<table>
<thead>
<tr>
<th><strong>Title</strong></th>
<th>On applications of orbit codes to storage</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Author(s)</strong></td>
<td>Liu, Shiqiu; Oggier, Frédérique</td>
</tr>
<tr>
<td><strong>Citation</strong></td>
<td>Liu, S., &amp; Oggier, F. (2016). On applications of orbit codes to storage. Advances in Mathematics of Communications, 10(1), 113-130.</td>
</tr>
<tr>
<td><strong>Date</strong></td>
<td>2016</td>
</tr>
<tr>
<td><strong>URL</strong></td>
<td><a href="http://hdl.handle.net/10220/41425">http://hdl.handle.net/10220/41425</a></td>
</tr>
<tr>
<td><strong>Rights</strong></td>
<td>© 2016 American Institute of Mathematical Sciences (AIMS) and Shandong University (SDU). This is the author created version of a work that has been peer reviewed and accepted for publication by Advances in Mathematics of Communications, American Institute of Mathematical Sciences (AIMS) and Shandong University (SDU). It incorporates referee’s comments but changes resulting from the publishing process, such as copyediting, structural formatting, may not be reflected in this document. The published version is available at: [<a href="http://dx.doi.org/10.3934/amc.2016.10.113">http://dx.doi.org/10.3934/amc.2016.10.113</a>].</td>
</tr>
</tbody>
</table>
ON APPLICATIONS OF ORBIT CODES TO STORAGE

SHIQIU LIU AND FRÉDÉRIQUE OGGIER

Abstract. We consider orbit codes, subspace codes obtained from orbits of the action of subgroups of $GL_n(F_q)$ on $m$-dimensional subspaces of $F_q^n$. We discuss their applicability to design storage codes, in particular in the context of collaborative repair: we identify known storage codes that can be interpreted as orbit codes, motivate why orbit codes provide good candidates for storage codes, and translate the storage code parameters into those of the algebraic objects involved. Two simple families of storage orbit codes are given.

1. Introduction

Consider an $(N,n)$ linear code $C$ over the finite field $F_q$, $q$ a prime power, that is, $C$ is an $n$-dimensional subspace of $F_q^N$. A codeword in $C$, in the context of communication, is sent over an erasure channel, which will result in an erroneous codeword, where some of its components have been erased. The code should be designed so as to permit the recovery of the codeword, assuming a given erasure pattern. The minimum Hamming distance $d_H$ of the code characterizes the erasure recovery capacity of the code, sometimes then denoted as an $(N,n,d_H)$ code.

This scenario can be easily translated into the context of distributed storage systems. Instead of sending a codeword over an erasure channel, store the $N$ coefficients of a codeword of $C$ into $N$ storage nodes. Node failures become erasures, and whether the stored encoded object can be retrieved given a failure pattern depends on $d_H$.

There is however a fundamental difference between erasure codes for communication, and erasure codes for distributed storage: in the former case, once the altered codeword is received, it is not deteriorated anymore, while in the latter case, a storage system is meant to last over months or years, and failures continue to happen. Therefore, no matter how good a linear code is, it cannot protect data from failures, without maintenance [1]. Maintenance means that in case of node failures, it is necessary to repair the data that became unavailable, namely: codeword coefficients which are erased need to be recomputed, and stored at live nodes. This can be done using a decoding algorithm, but often, it is preferred to have more efficient ways to compute codeword coefficients, typically as a linear combination of a small number of other codeword coefficients. When several nodes jointly repair $t$ failures, $t \geq 2$, the repair process is said to be collaborative [2, 3]. The metrics of interest to decide the efficacy of a storage code with respect to maintenance are typically the amount of bandwidth needed per repaired node, which is a function of data stored per node,
Shiqiu Liu and Frédérique Oggier

and the number of nodes contacted per repair. The latter is often in tension with the optimal trade-off between repair bandwidth and storage overhead.

To better handle the maintenance process, it is often useful to consider an \((Nm, n)\) linear code \(C\), with generator matrix \((\mathbf{v}_1, \ldots, \mathbf{v}_{Nm}) \in \text{Mat}_{n \times Nm}(F_q)\), with columns \(\mathbf{v}_i \in F_q^{n \times 1}\), \(i = 1, \ldots, Nm\) (This is also an approach used in the context of array codes, see e.g. [4]). Then a data object \(o \in F_q^{1 \times n}\) is encoded into a codeword \((o\mathbf{v}_1, \ldots, o\mathbf{v}_{Nm}) \in F_q^{1 \times Nm}\), and the \(Nm\) codeword coefficients are distributed into \(N\) storage nodes, each node storing \(m\) coefficients. This helps the maintenance process in that it allows the content of a node to be repaired with lesser communication cost than if the node content is considered as a single symbol. Each storage node is assumed to be equipped with computational power, therefore, a node storing \(o\mathbf{v}_{i_1}, \ldots, o\mathbf{v}_{i_m}\) is able to compute \(F_q\)-linear combinations \(\sum_{j=1}^m \alpha_j o\mathbf{v}_{i_j}\). From this point of view, it makes sense to therefore model each storage node as storing an \(m\)-dimensional subspace of \(F_q^m\) with \(F_q\)-basis \(\mathbf{v}_{i_1}, \ldots, \mathbf{v}_{i_m}\) (we may assume without loss of generality that \(\mathbf{v}_{i_1}, \ldots, \mathbf{v}_{i_m}\) are linearly independent), as explicitly suggested in [5].

With the above view point, the problem of designing storage codes then becomes that of finding \(N\) \(m\)-dimensional subspaces of \(F_q^m\), which naturally brings in the Grassmannian \(G_q(m, n)\), which is by definition the set of all subspaces of \(F_q^n\) of dimension \(m\). The question is then how to pick subspaces appropriately for the purpose of storage. A first tentative to treat storage codes in this manner was presented in [6], using Grassmannian graphs, which yielded some interesting constructions. A systematic approach to design subspace codes for storage has been recently proposed in [7], which relies on Plücker embeddings. Subspace codes were in general studied from a group action point of view in [8], which introduced the concept of orbit codes. In this paper, motivated by the natural interpretation of storage codes in terms of subspaces and by their properties, we adopt the approach of orbit codes, and question the applicability of such codes to distributed storage, hoping to exploit group actions as a way to characterize good storage codes.

We are particularly interested in the collaborative case, where nodes involved in the repair of \(t\) failures, \(t \geq 2\), download data from live nodes, and exchange some of their obtained data among a subset of each others. When this subset is not proper and involves all \(t-1\) other nodes, it is referred to as collaborative, while partial collaboration [9] emphasizes that data is exchanged among a proper subset of strictly less than \(t-1\) nodes. One reason for choosing this setting is that the one failure case has been already well studied and many codes are available (see [1] and reference therein), while the scenarios of collaboration, and especially partial collaboration, have been less studied, and few constructions are known for partial collaboration [10]. We focus on two classical extreme regimes: one where the amount of repair bandwidth per repaired node is minimized, and one where the amount of storage per node is minimized.

