - \frac{1}{5}\right) - 1 \\
&= 250\left(\frac{4}{5}\right) - 1 \\
&= 50(4) - 1 \\
&= 199\ .
\end{aligned}$$

Thus, $\tilde{k} \leq |F|\left(m - \frac{\deg(g)}{|B|}\right) - m + 1$.

Hence, by Corollary 5.4, for any $\boldsymbol{\gamma}^* \in F^2$, there is a repair scheme for $C_{\tilde{k}}(F, F^2)$ with respect to $\boldsymbol{\gamma}^*$ with repair bit-bandwidth of
$(N-1)\log_2(|B|) = (15625-1)\log_2(5) = 15624\log_2(5)$, as desired.

□

Notice, any finite field F that admits a polynomial $g(\vec{x}) \in F[x_1, \ldots, x_m]$ such that the only solution for $g(\vec{x}) = 0$ is $\vec{0}$ has an analogous exact repair scheme to the one presented in the above example. This is a very real way to construct exact repair schemes for Cartesian codes that have comparable dimensions and repair bit-bandwidths to previously constructed exact repair schemes for Reed-Solomon codes.

6 Conclusion

This research has been a mere introduction to the much wider and deeper study of finding exact repair schemes for Cartesian codes. We started with the foundational research done by Guruswarmi and Wooters in 2017 [1] on finding exact repair schemes for Reed-Solomon codes. Their research showed that certain codes, like Reed-Solomon codes, with traditionally high-bandwidth recovery schemes could be intelligently adapted to admit low-bandwidth exact repair schemes with manageable dimension. In fact, for Reed-Solomon codes with an evaluation set matching the whole field, they found an exact repair scheme with optimal bandwidth [1]. These results for Reed-Solomon codes provided a proof of concept that low-bandwidth exact repair schemes could be developed for other evaluation codes. We concerned ourselves specifically with Cartesian codes. Firstly, we showed that the general linear exact repair scheme algorithm presented in 2017 by Guruswarmi and Wooters [1] could be directly generalized for Cartesian codes. Then, our research shifted to focusing on finding these repair polynomials that make the linear exact repair scheme algorithm effective. We have shown two methods for doing just that, both closely tied to Reed-Solomon codes.

The first, "Extracting Reed-Solomon Codes" entails finding univariate polynomials that produce the same codewords as multivariate polynomials used by a Cartesian codes. Then, we can consider the Reed-Solomon code that uses the univariate polynomial, and use those repairing properties on our original Cartesian Code.

The next approach, the "Pure Multivariate Approach", involved finding legitimate repair polynomials for our Cartesian codes, via repair polynomials for Reed-Solomon codes. This method relies heavily on finding particular polynomials that have only one solution. We showed these polynomials are relatively abundant, and can be used to create low-bandwidth repair schemes for Cartesian codes of the form $C_k(F, F^m)$ with reasonable dimension.

Going forward, there is another hybrid approach which connects the two previous methods we left undiscussed. Like the "Extracting Reed-Solomon Codes" approach, given a Cartesian code that uses multivariate polynomials in m variables, we can instead look at multivariate polynomials in m' variables that produce the same codewords. Then, using the "Pure Multivariate Approach" on the Cartesian code using the polynomials in m' variables, we can repair failures, and apply those results to the original Cartesian code.

Beyond this hybrid approach, there is much immediate research left to be done on this topic. First and foremost, doing deeper research into these special polynomials with

exactly one solution. From our programmatic approach, we found many such polynomials for every finite field tested, and we even found patterns in those polynomials. However, we currently do not know how to construct these polynomials, or determine if a given polynomial works, without actually evaluating it at every point in the field. We also look forward to trying to show our schemes are optimal. There are bounds for the amount of bandwidth required to repair Reed-Solomon codes, but those rely on Reed-Solomon codes being maximum distance separable codes. For Cartesian codes, it is less clear what the optimal bandwidth for a repair scheme is without more careful study. Even if we cannot find an optimal bandwidth for repair schemes over a Cartesian code, it would still be useful and relevant to determine some metric by which to gauge the bandwidth-efficiency of an exact repair scheme over a Cartesian code. Also of course, our paper only looks into repairing one erasure in a Cartesian code. A natural next step is to try to adapt these methods to fix multiple erasures. Finally, we also want to research other evaluation codes, slightly different from Cartesian codes. Mainly, we should notice Cartesian codes use multivariate polynomials that have a *total degree* bounded by some integer k. We would like to study codes that use multivariate polynomials with more complex conditions on their degrees. For example, codes that use polynomials where the max degree in every variable is less than k. There is a whole class of codes to unpack, as well as a whole class of ideas, meaning there is much research left to be done.

References

[1] Guruswami, V., & Wootters, M. (2017). Repairing reed-solomon codes. IEEE transactions on Information Theory, 63(9), 5684-5698.

[2] López, H. H., Manganiello, F., & Matthews, G. L. (2019). Affine Cartesian codes with complementary duals. Finite Fields and Their Applications, 57, 13-28.

[3] Tamo, I., Ye, M., & Barg, A. (2018). The repair problem for Reed-Solomon codes: Optimal repair of single and multiple erasures, asymptotically optimal node size. arXiv preprint arXiv:1805.01883.

[4] Zilic, Z., & Vranesic, Z. G. (2002). A deterministic multivariate interpolation algorithm for small finite fields. IEEE Transactions on Computers, 51(9), 1100-1105.

[5] Dimakis, A. G., Godfrey, P. B., Wu, Y., Wainwright, M. J., & Ramchandran, K. (2010). Network coding for distributed storage systems. IEEE transactions on information theory, 56(9), 4539-4551.