This article has been accepted for publication in IEEE Transactions on Information Theory. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TIT.2024.3370410

Computing Efficiently a Parity-Check Matrix for $\mathbb{Z}_{p^s}$-Additive Codes

Cristina Fernández-Córdoba, Adrián Torres, Carlos Vela, and Mercè Villanueva

***Abstract*—The $\mathbb{Z}_{p^s}$-additive codes of length n are subgroups of $\mathbb{Z}_{p^s}^n$, with p prime and $s \geq 1$. They can be seen as a generalization of linear codes over $\mathbb{Z}_2$, $\mathbb{Z}_4$, or more general over $\mathbb{Z}_{2^s}$. In this paper, we show two methods for computing a parity-check matrix of a $\mathbb{Z}_{p^s}$-additive code from a generator matrix of the code in standard form. We also compare the performance of our results implemented in Magma with the current available function in Magma for linear codes over finite rings in general. Complementing this comparison, we also show a time complexity analysis of the algorithms. The rings $\mathbb{Z}_{p^s}$ belong to a more general class of rings: finite chain rings. Along the paper, we observe that the same results can be applied to any linear code over a finite commutative chain ring.**

***Index Terms*—Additive code, chain ring, parity-check matrix, performance, time complexity.**

I. Introduction

Let R be a finite commutative ring with identity. The ring R is called a chain ring if all its left (right) ideals form a unique chain under inclusion. It is well known that if R is a finite chain ring, then R is a principal ideal ring and it has a unique maximal ideal $\langle\gamma\rangle$. Its chain of ideals is

$$\langle 0\rangle = \langle\gamma^s\rangle \subsetneq \langle\gamma^{s-1}\rangle \subsetneq \ldots \subsetneq \langle\gamma\rangle \subsetneq R.$$

The integer s is called the nilpotency index of $\langle\gamma\rangle$.

Let $\mathbb{Z}_{p^s}$ be the ring of integers modulo p^s with p prime and $s \geq 1$. The set of n-tuples over $\mathbb{Z}_{p^s}$ is denoted by $\mathbb{Z}_{p^s}^n$. In this paper, the elements of $\mathbb{Z}_{p^s}^n$ will also be called vectors. The order of a vector u over $\mathbb{Z}_{p^s}$, denoted by $o(u)$, is the smallest positive integer m such that $mu = \mathbf{0}$. Note that $\mathbb{Z}_{p^s}$ is a chain ring with the unique maximal ideal $\langle p\rangle$ and nilpotency index s.

A code over $\mathbb{Z}_p$ of length n is a nonempty subset of $\mathbb{Z}_p^n$, and it is linear if it is a subspace of $\mathbb{Z}_p^n$. Similarly, a nonempty subset of $\mathbb{Z}_{p^s}^n$ is a $\mathbb{Z}_{p^s}$-additive if it is a subgroup of $\mathbb{Z}_{p^s}^n$. Note that, when $p = 2$ and $s = 1$, a $\mathbb{Z}_{p^s}$-additive code is a binary linear code and, when $p = 2$ and $s = 2$, it is a quaternary linear code or a linear code over $\mathbb{Z}_4$. $\mathbb{Z}_{p^s}$-additive codes were first defined as p-adic codes in [8]. Let $\mathcal{C}$ be a $\mathbb{Z}_{p^s}$-additive code of length n. Since $\mathcal{C}$ is a subgroup of $\mathbb{Z}_{p^s}^n$, it is isomorphic to an abelian group $\mathbb{Z}_{p^s}^{t_1} \times \mathbb{Z}_{p^{s-1}}^{t_2} \times \cdots \times \mathbb{Z}_p^{t_s}$, and we say that $\mathcal{C}$ is of type $(p^s)^{t_1}(p^{s-1})^{t_2}\ldots p^{t_s}$ or briefly $(n; t_1, \ldots, t_s)$. It is clear that a $\mathbb{Z}_{p^s}$-additive code of type $(n; t_1, \ldots, t_s)$ has $p^{st_1+(s-1)t_2+\cdots+t_s}$ codewords. These codes have been considered and studied for example in [6], [7], [9], [14], [16], [18], [19], [21].

In general, $\mathbb{Z}_{p^s}$-additive codes are not free as submodules of $\mathbb{Z}_{p^s}^n$ (they are free only when $t_2 = \cdots = t_s = 0$), which means that they usually do not have a basis, that is, a generating set consisting of linearly independent vectors. However, there exists a set of codewords $S = \{\mathbf{u}_i^{(j)} \mid 1 \leq j \leq s, 1 \leq i \leq t_j\} \subseteq \mathcal{C}$, where $o(\mathbf{u}_i^{(j)}) = p^{s-j+1}$ for all i and j, satisfying that any codeword in $\mathcal{C}$ can be expressed uniquely in the form

$$\sum_{j=1}^{s}\sum_{i=1}^{t_j} \lambda_i^{(j)}\mathbf{u}_i^{(j)}, \tag{1}$$

for $\lambda_i^{(j)} \in \mathbb{Z}_{p^{s-j+1}}$. The matrix whose rows are the codewords in S is a generator matrix of $\mathcal{C}$ having a minimum number of rows, that is, $t_1 + \cdots + t_s$ rows.

The inner product of $\mathbf{u} = (u_1, \ldots, u_n)$ and $\mathbf{v} = (v_1, \ldots, v_n)$ in $\mathbb{Z}_{p^s}^n$ is defined as $\mathbf{u} \cdot \mathbf{v} = \sum_{i=1}^n u_i v_i \in \mathbb{Z}_{p^s}$. Then, if $\mathcal{C}$ is a $\mathbb{Z}_{p^s}$-additive code of length n, its dual code is

$$\mathcal{C}^\perp = \{\mathbf{v} \in \mathbb{Z}_{p^s}^n \mid \mathbf{u} \cdot \mathbf{v} = 0 \text{ for all } \mathbf{u} \in \mathcal{C}\}.$$

In [8], it is proved that if $\mathcal{C}$ is a $\mathbb{Z}_{p^s}$-additive code of type $(n; t_1, \ldots, t_s)$, then $\mathcal{C}^\perp$ is a $\mathbb{Z}_{p^s}$-additive code of type $(n; n-t, t_s, t_{s-1}, \ldots, t_2)$, where $t = \sum_{i=1}^s t_i$.

