Abstract—We study the exact and optimal repair of multiple failures in codes for distributed storage. More particularly, we examine the use of interference alignment to build exact scalar minimum storage coordinated regenerating codes (MSCR). We show that it is possible to build codes for the case of $k=2$ and $d \geq k$ by aligning interferences independently but that this technique cannot be applied as soon as $k \geq 3$ and $d > k$. Our results also apply to adaptive regenerating codes.

I. INTRODUCTION

Codes allow distributed storage systems to tolerate failures of some devices. Yet, repairing naively by downloading and decoding the whole code induces high repair costs. Regenerating codes reduce this cost by repairing without decoding. Optimal tradeoffs between storage and repair cost (network bandwidth) have been established both for the single failure case [1] and for the multiple failures case [2]–[4] using network information theory. Adaptive regenerating codes, which allow the number of devices involved to differ between repairs, have been defined in [2]. The two extreme points of the optimal tradeoffs are Minimum Bandwidth (MBR/MBCR), which minimizes repair cost first, and Minimum Storage (MSR/MSCR), which minimizes storage first as shown on Figure 1c. Codes matching these theoretical tradeoffs can be built using non-deterministic random linear network codes.

In this case, repairs are termed as functional repair (Figure 1a) for the regenerated data is not strictly equal to the lost data. However, such non-deterministic schemes are not desirable for they (i) require homomorphic hash functions to provide basic security (integrity checking), (ii) cannot be turned into systematic codes, which offer access to data without decoding, and (iii) they can only provide probabilistic guarantees. Deterministic schemes overcome these issues by offering exact repair (i.e., during a repair, the regenerated block is equal to the lost block and not only equivalent as shown on Figure 1b). Yet, it has been shown that exact repair is strictly harder than functional repair [5], [6], which means that the existence of functional regenerating codes does not imply that exact regenerating codes exist. Hence, an interesting question is whether the previous tradeoffs, which apply to functional repairs, can still be achieved for exact repairs. The problem of repairing exactly a single failure has been well studied [5]–[9]. However, the exact repair of multiple failures remains an open question since it has been studied only for the very specific setting $d = k$ [4], [10].

We focus on this problem, and extend our previous work on coordinated regenerating codes [2] with exact repair. We consider the case of $n, k, d > k, t > 1$ for scalar constructions (i.e., $\beta = 1$) and make the following contributions:

- In the line of exact scalar minimum storage regenerating codes [5], [8], [9], we propose exact scalar minimum storage coordinated regenerating codes (MSCR) for $n, k = 2, d \geq k, t = n - d$. This interference alignment based construction is inspired by [5], [9]. (Section III)
- Scalar MSR codes can be repaired exactly by aligning interferences independently [5], [7]. We show that when $k \geq 3$, aligning interferences independently is not sufficient to repair exactly scalar MSCR codes. (Section IV).

Note that these results, which correspond to the MSCR point, also apply to exact scalar adaptive regenerating codes [2].

Most previous works have been limited to single failures ($t = 1$). For multiple failures, there only exist results for the case $n, k, d = k, t = n - k$, a degenerated case where the repair of regenerating codes boils down to repairing, in parallel, $t$ independent erasure correcting codes [4]. A similar construction exists for MBCR codes [10].

The extended version of the article [11] can be referred to for more details on the codes constructions and the related work.

II. MODEL

We consider a system storing a file of $M$ bits spread onto $n$ devices (each storing $\alpha = \frac{M}{n}$ bits) such that the file can be recovered by collecting enough data from any $k$ devices. For repairing coordinated regenerating codes, each failed device contacts $d \geq k$ live devices and gets $\beta$ bits from each. The $t$ failed devices coordinate by exchanging $\beta'$ bits. The data is then processed and $\alpha$ bits are stored. These studies lead to the definition of the optimal tradeoffs between storage $\alpha$ and repair costs $\gamma = d\beta + (t - 1)\beta'$. The values of $\alpha$, $\beta$ and $\beta'$ corresponding to MSCR and MBCR codes are recalled on Figure 1c. In this paper, we will focus on MSCR constructions for they are very close to classical erasure correcting codes and are highly related to adaptive regenerating codes.

In the sequel of the article, we will study the exact repair of regenerating codes when multiple failures occur. We study the non-degenerated case of $d > k$ and use scalar codes ($\beta = 1$).

