Solving Monoshift Systems and Applications in Random Coding

Yiheng Zhang, Ximing Fu, Xuanchen Wu, Shenghao Yang and Kenneth W. Shum

***Abstract*—A monoshift matrix is a matrix that has binary polynomials of degree at most 1 as entries, and a monoshift system is a system of linear equations over polynomials with a monoshift coefficient matrix. We propose an algorithm called augmented elimination to reduce a monoshift matrix to a form called augmented echelon form of degree at most 1. The monoshift system in augmented echelon form can be solved efficiently by successive cancellation. We further derive a recursive formula of the rank distribution of a uniformly random monoshift matrix. For a square uniformly random monoshift matrix, the deficient-rank probability decreases to 0 almost exponentially fast as the matrix size increases. This is quite different compared with the square random matrices over a fixed finite field, where the deficient-rank probability increases when the matrix size increases. Certain coding problems can benefit from this special property of monoshift systems, as demonstrated by the applications in distributed storage systems with decentralized encoding and in batched network coding.**

I. Introduction

Finite fields have been extensively used for error correcting codes, storage codes and network codes (see e.g., [1]–[4]). Its rich algebraic structure adds to the prosperity of the coding theory. Its properties, however, may also limit the performance of codes in certain aspects. It is well known that for a $k \times k$ uniformly random matrix M over the finite field $\mathbb{F}_q$ of q elements (i.e., each entry is uniformly at random chosen from $\mathbb{F}_q$), the probability to have M full rank is lower than $1 - 1/q$ and decreases as k increases. Due to this property, many results in coding theory may require a sufficiently large finite field. For example, in the random linear network coding theory of [5], the field size is required to be sufficiently large so as to achieve the cut-set bound.

Using a larger finite field may incur a higher cost of encoding and decoding computation. In an attempt to reduce the complexity of field operations, alternative algebraic structures have been studied in literature for codes with lower computation costs [6], [7]. For example, shift and XOR operations have been successfully applied in designing structured storage codes [8], [9] and regenerating codes [10], [11], where the generator matrices include only monomials of unbounded degrees. Circular shift and XOR operations have been applied to construct network codes [12].

In this paper, we study an algebraic structure that can potentially have a better random coding performance and lower computation costs than finite fields. A *monoshift matrix* is a matrix of polynomials of degree at most 1, and a *monoshift system* is a system of linear equations over polynomials with a monoshift coefficient matrix. Compared with the generator matrices of the shift-XOR codes, a monoshift matrix has a degree bound and can include polynomial $1 + z$ that cannot be used in shift-XOR codes [8], [9], [11], where z is the indeterminate.

The efficient zigzag algorithms of solving shift-XOR codes cannot be applied directly to solve a monoshift system. In existing approaches of solving a general system of linear equations over polynomials (e.g., [13]), the coefficient matrix is transformed to the row echelon form. As the row echelon form of a monoshift matrix may have a degree larger than 1, this approach may induce a higher computation cost. For monoshift systems, we propose an algorithm called *augmented elimination* that reduces the coefficient matrix to what we call an augmented echelon form of degree at most 1 (see Sec. II). From the augmented echelon form, the rank of a monoshift matrix is determined and whether the corresponding monoshift system has a unique solution is known. If uniquely solvable, the monoshift system in augmented echelon form can be solved efficiently by successive cancellation similar to the zigzag algorithms discussed in [8], [9], [11].

We further derive a recursive formula of the rank distribution of a uniformly random monoshift matrix (see Sec. III-C). For a $k \times k$ uniformly random monoshift matrix G, our evaluation shows that the deficient-rank probability of G decreases to 0 almost exponentially fast as k increases. To illustrate, the deficient-rank probability is 0.0002 for $k = 16$ and more than 10^{-8} for $k = 32$. In contrast, for the uniformly random $k \times k$ matrix M over finite fields to achieve at least the same full-rank probability, the field size is at least 2^{13} for $k = 16$ and 2^{28} for $k = 32$.

Two coding applications of monoshift systems are introduced in Sec. IV. The first is a distributed storage system with decentralized encoding, for which a sparse random linear coding scheme using finite fields was proposed [14]. A similar scheme based on monoshift operations can guarantee a high probability of successful decoding when the number of data nodes is slightly large, e.g., 16. Due to the shift-operation, the coded sequence is 1 bit longer than that of the data sequences, which is a much smaller storage overhead compared with the structured shift-XOR codes [8], [9], [11].