Let $\mathcal{C}$ be a $\mathbb{Z}_{p^s}$-additive code with generator matrix G. A matrix H is a parity-check matrix of $\mathcal{C}$ if it is a generator matrix of its dual code $\mathcal{C}^\perp$. In this sense, the code $\mathcal{C}$ can be generated from H by computing all the orthogonal vectors to it, that is,

$$\mathcal{C} = \{\mathbf{v} \in \mathbb{Z}_{p^s}^n \mid H\mathbf{v}^T = \mathbf{0}\}.$$

A parity-check matrix H holds that $GH^T = (\mathbf{0})$, which is a crucial property that plays the main role in syndrome decoding. It can be used to correct errors but also to correct erasures, since it provides a linear system of equations that can be solved to recover the sent information.

Two codes C_1 and C_2 over $\mathbb{Z}_p$ of length n are said to be monomially equivalent (or just equivalent) provided there is a monomial matrix M such that $C_2 = \{\mathbf{c}M \mid \mathbf{c} \in C_1\}$. Recall that a monomial matrix is a square matrix with exactly one nonzero entry in each row and column. They are said to

This work has been partially supported by the Spanish MINECO under grants PID2019-104664GB-I00, PID2022-137924NB-I00, and RED2022-134306-T (AEI / 10.13039/501100011033), by the Catalan AGAUR under grant 2021 SGR 00643, and by the Portuguese Foundation for Science and Technology (FCT-Fundação para a Ciência e a Tecnologia), through CIDMA - Center for Research and Development in Mathematics and Applications, within project UIDB/04106/2020.

Carlos Vela is with the Department of Mathematics, University of Aveiro, 3810-197 Aveiro, Portugal (e-mail: carlos.velacabello@unisg.ch)

Cristina Fernández-Córdoba, Adrián Torres and Mercè Villanueva are with the Department of Information and Communications Engineering, Universitat Autònoma de Barcelona, 08193 Cerdanyola del Vallès, Spain (e-mails: cristina.fernandez@uab.cat, adrian.torres@uab.cat, merce.villanueva@uab.cat)

The material in this paper was submitted in part at the 2024 IEEE International Symposium on Information Theory in Athens, Greece, 2024 [12].

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in IEEE Transactions on Information Theory. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TIT.2024.3370410

be permutation equivalent if there is a permutation matrix P such that $C_2 = \{\mathbf{c}P \mid \mathbf{c} \in C_1\}$. Recall that a permutation matrix is a square matrix with exactly one 1 in each row and column and 0s elsewhere. Let $\mathcal{S}_n$ be the symmetric group of permutations on the set $\{1, \ldots, n\}$. A permutation matrix represents a permutation of coordinates, so we can also say that they are permutation equivalent if there is a permutation of coordinates $\pi \in \mathcal{S}_n$ such that $C_2 = \{\pi(\mathbf{c}) \mid \mathbf{c} \in C_1\}$. Similarly, two $\mathbb{Z}_{p^s}$-additive codes, $\mathcal{C}_1$ and $\mathcal{C}_2$, of length n are said to be permutation equivalent if they differ only by a permutation of coordinates, that is, if there is a permutation of coordinates $\pi \in \mathcal{S}_n$ such that $\mathcal{C}_2 = \{\pi(\mathbf{c}) \mid \mathbf{c} \in \mathcal{C}_1\}$.

Let Id_k be the identity matrix of size $k \times k$. In [15], it is shown that any quaternary linear code code of type $(n; t_1, t_2)$ is permutation equivalent to a quaternary linear code with a generator matrix of the form

$$G = \begin{pmatrix} \mathrm{Id}_{t_1} & R & S \\ \mathbf{0} & 2\mathrm{Id}_{t_2} & 2T \end{pmatrix}, \tag{2}$$

where R and T are matrices over $\mathbb{Z}_4$ with all entries in $\{0, 1\} \subset \mathbb{Z}_4$, and S is a matrix over $\mathbb{Z}_4$. In this case, we say that G is in standard form. In the same paper, it is shown that if the generator matrix G of a quaternary linear code $\mathcal{C}$ is as in (2), then a parity-check matrix of $\mathcal{C}$ can be computed as follows:

$$H = \begin{pmatrix} -(S + RT)^T & T^T & \mathrm{Id}_{n-t_1-t_2} \\ 2R^T & 2\mathrm{Id}_{t_2} & \mathbf{0} \end{pmatrix}. \tag{3}$$

In [8], a generator matrix in standard form for $\mathbb{Z}_{p^s}$-additive codes is given as a generalization of matrix (2) for quaternary linear codes. Specifically, if $\mathcal{C}$ is a $\mathbb{Z}_{p^s}$-additive code of type $(n; t_1, \ldots, t_s)$, then $\mathcal{C}$ is permutation equivalent to a $\mathbb{Z}_{p^s}$-additive code $\mathcal{C}'$ generated by a generator matrix G in standard form, that is, of the form

$$\begin{pmatrix} \mathrm{Id}_{t_1} & A_{1,2} & A_{1,3} & \cdots & A_{1,s} & A_{1,s+1} \\ \mathbf{0} & p\mathrm{Id}_{t_2} & pA_{2,3} & \cdots & pA_{2,s} & pA_{2,s+1} \\ \mathbf{0} & \mathbf{0} & p^2\mathrm{Id}_{t_3} & \cdots & p^2A_{3,s} & p^2A_{3,s+1} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & p^{s-1}\mathrm{Id}_{t_s} & p^{s-1}A_{s,s+1} \end{pmatrix}, \tag{4}$$

where $A_{i,j}$ are matrices over $\mathbb{Z}_{p^s}$ for all $1 \leq j \leq s$ and $1 \leq i \leq s + 1 - j$. In [8], it is also shown that a parity-check matrix H for $\mathcal{C}'$ is of the form

$$\begin{pmatrix} H_{1,1} & H_{2,1} & H_{3,1} & \cdots & H_{s,1} & \mathrm{Id}_{n-t} \\ pH_{1,2} & pH_{2,2} & pH_{3,2} & \cdots & p\mathrm{Id}_{t_s} & \mathbf{0} \\ \vdots & \vdots & \vdots & \iddots & \mathbf{0} & \mathbf{0} \\ p^{s-2}H_{1,s-1} & p^{s-2}H_{2,s-1} & p^{s-2}\mathrm{Id}_{t_3} & \cdots & \mathbf{0} & \mathbf{0} \\ p^{s-1}H_{1,s} & p^{s-1}\mathrm{Id}_{t_2} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} \end{pmatrix}, \tag{5}$$

where the column blocks have the same size as in (4).