In the article, we use failed devices to designate either the devices that have failed, or the new spare devices that hold the repaired data. The meaning will be clear from the context.
We adopt the following conventions: data \( v \) and codewords \( w \) are column vectors, the generator matrix \( G \) is rectangular and the encoding operation \( w = Gv \) gives a column vector.

III. EXACT MSCR CODES FOR \( k = 2 \)

We consider a system storing a file of size \( M = k(d - k + t) \) split in \( k = 2 \) blocks \((a, b)\), each of size \( \alpha = d - k + t \) sub-blocks. The system consists of \( n = d + t \) devices as we assume that all failed devices and all live devices take part to the repair\(^2\). In the sequel of the article, we consider a finite field \( \mathbb{F}_q \) of size \( q \) and having a generator element \( \omega \).

The system is compounded of two devices storing the systematic part and \( s = n - 2 \) devices storing the redundancy part.

- The first systematic device stores \( a = (a_1, \ldots, a_\alpha)^t \).
- The second systematic device stores \( b = (b_1, \ldots, b_\beta)^t \).
- The \( i \)-th redundancy device, \( i \in \{0, \ldots, \alpha - 1\} \), stores
  \[ r_i = (a_1 + \omega^i \mod \alpha, a_\alpha + \omega^{i+\alpha-1} \mod \alpha) \]

An example for \( k = 2, d = 3 \) and \( t = 2 \) is given on Figure 3.

Using the previously defined code, we can state the two following theorems:

**Theorem 1.** Minimum storage coordinated regenerating codes can be repaired exactly (for the systematic devices) when \( n = d + t, k = 2 \) and \( t = 2 \) (i.e., multiple repairs are performed simultaneously). Similarly, adaptive regenerating codes with \( k = 2 \) can be repaired exactly.

**Proof:** In the sequel of this section, we show that (i) the code we define is an MDS code (i.e., the original data can be recovered from any \( k = 2 \) devices) and (ii) the two systematic devices can be repaired exactly. To repair exactly adaptive regenerating codes, we require that any single failure can be repaired, in addition to the two aforementioned properties.

A. The MDS Property

This property is trivially satisfied since, when fetching data from any two devices, we get \( \alpha \) groups of 2 equations over 2 unknowns, where each group concerns different unknowns. The \( i\)th group is about \( a_i \) and \( b_i \) and consist of 2 independent equations. Hence, the unknowns of each group can be recovered and the MDS property is satisfied.

\(^2\)The code we define and the proofs are given for \( n = d + t \) for the sake of clarity. However, the method can also be applied to codes where \( n > d + t \)

B. Repairing two failures

The repair consists of the following steps, which map onto the process defined in [2]. In this scheme, illustrated in Figure 2, we do not rely on random linear network coding but give a method for repairing exactly.

1. **Identify lost data.** Given the failure of any two devices (systematic or redundancy), we perform a change of variables to transform the actual code \( C \) into a code \( C' \), in which the failed devices are the systematic ones storing \( a = (a_1 \ldots a_\alpha)^t \) and \( b = (b_1 \ldots b_\beta)^t \). Such a code is guaranteed to exist since the original code is MDS (same argument as in [5]).

2. **Prepare (Collect).** Each live device that participates to the repair computes a sub-block to be sent to the first device and a sub-block to be sent to the second device. All the sub-blocks to be sent to the first device have the common property that the interfering information about \( b \) is aligned (i.e., the \( i\)-th live device, storing \( r_i \), uses \( v_{\alpha i} \) as a repair vector and sends \( v_{\alpha i} r_i = w_{\alpha i} a + z_i b \) so that the spare device receives different information about \( a \) but the same about \( b \). To build \( v_{\alpha i} \), given some arbitrary alignment vector \( z_0 \), and given that
  \[ r_i = A_i a + B_i b, \]
  the repair vector is \( v_{\alpha i} = z_i B_i^{-1} \). Since the MDS property is satisfied (i.e., we can recover from \( a \) and \( r_i \)), \( B_i \) is invertible, and the repair vector exists. The role of \( a \) and \( b \) are reversed for sub-blocks to be sent to the second device.

\[^3\]In this description, \( v_{\alpha i} r_i, v_{\alpha i} a \) or \( z_i, b \) are of scalars (i.e., the resulting matrices are of dimension \( 1 \times 1 \)). As a result \( c_a = (v_{\alpha i} r_i_1, \ldots, v_{\alpha i} r_i^n)^t \) is a matrix of size \( d \times 1 \) and \( (c_a | c_b v_{\alpha i}^t) \) a matrix of dimension \( (d + 1) \times 1 \).
3. **Transfer (Collect).** The sub-blocks are sent and the first (resp. second) device stores them temporarily as $c_a = (v_{a1}t_1, \ldots, v_{a\ell}t_d)^t$ (resp. $c_b$) for use in steps 4 and 6.

