Abstract—In distributed storage systems that employ erasure coding, the issue of minimizing the total communication required to exactly rebuild a storage node after a failure arises. This repair bandwidth depends on the structure of the storage code and the repair strategies used to restore the lost data. Designing high-rate maximum-distance separable (MDS) codes that achieve the optimum repair communication has been a well-known open problem. In this work, we use Hadamard matrices to construct the first explicit 2-parity MDS storage code with optimal repair properties for all single node failures, including the parity nodes. Our construction relies on a novel method of achieving perfect interference alignment over finite fields with a finite file size, or number of symbol extensions.

I. INTRODUCTION

Distributed storage systems have reached a massive scale and recovery from failures is now part of regular operation rather than a rare exception [4]. Large scale deployments typically need to tolerate multiple failures, both for high availability and to prevent data loss. Erasure coded storage achieves high failure tolerance without requiring a large number of replicas that increase the storage cost [6]. Three application contexts where erasure coding techniques are being currently deployed or under investigation are Cloud storage systems [5], archival storage, and peer-to-peer storage systems like Cleversafe and Wuala.

One central problem in erasure coded distributed storage systems is that of maintaining an encoded representation when failures occur. To maintain the same level of redundancy when a storage node leaves the system, a newcomer node has to join the array, access some existing nodes, and exactly reproduce the contents of the departed node. In its most general form this problem is known as the Exact Code Repair Problem [2], [1]. Currently, the most well-understood metric that can be optimized during a repair is that of repair bandwidth, i.e., the total information communicated in the network. For designing \((n, k)\) MDS erasure codes that have \(n\) storage nodes and can tolerate any \(n - k\) failures, the information theoretic cut-set bounds for repair communication were specified in [2] and shown to be achievable for all values of \(n, k\) in a series of recent papers [3], [9]–[16]. In particular, it was shown that for an \((n, k)\) code, if a single node fails, downloading \(\frac{1}{n-k}\) fraction of every surviving disk is sufficient and optimal in terms of repair bandwidth for the repair of a failed node.

For code rates \(k/n \leq 1/2\), explicit repair optimal MDS codes were designed by Shah et al. [12], Rashmi et al. [16], and Suh et al. [11]. These schemes employ exact interference alignment (IA) techniques to minimize the repair bandwidth. For the high-rate regime, the only known complete constructions [14], [15] are asymptotic and require large file sizes (symbol extensions) and field sizes. These constructions use the symbol extension IA technique of [7] to establish the existence of MDS storage codes, that come arbitrarily close to (but do not exactly match) the information theoretic lower bound for the repair bandwidth for all \(n, k\). These constructions are impractical due to the arbitrarily large finite field and file size that they require, even for small \(n\) and \(k\). For the high-rate regime, there has recently been a substantial progress in designing explicit codes that have optimal repair properties. Tamo et al. [17] and Cadambe et al. [18], [19] designed MDS codes for any \((n, k)\) parameters that have optimal repair for the systematic nodes, but not the code parities. In fact, [19] provides a subspace-interference-alignment framework for the design of codes with optimal repair of systematic nodes. However, it is not clear whether this framework can be extended to optimal repair of the parity nodes. As a matter of fact, in current literature, there does not exist a high-rate \((k/n \geq 1/2)\) MDS code that can perform better than the trivial strategy of downloading the entire contents of \(k\) nodes to repair a single parity failure. It is this open problem that is resolved in this paper.

Our Contribution: We introduce the first explicit high-rate \((k + 2, k)\) MDS storage code with optimal repair communication for all nodes. Our storage code exploits fundamental properties of Hadamard designs and perfect IA instances that can be understood through the use of a lattice representation of the symbol extension technique of Cadambe et al. [7], [14], [15]. Our key technical contribution is a scheme that achieves perfect interference alignment with a finite number of extensions. This was first developed in [21] and used in a 2 parity code with optimal systematic repair and near optimal parity repair, that can handle any single node failure. Here, we extend the codes developed in [21] in two ways. First, the codes of [21] were not MDS codes whereas the codes of this paper are MDS codes. Second, the repair strategy of [21] for parity nodes was sub-optimal. In this paper, we construct a
novel code and repair strategy such that it is optimal for any single failed node including single parity node failures. We use a combinatorial view of the interference alignment scheme of [7] using a framework we call dots-on-a-lattice. Hadamard matrices are shown to be crucial in achieving finite perfect alignment and ensuring the full-rank of desired subspaces.

II. MDS STORAGE CODES WITH 2 PARITY NODES

In this section, we consider the code repair problem for MDS storage codes with 2 parity nodes. After we lay down the model for repair, we continue with introducing our code construction. Let a file of size $M = kN$ denoted by the vector $f \in \mathbb{F}_q^k$. Each of size $N$. We wish to store $f$ with redundancy across $k + 2$ nodes $i = 1, 2, \ldots, k + 2$. We require that the encoded storage array is resilient up to any $k$ node erasures. To satisfy the redundancy and erasure resiliency properties, the file is encoded using a $(k + 2, k)$ MDS storage code.

$$\begin{array}{|c|c|}
\hline
\text{systematic node} & \text{systematic data} \\
\hline
1 & f_1 \\
\vdots & \vdots \\
k & f_k \\
\hline
\text{parity node} & \text{parity data} \\
\hline
1 & f_1 + \ldots + f_k \\
2 & A_1^T f_1 + \ldots + A_k^T f_k \\
\hline
\end{array}$$

Fig. 1. A $(k + 2, k)$ CODED STORAGE ARRAY.

In Fig. 1 we provide a systematic representation of a generic 2-parity MDS encoded storage array. The first $k$ storage nodes store the systematic file parts. Without loss of generality, the first parity stores the sum of all $k$ systematic parts $f_1 + \ldots + f_k$ and the second parity stores an additional linear combination $A_1^T f_1 + \ldots + A_k^T f_k$. Here, $A_i$ denotes an $N \times N$ matrix of coding coefficients used by the second parity node to “scale and mix” the contents of the $i$th file piece $f_i$, $i \in [k]$, and $[k] = \{1, \ldots, k\}$.