The paper is organized as follows. In Section 2, we translate code parameters in terms of orbit codes, and give a motivating example. In Section 3, we focus on orbit codes generated by the companion matrix of a polynomial, and propose a simple family of codes, for partial collaborative repair, which minimizes the repair bandwidth. We further identify a known class of storage codes [11] as being one instance of cyclic orbit codes. In Section 4, we present another construction of orbit codes for partial collaboration, aiming this time at minimizing the storage. Though this instance of orbit codes is simple, it does achieve the optimal performance in...
terms of trade-off between storage and repair bandwidth. In Section 5, we compare constructions with known bounds: the first construction is shown to achieve the minimum repair bandwidth, but trade the corresponding optimal storage with the minimal number of nodes contacted per repair. Note that as of now, optimal families for partial collaboration at the minimum repair bandwidth case are not known. The second construction is indeed offering the minimum storage together with its corresponding minimum repair bandwidth. The number of contacted nodes per repair is large. The presented codes have the further benefit of being easily generated, as orbit codes are likely to be in general. It turns out that though the framework of orbit codes seem fairly natural to address storage codes, and some known storage codes can be interpreted as orbit codes, the dependency of the code parameters not only on the cyclic group which is acting but also on the original subspace on which the cyclic group acts complicates the study of such codes, which therefore leaves many open questions.

2. Storage Codes from Orbit Codes

As above, let $\mathbb{F}_q$ be the finite field with $q$ elements, with $q$ a prime power, and let $G_q(m,n)$ be the set of all subspaces of $\mathbb{F}_q^n$ of dimension $m$, called Grassmannian:

$$G_q(m,n) = \{ \mathcal{U} \text{ subspace of } \mathbb{F}_q^n, \dim(\mathcal{U}) = m \}.$$  

We denote by $GL_n(\mathbb{F}_q)$ the set of $n \times n$ invertible matrices with coefficients in $\mathbb{F}_q$. Multiplication by elements of $GL_n(\mathbb{F}_q)$ defines a group action from the right on $G_q(m,n)$ by

$$G_q(m,n) \times GL_n(\mathbb{F}_q) \rightarrow G_q(m,n)$$

$$(\mathcal{U}, A) \mapsto A\mathcal{U} = \{ uA, u \in \mathcal{U} \}$$

since any element of $GL_n(\mathbb{F}_q)$ maps an $m$-dimensional subspace to an $m$-dimensional subspace. In fact, as pointed out in [8], since any two $m$-dimensional subspaces can be mapped onto each other by an element of $GL_n(\mathbb{F}_q)$, $GL_n(\mathbb{F}_q)$ acts transitively on $G_q(m,n)$.

Fix $\mathcal{U} \in G_q(m,n)$, let $G$ be a subgroup of $GL_n(\mathbb{F}_q)$, and let $\mathcal{U}G = \{ \mathcal{U}g, g \in G \}$ be the orbit of $\mathcal{U}$ under the right action of $G$. We will refer to $\mathcal{U}G$ as an orbit code, following the terminology of [8]. If furthermore $G$ is cyclic, we call $\mathcal{U}G$ a cyclic orbit code.

In order to represent an $m$-dimensional subspace $\mathcal{U}$ of the vector space $\mathbb{F}_q^n$, we fix a basis of $\mathcal{U}$, and use an $m \times n$ matrix $\mathcal{U}$ whose row space $\{v\mathcal{U}, v \in \mathbb{F}_q^m\}$ is $\mathcal{U}$.

 Orbit codes form collections of $m$-dimensional subspaces of $\mathbb{F}_q^n$ obtained by group action, which have been successfully used in the context of network coding [8]. This makes them potential candidates to provide codes for distributed storage. Let us see how the storage parameters are translated into parameters for an orbit code $\mathcal{U}G$, assuming that an object $\mathbf{o}$ of length $n$ is stored.

2.1. Code Length. The number $N$ of storage nodes is the length of the code, that is the cardinality $|\mathcal{U}G|$ of the orbit $\mathcal{U}G$, and it is well known that

$$|\mathcal{U}G| = \frac{|G|}{|\text{Stab}_G(\mathcal{U})|}.$$
where \( \text{Stab}_G(\mathcal{U}) = \{ g \in G, \mathcal{U}g = \mathcal{U} \} \) is a subgroup of \( G \) called the stabilizer of \( \mathcal{U} \). Alternatively

\[
|\mathcal{U}G| = \frac{|G|}{|G \cap \text{Stab}_{\text{GL}_n(F_q)}(\mathcal{U})|}.
\]

Indeed, if \( g \in \text{Stab}_G(\mathcal{U}) \), then \( g \in G \) and \( g \) is also an element of \( \text{GL}_n(F_q) \) such that \( \mathcal{U}g = \mathcal{U} \). For the reverse inclusion, similarly, if \( g \in G \) and \( \mathcal{U}g = \mathcal{U} \), then \( g \in \text{Stab}_G(\mathcal{U}) \).

The following two results give a way to compute the stabilizer of \( \mathcal{U} \), and in turn the number of nodes over which the storage code is used.

**Lemma 1.** Let \( \mathcal{U}_I \) be the \( m \)-dimensional subspace generated by the first \( m \) unit vectors, that is, it has \( \mathcal{U}_I = (I_m, 0_{m \times (n-m)}) \) as an \( F_q \)-basis. Then

\[
\text{Stab}_{\text{GL}_n(F_q)}(\mathcal{U}_I) = \left\{ \begin{bmatrix} A & 0 \\ B & C \end{bmatrix}, A \in \text{GL}_m(F_q), B \in F_q^{(n-m) \times m}, C \in \text{GL}_{n-m}(F_q) \right\}.
\]

**Proof.** Take \( g \) in \( \text{Stab}_{\text{GL}_n(F_q)}(\mathcal{U}_I) \). Note that \( U_1g = (A, 0_{m \times (n-m)}) \) and since \( A \) is invertible, then \( (A, 0) \) generates \( \mathcal{U}_I \). Then \( B \) can take any value, but \( C \) needs to be invertible, to ensure that \( g \) is. \( \square \)

We keep the above notation in what follows.

**Lemma 2.** For any \( m \)-dimensional subspace \( \mathcal{U} \), we have

\[
\text{Stab}_{\text{GL}_n(F_q)}(\mathcal{U}) = L^{-1}\text{Stab}_{\text{GL}_n(F_q)}(\mathcal{U}_I)L,
\]

for \( L \) such that \( \mathcal{U} = \mathcal{U}_I L \), \( L \in \text{GL}_n(F_q) \).

**Proof.** Note first that any \( m \)-dimensional subspace \( \mathcal{U} \) is of the form \( \mathcal{U} = \mathcal{U}_I L \) for some \( L \) in \( \text{GL}_n(F_q) \). Take \( g \in \text{Stab}_{\text{GL}_n(F_q)}(\mathcal{U}) \), then by definition

\[
\mathcal{U}g = \mathcal{U} \Leftrightarrow \mathcal{U}_IgL = \mathcal{U}_IL \Leftrightarrow \mathcal{U}_IgL^{-1} = \mathcal{U}_I,
\]

showing that \( LgL^{-1} \in \text{Stab}_{\text{GL}_n(F_q)}(\mathcal{U}_I) \), that is \( g \in L^{-1}\text{Stab}_{\text{GL}_n(F_q)}(\mathcal{U}_I)L \).