We also show how to use monoshift operations in BATS codes [15], which is a class of efficient random linear network coding schemes for networks with packet loss. For the existing outer code of a BATS code employing finite field operations, it is not efficient to use a large batch size, e.g., 64 and 128, due to the computation cost. Using monoshift operations in place

The authors are with The Chinese University of Hong Kong, Shenzhen.

of the finite field operations, outer codes with a relatively large batch size becomes feasible. Together with a proper recoding scheme, BATS codes with monoshift operations can have a better tradeoff between achievable rate and computation cost than the existing ones.

II. Preliminaries

A polynomial x over the binary field $\{0,1\}$ of degree less than L is

$$x(z) = \sum_{i=0}^{L-1} x[i]z^i, \quad x[i] \in \{0,1\},$$

where $x[i]z^i$ is called the degree-i term. A *right shift* of $x(z)$ is defined as the product $zx(z)$. If $x(z)$ has the constant term $x[0]=0$, then it is said to be a *multiple* of z, or *left shiftable*, and its *left shift* is defined as $z^{-1}x(z)$.

We study systems of polynomial arithmetic that involve vectors and matrices of polynomials. If $x_1(z),\dots,x_k(z)$ are a sequence of polynomials, they form a column vector $\mathbf{x} = \begin{pmatrix} x_1 & \cdots & x_k \end{pmatrix}^\top$. The *degree* of the vector $\mathbf{x}$ is defined as the highest degree of polynomials in $\mathbf{x}$. Write $\mathbf{x}[i]$ as the vector of coefficients of the degree-i-terms in $\mathbf{x}$, i.e., $\mathbf{x}[i] = \begin{pmatrix} x_1[i] & \cdots & x_k[i] \end{pmatrix}^\top$. Immediately, $\mathbf{x} = \sum_{i=0}^{L-1} \mathbf{x}[i]z^i$.

In addition, our definition of right/left shift applies to vectors of polynomial element-wise. For $\mathbf{x} = (x_i)$, $z\mathbf{x} = (zx_i)$ and if $\mathbf{x}[0] = 0$, then $\mathbf{x}$ is said to be a *multiple* of z or *left shiftable*, and $z^{-1}\mathbf{x} = (z^{-1}x_i)$.

The above discussions about column vectors extend naturally to row vectors and also to matrices of polynomials.

III. Solving Monoshift Systems

Definition 1 (monoshift matrix)**.** A matrix G whose entries are composed of polynomials in the set $\{0,1,z,z+1\}$ is called a *monoshift* matrix. G admits the decomposition $G = A + Bz$ for some binary matrices A, B.

Definition 2 (monoshift system)**.** For a vector $\mathbf{y}$ of n polynomials, and an $n \times k$ monoshift matrix G, the equation

$$G\mathbf{x} = \mathbf{y} \tag{1}$$

is called a *monoshift system.* A vector $\mathbf{x}$ of k binary polynomials satisfying (1) is said to be a *solution* to the system.

In the existing approaches of solving a general system of linear equations over polynomials (e.g., [13]), the coefficient matrix is transformed to the *row echelon form* using a Gaussian-elimination-like process. The row echelon form of a monoshift matrix, however, may have a degree larger than 1. Therefore, this approach may incur a higher computation cost. For monoshift system, we present a more efficient way to find solutions.

A. Back Substitution

We start with a special case where $G = A + Bz$ is square and its constant term A is unit upper triangular, i.e., upper triangular with diagonal entries all 1. Note that G may not be upper triangular as B can be an arbitrary binary matrix. Suppose for certain integer $L > 1$,

$$\mathbf{x} = \sum_{i=0}^{L-1} \mathbf{x}[i]z^i \quad \text{and} \quad \mathbf{y} = \sum_{i=0}^{L} \mathbf{y}[i]z^i.$$

The system (1) can be written as

$$\sum_{i=1}^{L-1} (B\mathbf{x}[i-1] + A\mathbf{x}[i])z^i + B\mathbf{x}[L-1]z^L + A\mathbf{x}[0] = \sum_{i=0}^{L} \mathbf{y}[i]z^i.$$

As the coefficients on both sides match by degree, we obtain

$$\begin{aligned} A\mathbf{x}[0] &= \mathbf{y}[0] \\ A\mathbf{x}[1] &= \mathbf{y}[1] + B\mathbf{x}[0] \\ &\vdots \\ A\mathbf{x}[L-1] &= \mathbf{y}[L-1] + B\mathbf{x}[L-2]. \end{aligned} \tag{2}$$

Since A is unit upper triangular, one applies back substitution to uniquely solve the above system of equations from top to bottom.