To maintain the same level of redundancy when a storage component fails or leaves the system, the code repair process has to take place to exactly regenerate the lost data in a newcomer storage node. Let, for example, a systematic node $i \in [k]$ fail. Then, a newcomer joins the storage network, connects to the remaining nodes, and has to download a sufficient amount of data to reconstruct $f_i$.

Note that the lost systematic part $f_i$, exists only as a term of a linear combination at each parity node, as seen in Fig. 1. To regenerate the $N$ elements of $f_i$, the newcomer has to download from the parity nodes a size of data equal to the size of the lost piece, that is, $N$ linearly independent coded elements. Assuming that it downloads the same amount of data from both parities, the downloaded contents can be represented as a stack of $N$ equations

$$\begin{bmatrix}
p_1^{(1)} \\
p_1^{(2)}
\end{bmatrix} \cong \begin{bmatrix}
(V_i^{(1)})^T \\
(A_i V_i^{(2)})^T
\end{bmatrix} f_i + \sum_{i=1, i \neq s}^k \begin{bmatrix}
(V_i^{(1)})^T \\
(A_i V_i^{(2)})^T
\end{bmatrix} f_s, \tag{1}
$$

where $p_i^{(1)}, p_i^{(2)} \in \mathbb{F}_q^N$ are the equations downloaded from the first and second parity node, respectively, and $V_i^{(1)}, V_i^{(2)} \in \mathbb{F}_q^{N \times N}$ are the repair matrices. Each repair matrix is used to mix the $N$ parity contents so that a set of $N$ equations is formed. Then, retrieving $f_i$ from (1) is equivalent to solving an under-determined set of $N$ equations in the $kN$ unknowns of $f_i$ with respect to the $N$ desired unknowns of $f_s$, $s \in [k]\setminus i$, as noted in (1). These interference terms corrupt the desired data and need to be canceled. To do that, the newcomer needs to download additional data from the remaining $k - 1$ systematic nodes, that will “replicate” and cancel the interference terms from the downloaded equations.

To cancel a single interference term of (1) that has size $N$, it suffices to download a basis of equations that generates it. The dimensions of this basis does not need to be equal to $N$. For example, to erase the interference component generated by the file part $f_s$, the newcomer needs to connect to systematic node $s$ and download a number of linear equations equal to

$$\frac{N}{2} \leq \text{rank} \left[ \begin{bmatrix} (V_i^{(1)})^T \\ (A_i V_i^{(2)})^T \end{bmatrix} \right] \leq N, \tag{2}
$$

This is exactly the communication bandwidth price we are paying to delete a single interference term. The lower bound in (2) comes from the fact that $\frac{N}{2}$ linearly independent equations need to be downloaded from each of the parities, hence $\text{rank}(V_i^{(1)}) = \text{rank}(V_i^{(2)}) = \frac{N}{2}$ for any $i \in [k]$. Eventually, we need to generate all undesired terms in the newcomer, so to subtract them from (1). Then, a full rank system of $N$ equations in the $N$ unknowns has to be formed. A generic example of a code repair instance for a $(4, 2)$ storage code is given in Fig. 2.
In general, to repair a systematic node \( i \in [k] \) of an arbitrary \((k + 2, k)\) MDS storage code, we need to obtain a feasible solution to the following optimal rank minimization problem over \( \mathbb{F}_q \):

\[
\mathcal{R}_i: \min_{V_i^{(1)}, V_i^{(2)}} \sum_{s=1, s \neq i}^{k} \text{rank} \left( \left[ V_i^{(1)} A_s V_i^{(2)} \right] \right) \\
\text{s.t.:} \quad \text{rank} \left( \left[ V_i^{(1)} A_s V_i^{(2)} \right] \right) = N,
\]

where \( i \) the full rank constraints correspond to the requirement that the \( N \) equations downloaded from the parities are linearly independent, when viewed as equations in the \( N \) components of \( f_i \), and ii) the rank minimization corresponds to minimizing the sum of bases dimensions needed to cancel each interference term. For a specific feasible selection of repair matricies the repair bandwidth to exactly regenerate systematic node \( i \) is given by

\[
\gamma_i = \sum_{\text{Reequations lost}} N + \sum_{s=1, s \neq i}^{k} \text{dim of interference equations by } f_s
\]

where the sum rank term is the aggregate of interference dimensions. An optimal solution to \( \mathcal{R}_i \) is guaranteed to minimize the repair bandwidth we need to communicate to the repair systematic node \( i \in [k] \).

**Remark 1:** From [2] it is known that the theoretical minimum repair bandwidth, for any single node repair of an optimal (linear or nonlinear) \((k + 2, k)\) MDS code is exactly \( \frac{1}{2} \) of the information stored across the \( k + 1 \) remaining nodes, i.e. \((k + 1) \frac{N}{2} \), and \( N \) has to be an even number. This bound is proven using cut-set bounds on infinite flow graphs. Here, we provide an interpretation of this bound in terms of linear codes by calculating the minimum possible sum of ranks in \( \mathcal{R}_i \): since each repair matrix has to have full column rank \( \frac{N}{2} \) to be a feasible solution, the minimum number of dimensions each interference can be suppressed to is \( \frac{N}{2} \). This aggregates in a minimum repair bandwidth of \((k + 1) \frac{N}{2} \) repair equations. If we wish to achieve this bound, interference alignment has to be employed, so that undesired components are confined to the minimum number of dimensions. Interestingly, linear codes suffice to asymptotically achieve this bound [14], [15].

The difficulty in designing optimal MDS storage codes lies in a threefold requirement: i) the code has to satisfy the MDS property, ii) systematic nodes of the code have to be optimally repaired, and iii) parity nodes of the code have to be optimally repaired. Currently, there exist MDS codes for rates \( \frac{k}{k+1} \leq \frac{1}{2} \) [11], [16] for which all nodes can be optimally repaired. For the high data rate regime, Tamo et al. [17] and Cadambe et al. [19] presented the first MDS codes where any systematic node failure can be optimally repaired. However, prior to this work, there do not exist MDS storage codes of arbitrarily high rate that can optimal repair any node.