Conversely, take an element \( L^{-1}gL \in L^{-1}\text{Stab}_{\text{GL}_n(F_q)}(\mathcal{U}_I)L \), then

\[
\mathcal{U}L^{-1}gL = \mathcal{U}_IL^{-1}gL = \mathcal{U}_IgL = \mathcal{U}_IL = \mathcal{U}
\]

and \( L^{-1}gL \) indeed belongs to the stabilizer of \( \mathcal{U} \). \( \square \)

Using the above lemma, we have that

(1) \[
|\text{Stab}_{\text{GL}_n(F_q)}(\mathcal{U})| = |L^{-1}\text{Stab}_{\text{GL}_n(F_q)}(\mathcal{U}_I)L| = |\text{Stab}_{\text{GL}_n(F_q)}(\mathcal{U}_I)|.
\]

2.2. **Storage Capacity.** The storage capacity (or number of stored symbols in \( F_q \)) for every node is \( m \). This is because a subspace \( \mathcal{U} \) of dimension \( m \) means that the corresponding node actually stores \( \mathcal{U}o^T \in F_q^m \), where we recall that \( o \in F_q^{1 \times n} \) is the data object to be stored. This notion was introduced in [5]. Under the hypothesis that a stored object \( o \in F_q^{1 \times n} \) should be recoverable from any \( k \) nodes, the minimal value of \( m \) is \( n/k \).
2.3. Fault Tolerance. The ability to recover a stored object \( o \in \mathbb{F}_q^{1 \times n} \) from any \( k \) nodes, thus providing protection against any \( N - k \) failures is rephrased as:
\[
\dim(\mathcal{U}_1 + \ldots + \mathcal{U}_n) = n.
\]
Note that when \( m = 1 \), then, recalling the connection with classical codes presented in the introduction, every node stores one column of a generator matrix of a linear code, and the property of fault tolerance just discussed is that of a maximum distance separable (MDS) code.

For arbitrary \( k \), there is no formula to evaluate \( \dim(\mathcal{U}_1 + \ldots + \mathcal{U}_n) \), apart from using iteratively the well known formula when \( k = 2 \) which states that
\[
\dim(\mathcal{U}^{g^i} + \mathcal{U}^{g^j}) = \dim(\mathcal{U}^{g^i}) + \dim(\mathcal{U}^{g^j}) - \dim(\mathcal{U}^{g^i} \cap \mathcal{U}^{g^j}).
\]

**Example 1.** Consider the Grassmannian \( G_2(3,5) \), and let \( G = \langle g \rangle \) be the subgroup of \( GL_5(\mathbb{F}_2) \) generated by
\[
g = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix}.
\]
Consider the orbit code \( \mathcal{U}G \), where \( \mathcal{U} = \mathcal{U}_1 \) has for basis
\[
\mathcal{U} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}.
\]
The length \( N \) of the orbit code \( \mathcal{U}G \) is
\[
N = |\mathcal{U}G| = \frac{|G|}{|\text{Stab}_G(\mathcal{U})|} = 4.
\]
Indeed, \( g \) has order 4
\[
g^2 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix},
\]
and \( \text{Stab}_G(\mathcal{U}) \) is trivial, since \( G \) intersects trivially \( \text{Stab}_{GL_5(\mathbb{F}_2)}(\mathcal{U}) \) which is formed of matrices whose first \( 3 \times 3 \) block is invertible (see Lemma 1). The orbit code \( \mathcal{U}G \) is
\[
\mathcal{U}, \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}.
\]
This corresponds to a storage code where the size of the object \( o \) is \( n = 5 \), every storage node stores \( m = 3 \) symbols, e.g., the first node corresponding to the subspace \( \mathcal{U} \) stores \( (1,0,0,0,0) o^T = o_1 \), \( (0,1,0,0,0) o^T = o_2 \), \( (0,0,1,0,0) o^T = o_3 \) for \( o = (o_1, \ldots, o_5) \in \mathbb{F}_2^5 \), giving the following storage allocation:

- Node 1: \( o_1, o_2, o_3 \)
- Node 2: \( o_3, o_5, \sum_i o_i \)
- Node 3: \( \sum_i o_i, o_2, o_4 \)
- Node 4: \( o_4, o_1, o_5 \).

The data object \( o \) may be retrieved out of any two nodes, since, from (2)
\[
\dim(\mathcal{U}^{g^i} + \mathcal{U}^{g^j}) = 2 \dim(\mathcal{U}^{g^i}) - 1 = 5
\]
for all $i \neq j$. Indeed, two distinct subspaces intersect in a subspace of dimension 1.

2.4. Repair. As pointed out in the introduction, the main difference between a storage code and an erasure code is that a storage code should be amenable to repair, namely, the code should be such that the data stored at one node can be computed from a (small) subset of other nodes, without (necessarily) having to decode the object first. Concretely, a repair of one failure is done as follows: a node downloads data from several, possibly all live nodes, and from the data gathered, recomputes the codeword coefficient lost because of the failure. The parameters of importance are typically the communication cost of the repair, called repair bandwidth, which is also related to the number of live nodes contacted.

- Decomposing the stored data per node into $m$ pieces is beneficial to the communication cost: it allows a repair to download part of the stored data. The optimal repair bandwidth per failed node is $m$, since at least the content of a failed node must be communicated for a repair to be successful.
- To speed up repairs, contacting few live nodes is beneficial [1]. The optimal number of nodes is thus 2 in the case of one failure (since less than 2 means that a repetition code is used), or 1 in the (partial) collaborative case.

One may consider either repair of one failure at a time, or (possibly) wait for several failures to occur before repairing $t$ of them together. Joint repair of multiple failures is often done collaboratively [2, 3] as follows: a subset of nodes handle the repair, they each download some amount of information, perform the necessary computations on their data, but also further exchange data among each other, thus involving $t - 1$ nodes in the exchange process. If strictly less than $t - 1$ nodes are involved in the exchange, we speak of partial collaboration. Collaboration is known to be beneficial to the repair process [2, 3] in terms of reducing the repair bandwidth.

Example 2. We continue Example 1, where we have for $\mathbf{o}$ of size 5

Node 1 : $o_1, o_2, o_3$
Node 2 : $o_3, o_5, \sum_i o_i$
Node 3 : $\sum_i o_i, o_2, o_4$
Node 4 : $o_4, o_1, o_5$.

In case of one node failure, the subspace $Ug^i$ may be computed from the knowledge of a subspace of dimension 1 from each of the other $Ug^j$, $j \neq i$, which follows from the fact that $\dim(Ug^i \cap Ug^j) = 1$ for all $i \neq j$. For example, if Node 1 were to fail, it can be recomputed using $o_1$ from Node 4, $o_2$ from Node 3, and $o_3$ from Node 2. This code instance has been reported in [12], not in the language of orbit codes, but as a code whose repair bandwidth is minimal.

While the motivation example presented above is optimal for the case of one failure, most of our focus next will be on the collaborative case.

3. Instances of Cyclic Orbit Codes using Companion Matrices

We next propose a family of cyclic orbit codes based on companion matrices and compute the parameters of the corresponding storage codes.
Definition 1. Let \( p(x) = \sum_{i=0}^{n-1} p_i x^i + x^n \) be a monic polynomial in \( \mathbb{F}_q[x] \). Then the companion matrix \( M_p \) of \( p(x) \) is defined by

\[
M_p = \begin{bmatrix}
0 & 1 & 0 & \cdots & 0 \\
0 & 0 & 1 & \cdots & 0 \\
\vdots & \ddots & \ddots & \ddots & \vdots \\
0 & 0 & 0 & \cdots & 1 \\
-p_0 & -p_1 & -p_2 & \cdots & -p_{n-1}
\end{bmatrix} \in \mathbb{F}_q^{n \times n}.
\]

We first discuss basic properties of codes, namely length and fault tolerance, before proposing code constructions.