Theorem 1. *If $G = A + Bz$ is a $k \times k$ monoshift matrix where A is unit upper triangular, the monoshift system $G\mathbf{x} = \mathbf{y}$ admits a unique solution for all polynomial vector $\mathbf{y}$, and the solution is solved within $(k^2 + k(k-1)/2)L$ XOR operations.*

Proof. Solving each equation (except the first) requires one vector multiplication, one vector addition, and one Gaussian back substitution, which cost $k(k-1)/2$, k, and $(k-1)k$ XOR operations, respectively. In total there are L such substitution, $L-1$ such additions and multiplications. Therefore $\left(k^2 + k(k-1)/2\right) L$ XOR operations. □

B. Augmented Elimination

Analogous to Gaussian elimination, we introduce a process that transforms an $n \times k$ monoshift matrix to a standard form so that we can determine whether the corresponding monoshift system admits a unique solution. The *augmented matrix* of a monoshift matrix $G = A + Bz$ is defined as $(A \mid B)$.

Definition 3 (Augmented echelon form)**.** A monoshift matrix $G = A + Bz$ is of *augmented echelon form* if its augmented matrix is of the form

$$\left(\begin{array}{c|c} D & E \\ 0 & 0 \end{array}\right), \tag{3}$$

where D is a binary matrix of row echelon form, subject to perhaps a certain row interchanging, and has no zero rows; and 0 denotes an all-zero block.

Notice that a monoshift matrix of augmented echelon form may not be of the row echelon form as the matrix E can be any binary matrix. For a general $n \times k$ monoshift matrix G, if G is of augmented echelon form as in (3), and D has k rows, then D (subject to row interchanging) is unit upper triangular. Hence, a monoshift system generated by G is uniquely solvable by back substitution. So we discuss how to transform G to augmented echelon form.

Definition 4 (Elementary row operations)**.** An *elementary row operation* on a monoshift matrix has one of the following types:

1) Add a row to one of the other rows;
2) Interchange two rows;
3) For a row that is a multiple of z, divide it by z, i.e. perform a left shift.

The first two types of elementary row operations on G are equivalent to the corresponding elementary row operations on the augmented matrix of G. Suppose the ith row of G is $\mathbf{r}z$. If we perform the third type of elementary row operation on the ith row of G, the effect on the augmented matrix can be illustrated as follows:

$$\left(\begin{array}{c|c} * & * \\ 0 & \mathbf{r} \\ * & * \end{array}\right) \xrightarrow{\text{Type 3 operation on row } i} \left(\begin{array}{c|c} * & * \\ \mathbf{r} & 0 \\ * & * \end{array}\right).$$

The three types of elementary row operations are all invertible. If we perform an elementary row operations on both sides of a monoshift system, the solution set is preserved. Now we start to introduce *augmented elimination*, which performs a sequence of elementary row operations to reduce an $n \times k$ monoshift matrix $G = A + Bz$ to augmented echelon form. We use the following augmented matrix as a run-through example:

$$\left(\begin{array}{ccc|ccc} 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 & 1 \end{array}\right). \tag{4}$$

The augmented elimination has the following steps.

1) Gaussian elimination: Perform Gaussian elimination on the augmented matrix of G so that A is reduced into row echelon form:

$$(A \mid B) \to G_1 := \left(\begin{array}{c|c} D & E \\ 0 & F \end{array}\right)$$

where all rows in D are nonzero. Stop the Gaussian elimination at the moment the left half of the augmented matrix is in row echelon form. If F is zero or empty, then the resulting matrix is in augmented echelon form and the augmented elimination stops. For the example in (4), this step generates

$$\left(\begin{array}{ccc|ccc} 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{array}\right). \tag{5}$$

2) Left shift (special form): If F is nonempty, pick any row $\mathbf{r}$ of F, and perform a Type 3 elementary row operation (left shift) on the row of G_1 containing $\mathbf{r}$. Denote by F' the matrix obtained by removing row $\mathbf{r}$ from F. There are two cases:

- If $\mathbf{r} = 0$, move the all-zero row to the bottom and obtain

$$G_2 := \left(\begin{array}{c|c} D & E \\ 0 & F' \\ 0 & 0 \end{array}\right).$$

In this case, this step is repeated with F' in place of F.