In the following, we present the first explicit, high-rate, repair optimal \((k + 2, k)\) MDS storage code that achieves the minimum repair bandwidth bound for the repair of any single systematic or parity node failure.

### III. A Repair Optimal 2 Parity Storage Code

Let a \((k + 2, k)\) MDS storage code for file size \( M = k2^{k+1} \), with coding matrices

\[
A_i = a_i X_i + b_i X_{k+1} + I_N, \quad i \in [k]
\]

where \( X_i = \text{blkdiag} \left( \frac{I_{M/2}}, -I_{M/2} \right), N = 2^{k+1}, \) and \( a_i, b_i \) satisfy \( a_i^2 - b_i^2 = -1 \), for all \( i \in [k] \).

**Theorem 1:** There exists a finite field \( \mathbb{F}_q \) of order \( q \geq 2k+3 \) and explicit constants \( a_i, b_i \in \mathbb{F}_q, \forall i \in [k] \), such that the \((k+2, k)\) storage code in (5) is a repair optimal MDS storage code.

In Fig. 3, we give the coding matrices of a \((5,3)\) MDS code over \( \mathbb{F}_{11} \) based on our construction.

**Remark 2:** The code constructions presented here have generator matrices that are as sparse as possible, since any additional sparsity would violate the MDS property. This creates the additional benefit of minimum update complexity when some bits of the stored data object change.

Before we proceed with proving Theorem 1, we state the intuition behind our code construction and the tools that we use. Motivated by the asymptotic IA schemes, we use similar concepts provided by a combinatorial explanation of interference alignment in terms of dots on lattices. In contrast to the asymptotic IA codes, here, instead of letting randomness choose the coding matrices, we select particular constructions based on Hadamard matrices that achieve exact interference alignment for fixed in \( k \) file sizes (symbol extensions). In section V we prove the optimal repair of systematic nodes, in Section VII we show the optimal repair of parity nodes, and in Section VIII we state explicit conditions for the MDS property.

### IV. Dots-on-a-lattice and Hadamard Designs

In this section, we simplify our ultimate goal of finding codes with minimal repair bandwidth defined in problem \( \mathcal{R}_i \) in Section II. On simplifying the problem at hand, we will explain our Hadamard matrix based design which lies at the heart of the code constructions described in (5). Consider (3) for the code in (5). Let us say that node \( i = 1 \) fails. To minimize

\[
\sum_{s=2}^{k} \text{rank} \left( \left[ A_s V_1 \ V_1 \right] \right)
\]

we would like to maximize the overlap (alignment) of \( V_1 \) and \( A_s V_1 \) for \( s \in [k] \setminus \{1\}, \) for the repair of node \( i = 1 \). Ideally, we would like to have all the columns of \( A_s V_1 \) to lie in the column space of \( V_1 \), so that the rank of \( A_s V_1 \) is as small as the rank of \( V_1 \). In other words, we would like \( V_1 \) to be an invariant subspace of \( A_s \), \( s \in [k] \setminus \{1\} \). Now, we make the following observation. If we find \( V_1 \) such that it is a common invariant subspace of \( X_0, X_3, \ldots, X_{k+1}, \) i.e., if

\[
\text{colspan}(V_1) = \text{colspan}(X_0 V_1) = \ldots = \text{colspan}(X_{k+1} V_1)
\]

then, \( V_1 \) would completely overlap with \( A_s V_1, s \in [k] \setminus \{1\} \) as desired. To see this, note that \( A_s V_1 = a_s X_s V_1 + \)

\[2\text{We use } -1 \text{ to denote the field element } q - 1 \text{ over } \mathbb{F}_q.\]
Indeed, this idea is central to our constructions. In this section the span of \( b s X_{k+1} V_1 + V_1 \). Therefore, every column vector of \( A_s V_1, s \neq 1 \) is the sum of three column vectors: one from the span of \( X_s V_1 \), one from the span of \( X_{k+1} V_1 \) and one from \( V_1 \) itself. If \( V_1 \) is invariant to both \( X_s \) and \( X_{k+1} \), then every column vector of \( A_s V_1 \) is a sum of three column vectors from the span of \( V_1 \) and therefore lies in the span of \( V_1 \). This means that if we somehow satisfy (6), then \( A_s V_1 \) lies in the span of \( V_1 \). Thus, from the perspective of repair of node \( i = 1 \), our goal is to find \( k \) matrices \( X_2, X_3, \ldots, X_{k+1} \) and a matrix \( V_1 \) which is invariant to the aforementioned matrices. Indeed, this idea is central to our constructions. In this section we shall pursue a simpler problem whose solution captures the main idea of our constructions. We attempt to find two matrices which we denote by \( T_1 \) and \( T_2 \) and a matrix \( V \) which is invariant to \( T_1, T_2 \).

Assume two arbitrary \( N \times N \) full rank matrices \( T_1 \) and \( T_2 \) that commute. We wish to construct a full rank matrix \( V \), with at most \( \frac{N}{2} \) columns, such that the span of \( T_1 V \) aligns as much as possible with the span of \( T_2 V \); we have to pick \( V \) such that it minimizes the dimensions of the union of the two spans, that is the rank of \( [T_1 V \ T_2 V] \). How can we construct such a matrix? Assume that we start with one vector with nonzero entries, i.e., \( V = w \), and for simplicity we let it be the all-ones vector. Then in the general case, \( T_1 w \) and \( T_2 w \) have zero intersection which is not desired. However, we can augment \( V \) such that it has as columns the elements of the set \( \{w, T_1 w, T_2 w, T_1 T_2 w\} \). This idea of augmenting the set \( V \) in this manner to increase the overlap is presented in [8]. Observe that each vector \( T_1^i T_2^j w \) of \( V \) can be represented by the power tuple \((x_1, x_2)\). This helps us visualize \( V \) as a set of dots on the 2-dimensional integer lattice as shown in Fig. 4.