3.1. Basic Properties. The following is probably well known and is proven for the sake of completeness.

Lemma 3. Let \( G = \langle g \rangle \) be the subgroup of \( \text{GL}_n(\mathbb{F}_q) \) generated by the \( n \times n \) companion matrix

\[
g = \begin{bmatrix}
0 & 1 & 0 & \cdots & 0 \\
0 & 0 & 1 & \cdots & 0 \\
\vdots & \ddots & \ddots & \ddots & \vdots \\
0 & 0 & 0 & \cdots & 1 \\
-p_0 & -p_1 & -p_2 & \cdots & -p_{n-1}
\end{bmatrix}
\]

of a polynomial \( p(x) = \sum_{i=0}^{n-1} p_i x^i + x^n \in \mathbb{F}_q[x] \), \( p_0 \neq 0 \). Then the order of \( g \) is the smallest positive integer \( l \) such that \( p(x) \) divides \( x^l - 1 \).

Proof. It is well known that \( p(x) \) is the minimal polynomial of the companion matrix \( M_p \) defined in (3), namely \( p(x) \) is the monic polynomial of least degree such that \( p(M_p) = 0 \). Now since \( p_0 \neq 0 \), there exists a positive integer \( e \leq q^n - 1 \) such that \( p(x) \) divides \( x^e - 1 \) (see [13, Lemma 3.1]), so take \( l \) to be the smallest positive integer with the property that \( p(x)|x^l - 1 \). Consequently, \( M_p^l = I_n \) if and only if \( p(x)|x^l - 1 \). \qed

Corollary 1. [14] Let \( G = \langle g \rangle \) be the subgroup generated by the companion matrix of the polynomial \( p(x) = \sum_{i=0}^{n-1} x^i \in \mathbb{F}_q[x] \). Then \( |G| = n + 1 \).

Proof. We have \((x - 1) \sum_{i=0}^{n} x^i = x^{n+1} - 1\). \qed

One advantage to deal with companion matrices is that we are sure that no matter the choice of \( \mathcal{U} \), the resulting code will not be trivial (that is, containing \( \mathcal{U} \) only).

Proposition 1. Let \( g \) be the companion matrix of \( p(x) \) as in (4), and let \( \mathcal{U} \) be an \( m \)-dimensional subspace in \( G_q(m, n) \). Then the orbit code \( \mathcal{U}(g) \) is not trivial (\( |\mathcal{U}(g)| \geq 2 \)).

Proof. Consider \( g \) as a linear map from \( \mathbb{F}_q^n \) to \( \mathbb{F}_q^n \), and \( \mathbb{F}_q^n \) as an \( \mathbb{F}_q[x] \)-module over the polynomial ring \( \mathbb{F}_q[x] \) [15], its invariant subspaces are given by the Smith normal form of \( xI_n - g \), a diagonal matrix whose diagonal coefficients are called elementary divisors, which we denoted by \( d_i \). They are computed as follows. Compute the \( i \times i \)
minors of
\[ xI_n - g = \begin{bmatrix}
  x & -1 & 0 & \cdots & 0 \\
  0 & x & -1 & \cdots & 0 \\
  \vdots & \ddots & \ddots & \ddots & \vdots \\
  0 & 0 & 0 & \cdots & x + p_{n-1}
\end{bmatrix} \\
\begin{bmatrix}
p_0 \\
p_1 \\
p_2 \\
\vdots \\
1
\end{bmatrix} \]
and let \( d_i \) denote the greatest common divisor of all the \( i \times i \) minors. Then \( d_1 = d_2 = \ldots = d_{n-1} = \pm 1 \), which shows that there is no invariant subspace of dimension \( m, 1 \leq m \leq n - 1 \).

While any choice of \( U \) does give a non-trivial orbit code, different choices of \( U \) may give different sizes of orbit codes, as illustrated next.

**Example 3.** Choose \( G = \langle g \rangle \) with
\[
g = \begin{bmatrix}
  0 & 1 & 0 & 0 \\
  0 & 0 & 1 & 0 \\
  0 & 0 & 0 & 1 \\
  1 & 1 & 0 & 0
\end{bmatrix},
\]
which is the companion matrix of \( x^4 + x + 1 \in \mathbb{F}_2[x] \). Since \( x^4 + x + 1 \) is an irreducible polynomial, let \( \omega \) be a root, that is \( \omega^4 = \omega + 1 \), and \( \mathbb{F}_2[x]/(x^4 + x + 1) \cong \mathbb{F}_2(\omega) \cong \mathbb{F}_{2^4} \).

Since \( \omega \) is a primitive element of \( \mathbb{F}_{2^4} \), the order of \( G \) is 15.

Consider \( U_I \in G_2(2, 4) \) with canonical basis \( U_I = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix} \). By Lemma 1, the stabilizer of \( U_I \) under \( G \) is trivial, since \( G \) intersects trivially \( \text{Stab}_{G_2(2, 4)}(U_I) \), which is formed of matrices whose first \( 2 \times 2 \) block is invertible. Therefore the length of the code \( U_I G \) is 15. The length of the code can more directly be seen to be 15 by invoking Lemma 4 below.

To illustrate the fact that a change in the choice of \( U \) may change the length of the corresponding orbit code, we look for powers of \( g^i \) whose set of of invariant subspaces is non-empty. Take \( g^5 \), given by
\[
g^5 = \begin{bmatrix}
  0 & 1 & 1 & 0 \\
  0 & 0 & 1 & 1 \\
  1 & 1 & 0 & 1 \\
  1 & 0 & 1 & 0
\end{bmatrix},
\]
and as in Proposition 1, and keeping the same notation, compute the Smith normal form of \( xI_4 - g^5 \), to find that \( d_1 = d_2 = 1 \), while \( d_3 = x^2 + x + 1 \), and \( d_4 = x^4 + x^2 + 1 \). Therefore the invariant factors are
\[
d_1 = 1, \quad \frac{d_2}{d_1} = 1, \quad \frac{d_3}{d_2} = x^2 + x + 1, \quad \frac{d_4}{d_3} = x^4 + x^2 + 1,
\]
revealing the presence of invariant subspaces of dimension 2. Thus, let \( U \in G_2(2, 4) \) and \( U = U_I L \), where
\[
L = \begin{bmatrix}
  1 & 0 & 0 & 0 \\
  0 & 1 & 1 & 0 \\
  0 & 0 & 1 & 0 \\
  0 & 0 & 0 & 1
\end{bmatrix},
\]
so that a basis is given by
\[
U = \begin{bmatrix}
  1 & 0 & 0 & 0 \\
  0 & 1 & 1 & 0
\end{bmatrix}.
Since $Ug^5 = U, Ug^{10} = (Ug^5)g^5 = U$, and the stabilizer of $U$ under $G$ has cardinality 3, given by

$$Stab_G(U) = \{I_4, g^5, g^{10}\},$$

so that the orbit code $UG$, with $U = UI$, is of length 5, while $UIG$ is of length 15.

The following result tells the length of the considered orbit codes from companion matrices when $q = 2$.

**Lemma 4.** Let $g$ be the companion matrix of $p(x)$, with $g$ and $p(x)$ as in (4), $G = \langle g \rangle$, and let $U_1 \in G_q(m, n)$ be the subspace generated by a canonical basis as above. If $q = 2$, then the stabilizer $Stab_G(U_1)$ is trivial. Consequently the size of the orbit code is the order of $g$.