- If $\mathbf{r} \neq 0$, the augmented matrix after the left shift becomes

$$G_3 := \left(\begin{array}{c|c} \mathbf{r} & 0 \\ D & E \\ 0 & F' \end{array}\right),$$

where we also perform row interchangings to move the row containing $\mathbf{r}$ to the first row.

For the example in (5), this step gives

$$\left(\begin{array}{ccc|ccc} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 \end{array}\right). \tag{6}$$

3) All-in elimination (special form): Find a row (called a pivot row) in $(D \mid E)$ block of G_3 that has the leading entry at the same column as the leading entry of the first row of G_3. Since D is in the row echelon form, there exists at most one such a row. If a pivot row exists, we repeat the following operations: Find a row (called the jth row) in G_3 different from the pivot row whose leading entry shares the same column as the leading entry of the pivot row, and add the jth row to the pivot row (i.e., eliminate the leading entry of the pivot row). When the leading entry of the pivot row cannot be further eliminated, the augmented matrix becomes

$$G_4 := \left(\begin{array}{c|c} \mathbf{r} & 0 \\ D' & E' \\ 0 & F'' \end{array}\right),$$

where the pivot row is moved by row interchanging so that D' is also in row echelon form. If the pivot row has the first k entries 0, it is included the block of $(0 \mid F'')$, so that all the rows of D' are nonzero. For the example in (6), this step generates

$$\left(\begin{array}{ccc|ccc} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{array}\right). \tag{7}$$

Define a general form of the augmented matrix:

$$G_5 := \left(\begin{array}{c|c} C & 0 \\ D & E \\ 0 & F \\ 0 & 0 \end{array}\right) \tag{8}$$

where C, D, E, F are binary matrices, and $\begin{pmatrix} C \\ D \end{pmatrix}$ is of row echelon form by perhaps some interchanging of rows and contains no all-zero rows. We see that G_1, G_2 and G_4 are special cases of G_5. So after the processing of the first three steps, we would obtain an augmented matrix of the form G_5. The following three steps of augmented elimination are performed on G_5 to generate a new matrix of the same form, and are performed iteratively until the F block is empty.

4) Left shift (general form): This step is similar to Step 2 except that a matrix of more general form G_5 is processed. If F is nonempty, pick arbitrarily a row $\mathbf{s}$ of F. We perform a left shift on this row and move this row to the top:

$$G_6 := \left(\begin{array}{c|c} \mathbf{s} & 0 \\ C & 0 \\ D & E \\ 0 & F' \\ 0 & 0 \end{array}\right)$$

where F' denotes F with row $\mathbf{s}$ removed. For the example in (7), this step generates

$$\left(\begin{array}{ccc|ccc}1 & 0 & 0 & 0 & 0 & 0\\ 1 & 1 & 1 & 0 & 0 & 0\\ 0 & 1 & 0 & 1 & 1 & 0\end{array}\right). \tag{9}$$

5) Half-in elimination: Repeat the following operations to cancel the leading entry of the first row of G_6: Find a row in $(C \mid 0)$ with the leading entry sharing the same column as that of the first row, and add the row to the first row. Denote by $(\mathbf{r} \mid 0)$ the first row until the leading entry cannot be further cancelled. There are two cases:

- If $\mathbf{r} = 0$, the first row is moved to the bottom, obtaining
$$G_7 := \left(\begin{array}{c|c}C & 0\\ D & E\\ 0 & F'\\ 0 & 0\end{array}\right)$$
which is of the same form as G_5. Continue with Step 4.
- If $\mathbf{r} \neq 0$, the augmented matrix is now of the form
$$G_7' := \left(\begin{array}{c|c}\mathbf{r} & 0\\ C & 0\\ D & E\\ 0 & F'\\ 0 & 0\end{array}\right).$$