For this new selection of \( V \), we have

\[
T_1 V = [T_1 w \ T_1^2 w \ T_1 T_2 w \ T_1^2 T_2 V] \quad (7)
\]

and

\[
T_2 V = [T_2 w \ T_2 T_1 w \ T_2^3 w \ T_1 T_2^2 V]. \quad (8)
\]

The intersection of the spans of these two matrices is now nonzero: the matrix \([T_1 V \ T_2 V]\) has rank 7 instead of the maximum possible of 8. This happens because the vector \( T_1 T_2 w \) is repeated in both matrices \( T_1 V \) and \( T_2 V \). In Fig. 5 we illustrate this concatenation, in terms of dots on \( \mathbb{Z}^2 \), where the intersection between the two spans is manifested as an overlap of dots.

\[
\begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1
\end{bmatrix}
\]

Fig. 4. A repair optimal (5,3) MDS code over \( \mathbb{F}_{11} \)

Remark 3: Observe how matrix multiplication of \( T_1 \) and \( T_2 \) with the vectors in \( V \) is pronounced through the dots representation: the dots representations of \( T_1 V \) and \( T_2 V \) matrices are shifted versions of \( V \) along the \( x_1 \) and \( x_2 \) axes. The key idea behind choosing a new \( V \) at each step is to iteratively augment the old one with products of the \( T_i \) matrices raised to specific powers times the current \( V \).

\[
\text{initialize: } V \leftarrow w.
\]

\[
\text{multiply with powers of } T_1: V \leftarrow [V \ T_1 V \ldots \ T_1^{m-1} V]
\]

\[
\text{multiply with powers of } T_2: V \leftarrow [V \ T_2 V \ldots \ T_2^{m-1} V] .
\]

In general, by using powers up to \( m - 1 \), with \( m^2 \leq \frac{N}{2} \), we obtain \( V \) with \( m^2 \) columns that are the elements of the set

\[
V = \{T_1^i T_2^j w : x_s \in \{0, \ldots, m - 1\}\} , \quad (9)
\]

where \( w = 1_{N \times 1} \). Then, matrix \( V \) achieves

\[
m^2 < \text{rank } ([T_1 V \ T_2 V]) \leq (m + 1)^2 . \quad (10)
\]

This means that, if we let \( N \) grow (i.e., the number of "symbol extensions") such that it can support arbitrarily large \( m \), we can asymptotically create as much alignment as we desire: for sufficiently large \( N \), \((m + 1)^2/m^2 \) goes to 1.
we give the \( m = 4 \) case in Fig. 6, where we observe that the alignment is more substantial (with respect to the size of \( \mathbf{V} \)) compared to Fig. 5. This alignment scheme, in a more general form, was presented by Cadambe and Jafar in [7] to prove the Degrees-of-Freedom of the \( K \)-user interference channel. For that wireless scenario, the \( \mathbf{T}_i \) matrices are given by nature and are i.i.d. diagonals. Perfect alignment of spaces for these matrices is not known to be possible for finite \( m \) [7], [22].

For network coding problems, and in particular, for storage coding problems, the analogous \( \mathbf{T}_i \) matrices (our coding matrices) are free to design under some specific constraints that ensure the MDS property of the code. Before, we give explicit matrices that achieve alignment in a finite number of extensions, we answer the analogous question considering our toy example: do there exist \( \mathbf{T}_1 \) and \( \mathbf{T}_2 \) matrices such that we can construct a full-rank \( \mathbf{V} \) that achieves perfect intersection (exact alignment) of the spans of \( \mathbf{T}_1 \mathbf{V} \) and \( \mathbf{T}_2 \mathbf{V} \), for some \( m \) and \( N = m^3 \)? That is, can we find matrices such that

\[
\text{span}(\mathbf{T}_1 \mathbf{V}) = \text{span}(\mathbf{T}_2 \mathbf{V}) \quad \text{and} \quad \text{rank}(\mathbf{V}) = m^2
\]

(11)
is possible? Perfect and finite symbol extension IA instances are possible when we enforce the dots representation of the \( \mathbf{V} \) matrix, or the set of power tuples that generate it, to wrap-around itself. This wrap-around property is crucial in enabling perfect alignment of spaces. We see that this property is obtained when the elements of the matrices are \( m^6 \) roots of unity, i.e.,

\[
\mathbf{T}_i^m = \mathbf{I}_N.
\]

To see that, we formally state the dots on a lattice representation. Let a map \( \mathcal{L} \) from a matrix with \( r \) columns, each generated as \( \mathbf{T}_1^x \mathbf{T}_2^z \mathbf{w} \), to a set of \( r \) points, such that the column \( \mathbf{T}_1^x \mathbf{T}_2^z \mathbf{w} \) maps to the point \((x_1, x_2)\). Then, we have for \( \mathbf{V} \)

\[
\mathcal{L}(\mathbf{V}) \triangleq \{ x_1 \mathbf{e}_1 + x_2 \mathbf{e}_2; \ x_1, x_2 \in \{0, \ldots, m - 1\} \},
\]

(13)

where \( \mathbf{e}_i \) is the \( i \)-th column of the identity matrix. Using this representation, the products \( \mathbf{T}_1 \mathbf{V} \) and \( \mathbf{T}_2 \mathbf{V} \) map to

\[
\mathcal{L}(\mathbf{T}_1 \mathbf{V}) = \{ (x_1 + 1) \mathbf{e}_1 + x_2 \mathbf{e}_2; \ x_1, x_2 \in \{0, \ldots, m - 1\} \}
\]

and \( \mathcal{L}(\mathbf{T}_2 \mathbf{V}) = \{ x_1 \mathbf{e}_1 + (x_2 + 1) \mathbf{e}_2; \ x_1, x_2 \in \{0, \ldots, m - 1\} \}
\]

respectively. For perfect alignment, we have to design the \( \mathbf{T}_i \) matrices such that \( \mathcal{L}(\mathbf{T}_1 \mathbf{V}) = \mathcal{L}(\mathbf{T}_2 \mathbf{V}) \). A sufficient set of conditions for perfect span intersection is that the power tuples of \( \mathbf{V}, \mathbf{T}_1 \mathbf{V}, \) and \( \mathbf{T}_2 \mathbf{V} \) perfectly intersect, for example, consider the condition for the \( \mathbf{T}_1 \) matrix \( \mathcal{L}(\mathbf{T}_1 \mathbf{V}) = \mathcal{L}(\mathbf{V}) \), i.e.,

\[
\{(x_1 + 1)\mathbf{e}_1 + x_2 \mathbf{e}_2; \ x_1, x_2 \in \{m\}\} = \{\mathbf{e}_1 + x_2 \mathbf{e}_2; \ x_1, x_2 \in \{m\}\}
\]