4. **Prepare (Coordinate).** Using what has been received in step 3, the second spare device prepares a sub-block $v_{a1}c_b = w_{a1}a + z_{a}b$ to be sent to the first spare device. The information about $b$ is aligned as in sub-blocks prepared during step 2. Again, the role $a$ and $b$ are reversed for the sub-block to be sent from the first to the second spare device.

5. **Transfer (Coordinate).** The sub-blocks are sent and the first (resp. second) spare devices add them to blocks received in step 3 thus storing $(c_a|v_{a0}c_b)^t$ (resp. $(c_b|v_{b0}c_a)^t$).

6. **Recove and Store.** The $d+1$ sub-blocks $(c_a|v_{a0}c_b)^t = (w_{a1}a + z_{a}b, \ldots, w_{a\ell}a + z_{a}b, w_{a0}a + z_{a}b)^t$ allow recovering both the interfering information $z_{a}b$ (but not the individual values of $b_i$), and the desired information $a = (a_1\ldots a_d)^t$. Indeed, the received sub-blocks define $d+1$ equations over $d+1$ unknowns $(z_{a}b, a_1, \ldots, a_d)$. The second spare device performs a similar processing with the role of $a$ and $b$ reversed.

We now apply this general method to the repair of the systematic part of the code we define, as shown on Figure 3. A finite field of size $q = n - 1$ is sufficient to offer the repair of the systematic devices using the previously defined code. In order to repair the two systematic devices on the code we defined, during the collecting step the $i$-th redundancy device sends $(\omega^{-(i \mod \alpha)}a_1, \ldots, \omega^{-(i + \alpha - 1 \mod \alpha)}a_{\alpha})t_i$ to the first device being repaired and $(1 \ldots 1)t_i$ to the second device being repaired. The vectors $v_{a\beta}$ (resp. $v_{b\beta}$) are chosen so that $z_{a\beta} = \sigma$ (resp. $z_{b\beta} = \sigma$) with $\sigma = (1 \ldots 1)$ ($\alpha$ terms). We note $c_a$ (respectively $c_b$) the vector of $d$ symbols received by the device repairing $a$ (respectively $b$).

At the coordination step, the first systematic device sends $(\omega^{0}a_1 + \cdots + \omega^{-(\alpha - 1)}a_{\alpha} + \sigma b)$ to the second one, while the second one sends $(\omega^{0}a_1 + \cdots + \omega^{\alpha - 1}a_{\alpha} + \sigma b)$ to the first one.

At the end of these steps, the first device has received $\alpha + 1$ equations. Let us note $\mu = 1 + \cdots + 1$ ($\alpha$ terms). Since the interfering information about $b_i$ is aligned, it writes as

\[
\begin{pmatrix}
\omega^0 a_1 + \cdots + \omega^{-(\alpha-1)} a_\alpha + \sigma b \\
\vdots \\
\omega^{-(\ell \mod \alpha)} a_1 + \cdots + \omega^{-(\ell + \alpha - 1 \mod \alpha)} a_\alpha + \sigma b \\
\vdots \\
\omega^{-(\alpha-1 \mod \alpha)} a_1 + \cdots + \omega^{-(2\alpha - 2 \mod \alpha)} a_\alpha + \sigma b \\
\end{pmatrix}
\]

As a consequence, it consists of a system of $\alpha + 1$ independent equations and $\alpha + 1$ unknowns $(a_{\alpha}s$ and $\sigma b)$. As a result, the $\alpha$ unknowns $a_i$ can be recovered. The second device has received something similar with the roles $a$ and $b$ exchanged.

This repair method applied to a code $(n = d + t, k = 2, d > k, t = 2)$ $(n = 5$ and $d = 3$ on Figure 3) naturally extends to other cases such as codes $n > d + t, k = 2, d > k, t = 2$.