In this paper, we show how to construct a parity-check matrix of a $\mathbb{Z}_{p^s}$-additive code from a generator matrix in standard form, as a generalization of matrix (3) for quaternary linear codes. This paper is organised as follows. In Section II, we describe two methods to obtain a parity-check matrix, one is based on the computation of block-minors of an associated matrix to the generator matrix of the $\mathbb{Z}_{p^s}$-additive code in standard form, and another one is based on computing the block-minors in a recursive way using previous computed matrices. In Section III, we describe algorithms for both methods. We also compare the computational time of an implementation of these algorithms in Magma, for some values of the parameters, together with the implementation given by a function in Magma that works for any linear code over a finite ring. Finally, in Section IV, we give some conclusions and future research on this topic.

II. Computation of a parity-check matrix

In this section, we present two different approaches to construct a parity-check matrix for $\mathbb{Z}_{p^s}$-additive codes from a generator matrix in standard form (see Theorems 1 and 2). First, we present some results on the computation of determinants for square matrices with a certain structure.

Let $A = (a_{r,s})_{1 \leq r,s \leq n}$ be a square $n \times n$ matrix. It is well-known that the determinant of A can be computed as

$$|A| = \sum_{\sigma \in \mathcal{S}_n} \mathrm{sgn}(\sigma) \prod_{h=1}^{n} a_{h,\sigma(h)}, \tag{6}$$

where $\mathrm{sgn}(\sigma)$ is the parity of σ. If A is a square $n \times n$ matrix of the form

$$\begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n-1} & a_{1,n} \\ 1 & a_{2,2} & \cdots & a_{2,n-1} & a_{2,n} \\ 0 & 1 & \cdots & a_{3,n-1} & a_{3,n} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & a_{n,n} \end{pmatrix}, \tag{7}$$

then we can improve the equation given in (6) for the determinant of A computing the additions of the products that do not consider any element under the diagonal with ones. Therefore, we just need to consider permutations $\sigma \in \mathcal{S}_n$ such that $\sigma(h) \geq h - 1$ for all $h \in \{1, \ldots, n\}$. Moreover, we can avoid multiplying by one by considering only the products of elements $a_{i,j}$ with $j \geq i$. In order to write this more formally, we introduce the following definitions.

Definition 1: Let $\hat{\mathcal{S}}_n = \{\sigma \in \mathcal{S}_n \,|\, \sigma(h) \geq h - 1 \text{ for all } h \in \{1, \ldots, n\}\}$, and $J_\sigma = \{h_1, \ldots, h_r\} = \{h \in \{1, \ldots, n\} \mid \sigma(h) \geq h\}$. Note that J_σ is not empty, since $\sigma(1) \geq 1$ for any $\sigma \in \hat{\mathcal{S}}_n$. We consider the elements in J_σ ordered, i.e., $h_1 < h_2 < \ldots < h_r$.

Proposition 1: Let A be a matrix as in (7). Then, the determinant of A is given by

$$|A| = \sum_{\sigma \in \hat{\mathcal{S}}_n} \mathrm{sgn}(\sigma) \prod_{h \in J_\sigma} a_{h,\sigma(h)},$$

where $\hat{\mathcal{S}}_n$ and J_σ are as in Definition 1.

Proof: Straightforward from (6) and Definition 1. ∎

Example 1: Consider the matrix A as in (7) with $n = 3$, that is,

$$A = \begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ 1 & a_{2,2} & a_{2,3} \\ 0 & 1 & a_{3,3} \end{pmatrix}.$$

In this case, we have that $\mathcal{S}_3 = \{Id, (1,2), (1,3), (2,3), (1,2,3), (1,3,2)\}$ and $\hat{\mathcal{S}}_3 = \{Id, (1,2), (2,3), (1,3,2)\}$.

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in IEEE Transactions on Information Theory. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TIT.2024.3370410

Then, $J_{Id} = \{1,2,3\}$, $J_{(1,2)} = \{1,3\}$, $J_{(2,3)} = \{1,2\}$, and $J_{(1,3,2)} = \{1\}$. Therefore,

$$|A| = a_{1,1}a_{2,2}a_{3,3} - a_{1,2}a_{3,3} - a_{1,1}a_{2,3} + a_{1,3}.$$

Now, we give an alternative expression for the computation of the determinant of a matrix A as in (7) by using the minors of the diagonal of A.

Definition 2: Let $A = (a_{r,s})_{1\leq r,s\leq n}$ be a square $n \times n$ matrix. The *i-th minor of the diagonal of A of order j*, denoted by O^i_j, is the determinant of the i-th submatrix of size j in the diagonal of A, that is,

$$O^i_j = |(a_{r,s})_{i\leq r,s\leq i+j-1}|.$$

We consider that $O^i_0 = 1$ for all $i \geq 1$. Note that $O^i_1 = a_{i,i}$ and $O^1_n = |A|$.

Proposition 2: Let A be a matrix as in (7). Then, the determinant of A is given by

$$|A| = O^1_n = \sum_{k=1}^{n}(-1)^{k-1}a_{1,k}O^{k+1}_{n-k},$$

where O^i_j is the i-th minor of the diagonal of A of order j.