**Proof.** Let $l$ be the order of $g$, and suppose that the stabilizer of $U_1$ under $G = \langle g \rangle$ is not trivial, that is, there exists an $s$, $0 < s < l$, such that $U_1g^s = U_1$. Then by Lemma 1, there exist $A \in GL_m(F_q), B \in F_q^{(n-m)\times m}, C \in GL_{n-m}(F_q)$ such that

$$g^s = \begin{bmatrix} A & 0 \\ B & C \end{bmatrix}.$$

Note that $g^s \cdot g = g \cdot g^s$, therefore the first $m$ rows of $g^s \cdot g$ and $g \cdot g^s$ are equal, and given respectively by

$$\begin{bmatrix} 0_{m\times 1} & A & 0_{m\times(n-m-1)} \end{bmatrix} = \begin{bmatrix} a_2 & 0 \\ a_3 & 0 \\ \vdots & \vdots \\ a_m & 0 \\ b_1 & c_1 \end{bmatrix},$$

where $a_i, b_i, c_i$ denote the respective $i$th row of $A, B$ and $C$. Equating coefficientwise shows that

$$A = a_{11}I_m, \quad b_1 = 0, \quad c_1 = (a_{11}, 0, ..., 0).$$

Repeating the same computation by noting that the first $m$ rows of $g^{s+1} \cdot g$ equal those of $g \cdot g^{s+1}$, we get

$$b_2 = 0, \quad c_2 = (0, a_{11}, 0, ..., 0),$$

and more generally, comparing the first $m$ rows of $g^{s+t} \cdot g$ and $g \cdot g^{s+t}$, $t = 1, \ldots, n-m-1$, yields

$$b_j = 0, \quad c_j = (0, ..., a_{11}, ...0), \quad j = 1, \ldots, n-m,$$

where only the $j$th coordinate of $c_j$ is non-zero.

Therefore

$$g^s = a_{11}I_n,$$

and since $q = 2$, the only choice is $a_{11} = 1$. This is a contradiction, since $g$ has order $l > s$. \qed

Note that the same argument breaks at $g^s = a_{11}I_n$ for $q \neq 2$. For example, when $q = 3$, $p(x) = x^3 + 2$, then $g$ has order 6, and $g^3 = 2I_3$. 

Advances in Mathematics of Communications Volume X, No. X (200X), X–XX
3.2. Code Construction I: $t$ Failures with Minimized Bandwidth. We saw above that under the same group action, $U G$ and $U_I G$ may yield different code lengths, due to different stabilizers. In this section, we thus consider the canonical case $U_I G$.

**Lemma 5.** Take $U, U' \in G_q(n-1,n)$, $U \neq U'$. Then $U$ and $U'$ intersect in a subspace of dimension $n-2$.

**Proof.** Since by (2)
\[
\dim(U + U') = 2(n-1) - \dim(U \cap U') \leq n,
\]
it must be that
\[
n - 2 \leq \dim(U \cap U')
\]
which concludes the proof, since $U$ and $U'$ are distinct.

The following lemma describes the behavior of intersection of three subspaces, and shows that the considered orbit codes exhibit a particular structure, namely any three subspaces intersect in a subspace of dimension $n - 3$.

**Lemma 6.** Take $U, U', U''$ three distinct elements of $G_q(n-1,n)$. Then $U, U'$ and $U''$ intersect in a subspace of dimension either $n-2$ or $n-3$. When $U, U', U''$ belong to an orbit code $U(g)$ with $g$ the companion matrix of $p(x)$ as in (4), the dimension of their intersection is $n - 3$.

**Proof.** Since by (2)
\[
\dim(U \cap U' \cap U'') = \dim(U) + \dim(U' \cap U'') - \dim(U + (U' \cap U'')),
\]
and we know that $\dim(U) = n - 1$ and $\dim(U' \cap U'') = n - 2$ by the above lemma, we have that
\[
\dim(U \cap U' \cap U'') = 2n - 3 - \dim(U + (U' \cap U'')).
\]
Now either $(U' \cap U'') \subset U$ and $\dim(U + (U' \cap U'')) = n - 1$, or $(U' \cap U'') \not\subset U$ and $\dim(U + (U' \cap U'')) = n$, which proves the first claim.

Consider now the case where the intersection is $n - 2$. Suppose there exist three subspaces $A, B, C$ which intersect in a subspace of dimension $n - 2$. Let $D$ be another subspace. Then $D$ intersects $B$ in a subspace of dimension $n - 2$. Either this subspace is $A \cap B \cap C$, or not. We will suppose not, and show that this yields a contradiction. Then since $B$ and $D$ each are $(n-1)$-dimensional, it must be that $B \cap D$ is formed by an $(n-3)$-dimensional subspace of $A \cap B \cap C$, and a one-dimensional subspace that belongs to $B$, and not to $A$, or to $C$. Now $D$ also intersects $C$ in an $(n-2)$ subspace, and it already intersects it in a $(n-3)$-subspace, therefore the intersection of $C$ and $D$ further contains a one dimensional subspace that belongs to $C$, and not to $A$ and $B$. But by now, the $(n-1)$-dimensional subspace $D$ is fully determined, it is made of a $(n-3)$-dimensional subspace that belongs to $A \cap B \cap C$, and two other one-dimensional subspaces, one, say $c$, belonging to $C$ (no to $A$ or $B$) and one, say $b$, belonging to $B$ (and not to $C$ or $A$). Finally, $D$ must intersect $A$ in a $(n-2)$-dimensional subspace. So either $b$ or $c$ must belong to $A$. But this not possible, because if $b$ belongs to $A$, then the intersection between $B$ and $A$ becomes of dimension $n - 1$, and similarly for if $c$ were to belong to $A$, then the intersection of $A$ and $C$ would be $n - 1$. 

10 Shiqiu Liu and Frédérique Oggier

**Advances in Mathematics of Communications**

**Volume X, No. X (200X), X–XX**
This shows that if there exist three subspaces that intersect in a subspace of dimension \( n - 2 \), then any other subspace will intersect them in the same subspace of dimension \( n - 2 \). Clearly this configuration cannot happen for the orbit code generated from a companion matrix. To see this, it is enough to consider \( Ug, Ug^2 \) and \( Ug^3 \) to see that \( Ug^3 \) contains a one-dimensional subspace which is common to \( Ug^2 \) but not to \( Ug \).

We propose next a simple construction which will turn out to provide the optimal repair bandwidth, since it requires per failure the download of exactly the stored amount per node.

**Proposition 2.** Let \( g \) be the companion matrix of \( p(x) \) as in (4) and let \( U_I \) contain a canonical basis of \( U_I \in G_2(n - 1, n) \). Then the orbit code \( UG \) has the following properties:

1. it is of length \( l \) which is the order of \( g \),
2. the object is retrieved by contacting any two nodes,

Furthermore, for \( n \geq 4 \), if \( p(x) \) is chosen as in Corollary 1, then the repair of \( t \) failures, \( t \geq 2 \), is done by downloading \( n - 2 \) elements of \( F_2 \) from one node for each failure, and exchanging one element of \( F_2 \) between two repair nodes.

**Proof.** The claim on the length comes from Lemma 4. Label the \( l \) storage nodes from 0 to \( l - 1 \). By assumption, the \( i \)th node stores the element \( Ug^i \) of the orbit \( UG \) of \( U \) under the right action of \( G \), \( i = 0, 1, \ldots, l - 1 \).