These conditions are satisfied when the matrix powers “wrap around” upon reaching their modulus, \( m \). This wrap-around property is obtained when the \( \mathbf{T}_1 \) and \( \mathbf{T}_2 \) are diagonals of \( m^6 \) roots of unity \( \mathbf{T}_1^m = \mathbf{T}_2^m = \mathbf{T}_2^m = \mathbf{I}_N \). However, arbitrary selection of their elements is not sufficient to ensure the full rank property of \( \mathbf{V} \). To hint on a general procedure which outputs “good” matrices, we see an example where we tune our parameters such that \( \mathbf{V} \) has orthogonal columns. Let us briefly consider the case where \( m = 2 \) and \( N = 2^3 \), for which we choose

\[
\mathbf{T}_1 = \text{diag} \left( \begin{array}{ccc} 1 & -1 & 1 \\ -1 & 1 & -1 \\ 1 & -1 & 1 \end{array} \right) \quad \text{and} \quad \mathbf{T}_2 = \text{diag} \left( \begin{array}{ccc} 1 & -1 & 1 \\ -1 & 1 & -1 \\ 1 & -1 & 1 \end{array} \right).
\]

(14)

For these matrices, \( \mathbf{V} \) has \( m^2 = 4 \) orthogonal columns

\[
\mathbf{V} = [\mathbf{w} \ T_1 \mathbf{w} \ T_2 \mathbf{w} \ T_1 \mathbf{T}_2 \mathbf{w}] = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}
\]

(15)

and \( \mathbf{T}_1 \mathbf{V} = [\mathbf{T}_2 \mathbf{w} \ T_1 \mathbf{T}_2 \mathbf{w} \ T_2 \mathbf{w}] \), \( \mathbf{T}_2 \mathbf{V} = [\mathbf{T}_1 \mathbf{w} \ T_1 \mathbf{T}_2 \mathbf{w} \ T_2 \mathbf{w}] \) have fully overlapping spans. We observe that for the additional matrix

\[
\mathbf{T}_3 = \text{diag} \left( \begin{array}{ccc} 1 & -1 & 1 \\ -1 & 1 & -1 \\ 1 & -1 & 1 \end{array} \right)
\]

(16)

we have that \( [\mathbf{V} \ T_1 \mathbf{V}] = \mathbf{H}_8 \), where \( \mathbf{H}_8 \) is the \( 8 \times 8 \) Hadamard matrix. We will see that Hadamard designs provide the conditions for perfect alignment and linear independence for our problem.

Let \( m = 2, \ N = 2^L \), and \( \mathbf{X}_i = \mathbf{I}_2 \otimes \text{blkdiag} \left( \frac{\mathbf{I}_2}{\sqrt{2}}, -\frac{\mathbf{I}_2}{\sqrt{2}} \right) \), for \( i \in \{L\} \), and consider the set

\[
\mathcal{H}_N = \left\{ \prod_{i=1}^L \mathbf{X}_i^z \mathbf{w} : \mathbf{w} \in \{0,1\} \right\}.
\]

(17)

Lemma 1: Let an \( N \times N \) Hadamard matrix of the Sylvester’s construction

\[
\mathbf{H}_N = \begin{bmatrix} \mathbf{H}_2^x & \mathbf{H}_2^y \\ \mathbf{H}_2^y & -\mathbf{H}_2^x \end{bmatrix}
\]

(18)

with \( \mathbf{H}_1 = 1 \). Then, \( \mathbf{H}_N \) is full rank with mutually orthogonal columns, that are the \( N \) elements of \( \mathcal{H}_N \).

For a proof of Lemma (1) is omitted due to lack of space.

Example To illustrate the connection between \( \mathcal{H}_N \) and \( \mathcal{H}_N \) we “decompose” the Hadamard matrix of order 4

\[
\mathbf{H}_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} = [\mathbf{w} \ X_2 \mathbf{w} \ X_1 \mathbf{w} \ X_3 \mathbf{w} \ X_5 \mathbf{w}],
\]

(19)
where $X_1 = \text{diag}\left( \frac{1}{-1} \right)$ and $X_2 = \text{diag}\left( \frac{1}{1} \right)$. Due to the commutativity of $X_1$ and $X_2$, the columns of $H_4$ are also the elements of $\mathcal{H}_4 = \{w, X_1w, X_2w, X_1X_2w\}$.

Now, consider the matrix $V_i$ that has as columns the elements of

$$V_i = \left\{ \prod_{s=1,s \neq i}^{L} X_s^xw : x_s \in \{0,1\} \right\}, \quad (20)$$

We know that the space of $V_i$ is invariant with respect to $X_j$ since the corresponding lattice representation wraps around itself due to $X_j^2 = I_N$. Additionally, we have

$$\mathcal{L}(X,V_i) = \left\{ e_i + \sum_{s=1,s \neq i}^{L} x_se_s : x_s \in \{0,1\} \right\},$$

and we observe that $\mathcal{L}(X,V_i) \cap \mathcal{L}(V_i) = \emptyset$, i.e., $\mathcal{L}(V_i)$ does not include any points with nonzero $x_i$ coordinates. Then, due to the orthogonality of elements within $\mathcal{H}_N$, we have

$$|\mathcal{L}(V_i)| = |\mathcal{L}(X,V_i)| = \text{rank}(V_i) = \text{rank}(X,V_i) = N/2,$$

for any $i,j \in \{1, \ldots, L\}$. Hence, we obtain the following lemma for the set $\mathcal{H}_N$ and its associated $\mathcal{L}$ map.