Proof: Using the Laplace expansion on the first column, the determinant $|A|$ can be calculated as follows:

$$|A| = a_{1,1}O^2_{n-1} - |A'_{n-1}|,$$

where

$$A'_{n-i} = \begin{pmatrix} a_{1,i+1} & a_{1,i+2} & \cdots & a_{1,n-1} & a_{1,n} \\ 1 & a_{i+2,i+2} & \cdots & a_{i+2,n-1} & a_{i+2,n} \\ 0 & 1 & \cdots & a_{i+3,n-1} & a_{i+3,n} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & a_{n,n} \end{pmatrix}.$$

We can repeat this process with A'_{n-1} obtaining $|A| = a_{1,1}O^2_{n-1} - a_{1,2}O^3_{n-2} + |A'_{n-2}|$, and so on so forth until we have

$$|A| = \sum_{k=1}^{n}(-1)^{k-1}a_{1,k}O^{k+1}_{n-k}.$$

■

Corollary 1: Let A be a matrix as in (7). Then,

$$O^i_j = \sum_{k=i}^{i+j-1}(-1)^{i-k}a_{i,k}O^{k+1}_{i+j-1-k}, \tag{8}$$

where O^i_j is the i-th minor of the diagonal of A of order j, for all $1 \leq i \leq n$ and $i \leq j \leq n$.

From now on, we consider block-matrices, that is, matrices whose entries are submatrices instead of scalars. Indeed, we first define the reduced associated matrix G^{RA} of a generator matrix G in standard form, which is a block-matrix.

Definition 3: Let G be the generator matrix of a $\mathbb{Z}_{p^s}$-additive code $\mathcal{C}$ in standard form, that is, as in (4). The *reduced associated matrix* G^{RA} of G is the matrix

$$G^{RA} = \begin{pmatrix} A_{1,2} & A_{1,3} & \cdots & A_{1,s} & A_{1,s+1} \\ \mathrm{Id}_{t_2} & A_{2,3} & \cdots & A_{2,s} & A_{2,s+1} \\ \mathbf{0} & \mathrm{Id}_{t_3} & \cdots & A_{3,s} & A_{3,s+1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathrm{Id}_{t_s} & A_{s,s+1} \end{pmatrix}. \tag{9}$$

In general, a blockwise determinant of a square block-matrix, computed by performing multiplications and additions of blocks, is not well-defined due to the noncompatibility of the different dimensions of each block and the fact that the product of matrices is noncommutative. However, we propose a notion of determinant, called block-determinant, for any block-matrix of the form as in (9), by defining an analogous expression to the one given in Proposition 1. Since the block-submatrices $(A_{r,s+1})_{i\leq r,s\leq i+j-1}$ are also in the form of (9), we can also provide a notion of block-minors of the block-diagonal of G^{RA}, analogous to the minors O^i_j described in Definition 2. Then, we give an analogue of Proposition 2 to obtain another expression to compute these block-minors.

Definition 4: Let A be a block-matrix as in (9). The i-th *block-minor* of the block-diagonal of A of order j, denoted also by O^i_j, is the block-determinant of the i-th submatrix of size j in the block-diagonal of A, that is,

$$\begin{aligned} O^i_j = |(A_{r,s+1})_{i\leq r,s\leq i+j-1}| = \\ = \begin{vmatrix} A_{i,i+1} & A_{i,i+2} & \cdots & A_{i,i+j-1} & A_{i,i+j} \\ \mathrm{Id}_{t_{i+1}} & A_{i+1,i+2} & \cdots & A_{i+1,i+j-1} & A_{i+1,i+j} \\ \mathbf{0} & \mathrm{Id}_{t_{i+2}} & \cdots & A_{i+2,i+j-1} & A_{i+2,i+j} \\ \vdots & \mathbf{0} & \ddots & \vdots & \vdots \\ \mathbf{0} & \cdots & \mathbf{0} & \mathrm{Id}_{t_{i+j-1}} & A_{i+j-1,i+j} \end{vmatrix} \\ = \sum_{\sigma\in\hat{\mathcal{S}}_j}\mathrm{sgn}(\sigma)\prod_{h\in J_\sigma}A_{i+h-1,i+\sigma(h)}, \end{aligned} \tag{10}$$

where $\hat{\mathcal{S}}_j$ and J_σ are as in Definition 1.

The following results are used to show that the products in (10) are well-defined.

Remark 1: Let $\sigma \in \hat{\mathcal{S}}_j$ and $h \in \{1,\ldots,j\}$. Then,

1) $\sigma(h) = h-1$ if $h \notin J_\sigma$,
2) $\sigma^{-1}(h) \leq h+1$.

Lemma 1: Let $\sigma \in \hat{\mathcal{S}}_j$ and $J_\sigma = \{h_1,\ldots,h_r\}$. Then, for any $k \in \{1,\ldots,r-1\}$, $h_{k+1} = \sigma(h_k)+1$.

Proof: First, we see that $\sigma(h_k)+1 \in J_\sigma$. Assume $\sigma(h_k)+1 \notin J_\sigma$. By Remark 1-1, we have that $\sigma(\sigma(h_k)+1) = (\sigma(h_k)+1)-1 = \sigma(h_k)$. Therefore, since σ is a one-to-one map, $\sigma(h_k)+1 = h_k$, that is, $\sigma(h_k) = h_k-1$ which is not possible because $h_k \in J_\sigma$. Therefore, $\sigma(h_k)+1 \in J_\sigma$, and the result follows if we prove that $\{h_k+1,\ldots,\sigma(h_k)\} \cap J_\sigma = \emptyset$.

Consider $i \in \{1,\ldots,h_k-1\}$. By Remark 1-2, $\sigma^{-1}(i) \leq i+1 \leq h_k$. If $\sigma^{-1}(i) = h_k$, then $\sigma(h_k) = i \leq h_k-1$ which is not possible since $h_k \in J_\sigma$. Therefore, $\sigma^{-1}(i) \leq h_k-1$. Since there are h_k-1 different values of both $\sigma^{-1}(i)$ and $i \in \{1,\ldots,h_k-1\}$, we have that $\sigma^{-1}(h_k) \notin \{1,\ldots,h_k-1\}$. By Remark 1-2, $\sigma^{-1}(h_k) \leq h_k+1$. Thus, there are only two possible values remaining for $\sigma^{-1}(h_k)$, say h_k or h_k+1.