To retrieve a stored object \( o \in F_2^{l \times l} \) from any 2 nodes, or equivalently to protect against any \( l - 2 \) failures, we need that for a choice of 2 subspaces from the code \( UG \),

\[
\dim(U_{i_1} + U_{i_2}) = n,
\]

and the claim follows from Lemma 5.

Recall Subsection 2.2 for counting the size of elements downloaded/repaired.

Consider any four distinct nodes say \( s, v, i, j \), respectively storing \( Ug^s, Ug^v, Ug^i, Ug^j \). Let \( A = Ug^s \cap Ug^v \) be the intersection of node \( i \) and \( s \), it is of dimension \( n - 2 \) by the above lemma, and similarly, set \( B = Ug^i \cap Ug^j \). We claim that \( \dim(A + B) = n \). Since

\[
\dim(A + B) = \dim(A) + \dim(B) - \dim(A \cap B) = 2(n - 2) - \dim(A \cap B),
\]

we need to show that \( \dim(A \cap B) = n - 4 \). Write \( Ug^i = \langle g_1, \ldots, g_{n-2}, g_{v_0} \rangle \), \( Ug^j = \langle g_1, \ldots, g_{n-2}, g_{j_0} \rangle \). Then \( A = Ug^s \cap Ug^v = \langle g_1, \ldots, g_{s_0-1}, g_{s_0+1}, \ldots, g_{n-2}, g_{v_0} \rangle \), otherwise the nodes \( i, j \) and \( s \) would intersect in the same subspace of dimension \( n - 2 \), which is not possible by Lemma 6, and for the same reason \( B = Ug^i \cap Ug^j = \langle g_1, \ldots, g_{v_0-1}, g_{v_0+1}, \ldots, g_{n-2}, g_{j_0} \rangle \). Without loss of generality, suppose that \( s_0 < v_0 \). Then

\[
A \cap B = \langle g_1, \ldots, g_{s_0-1}, g_{s_0+1}, \ldots, g_{v_0-1}, g_{v_0+1}, \ldots, g_{n-2} \rangle
\]

which has dimension \( n - 4 \).

Suppose that \( t \) nodes have failed, say node \( w, w = 0, \ldots, t - 1 \), which each downloads from node \( w + t \) the subspace \( Ug^w \cap Ug^{w+t} \) of dimension \( n - 2 \) (the labeling of the nodes is done for simplicity and without loss of generality). Each repaired node, after the download phase, is missing only one 1-dimensional subspace. By the above claim, any pair \( A := Ug^w \cap Ug^{w+t}, B := Ug^w \cap Ug^{w+t} \) has the property that \( \dim((Ug^w \cap Ug^{w+t}) + (Ug^w \cap Ug^{w+t})) = n \), thus every repaired node can contact one other node to complete the repair process. Indeed, since \( \dim(A + B) = n \), write the
missing basis vector $a$ at node $w$ in a basis $\{v_1, \ldots, v_n\}$ of $A + B$ as $a = \sum_{i=1}^{n} a_i v_i$. If $v_1, \ldots, v_l \in A$, ask the symbol $\sum_{i=l+1}^{n} a_i v_i$ from $B$ in node $w'$. Iterate this process for the missing basis vector at other nodes.

We remark that this code requires the download of $n-1$ symbols per failure, for a storage of $n-1$ symbols per node, thus is using the minimal possible bandwidth. It is a collaborative code for $t = 2$, since two nodes are participating to the repair, and they are exchanging data among each other, and a partially collaborative code for $t \ge 3$, since then each repaired node only needs to communicate with one other node.

**Example 4.** Consider $G_2(3,4)$, and the subspace $\mathcal{U}$, with basis

$$U = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$ 

Let $G = \langle g \rangle$ be the cyclic subgroup of $GL_4(\mathbb{F}_q)$ generated by

$$g = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}.$$ 

The order of $g$ is 5 and the elements of the orbit codes are explicitly given by

$$U, \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}.$$ 

We store them in node 0 to 4. Assume that node 1 and node 4 failed. To repair node 1, download $\{(0010), (0001)\}$ from node 2, to repair node 4, download $\{(1000), (0100)\}$ from node 0, then during collaboration, the node repairing node 1 can compute (0011) and send it, and the node repairing node 4 sends (0100) in exchange. Note that the strategy is not unique. To repair node 1, download alternatively $\{(0001), (0110)\}$ from node 3, to repair node 4, get $\{(1100), (1111)\}$ from node 2, then the node repairing node 1 can get (0011) and the one repairing node 4 can get (0111) from the collaboration.

### 3.3. Code Construction III: Spreads.

In this section, we interpret a known family of storage codes [11] in the language of orbit codes, using the result from [8] that shows the connection between spreads and orbit codes.

Consider the finite fields $\mathbb{F}_q \subset \mathbb{F}_{q^m} \subset \mathbb{F}_{q^n}$, with $m|n$. It is well known that $\mathbb{F}_{q^n}$ is an $n$-dimensional vector space over $\mathbb{F}_q$, of which $\mathbb{F}_{q^m}$ and its cosets are $m$-dimensional subspaces. They thus form a spread, that is a collection of disjoint subspaces whose union is $\mathbb{F}_{q^n}$. Spreads to design storage codes have been investigated in [11], and can be obtained as a particular case of the proposed construction of orbit codes using companion matrices. For this, two ingredients are needed: (1) $g$ must be the companion matrix of a primitive polynomial $p(x)$, meaning that $p(x)$ is monic and irreducible of degree $n$, and of order $l = q^n - 1$, namely, roots of $p(x)$ are primitive elements of $\mathbb{F}_{q^n}$, they generate the cyclic group $\mathbb{F}_{q^n}^*$, and (2) $\mathcal{U}$ is chosen so as to contain a basis for $\mathbb{F}_{q^n}$.

**Proposition 3.** Suppose that $m|n$, and consider the field extension $\mathbb{F}_{q^n}/\mathbb{F}_{q^m}$. Let $\mathcal{U}$ contain a basis of $\mathbb{F}_{q^m}$, and $g$ be the companion matrix of a primitive polynomial $p(x)$.
On Applications of Orbit Codes to Storage 13

\[ p(x) \text{ such that } p(\omega) = 0 \text{ and } \langle \omega \rangle = \mathbb{F}_{q^m}^*. \] Then \( U(g) \) is an orbit code of length \( \frac{q^n - 1}{q - 1} \) such that \( Ug^i \cap Ug^j \) is trivial whenever \( i \neq j \), and the union of \( Ug^i \), \( i = 1, \ldots, \frac{q^n - 1}{q - 1} \) is \( \mathbb{F}_{q^n}^* \).

**Proof.** Note that \( \mu := \omega^{n/m} \) is a generator of \( \mathbb{F}_{q^m}^* \), and that \( \{1, \mu, \ldots, \mu^{m-1}\} \) forms an \( \mathbb{F}_q \)-basis of \( \mathbb{F}_{q^m} \). Form a matrix \( U \) where each row contains \( \mu^i = \omega^{in/m} \) written in an \( \mathbb{F}_q \)-basis of \( \mathbb{F}_{q^n} \), \( i = 0, \ldots, m-1 \). Then the action of \( g \) corresponds to multiplication by \( \omega \), and the claims follow from the fact that as a \( \mathbb{F}_{q^n}^* \) can be written as the union of \( \omega^i \mathbb{F}_{q^m}^*, i = 1, \ldots, \frac{q^n - 1}{q - 1} \).