Lemma 2: For any $i,j \in \{1, 2, \ldots, L\}$ we have that

$$\text{rank}(\{V_i \quad X_jV_i\}) = |\mathcal{L}(V_i) \cup \mathcal{L}(X_jV_i)| = \begin{cases} N, & i = j; \\ \frac{N}{2}, & i \neq j. \end{cases}$$

In Fig. 7, we give an illustrative example of the aforementioned definitions and properties. For $N = 2^3$, we consider $H_8$ and $V_3$ along with the matrix product $X_2V_3$ and their corresponding lattice representations.

![Fig. 7](image)

**Remark 4:** Notice that equations (14) and (15) are respectively analogous to the channel matrices and beamforming vectors used in wireless channels for ergodic interference alignment [20]. In particular, for the $K$ user interference channel, the channel matrices used for ergodic alignment are diagonalized versions of the column vectors of $H_2$.

V. OPTIMAL SYSTEMATIC NODE REPAIR

Let systematic node $i \in [k]$ of the code in (5) fail. The coding matrix $A_i$, corresponding to the lost systematic piece $f_i$, holds one matrix, that is, $X_i$, which is unique among all other coding matrices, $A_s$, $s \in [k]\{i\}$. We pick the repair matrix as a set of $\frac{N}{2}$ vectors that span a subspace invariant to all $X_s$ matrices but to one key matrix: the unique $X_i$ component of $A_i$. We construct the $N \times \frac{N}{2}$ repair matrix $V_i$ whose columns are the elements of the set

$$V_i = \left\{ \prod_{s=1,s \neq i}^{k+1} X_s^xw : x_s \in \{0,1\} \right\}. \quad (21)$$

This repair matrix is used to multiply the contents of parity node 1 and 2, that is, $V_i^{(1)} = V_i^{(2)} = V_i$. During the repair, the useful (desired signal) space pulated by $f_i$ is $[V_i \quad A_i V_i]$ and the interference space due to file part $f_s$, $s \in [k]\{i\}$, is $[V_i \quad A_i V_i]$, $V_i$. Remember that an optimal solution to $R_i$ requires the useful space to have rank $N$ and each of the interference spaces rank $\frac{N}{2}$. Observe that the following holds for each of the interference spaces

$$N/2 \leq \text{rank}([V_i \quad (a_iX_s + b_iX_{k+1} + I_N)V_i]) \leq |\mathcal{L}(V_i) \cup \mathcal{L}(X_jV_i) \cup \mathcal{L}(X_{k+1}V_i)| = |\mathcal{L}(V_i)| = N/2,$$

for $s \in [k]\{i\}$, since $\mathcal{L}(X_jV_i) = \mathcal{L}(V_i)$, $s \in [k+1]\{i\}$. Then, for the useful data space we have

$$\text{rank}([V_i \quad A_i V_i]) = \text{rank}([V_i \quad (a_iX_s + b_iX_{k+1} + I_N)V_i]) \leq |\mathcal{L}(V_i) \cup \mathcal{L}(X_jV_i) \cup \mathcal{L}(X_sV_i)| = |\mathcal{L}(H_N)| = N,$$

for any $a_i \neq 0$, where (*) comes from the fact that $(a_iX_s + b_iX_{k+1} + I_N)V_i$ is a linear combination of columns from $V_i$, $X_{k+1}V_i$, and $X_sV_i$, and the column spaces of $V_i$ and $X_{k+1}V_i$ are identical.

Therefore, we are able to generate the minimum amount of interference and at the same time satisfy the full rank constraint of $R_i$. The repair matrix in (21) is an optimal solution for $R_i$ and systematic node $i$ can be optimally repaired by downloading $(k+1)\frac{N}{2}$ worth data equations, for all $i \in [k]$.

VI. OPTIMAL PARITY REPAIR

A. Repairing the first parity

Let the first parity node fail. We make a change of variables to obtain a new representation for our code in (5), where the first parity is a systematic node in an equivalent representation. We start with our $(k, k+2)$ MDS storage code of (5)

$$\begin{pmatrix} I_N & 0_N & \ldots & 0_N \\ 0_N & I_N & \ldots & 0_N \\ \vdots & \vdots & \ddots & \vdots \\ 0_N & 0_N & \ldots & I_N \\ A_1 & A_2 & \ldots & A_k \end{pmatrix} f. \quad (22)$$

and make the following change of variables $\sum_{i=1}^{k} f_i = y_1$, $f_s = y_s$, $s \neq 1$ and solve for $f_1$ in terms of the $y_i$ to obtain $f_1 = y_1 - \sum_{s=2}^{k} y_s$. This results to the equivalent representation

$$\begin{pmatrix} I_N & -I_N & \ldots & -I_N \\ 0_N & I_N & \ldots & 0_N \\ \vdots & \vdots & \ddots & \vdots \\ 0_N & 0_N & \ldots & I_N \\ I_N & 0_N & \ldots & 0_N \\ A_1 & A_2 & \ldots & A_k - A_1 \end{pmatrix} y, \quad (23)$$

where $y = [y_1^T \ldots y_k^T]^T \in \mathbb{F}_q^{kN}$. The first parity node of the code in (5) now corresponds to the node which contains $y_1$ in the aforementioned representation.
Then, we construct the repair matrix $V_a$ with columns in the set

$$V_a = \left\{ \prod_{s=2}^{k+1} (X_1 X_s)^{x_s} w \in \{0, 1\} \right\}. \quad (24)$$