First, consider the case $\sigma(h_k) = h_k$. Since $h_k \in J_\sigma$, we have seen that $\sigma(h_k)+1 \in J_\sigma$ and hence $h_k+1 \in J_\sigma$. Then, since h_k and h_{k+1} are consecutive elements in J_σ, necessarily, $h_{k+1} = h_k+1 = \sigma(h_k)+1$ and the result holds. Finally, consider the case $\sigma(h_k+1) = h_k$. In this case, $h_k+1 \notin J_\sigma$. If $h_k+1 = \sigma(h_k)$, then clearly $\{h_k+1,\ldots,\sigma(h_k)\} \cap J_\sigma = \emptyset$ and we are done. If $h_k+1 \neq \sigma(h_k)$, by the same argument as before, $\sigma^{-1}(h_k+1) \notin \{1,\ldots,h_k-1\}$. By Remark 1-2, $\sigma^{-1}(h_k+1) \leq h_k+2$. Moreover, we have that $\sigma^{-1}(h_k+1)$

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in IEEE Transactions on Information Theory. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TIT.2024.3370410

cannot be h_k+1 or h_k, so $\sigma^{-1}(h_k+1) = h_k+2$. This implies that $h_k + 2 \notin J_\sigma$. This argument can be applied recursively, obtaining $\{h_k + 1, \ldots, \sigma(h_k)\} \cap J_\sigma = \emptyset$. ∎

Using Lemma 1, we see that all products in (10) are of the form

$$A_{i+h_k-1,i+\sigma(h_k)}A_{i+\sigma(h_k),i+\sigma(h_{k+1})},$$

for any $k \in \{1, \ldots, r-1\}$. Thus, O^i_j is a well-defined matrix of size $t_i \times z$, where z is the amount of columns that the matrices $A_{*,i+j}$ have, in this case t_{i+j}. We consider that $O^i_0 = Id$ for all $i \geq 1$. Note that $O^i_1 = A_{i,i+1}$.

Example 2: Let G^{RA} be a block-matrix as in (9). By Definition 1, we have $\hat{\mathcal{S}}_2 = \{Id, (1,2)\}$, $J_{Id} = \{1,2\}$ and $J_{(1,2)} = \{1\}$. Then, the block-minor O^{s-1}_2 can be computed as follows:

$$\begin{aligned} O^{s-1}_2 =& \begin{vmatrix} A_{s-1,s} & A_{s-1,s+1} \\ \mathrm{Id}_{t_s} & A_{s,s+1} \end{vmatrix} \\ =& A_{s-1,s}A_{s,s+1} - A_{s-1,s+1}. \end{aligned}$$

Clearly, $A_{s-1,s}A_{s,s+1}$ is a $t_{s-1} \times (n-t)$ matrix, where n is the total amount of columns of G and $t = \sum_{i=0}^{s} t_i$.

Similarly, taking into account the set $\hat{\mathcal{S}}_3$ and the corresponding sets of indices J_σ for $\sigma \in \hat{\mathcal{S}}_3$, given in Example 1, the block-minor O^{s-2}_3 can be computed as follows:

$$\begin{aligned} O^{s-2}_3 =& \begin{vmatrix} A_{s-2,s-1} & A_{s-2,s} & A_{s-2,s+1} \\ \mathrm{Id}_{t_{s-1}} & A_{s-1,s} & A_{s-1,s+1} \\ \mathbf{0} & \mathrm{Id}_{t_s} & A_{s,s+1} \end{vmatrix} \\ =& A_{s-2,s-1}A_{s-1,s}A_{s,s+1} - A_{s-2,s}A_{s,s+1} \\ & - A_{s-2,s-1}A_{s-1,s+1} + A_{s-2,s+1}. \end{aligned}$$

In order to compute the block-minor O^{s-3}_4, we consider the set of permutations $\hat{\mathcal{S}}_4 = \{Id, (3,4), (2,3), (2,4,3), (1,2), (1,2)(3,4), (1,3,2), (1,4,3,2)\}$. The corresponding sets of indices given in Definition 1 are: $J_{Id} = \{1,2,3,4\}$, $J_{(3,4)} = \{1,2,3\}$, $J_{(2,3)} = \{1,2,4\}$, $J_{(2,4,3)} = \{1,2\}$, $J_{(1,2)} = \{1,3,4\}$, $J_{(1,2)(3,4)} = \{1,3\}$, $J_{(1,3,2)} = \{1,4\}$, and $J_{(1,4,3,2)} = \{1\}$. Then the block-minor O^{s-3}_4 is

$$\begin{aligned} O^{s-3}_4 =& \begin{vmatrix} A_{s-3,s-2} & A_{s-3,s-1} & A_{s-3,s} & A_{s-3,s+1} \\ \mathrm{Id}_{t_{s-2}} & A_{s-2,s-1} & A_{s-2,s} & A_{s-2,s+1} \\ \mathbf{0} & \mathrm{Id}_{t_{s-1}} & A_{s-1,s} & A_{s-1,s+1} \\ \mathbf{0} & \mathbf{0} & \mathrm{Id}_{t_s} & A_{s,s+1} \end{vmatrix} \\ =& A_{s-3,s-2}A_{s-2,s-1}A_{s-1,s}A_{s,s+1} \\ & - A_{s-3,s-2}A_{s-2,s-1}A_{s-1,s+1} \\ & - A_{s-3,s-2}A_{s-2,s}A_{s,s+1} + A_{s-3,s-2}A_{s-2,s+1} \\ & - A_{s-3,s-1}A_{s-1,s}A_{s,s+1} + A_{s-3,s-1}A_{s-1,s+1} \\ & + A_{s-3,s}A_{s,s+1} - A_{s-3,s+1}. \end{aligned}$$

Proposition 3: Let A be a block-matrix as in (9). Then, the block-determinant of A is given by

$$|A| = O^1_s = \sum_{k=1}^{s} (-1)^{k-1} A_{1,k+1} O^{k+1}_{s-k},$$

where O^i_j is the i-th block-minor of the block-diagonal of A of order j.