For the example in (9), this step gives

$$\left(\begin{array}{ccc|ccc}0 & 1 & 1 & 0 & 0 & 0\\ 1 & 1 & 1 & 0 & 0 & 0\\ 0 & 1 & 0 & 1 & 1 & 0\end{array}\right). \tag{10}$$

6) All-in elimination (general form): This step is similar to Step 3 except that a matrix of more general form G_7' is processed, and the description is not repeated here. Denote C' as the matrix formed by C and $\mathbf{r}$ in G_7'. The matrix obtained after all-in elimination is of the form

$$G_8 := \left(\begin{array}{c|c}C' & 0\\ D' & E'\\ 0 & F''\\ 0 & 0\end{array}\right).$$

The leading entry of $\mathbf{r}$ does not share the column as that of any other row in C and D'. Therefore, the matrix $\begin{pmatrix}C'\\ D'\end{pmatrix}$ is in row echelon form (by perhaps some row interchanging), and hence G_8 has the same form as G_5. If F'' is nonempty, continue with Step 4. For the example in (10), this step gives

$$\left(\begin{array}{ccc|ccc}0 & 1 & 1 & 0 & 0 & 0\\ 1 & 1 & 1 & 0 & 0 & 0\\ 0 & 0 & 1 & 1 & 1 & 0\end{array}\right),$$

which is of augmented echelon form.

For each iteration of Step 4 – 6, either a zero row is added to the bottom block, or a row is added to the C block in G_5. Therefore, Step 4 – 6 can be repeated at most n times, resulting a matrix in the form of G_5 with empty F block and $\begin{pmatrix}C\\ D\end{pmatrix}$ of row echelon form by perhaps some interchanging of rows, fulfilling the criteria of augmented echelon form. Step 2 – 3 can be viewed as a simplified version of Step 4 – 6 applied on the special case G_1 of G_5.

The above process is summarized to the following theorem.

Theorem 2. *Suppose $G = A + Bz$ is an $n \times k$ monoshift matrix. There exists a sequence of elementary row operations, containing at most $n(n-1)$ row additions, that transforms G into augmented echelon form.*

Remark 1. In terms of a matrix of polynomials, the *rank* of a monoshift matrix G is defined as its maximal number of linearly independent rows. Applying elementary row operations on G does not affect its rank. Therefore, the number of nonzero rows in the augmented echelon form of G gives its rank. In addition, an $n \times k$ monoshift matrix is said to be full-rank if its rank is equal to $\min(n, k)$.

Now we summarize how to solve system (1) in general. First, apply a sequence of elementary row operations on both sides of (1) to transform G to augmented echelon form. If the rank of G is k, the system can be solved by back substitution provided there exists a solution. The total XOR operations for solving (1) is bounded by $(2n-1)n(k+L)+(k^2+k(k-1)/2)L$ to decode nL bits in $\mathbf{y}$, or $O(2n + 1.5k)$ per bit if $n, k \ll L$.

C. Rank Distribution of Random Monoshift Matrix

Denote by $\mathbb{F}_q$ the finite field of q elements. A matrix over $\mathbb{F}_q$ is said to be uniformly random if all the entries are uniformly at random chosen from $\mathbb{F}_q$. Define $\xi_0^k(q) = 1$ and for $r > 0$,

$$\xi_r^k(q) := (1 - q^{-k})(1 - q^{-k+1}) \cdots (1 - q^{-k+r-1}),$$

which is known as the probability that a $k \times r$ uniformly random matrix over $\mathbb{F}_q$ have rank r. Define

$$\xi_v^{n,k}(q) = \frac{\xi_v^n(q)\xi_v^k(q)}{\xi_v^v(q)q^{(n-v)(k-v)}},$$

which is the probability that an $n \times k$ uniformly random matrix over $\mathbb{F}_q$ have rank v.

By analyzing the augmented elimination process, we can determine the rank distribution of a uniformly random monoshift matrix.