C. **Repairing one device**

Finally, repairing one single device is easier, and interference alignment has already been used [5], [9]. However, we need to show that the code construction we present, which supports both repairs of single failures ($t = 2$) and repairs of two failures ($t = 2$), implies that it is possible to design exact scalar MSCR codes and exact scalar adaptive regenerating codes when $k = 2$, thus leading to Theorem 1.
IV. IMPOSSIBILITY OF INDEPENDENT INTERFERENCE ALIGNMENT FOR EXACT MSCR WHEN \( k \geq 3 \)

In this section, we examine whether the previous scheme, inspired by the repair of single failures [5], [9], can be applied to multiple failures when \( k \geq 3 \). This overall section is related to [5], [7] and similarly shows that requiring exact repairs over-constrains the system and requires alignment of information that cannot be aligned to maintain the MDS property thus leading to a contradiction.

When repairing a single failed systematic \(^4\) block \( a \), the information about the \( k - 1 \) other systematic blocks must be aligned as shown in [5]. In particular, it is required that blocks are aligned independently. Indeed, if we consider that the systematic devices send vectors \( v_\beta b, v_\gamma c, \ldots \), and that the \( i \)-th redundancy device sends \( v_\alpha a + v_\beta b + v_\gamma c, \ldots \), to the device repairing \( a \), then \( \text{colspan} (v_\beta b) = \text{colspan} (v_\beta b) \), \( \text{colspan} (v_\alpha a) = \text{colspan} (v_\alpha a) \) for all \( i \) (i.e., systematic blocks are considered independently and all the information about each interfering block received at the device performing the repair spans only one dimension).

We show that under this requirement, exact repair is not possible if \( k \geq 3 \). We give a proof, and explain the meaning of this impossibility on the information flow graph.

**Theorem 2.** When requiring interference alignment to be applied independently for each interfering systematic device, it is not possible to repair exactly MSCR codes with \( k \geq 3 \), \( t \geq 2 \), \( d > k \) in the scalar case (i.e., \( M = k(d - k + t) \) such that each device stores only \( d - k + t \) sub-blocks of size \( \beta = 1 \)). This impossibility also applies to adaptive regenerating codes.

**Proof:** Since any MDS code \( C \) can be turned into a systematic code \( C' \) (as explained in [5]), we base our proof on Lemma 3. Indeed, if it was possible to repair exactly MSCR codes with \( k \geq 3 \) and \( t \geq 2 \), it would be possible to build exact systematic MSCR codes. Moreover, if it was possible to exactly repair adaptive regenerating codes, it would be possible to derive MSCR codes by using the same repair method and restricting it to \( t = 2 \). Hence, the impossibility result extends to adaptive regenerating codes.

**Lemma 3.** When requiring interference alignment to be applied independently on all devices, it is not possible to repair exactly systematic MSCR codes with \( k \geq 3 \), \( t \geq 2 \), \( d > k \) in the scalar case (i.e., \( \beta = 1 \)).

**Proof:** Let us consider a code with \( k \geq 3 \), \( t \geq 2 \), \( d > k \), \( n > d + t \) and \( \alpha = d - k + t \). Let us assume that we want independent interference alignment (i.e., each interfering block spans only a sub-space of dimension 1).

The \( k \) first devices store systematic blocks as vectors \( a = (a_1)_{1 \leq i \leq \alpha}, b = (b_1)_{1 \leq i \leq \alpha}, c = (c_1)_{1 \leq i \leq \alpha} \ldots \) The \( n - k \) remaining devices store redundancy blocks as \( r_j = A_1 a + B_1 b + C_1 c + \ldots \). Thus leading to a set-up similar to the one depicted on Figure 4.

We are going to prove, by contradiction, that exact repair of systematic codes in the scalar case (i.e., \( \beta = 1 \)) is not achievable when \( k \geq 3 \) and \( t \geq 2 \). For the sake of clarity, our proof will describe the case of \( t = 2 \), \( k = 3 \) and \( d = 4 \) but it naturally extends to any larger value.

Assume that it is possible to repair exactly. Hence, it is possible to repair the simultaneous failure of devices storing \( a \) and \( b \). We consider this case and examine how exact repair constraints the system.