Proof: It is easy to prove this statement by following an analogous argument to that used in the proof of Proposition 2. ∎

Corollary 2: Let A be a block-matrix as in (9). Then,

$$O^i_j = \sum_{k=i}^{i+j-1} (-1)^{i-k} A_{i,k+1} O^{k+1}_{i+j-1-k}, \tag{11}$$

where O^i_j is the i-th block-minor of the block-diagonal of A of order j, for all $1 \leq i \leq s$ and $i \leq j \leq s$.

Now, we give the result that allows us to compute a parity-check matrix using the block-minors of the reduced associated matrix G^{RA} of G.

Theorem 1: Let $\mathcal{C}$ be a $\mathbb{Z}_{p^s}$-additive code of type $(n; t_1, \ldots, t_s)$ with a generator matrix G as in (4), and G^{RA} its reduced generator matrix. Then, the transpose of a parity-check matrix H of $\mathcal{C}$ is as follows:

$$\begin{pmatrix} H_{1,1} & pH_{1,2} & \cdots & p^{s-3}H_{1,s-2} & p^{s-2}H_{1,s-1} & p^{s-1}H_{1,s} \\ H_{2,1} & pH_{2,2} & \cdots & p^{s-3}H_{2,s-2} & p^{s-2}H_{2,s-1} & p^{s-1}\mathrm{Id}_{t_2} \\ H_{3,1} & pH_{3,2} & \cdots & p^{s-3}H_{3,s-2} & p^{s-2}\mathrm{Id}_{t_3} & \mathbf{0} \\ H_{4,1} & pH_{4,2} & \cdots & p^{s-3}\mathrm{Id}_{t_4} & \mathbf{0} & \mathbf{0} \\ \vdots & \vdots & \iddots & \vdots & \vdots & \vdots \\ H_{s,1} & p\mathrm{Id}_{t_s} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathrm{Id}_{n-t} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{pmatrix}, \tag{12}$$

where $t = \sum_{i=1}^{s} t_i$,

$$H_{i,j} = (-1)^{s+2-i-j} O^i_{s+2-i-j}, \tag{13}$$

for all $1 \leq j \leq s$ and $1 \leq i \leq s+1-j$, and O^i_k is the i-th block-minor of the block-diagonal of G^{RA} of order k.

Proof: Let $\mathcal{C}'$ be the $\mathbb{Z}_{p^s}$-additive code generated by matrix H given in (12). First, we prove that $GH^T = (\mathbf{0})$ and hence $\mathcal{C}' \subseteq \mathcal{C}^\perp$. Denote G_s and H_s the matrices G and H, respectively, corresponding to the value s. We prove that $G_s H_s^T = (\mathbf{0})$ by induction on $s \geq 2$. For $s = 2$, we have

$$\begin{aligned} &H_{2,1} = -A_{2,3}, \\ &H_{1,1} = -A_{1,3} - A_{1,2}H_{2,1} = -A_{1,3} + A_{1,2}A_{2,3}, \\ &H_{1,2} = -A_{1,2}. \end{aligned}$$

Clearly,

$$\begin{aligned} G_2 H_2^T = &\begin{pmatrix} \mathrm{Id}_{t_1} & A_{1,2} & A_{1,3} \\ \mathbf{0} & p\mathrm{Id}_{t_2} & pA_{2,3} \end{pmatrix} \cdot \\ &\begin{pmatrix} (A_{1,2}A_{2,3} - A_{1,3}) & -pA_{1,2} \\ -A_{2,3} & p\mathrm{Id}_{t_2} \\ \mathrm{Id}_{n-t_1-t_2} & \mathbf{0} \end{pmatrix} = (\mathbf{0}). \end{aligned}$$

By induction hypothesis, we assume that $G_k H_k^T = (\mathbf{0})$ for every integer $k \leq s-1$. Let us decompose the matrices G_s and H_s in terms of G_{s-1} and H_{s-1}:

$$G_s = \begin{pmatrix} & & & A_{1,s+1} \\ & G_{s-1} & & pA_{2,s+1} \\ & & & p^2 A_{3,s+1} \\ & & & \vdots \\ \mathbf{0} & \cdots & \mathbf{0} & p^{s-1}\mathrm{Id}_{t_s} \quad p^{s-1}A_{s,s+1} \end{pmatrix},$$

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in IEEE Transactions on Information Theory. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TIT.2024.3370410

$$H_s^T = \begin{pmatrix} H_{1,1} & & \\ H_{2,1} & & \\ H_{3,1} & & pH_{s-1}^T \\ H_{4,1} & & \\ \vdots & & \\ H_{s,1} & & \\ \mathrm{Id}_{n-t} & \mathbf{0} \quad \cdots \quad \mathbf{0} & \end{pmatrix}.$$

We can separate the product $G_s H_s^T$ in two parts. On the one hand, we have that

$$G_s \begin{pmatrix} pH_{s-1}^T \\ \mathbf{0} \end{pmatrix} = \begin{pmatrix} pG_{s-1}H_{s-1}^T \\ p^{s-1}\mathrm{Id}_{t_s} \cdot p\mathrm{Id}_{t_s} \end{pmatrix} = (\mathbf{0})$$

by the induction hypothesis and the structure of H_{s-1}^T. On the other hand, we have to prove that

$$G_s \begin{pmatrix} H_{1,1} \\ H_{2,1} \\ H_{3,1} \\ H_{4,1} \\ \vdots \\ H_{s,1} \\ \mathrm{Id}_{n-t} \end{pmatrix} = G_s H' = (\mathbf{0}),$$

where H' is the matrix having the first $n-t$ columns of H_s^T. Note that the i-th block-row of G_s is of the form