This set is again a subset of $H_N$. To repair the node of (23) that contains $y_1$ (i.e., the one that corresponds to the first parity node of (22)) we download $X_1 V_a$, times the contents of the first parity in (23) and $V_a$ times the contents of the second parity. During this repair process, the useful space is spanned by $[X_1 V_a, A_1 V_a]$ and the interference space due to file part $y_s, s \neq 1$, is $[-X_1 V_a, (A_s - A_1) V_a]$. Observe that the following hold

$$\mathcal{L}(X_1 X_s V_a) = \mathcal{L}(V_a)$$

$$\Rightarrow \mathcal{L}(X_1 V_a) = \mathcal{L}(V_a)$$

$$\Rightarrow \mathcal{L}(X_1 X_s V_a) = \mathcal{L}(X_2 V_a), \quad (25)$$

for any $s, s_1, s_2 \in [k + 1]$. That is, the repair matrix is an invariant subspace to the matrices $X_1 X_s$. The above equations imply that

$$\mathcal{L}(V_a) \cup \mathcal{L}(X_1 V_a) = \left\{ \sum_{s=1}^{k+1} x_s w : x_s \in \{0, 1\} \right\} = \mathcal{L}(H_N). \quad (26)$$

Here, we therefore have the following for each of the interference spaces $s \neq 1$

$$\text{rank}([X_1 V_a \ (a_s X_s + (b_s - b_1) X_{k+1} - a_1 X_1) V_a])$$

$$\leq |\mathcal{L}(X_1 V_a) \cup \mathcal{L}(V_a) \cup \mathcal{L}(X_{k+1} V_a)| = |\mathcal{L}(X_1 V_a)| = N/2.$$

Moreover, for the useful data space we have

$$\text{rank}([X_1 V_a \ (a_1 X_1 + b_1 X_{k+1} + I_N) V_a])$$

$$= \text{rank}([X_1 V_a V_a]) = |\mathcal{L}(V_a) \cup \mathcal{L}(X_1 V_a)| = |\mathcal{L}(H_N)| = N.$$

Thus, we can perform optimal repair of the node containing $y_1$ in (23), which is equivalent to optimally repairing the first parity of our code in (5).

### B. Repairing the second parity

Here, we first manipulate our coding matrices of (5) to obtain an equivalent representation of our code and then rewrite this code in a form where the second parity of (5) is a systematic node in a new representation. Without loss of generality, we can multiply any coding column block that multiplies the $i$th file part $[A_i]$ with a full rank matrix and maintain the same code properties, as shown in [16]. We multiply the $i$-th block of (5) with $a_i X_i - b_i X_{k+1} + I_N$ to obtain

$$\begin{bmatrix} I_N \\ A_i \\ \end{bmatrix} = \begin{bmatrix} a_i X_i - b_i X_{k+1} + I_N \\ (a_i X_i + I_N)^2 - b_i^2 I_N \\ \end{bmatrix} = \begin{bmatrix} a_i X_i - b_i X_{k+1} + I_N \\ 2a_i X_i + (a_i^2 - b_i^2 + 1) I_N, \end{bmatrix} \quad (27)$$

where in (*) we use the fact that $a_i^2 - b_i^2 + 1 = 0$. We continue by multiplying the $i$-th column block with $(a_i)^{-1} X_i$ to obtain

$$\begin{bmatrix} a_i X_i - b_i X_{k+1} + I_N \\ 2a_i X_i \\ \end{bmatrix} = \begin{bmatrix} I_N - a_i^{-1} b_i X_{k+1} X_i + a_i^{-1} I_N \\ I_N \\ \end{bmatrix}, \quad (28)$$

where in the last step we multiplied the contents of the second parity with $2^{-1}$. Hence, let

$$A'_i = I_N - a_i^{-1} b_i X_{k+1} X_i + a_i^{-1} X_i, \quad i \in [k]. \quad (29)$$

Then, we rewrite our original code as

$$\begin{bmatrix} I_N \\ I_N \\ \vdots \\ I_N \\ \end{bmatrix} f'$$

where $f'$ is a full rank row transformation of $f$, and make a change of variables $\sum_{i=1}^k f'_i = y'_i$ such that the second parity becomes a systematic node in a new representation. We thus obtain the equivalent form

$$\begin{bmatrix} I_N \\ 0_N \\ \vdots \\ 0_N \\ \end{bmatrix} y'_i \quad (30)$$

where

$$A'_i - A'_i = a_i^{-1} X_s - a_i^{-1} b_i X_{k+1} X_s + a_i^{-1} X_i = a_i^{-1} X_i.$$

Then, the parity node which corresponds to systematic node 1 here, can be repaired by multiplying with $X_1 V_b$ the first block and with $V_b$ the second to last block of the equivalent code, where $V_b$ has columns in the set

$$V_b = \left\{ X_{k+1} \prod_{s=2}^k (X_1 X_s)^{x_s} w : x_{k+1}, x_s \in \{0, 1\} \right\}. \quad (32)$$

Again, the following equations hold

$$\mathcal{L}(X_{k+1} V_b) = \mathcal{L}(V_b)$$

$$= \left\{ \left( \sum_{s=2}^k x_s \text{ (mod 2)} \right) e_1 + \sum_{s=2}^{k+1} x_s e_s : x_s \in \{0, 1\} \right\}, \quad (33)$$

$$\mathcal{L}(X_1 V_b) = \mathcal{L}(X_2 V_b)$$

$$= \left\{ \left( 1 + \sum_{s=2}^k x_s \text{ (mod 2)} \right) e_1 + \sum_{s=2}^{k+1} x_s e_s : x_s \in \{0, 1\} \right\}, \quad (34)$$

and

$$\mathcal{L}(X_1 X_{k+1} V_b) = \mathcal{L}(X_1 V_b), \quad (35)$$

for all $s_1, s_2 \in [k]$. Hence, we have for the interference space generated by component $y'_s, s \neq 1$

$$\text{rank}([X_1 V_b \ (A'_i - A'_i) V_b])$$

$$\leq |\mathcal{L}(X_1 V_b) \cup \mathcal{L}(X_2 V_b) \cup \mathcal{L}(X_{k+1} X_1 V_b) \cup \mathcal{L}(X_{k+1} X_s V_b)|$$

$$= |\mathcal{L}(X_1 V_b) \cup \mathcal{L}(X_2 V_b)| = N/2.$$

Moreover, the useful space is full rank

$$\text{rank}([X_1 V_b (I_N - a_i^{-1} b_i X_{k+1} X_i + a_i^{-1} X_i) V_b])$$

$$= \text{rank}([X_1 V_b V_b]) = N. \quad (37)$$

Thus, we can perform optimal repair for the second parity of the code in (5), with repair bandwidth $(k + 1)N/2$. 
VII. The MDS Property