For each device being repaired, all live devices project what they store onto a single vector and send this vector to the said device being repaired. Then, the devices being repaired coordinate by exchanging a single vector (a projection of what they have received so far). Hence, the device repairing \( a \) receives, at the end of both the collecting step and the coordination step:

\[
\begin{bmatrix}
0 \\
v_{a1} A_1 \\
v_{a2} A_2 \\
v_{a3} A_3 \\
w_A \\
\end{bmatrix} a + \begin{bmatrix}
0 \\
v_{a1} B_1 \\
v_{a2} B_2 \\
v_{a3} B_3 \\
w_B \\
\end{bmatrix} b + \begin{bmatrix}
v_\gamma \\
v_{a1} c_1 \\
v_{a2} c_2 \\
v_{a3} c_3 \\
w_C \\
\end{bmatrix} c
\]

(1)

To be able to recover \( a \), we must be able to decode the \( d - k + t + 3 \) desired unknowns of \( a \) out of the \( d + t - 1 = 5 \) equations containing a total of \( k(d - k + t) = 9 \) unknowns.

The same applies for \( b \). Hence, when aligning independently we must have,

\[
\text{rank} \left( \begin{bmatrix} v_\alpha c_1 \\ v_{a1} c_1 \\ v_{a2} c_2 \\ v_{a3} c_3 \\ w_C \end{bmatrix} \right) = 1, \quad \text{rank} \left( \begin{bmatrix} v_\gamma \\ v_{a1} b_1 \\ v_{a2} b_2 \\ v_{a3} b_3 \\ w_B \end{bmatrix} \right) = 1
\]

(2)

\[
\text{rank} \left( \begin{bmatrix} z_C \\ v_\gamma' \\ v_{a1} c_1 \\ v_{a2} c_2 \\ v_{a3} c_3 \end{bmatrix} \right) = 1, \quad \text{rank} \left( \begin{bmatrix} z_B \\ 0 \\ v_{a1} b_1 \\ v_{a2} b_2 \\ v_{a3} b_3 \end{bmatrix} \right) = 3
\]

(3)

Let us consider the choice of vectors \( v_\alpha, v_{a1}, v_\gamma, \) and of matrices \( C_i \) that allows exact repairs (i.e., such that constraints on ranks are satisfied) with coordination (i.e., \( k \geq 3 \) and \( t \geq 2 \)):

- All \( v_\alpha, C_i \) must be collinear according to (2).
- All \( v_{\beta i}, C_i \) must be collinear too according to (3).
- During the coordination step, what is sent by the device repairing \( a \) will necessarily be collinear to \( v_{a1} C_1 \) (i.e., what is stored) and to vector \( v_\gamma \). Let us name this vector, which is collinear to \( v_\gamma, z_c \). According to (3), \( z_c \), and hence \( v_\gamma \) must be collinear to all \( v_{\beta i} C_i \). Hence, we have:
  \( \forall i, v_{a1} = v_i (v_\gamma C_i^{-1}) \) and \( v_{\beta i} = \mu_i v_\gamma C_i^{-1} \). Note that the matrix \( C_i \) is invertible to guarantee the MDS property.

As a result, for all \( i \in \{1, \ldots, d\} \), vectors \( v_{a1} \) and \( v_{\beta i} \) are collinear since

\[
v_{a1} = \frac{v_i}{\mu_i} v_{\beta i}
\]

(4)

Let us consider the choice of matrices for \( B_i \) that allows exact repairs on the device repairing \( a \). According to (2), we

\[^4\]Again, the repair of a redundancy block in a code \( C \) is equivalent to the repair of a systematic block in a code \( C' \).
must have rank \((\mathbf{B}_1 \mathbf{v}_{\alpha_1}, \ldots, \mathbf{B}_d \mathbf{v}_{\alpha_d}) = 1\), which is equivalent to:

\[
\rho_1 \mathbf{v}_{\alpha_1} \mathbf{B}_1 = \rho_2 \mathbf{v}_{\alpha_2} \mathbf{B}_2 = \cdots = \rho_d \mathbf{v}_{\alpha_d} \mathbf{B}_d
\]

Combining (4) and (5), we can deduce that

\[
\frac{\mu_1}{\rho_1} \mathbf{v}_{\beta_1} \mathbf{B}_1 = \frac{\mu_2}{\rho_2} \mathbf{v}_{\beta_2} \mathbf{B}_2 = \cdots = \frac{\mu_d}{\rho_d} \mathbf{v}_{\beta_d} \mathbf{B}_d
\]