$$(G_s)_i = \begin{pmatrix} \mathbf{0} \cdots \mathbf{0} & p^{i-1}\mathrm{Id}_{t_i} & p^{i-1}A_{i,i+1} & \cdots & p^{i-1}A_{i,s+1} \end{pmatrix}$$

for all $1 \leq i \leq s$. Then,

$$\begin{aligned}
&(G_s)_i H' = \\
&= p^{i-1}\mathrm{Id}_{t_i} H_{i,1} + \sum_{j=1}^{s-i} p^{i-1} A_{i,i+j} H_{i+j,1} + p^{i-1} A_{i,s+1} \\
&= p^{i-1}(-1)^{s+1-i} O_{s+1-i}^{i} \\
&\quad + p^{i-1} \sum_{j=1}^{s-i} A_{i,i+j}(-1)^{s+1-i-j} O_{s+1-i-j}^{i+j} + p^{i-1} A_{i,s+1} \\
&= p^{i-1}\left[(-1)^{s+1-i} O_{s+1-i}^{i} + \sum_{j=1}^{s+1-i} A_{i,i+j}(-1)^{s+1-i-j} O_{s+1-i-j}^{i+j}\right] \\
&= p^{i-1}(-1)^{s+1-i}\left[O_{s+1-i}^{i} - \sum_{j=1}^{s+1-i} A_{i,i+j}(-1)^{j-1} O_{s+1-i-j}^{i+j}\right],
\end{aligned}$$

where the second equality is given by (13) and the third one by considering that, by definition, $O_0^{s+1} = Id$. Now, using Corollary 2, it is easy to see that

$$O_{s+1-i}^{i} = \sum_{j=1}^{s+1-i} A_{i,i+j}(-1)^{j-1} O_{s+1-i-j}^{i+j}.$$

Thus $(G_s)_i H' = (\mathbf{0})$ for all $1 \leq i \leq s$ and we conclude that $G_s H_s^T = (\mathbf{0})$.

We have proven that $\mathcal{C}' \subseteq \mathcal{C}^\perp$. For the other inclusion, we consider an arbitrary codeword $\mathbf{c} = (c_1, c_2, \ldots, c_n) \in \mathcal{C}^\perp$ and we have to see that it also belongs to $\mathcal{C}'$. The following argument was adapted from [22, Proposition 1.2]. The first $n-t$ rows of H have the identity matrix in the last $n-t$ columns. Therefore, we can add a linear combination of these $n-t$ first rows to $\mathbf{c}$ to obtain a new codeword of $\mathcal{C}^\perp$ of the form

$$\mathbf{c}^{(1)} = (c_1, c_2, \ldots, c_t, 0, \ldots, 0).$$

Since $\mathbf{c}^{(1)} \in \mathcal{C}^\perp$ is orthogonal to the last t_s rows of G_s, we obtain

$$\begin{aligned}
\mathbf{c}^{(1)}((G_s)_s)^T &= \mathbf{c}^{(1)}(\mathbf{0} \cdots \mathbf{0} \quad p^{s-1}\mathrm{Id}_{t_s} \quad p^{s-1}A_{s,s+1})^T \\
&= p^{s-1}(c_{t_1+\cdots+t_{s-1}+1}, \ldots, c_t) = \mathbf{0}.
\end{aligned}$$

This means that the components $c_{t_1+\cdots+t_{s-1}+1}, \ldots, c_t$ are all multiples of p. The next t_s rows of H, which are

$$\begin{pmatrix} pH_{1,2} & pH_{2,2} & \ldots & pH_{s-1,2} & p\mathrm{Id}_{t_s} & \mathbf{0} \end{pmatrix}, \qquad (14)$$

have zero entries in the last $n-t$ columns and $p\mathrm{Id}_{t_s}$ in the previous t_s columns. Therefore, we can add a linear combination of the rows in (14) to obtain a new codeword of $\mathcal{C}^\perp$ of the form

$$\mathbf{c}^{(2)} = (c_1, c_2, \ldots, c_{t_1+\cdots+t_{s-1}}, 0, \ldots, 0).$$

Since $\mathbf{c}^{(2)} \in \mathcal{C}^\perp$ is orthogonal to the second to last t_{s-1} rows of G, we obtain

$$\mathbf{c}^{(2)}((G_s)_{s-1})^T = p^{s-2}(c_{t_1+\cdots+t_{s-2}+1}, \ldots, c_{t_1+\cdots+t_{s-1}}) = \mathbf{0}.$$

This means that the components $c_{t_1+\cdots+t_{s-2}+1}, \ldots, c_{t_1+\cdots+t_{s-1}}$ are multiples of p^2. The same argument can be applied iteratively until we obtain the codeword of $\mathcal{C}^\perp$,

$$\mathbf{c}^{(s)} = (c_1, \ldots, c_{t_1}, 0, \ldots, 0).$$

Since $\mathbf{c}^{(s)} \in \mathcal{C}^\perp$ is orthogonal to the first t_1 rows of G, we obtain

$$\mathbf{c}^{(s)}((G_s)_1)^T = (c_1, \ldots, c_{t_1}) = \mathbf{0}.$$

Hence $\mathbf{c}^{(s)} = \mathbf{0}$ and $\mathbf{c} \in \mathcal{C}'$, since it can be obtained by a linear combination of the rows of H. Consequently, H is a generator and a parity-check matrix for $\mathcal{C}^\perp$ and $\mathcal{C}$, respectively. ■

Example 3: Let $p = 2$ and $s = 2$. Let G be the generator matrix in standard form of a $\mathbb{Z}_4$-additive code $\mathcal{C}$ of type $(n; t_1, t_2)$ and G^{RA} its reduced associated matrix:

$$G = \begin{pmatrix} \mathrm{Id}_{t_1} & A_{1,2} & A_{1,3} \\ \mathbf{0} & 2\mathrm{Id}_{t_2} & 2A_{2,3} \end{pmatrix}, \; G^{RA} = \begin{pmatrix} A_{1,2} & A_{1,3} \\ \mathrm{Id}_{t_2} & A_{2,3} \end{pmatrix}.$$