More discussion about spread codes is also available in [8].

**Example 5.** Take \( p(x) = x^4 + x + 1 \in \mathbb{F}_2[x] \), it is a primitive polynomial, and let \( g \) be its companion matrix

\[
\begin{bmatrix}
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
1 & 1 & 0 & 0
\end{bmatrix}
\]

Let \( \omega \) be a root of \( p(x) \), that is \( \omega^4 = \omega + 1 \), also \( \omega \) has order 15 and is a primitive element of \( \mathbb{F}_{16} \). Therefore \( \omega^5 \) is a generator of \( \mathbb{F}_{4}^* \), and an \( \mathbb{F}_2 \)-basis for \( \mathbb{F}_4 \) is

\[
\begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 1 & 0
\end{bmatrix}, \begin{bmatrix}
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 1
\end{bmatrix}, \begin{bmatrix}
1 & 1 & 0 & 0 \\
1 & 0 & 1 & 0
\end{bmatrix}
\]

Then the orbit code \( U(g) \) forms a spread, it has length 5, and we obtain a known storage code [11, Example 1], namely

\[
\begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 1 & 0
\end{bmatrix}, \begin{bmatrix}
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 1
\end{bmatrix}, \begin{bmatrix}
1 & 1 & 0 & 0 \\
1 & 0 & 1 & 0
\end{bmatrix}, \begin{bmatrix}
0 & 1 & 0 & 1
\end{bmatrix}
\]

4. **Instances of Cyclic Orbit Codes using Diagonal Matrices**

We describe in this section cyclic orbit codes based on diagonal matrices, and obtain Reed-Solomon codes as a particular case, and a family of storage codes which will turn out to give optimal parameters.

Let \( \omega \) be the primitive element of \( \mathbb{F}_q \). Consider the cyclic group \( G = \langle g \rangle \) of \( GL_n(\mathbb{F}_q) \) generated by the \( n \times n \) diagonal matrix

\[
g = \begin{bmatrix}
w & 0 & 0 & 0 \\
0 & w^2 & 0 & 0 \\
0 & 0 & 0 & 0 \\
\cdots & \cdots & \cdots & \cdots \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0
\end{bmatrix}
\]

where \( n < |\mathbb{F}_q| \). Then the order of \( g \) is that of \( w \), which is \( |\mathbb{F}_q| - 1 \). Indeed, for \( g^i \) to be the identity for some \( i \), we at least need \( w^i \) to be 1, but then all other powers \( (w^j)^i \) will be 1 as well, \( j = 2, \ldots, n - 1 \).

4.1. **Reed-Solomon Codes.** Consider the orbit code \( UG \) where \( U \) has for basis \( U = [1, w, \ldots, w^{n-1}] \).

The length \( N \) of the orbit code \( UG \) is

\[
N = |UG| = \frac{|G|}{|\text{Stab}_G(U)|} = |\mathbb{F}_q| - 1.
\]
In fact, the orbit code $\mathcal{U}G$ is given by $\mathcal{U}G = \{Ug^i, \ i = 0, \ldots, |\mathbb{F}_q| - 2\}$ where

$$Ug^i = [1, w, \ldots, w^{n-1}] \begin{bmatrix} 1 \\ w^i \\ w^{2i} \\ \vdots \\ w^{(n-1)i} \end{bmatrix} = [1, w^{i+1}, w^{2i+2}, \ldots, w^{(n-1)i+i(n-1)}].$$

This corresponds to a storage allocation where each node stores the inner product of the object $\mathbf{o}$ with a column of a generator matrix of a Reed-Solomon code.

4.2. A Code Construction for $t$ Failures, $t \geq 1$. A variation of the above construction can be obtained by tensor products. Consider the orbit code $\mathcal{U}G$ where $G = \langle g \otimes I_m \rangle$ for $g$ the diagonal matrix given by

$$g = \begin{bmatrix} 1 \\ w \\ w^2 \\ \vdots \\ w^{k-1} \end{bmatrix},$$

$k \geq 1$, and $I_m$ the $m$-dimensional identity matrix, and take for a basis of $\mathcal{U}$

$$U \otimes I_m = [1, w, \ldots, w^{k-1}] \otimes I_m.$$ 

Note that the generator $g$ chosen for this construction is similar to (5), but for the dimension $k \geq 1$. Since $n$ is the size of the object, we choose $k$ and $m$ such that $km = n$. Since the order of $g \otimes I_m$ is still $|\mathbb{F}_q| - 1$, this does not change the size $N$ of the orbit code $\mathcal{U}G$.

**Proposition 4.** Let $g$ be the $k$-dimensional diagonal matrix above, and let $U \otimes I_m = [1, w, \ldots, w^{k-1}] \otimes I_m$ be a basis of the $m$-dimensional subspace $\mathcal{U} \in G_q(m, n)$, $n = km$. Then the orbit code $\mathcal{U}G$ has the following properties:

1. the object is retrieved by contacting any $k$ nodes,
2. the repair of $t$ failures, $t \geq 1$, is done by contacting any $k$ nodes.

**Proof.**

1. If any $k$ nodes $i_1, \ldots, i_k$ are contacted, then we obtain $k$ columns of a Vandermonde matrix, each tensored by $I_m$, thus a $k \times k$ invertible submatrix tensored by $I_m$, and the data object $\mathbf{o}$ can be recovered.

2. Without loss of generality, we label the $t$ failed nodes from 1 to $t$. Then $t$ live nodes each contact $k$ nodes, say the $i$th node among those $t$ live nodes connects to nodes $i_1, \ldots, i_k$ and downloads only the one-dimensional subspace, corresponding to the $i$th row of $Ug^j \otimes I_m$, $j = 1, \ldots, k$. Again using the fact that we obtain $k$ columns of a Vandermonde matrix, the $i$th node computes a part of the object, given by the coefficients $\mathbf{o}_{i_1}, \mathbf{o}_{i_2+i}, \ldots, \mathbf{o}_{i_1+i(k-1)}$, and after the download phase, it can compute the inner product of the vector $(\mathbf{o}_{i_1}, \mathbf{o}_{i_2+i}, \ldots, \mathbf{o}_{i_1+i(k-1)})$ with the vector $Ug^j$, and thus obtains one of the $m$ coefficients it needs to store. The $m - 1$ missing coefficients are obtained through (partial) collaboration as follows: the $i$th node contacts...
nodes $i+1, \ldots, i+m-1$ (the indices are understood modulo the number of nodes). It already owns $(o_i, o_{m+i}, \ldots, o_{(k-1)m+i}) (Ug)^T$, and gets $(o_{i+1}, o_{m+i+1}, \ldots, o_{(k-1)m+i+1}) (Ug)^T$ from node $i+1$, and similarly from the other nodes, until $(o_{i+m-1}, o_{m+i+m-1}, \ldots, o_{(k-1)m+i+m-1}) (Ug)^T$ from node $i+m-1$.

5. Performance of Storage Codes

We briefly summarize known bounds on storage codes, to evaluate the proposed codes, and substantiate our earlier claim that they provide low repair bandwidth.

Recall that we encode an object $o$ of size $n$ over $\mathbb{F}_q$ into $Nm$ pieces and distribute them into $N$ storage nodes, each of the nodes stores $m$ coefficients in $\mathbb{F}_q$.