As a result, rank \((\mathbf{B}_1 \mathbf{v}_{\beta_1}, \ldots, \mathbf{B}_d \mathbf{v}_{\beta_d}) = 1\), which is in contradiction with (3) (i.e., \(b\) can be repaired too, or rank \((\mathbf{B}_1 \mathbf{v}_{\beta_1}, \ldots, \mathbf{B}_d \mathbf{v}_{\beta_d}) \geq d - 1\)). Hence, the exact repair of two failed devices when \(k \geq 3\) is impossible. The proof naturally extends to any higher value of \(k\) and \(t\). Hence, repairing exactly scalar (i.e., \(\beta = 1\)) codes with \(d > k\), \(k > 2\), and \(t > 1\) is impossible when relying on independent interference alignment.

This impossibility means that at some point, the amount of information that goes through the information flow graph [1], [2] is too low. Indeed, to ensure that the file is kept over time, all cuts between the source \(S\) and any data collector \(DC\) in a graph representing the transfer of data between devices during repairs must be greater than or equal to \(\mathcal{M}\) [2]. However, if we consider the graph of Figure 5 and force the device storing \(c\) to send the same \(\beta\) bits of information (by requiring alignment) to both the device storing \(a\) and the device storing \(b\), then the cut shown on the graph of Figure 5 has an insufficient capacity.

**V. CONCLUSION**

In this paper, we applied independent interference alignment to minimum storage coordinated regenerating codes (MSCR) and show that this technique allows exact repair if and only if \(k = 2\). Our results also apply to adaptive regenerating codes thus providing an interesting solution for the implementation of practical systems when \(k = 2\).

To overcome the impossibility shown in this paper, several tracks can be considered: (i) considering a technique that does not align the interferences independently, (ii) building vector codes (i.e., relying on sub-packetization with \(\beta > 1\) by opposition to scalar codes \(\beta = 1\) considered in this paper), or (iii) building minimum bandwidth coordinated regenerating codes (MBCR) (for single failure, codes exist for all parameters [8]).

**REFERENCES**

[1] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. O. Wainwright, and K. Ramchandran, “Network Coding for Distributed Storage Systems,” *IEEE Transactions On Information Theory*, vol. 56, pp. 4539–4551, 2010.

[2] A. Kermarrec, N. Le Scouarnec, and G. Straub, “Repairing Multiple Failures with Coordinated and Adaptive Regenerating Codes,” in *NetCod*, July 2011.

[3] Y. Hu, Y. Xu, X. Wang, C. Zhan, and P. Li, “Cooperative Recovery of Distributed Storage Systems from Multiple Losses with Network Coding,” *IEEE Journal on Selected Areas in Communications*, vol. 28, pp. 268–276, 2010.

[4] K. W. Shum, “Cooperative Regenerating Codes for Distributed Storage Systems,” in *ICC*, 2011.

[5] N. B. Shah, K. Rashmi, P. V. Kumar, and K. Ramchandran, “Interference Alignment in Regenerating Codes for Distributed Storage: Necessity and Code Constructions,” *Transaction on Information Theory*, vol. 58, pp. 2134–2158, 2012.

[6] ———, “Distributed Storage Codes with Repair-by-Transfer and Non-achievability of Interior Points on the Storage-Bandwidth Tradeoff,” *Transaction on Information Theory*, vol. 58, pp. 1837–1852, 2012.

[7] ———, “Explicit Codes Minimizing Repair Bandwidth for Distributed Storage,” in *ITW*, 2010.

[8] K. V. Rashmi, N. B. Shah, and P. V. Kumar, “Optimal Exact-Repair Regenerating Codes for Distributed Storage at the MSR and MBR Points via a Product-Matrix Construction,” *IEEE Transaction on Information Theory*, vol. 57, pp. 5227–5239, 2011.

[9] C. Sah and K. Ramchandran, “Exact-Repair MDS code construction using interference alignment,” *IEEE Transactions On Information Theory*, vol. 57, pp. 1425–1442, 2011.

[10] K. W. Shum and Y. Hu, “Exact Minimum-Repair-Bandwidth Cooperative Regenerating Codes for Distributed Storage Systems,” in *ISIT*, 2011.

[11] N. Le Scouarnec, “Exact Scalar Minimum Storage Coordinated Regenerating Codes,” *ArXiv e-prints*, 2012, arXiv:1202.0457.