We discuss the MDS property using the notion of data collectors (DCs), in the same manner that it was used in [2]. A DC can be considered as an external user that connects to the contents of some subset of $k$ nodes. A storage code where each node expends $\frac{M}{k}$ worth of storage, has the MDS property when all possible $\binom{k}{i}$ DCs can retrieve $f$. We can show that testing the MDS property is equivalent to checking the rank of a specific matrix associated with each DC. This DC matrix is the vertical concatenation of the $k$ stacks of equations stored by the nodes that the DC connects to. If all $\binom{k}{i}$ DC matrices are full rank, then the code is MDS.

As shown in [21] a DC that connects to $k$ systematic nodes, or $k-1$ systematic and 1 parity will receive a full rank set of $M$ equations and will be able to reconstruct the file. We need to also consider DCs that connect to $k-2$ systematic nodes and both parity nodes. Let a DC that connects to systematic node $1,\ldots,k-2$ and the two parities. The corresponding DC matrix is

$$
\begin{bmatrix}
I_N & \cdots & 0_{N \times N} & 0_{N \times N} & 0_{N \times N} \\
0_{N \times N} & \cdots & I_N & 0_{N \times N} & 0_{N \times N} \\
I_N & \cdots & I_N & 0_{N \times N} & 0_{N \times N} \\
A_1 & \cdots & A_{k-2} & A_{k-1} & A_k
\end{bmatrix}
$$

(38)

The leftmost $(k-2)N$ columns of the matrix in (38) are linearly independent, due to the upper-left identity block. Moreover, the leftmost $(k-2)N$ columns are linearly independent with the rightmost $2N$, using an analogous argument. Hence, we need to only check the rank of the sub-matrix $\begin{bmatrix} I_N & I_N \\ A_1 & A_2 \end{bmatrix}$. In general, a DC that connects to some $k-2$ subset of systematic nodes and the two parities has a corresponding matrix where $\begin{bmatrix} I_N & I_N \\ A_1 & A_2 \end{bmatrix}$ needs to be full rank so that the MDS property is ensured, for $i, j \in [k]$ and $i \neq j$. The code is MDS when

$$
\text{rank}\left(\begin{bmatrix} I_N & I_N \\ A_1 & A_j \end{bmatrix}\right) = \text{rank}\left(\begin{bmatrix} I_N & I_N \\ A_1 & A_j \end{bmatrix} \cdot \begin{bmatrix} I_N & I_N \\ 0_{N \times N} & -I_N \end{bmatrix}\right)
= N/2 + \text{rank}(a_ix_i + a_jx_j + (b_i - b_j)x_{k+1}) = N,
$$

(39)

for all $i, j \in [k]$, which is true if

$$
\text{rank}(a_ix_i + a_jx_j + (b_i - b_j)x_{k+1}) = N/2.
$$

Since the diagonal elements of $X_i$ are $\{\pm 1\}$, the previous requirement gives the lemma.

Lemma 3: The code in (5) is MDS when

$$
a_i \pm a_j \pm (b_i - b_j) \neq 0,
$$

(40)

for all $i \neq j \in [k]$.

Now, remember that our initial constraint on the $a_i$ and $b_i$ constants was

$$
a_i^2 - b_i^2 = -1 \iff (a_i - b_i)(a_i + b_i) = -1.
$$

(41)

one solution to the previous equation is $a_i - b_i = x_i$ and $a_i + b_i = -x_i^{-1}$. If we input this solution to (41), then the MDS equations of Lemma 3 become

$$
\begin{align*}
a_i - a_j + (b_i - b_j) &= -x_i^{-1} + x_j^{-1} \neq 0 \\
a_i + a_j - (b_i - b_j) &= x_i - x_j^{-1} \neq 0 \\
a_i - a_j - (b_i - b_j) &= x_i - x_j \neq 0 \\
a_i + a_j + (b_i - b_j) &= -x_i^{-1} + x_j \neq 0
\end{align*}
$$

The above conditions can be equivalently stated as

$$
x_i \neq x_j \text{ and } x_i x_j \neq 1,
$$

(42)

for any $i \neq j \in [k]$.

Then, consider a prime field $\mathbb{F}_q$ of size $q$. The set of $x_i$s that satisfies our MDS requirements, is such in which no two elements are inverses of each other. It is known that, over a prime field, half the nonzero elements are inverses of the other nonzero half. If we additionally do not consider $x_i \in \{1, q - 1\}$, then we are left with $\frac{q-2}{2}$ elements. Therefore, we can consider a prime field of size $q$ that has the property

$$
k \leq q - \frac{3}{2} \iff q \geq 2k + 3
$$

(43)

and obtain $x_1, \ldots, x_k$ such that our requirements are satisfied. Then, the elements $a_i$ and $b_i$, for all $i \in [k]$, can be obtained through the following equations

$$
a_i = 2^{-1}x_i - 2^{-1}x_i^{-1} \text{ and } b_i = -2^{-1}x_i - 2^{-1}x_i^{-1}.
$$

(44)

Observe that the above solutions yield $a_i \neq 0$ (that is needed for successful repair), for all $i \in [k]$, when $x_i \notin \{0, 1, q - 1\}$. Therefore a prime field of size greater than, or equal to $2k + 3$ always suffices to obtain the MDS property.

VIII. Acknowledgement

The authors would like to thank Changho Suh for insightful discussions.

References