Theorem 3. *The probability that a random monoshift matrix with entries uniformly and independently chosen from the set $\{0, 1, z, z+1\}$ have rank r is*

$$p_{n,k}(r) = \sum_{\substack{c+d=r\\ 0 \le d,c \le r}} P(c, d, 0),$$

for all $0 \le r \le \min\{n, k\}$, where the probability mass function $P(c, d, f)$ is defined recursively as follows:

$$\begin{aligned}P(c, d, f) = {} & 2^{c-k}P(c, d, f+1)\\ & +(1 - 2^{c+d-k-1})P(c-1, d, f+1)\\ & +((2^{d+1} - 1)/2^{k-c+1})P(c-1, d+1, f)\end{aligned}$$

for all $c > 0$ and $c+d+f \le n$ with the following boundary and initial conditions:

$$\begin{aligned} P(c,d,f) &= 0, && \textit{for } c+d+f > n \textit{ or } c < 0;\\ P(0,d,n-d) &= \xi_d^{n,k}(2), && \textit{for } 0 \le d \le n. \end{aligned}$$

Proof. The rank of G is determined by its augmented echelon form, which can be obtained by the augmented elimination process introduced in Sec. III-B. augmented elimination is applied iteratively to transform G to the form of G_5 in (8) until the F part is empty. We use a triple (c,d,f) to specify the state of augmented elimination, where c, d and f are the numbers of rows of C, D and F of G_5 , respectively. Denote $\Pr(c,d,f)$ as the probability to reach such a state.

Initially, after Step 1, P is empty, i.e. $p = 0$, and the state is $(0,d,n-d)$ with probability $\xi_d^{n,k}$. Note that the Gaussian elimination in Step 1 stops after transforming the first k columns to the row echelon form. Because elementary row operations are invertible, F and E in G_1 are still uniformly random, and stochastically independent with D.

Now we discuss the evolving of the state from Step 4. (Step 2 is a special case of Step 4 and hence is not discussed separately.) Assume that in G_5, E and F are uniformly random and is stochastic independent with C and D, and the state of G_5 is (c,d,f). At Step 4, a row $\mathbf{s}$ of S is picked uniformly at random and left-shifted. One of the following mutually exclusive events happens:

1) E_1: $\mathbf{s}$ is a linear combination of the rows of C.
2) E_2: not E_1 and $\mathbf{s}$ is a linear combination of the rows of C and D.
3) E_3: $\mathbf{s}$ is not a linear combination of the rows of C and D.

As the rows of C are linearly independent, E_1 occurs with probability 2^{c-k}. In this case, we reach the state of G_7, which is $(c,d,f-1)$, and has F' uniformly random.

As the rows of C and D are linearly independent, E_3 occurs with probability $1-2^{c+d-k}$. In this case, we reach G_8 where F'' is F with one row removed. The new state is $(c+1,d,f-1)$.

Last, E_2 occurs with probability $2^{c+d-k} - 2^{c-k}$. In this case, we reach the G_8 where F'' is F with one row removed and one row added from E. The new state is $(c+1,d-1,f)$.

When F part is empty, a state $(c,d,0)$ is reached, and the elimination process concludes the rank of G as the sum of rows in C and D, i.e., $c+d$. Therefore, the probability for G to have rank r is given by the sum of probability masses of all such states $(c,d,0)$ with $c+d=r$. □

For the $k \times k$ uniformly random matrix M over $\mathbb{F}_q$, the probability that M has rank k is $\xi_k^k(q)$, which for any fixed q, decreases as k increases. In contrast, using the formula in Theorem 3, the probability that the uniformly random $k \times k$ monoshift G is not full rank is calculated and plotted in Fig. 1. Observe from the figure that the deficient-rank probability of G tends to 0 almost exponentially fast.

Moreover, in Table I for some particular values of k, we give the values of $\Pr(\mathrm{rank}(G) = k)$. To make comparison, for each value of k, the minimum field size q as a power of 2 is given such that the probability of M to be full rank is at least $\Pr(\mathrm{rank}(G) = k)$.