5.1. Bounds. Denote by $\alpha$ the storage capacity of every storage node, to keep the usual notation found in the literature, here, $\alpha = m$. The typical known trade-off bounds for storage codes involve $\alpha$, and the amount of data communicated during repair. They all assume that the storage codes have the property that any $k$ out of $N$ storage nodes contain enough information to recover the object $o$.

(1) If the system repairs one failure at a time, the communication cost in terms of bandwidth, called repair bandwidth, is $\gamma = d\beta$ per node involved in the repair, where $d$ is the number of live nodes contacted, and from each of them, $\beta$ coefficients in $\mathbb{F}_q$ are downloaded.

(2) If the system waits for a threshold $t$ of failures before starting the repair, then $\gamma = d\beta + (t - 1)\beta'$ per node, where $t$ nodes each download $\beta$ symbols of $\mathbb{F}_q$ from $d$ live nodes, and exchange $\beta'$ symbols among each others. This scenario falls into the collaborative repair setting [2, 3].

(3) The collaborative repair process can be refined so that the exchange phase of data does not involve all the repair nodes, which is called partially collaborative repair [9]. This is the most general setting as of now, and it subsumes the previous two cases. We introduce the parameter $s$ to characterize the degree of collaboration, and $t - s$ counts the number of repair nodes involved in the exchange process.

There are two extreme regimes when it comes to storage capacity and repair bandwidth: one minimizes the storage capacity, it is called the Minimum Storage Repair (MSR) point, while the other minimizes the repair bandwidth $\gamma$ (this is Minimum Repair Bandwidth (MBR) point. At MSR point, as mentioned in Subsection 2.2, the storage capacity $\alpha = m$ is $n/k$: indeed, this follows from the assumption that any $k$ nodes contain enough information to recover $o$ of length $n$. At MBR point, the minimum repair bandwidth $\gamma$ per node is equal to $\alpha = m$: this surely is the least amount of data to be transferred for a node to be repaired (see also Subsection 2.4).

What the trade-off computations between $\alpha$ and $\gamma$ tell [9] which is not obvious is that once $\alpha = m = n/k$, $\gamma$ cannot be arbitrarily low, and vice-versa, once $\gamma = \alpha$, $\alpha$ cannot be arbitrarily low. The lowest values that the different parameters achieve are in fact:

**MSR:** \[ \begin{align*}
\alpha &= \frac{n}{k}, \\
\gamma &= \frac{n}{k} \frac{d+s-t-s}{d-k+t-s+1}, \\
\beta &= \beta' = \frac{n}{k} \frac{d-k+t-s+1}{d-k+t-s+1}.
\end{align*} \]

**MBR:** \[ \begin{align*}
\alpha &= \gamma = \frac{n}{k} \frac{2d+t-s}{2d-k+t-s+1}, \\
\beta &= \frac{n}{k} \frac{2d-k+t-s+1}{2d-k+t-s+1}, \\
\beta' &= \frac{n}{k} \frac{2d-k+t-s+1}{2d-k+t-s+1}.
\end{align*} \]
The meaning of the parameters involved is summarized for convenience in Table 1.

When \( s = 1 \), we get the full collaboration case ((2) above), while \( s = t \) corresponds to the situation with no collaboration ((1) above).

### 5.2. Comparison

Let us consider the collaborative code construction of Subsection 3.2 for \( t = 2 \). An object \( o \) of length \( n \) is stored in \( N \) nodes, each of the nodes storing one codeword of the orbit code \( U_I G, U_I \in G_2(n-1, n) \) with canonical basis \( U_I \), thus \( \alpha = n - 1 \). The group \( G \) acting is \( G = \langle g \rangle \) with \( g \) the companion matrix of the polynomial \( p(x) \) whose order is \( l \), so that \( N = l \). When \( p(x) = \sum_{i=0}^{n} x^i \), we know from Proposition 2 that \( k = 2, \beta = n - 2, \) and \( \beta' = 1 \). We notice that the total amount of data downloaded for repair is \( 2(n - 2) + 2 = 2n - 2 \), which is \( n - 1 \) per node, suggesting that the code is at MBR. At MBR, we should have, using \( d = 1, k = 2, t - s = 1 \):

\[
\alpha = \gamma = \frac{3n}{4}, \quad \beta = \frac{n}{2}, \quad \beta' = \frac{n}{4}.
\]

When \( n = 4 \), the code in Subsection 3.2 has

\[
\alpha = \gamma = n - 1 = 3 = \frac{3n}{4}, \quad \beta = n - 2 = 2 = \frac{n}{2}, \quad \beta' = 1 = \frac{n}{4},
\]

which is indeed at MBR (\( \alpha \) is minimized). When \( n > 4 \), the code has parameters:

\[
\alpha = \gamma = n - 1 > \frac{3n}{4}, \quad \beta = n - 2 > \frac{n}{2}, \quad \beta' = 1 < \frac{n}{4},
\]

which, as mentioned above, satisfies \( \alpha = \gamma \), but \( \alpha \) is not minimized. The number of contacted nodes is optimal.

Let us consider the partially collaborative code construction of Subsection 3.2 for \( t \geq 3 \). As discussed above for the collaborative code construction, this partially collaborative code construction only changes the value of \( t \), but \( t - s \) still equals to 1. We know then that the repair bandwidth is still \( n - 1 \), which suggests that the code is at MBR. Since the optimal values of \( \alpha, \gamma \) at MBR point are related to the value of \( t - s \) which is not changed, the minimized \( \alpha \) is obtained by \( n = 4 \) (which indeed is a collaborative phase since three failures cannot happen when \( n = 4 \)). For \( n \geq 4 \), still \( \alpha = \gamma \), but \( \alpha \) is not minimized, while the number of contacted nodes stays optimal. The gap between the optimal storage and the value of \( \alpha \) for this construction is shown on Figure 1, for different values of \( n \) between 4 and 10. We remark that optimal code constructions at MBR for partial collaboration are not known as of now. This suggests that a deeper study of automorphism groups as a future work to get new/better codes for partial collaboration.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( n )</td>
<td>size of an object</td>
</tr>
<tr>
<td>( k )</td>
<td>any choice of ( k ) nodes allow the object retrieval</td>
</tr>
<tr>
<td>( \alpha )</td>
<td>storage capacity per node</td>
</tr>
<tr>
<td>( \gamma )</td>
<td>repair bandwidth per node</td>
</tr>
<tr>
<td>( \beta )</td>
<td>download repair bandwidth</td>
</tr>
<tr>
<td>( \beta' )</td>
<td>exchange repair bandwidth</td>
</tr>
<tr>
<td>( t )</td>
<td>threshold at which a repair process is triggered</td>
</tr>
<tr>
<td>( t-s )</td>
<td>number of other repair nodes involved in collaboration</td>
</tr>
</tbody>
</table>

Table 1. Parameters involved in the bounds for storage codes.
Consider finally the code given in Subsection 4.2. Set \( m \) to be \( t - s + 1 \), which sets the level of collaboration, namely there is collaboration among \( t - s \) repair nodes. Then the size of the object is \( n = k(t - s + 1) \), each node has a storage capacity of \( \alpha = t - s + 1 \), \( t - s \geq 1 \). This construction can been seen as a new interpretation of the construction of [10] which fits the minimum storage regime (MSR), since \( \alpha = n/k \). The total repair bandwidth \( \gamma \) is

\[
\gamma = k + (t - s), \quad 1 \leq s \leq t,
\]

since every repair node downloads \( k \) amount of data, and exchanges \( t - s \) of it.

REFERENCES