TABLE I
THE FULL-RANK PROBABILITY OF THE RANDOM $k \times k$ MONOSHIFT MATRIX G, AND THE CORRESPONDING MINIMUM FIELD SIZE q SO THAT THE $k \times k$ MATRIX OVER THE FIELD ACHIEVES AT LEAST THE SAME FULL-RANK PROBABILITY

k	4	8	16	32
$\Pr(\mathrm{rank}(G) = k)$	0.8556	0.9771	0.9998	$> 1 - 10^{-8}$
minimum field size q	2^3	2^6	2^{13}	2^{28}

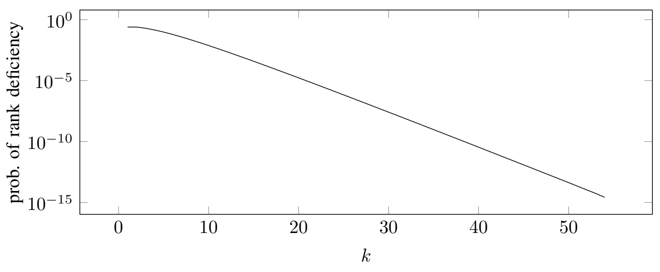

Fig. 1: Probability that a $k \times k$ random monoshift matrix is not full rank.

IV. Applications of Monoshift Systems in Coding

A. Random Storage Codes

Consider a system of k data nodes and n storage nodes, where $k \le n$. Each data node generates a data sequence of L bits. The data sequences can be encoded and stored at the n storage nodes. It is required that all the data sequences can be decoded by retrieving the data stored at any k storage nodes. There is not a centralized encoder for this system. This problem has been studied in [14], where a sparse random linear coding scheme using finite fields is proposed.

We here propose a scheme using monoshift systems. Denote by polynomials x_i of degree $L-1$, $i = 1,\ldots,k$ the k data sequences, and by polynomials y_i, $i = 1,\ldots,n$ the n coded sequences each stored at one storage node. The coded sequences are formed by $\mathbf{y} = G\mathbf{x}$, where G is an $n \times k$ monoshift matrix with each entry uniformly at random chosen from $\{0, 1, z, z+1\}$. Due to the shift operation, the storage cost at each storage node is $L+1$ bits.

For decoding, the coded sequences stored at any k storage nodes are retrieved, and the data sequences can be decoded if the corresponding monoshift system is uniquely solvable. When $k = 16$ for example, the probability that the monoshift system has a unique solution is 99.98% (see Table I).

B. Batched Network Codes

We introduce another application of monoshift systems in batched network coding (see e.g., [17]–[19]), which is a class of efficient random linear network coding schemes with low computation costs and coefficient vector overheads. Most sophisticated designs of batched network coding schemes are based on finite field operations [15], [20]. Here we discuss a variation of BATS codes [15] using monoshift operations.

Suppose a sequence of K input polynomials of the same degree are to be transmitted from a source node to a destination node through a network using a monoshift BATS code, which has an outer code and an inner code. The source node uses the outer code to generate an unlimited number of batches $\mathbf{b}_1, \mathbf{b}_2, \ldots$ generated using monoshift matrices. In particular,

$$\mathbf{b}_i = G_i \mathbf{x}_i$$

where G_i, called *generator matrix* is an $M \times d_i$ uniformly random monoshift matrix, and $\mathbf{x}_i$ is a vector of d_i polynomials uniformly chosen from the K inputs polynomials. The number M is predetermined and referred to as the *batch size* of the outer code. The degrees d_i is obtained by sampling a *degree distribution* $\Psi = (\Psi_1, \ldots, \Psi_K)$ so that d_i takes value d with probability Ψ_d.

The batches are then transmitted through the network where *binary* random linear network coding is allowed among polynomials belonging to the same batch. At the destination node, the received polynomials of the ith batch is in the form

$$\mathbf{y}_i = H_i G_i \mathbf{x}_i \tag{11}$$

where H_i, called *transfer matrix*, is an M-column binary random matrix whose number of rows corresponds to the number of polynomials received for the ith batch. The recoding scheme within batches is known as the *inner code*.

The monoshift outer code and the binary inner code provides an efficient combination for implementing BATS codes. This combination enables the use of a relatively large batch size, e.g., 64 and 128. Binary recoding has the advantages of lower recoding computation costs and shorter coefficient vector overheads. It has been shown that using binary recoding with batch size 64, a better expected rank can be achieved compared with recoding over $GF(2^8)$ with batch size 32 [21]. Due to the computation cost, it is less efficient to use batch sizes 64 or higher for large finite field operations.

Last, we remark that based on the probability characterized in Theorem 3, the existing analysis of the asymptotic and finite-length performances of BATS codes [21] can be extended to monoshift BATS codes as well.

V. Concluding Remarks

A monoshift system demonstrates certain unique properties among various algebraic linear systems. Based on the efficient solving algorithm and the rank distribution studied in this paper, monoshift systems are ready to be applied in various coding problems, especially decentralized scenarios where random coding is preferred.

References

[1] I. S. Reed and G. Solomon, "Polynomial codes over certain finite fields," *Journal of the Society for Industrial and Applied Mathematics*, vol. 8, pp. 300–304, Jun. 1960.

[2] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*. North-Holland publishing, 1978.

[3] O. Pretzel, *Error-correcting Codes and Finite Fields*, ser. Oxford applied mathematics and computing science series. Clarendon Press, 1996.

[4] R. Koetter and M. Médard, "An algebraic approach to network coding," *IEEE/ACM transactions on networking*, vol. 11, no. 5, pp. 782–795, 2003.

[5] T. Ho, M. Medard, R. Koetter, D. R, Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to multicast," *IEEE Trans. Inf. Theory*, vol. 52, no. 10, pp. 4413–4430, Oct. 2006.

[6] J. Bloemer, M. Kalfane, M. Karpinski, R. Karp, M. Luby, and D. Zuckerman, "An XOR-based erasure-resilient coding scheme," ICSI, Tech. Rep. TR-95-048, 1995.

[7] J. S. Plank and L. Xu, "Optimizing Cauchy reed-solomon codes for fault-tolerant network storage applications," in *Fifth IEEE International Symposium on Network Computing and Applications (NCA'06)*. IEEE, 2006, pp. 173–180.

[8] C. W. Sung and X. Gong, "A ZigZag-decodable code with the MDS property for distributed storage systems," in *IEEE Int. Symp. Inf. Theory*, Jul. 2013, pp. 341–345.

[9] X. Gong and C. W. Sung, "Zigzag decodable codes: Linear-time erasure codes with applications to data storage," *J. Comput. Syst. Sci.*, vol. 89, pp. 190–208, 2017.

[10] H. Hou, K. W. Shum, M. Chen, and H. Li, "BASIC codes: Low-complexity regenerating codes for distributed storage systems," *IEEE Trans. Information Theory*, vol. 62, no. 6, pp. 3053–3069, 2016.

[11] X. Fu, S. Yang, and Z. Xiao, "Decoding and repair schemes for shift-XOR regenerating codes," *IEEE Transactions on Information Theory*, vol. 66, no. 12, pp. 7371–7386, 2020.

[12] H. Tang, Q. T. Sun, Z. Li, X. Yang, and K. Long, "Circular-shift linear network coding," *IEEE Trans. Inf. Theory*, vol. 65, no. 1, pp. 65–80, 2019.

[13] R. Kannan, "Solving systems of linear equations over polynomials," *Theoretical Computer Science*, vol. 39, pp. 69–88, 1985.

[14] A. G. Dimakis, V. Prabhakaran, and K. Ramchandran, "Decentralized erasure codes for distributed networked storage," *IEEE Transactions on Information Theory*, vol. 52, no. 6, pp. 2809–2816, 2006.

[15] S. Yang and R. W. Yeung, "Batched sparse codes," *IEEE Trans. Inf. Theory*, vol. 60, no. 9, pp. 5322–5346, Sep. 2014.

[16] Y. Zhang, X. Fu, X. Wu, S. Yang, and K. W. Shum, "Solving monoshift systems and applications in random coding," full version. [Online]. Available: https://shhyang.github.io/research/monoshift-full.pdf

[17] P. A. Chou, Y. Wu, and K. Jain, "Practical network coding," in *Proc. Allerton Conf. Comm., Control, and Computing*, Oct. 2003.

[18] D. Silva, W. Zeng, and F. R. Kschischang, "Sparse network coding with overlapping classes," in *Proc. NetCod '09*, 2009, pp. 74–79.

[19] A. Heidarzadeh and A. H. Banihashemi, "Overlapped chunked network coding," in *Proc. ITW '10*, 2010, pp. 1–5.

[20] B. Tang and S. Yang, "An LDPC approach for chunked network codes," *IEEE/ACM Transactions on Networking*, no. 1, pp. 605–617, Feb. 2018.

[21] S. Yang and R. W. Yeung, *BATS Codes: Theory and Practice*, ser. Synthesis Lectures on Communication Networks. Morgan & Claypool Publishers, 2017, vol. 26.