Then, by Theorem 1, the transpose of a generator matrix H of $\mathcal{C}^\perp$ can be constructed as follows:

$$H^T = \begin{pmatrix} A_{1,2}A_{2,3} - A_{1,3} & -2A_{1,2} \\ -A_{2,3} & 2\mathrm{Id}_{t_2} \\ \mathrm{Id}_{n-t_1-t_2} & \mathbf{0} \end{pmatrix},$$

since $H_{1,1} = O_2^1 = A_{1,2}A_{2,3} - A_{1,3}$, $H_{2,1} = -O_1^2 = -A_{2,3}$, and $H_{1,2} = -O_1^1 = -A_{1,2}$. Note that this parity-check matrix H of $\mathcal{C}$ generates the same code as the matrix (3). Indeed, both matrices are equal if we consider $-A_{2,3}$ instead of $A_{2,3}$. Note that, in both cases, G generates the same code $\mathcal{C}$.

Example 4: Let $p = 2$ and $s = 3$. Let G be the generator matrix in standard form of a $\mathbb{Z}_8$-additive code $\mathcal{C}$ of type $(n; t_1, t_2, t_3)$ and G^{RA} its reduced associated matrix:

$$G = \begin{pmatrix} \mathrm{Id}_{t_1} & A_{1,2} & A_{1,3} & A_{1,4} \\ \mathbf{0} & 2\mathrm{Id}_{t_2} & 2A_{2,3} & 2A_{2,4} \\ \mathbf{0} & \mathbf{0} & 4\mathrm{Id}_{t_3} & 4A_{3,4} \end{pmatrix},$$

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in IEEE Transactions on Information Theory. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TIT.2024.3370410

$$G^{RA} = \begin{pmatrix} A_{1,2} & A_{1,3} & A_{1,4} \\ \mathrm{Id}_{t_2} & A_{2,3} & A_{2,4} \\ \mathbf{0} & \mathrm{Id}_{t_3} & A_{3,4} \end{pmatrix}.$$

Then, the transpose of a generator matrix H of $\mathcal{C}^{\perp}$ can be constructed as follows:

$$H^T = \begin{pmatrix} -(A_{1,2}A_{2,3}A_{3,4} + A_{1,4} - A_{1,2}A_{2,4} - A_{1,3}A_{3,4}) & 2(A_{1,2}A_{2,3} - A_{1,3}) & -4A_{1,2} \\ A_{2,3}A_{3,4} - A_{2,4} & -2A_{2,3} & 4\mathrm{Id}_{t_2} \\ -A_{3,4} & 2\mathrm{Id}_{t_3} & \mathbf{0} \\ \mathrm{Id}_{n-t_1-t_2-t_3} & \mathbf{0} & \mathbf{0} \end{pmatrix}$$

by Theorem 1 and Corollary 2. For example, we have that $H_{1,1} = -O_3^1 = A_{1,2}O_2^2 - A_{1,3}O_1^3 + A_{1,4}O_0^4 = A_{1,2}(A_{2,3}A_{3,4} - A_{2,4}) - A_{1,3}A_{3,4} + A_{1,4}$.

The computation of a parity-check matrix by using Theorem 1 requires the reckoning of many minors. The computation of these minors, $O^i_{s+2-i-j}$, is carried out using Corollary 2, so it requires the computation of j different minors of lower order. In this case, we assume that we compute each one of the blocks of the parity-check matrix $H_{i,j}$ independently. However, now, we show that following an appropriate order in the computation of the different matrices $H_{i,j}$, we are able to obtain an expression to compute $O_{s+2-i-j}$, where all the minors of lower order in (11) have already been computed in a previous step. In fact, we can obtain a similar expression, which directly relates $H_{i,j}$ with all others $H_{i,k}$ such that $k \geq j$. This is shown in Theorem 2.

Theorem 2: Let $\mathcal{C}$ be a $\mathbb{Z}_{p^s}$-additive code of type $(n; t_1, \ldots, t_s)$ with a generator matrix G as in (4). Then, the blocks of the matrix (12) given in Theorem 1, which is the transpose of a parity-check matrix H of $\mathcal{C}$, can be calculated as follows:

$$H_{i,j} = -\left(A_{i,s-j+2} + \sum_{k=i+1}^{s-j+1} A_{i,k}H_{k,j}\right) \tag{15}$$

for all $1 \leq j \leq s$ and $1 \leq i < s-j+1$. Note that $H_{s-j+1,j} = -A_{s-j+1,s-j+2}$.

Proof: We prove this statement by seeing that the matrix computed by using (15) is the same as the one in (12). To achieve that, we show that $H_{i,j} = \hat{H}_{i,j}$ for all $1 \leq j \leq s$ and $1 \leq i \leq s-j+1$, where $\hat{H}_{i,j}$ is as in (13), that is,

$$\begin{aligned} H_{i,j} &= -\left(A_{i,s-j+2} + \sum_{k=i+1}^{s-j+1} A_{i,k}H_{k,j}\right) \\ &= (-1)^{s+2-i-j}O^i_{s+2-i-j} = \hat{H}_{i,j}. \end{aligned} \tag{16}$$

We prove this by induction on i for any $j \in \{1, \ldots, s\}$. For the case $i = s+1-j$, we have that

$$H_{s+1-j,j} = -\left(A_{s+1-j,s-j+2}\right) = (-1)O_1^{s+1-j} = \hat{H}_{s+1-j,j}.$$

By induction hypothesis, we assume that (16) is true for $i \leq s-j+1$ and we want to see that it is true for $i-1$, i.e., $H_{i-1,j} = \hat{H}_{i-1,j}$. We have that

$$\begin{aligned} \hat{H}_{i-1,j} &= (-1)^{s+2-j-(i-1)}O^{i-1}_{s+2-j-(i-1)} \\ &= (-1)^{s+2-j-(i-1)} \sum_{k=i-1}^{s+1-j} A_{i-1,k+1}(-1)^{i-1-k}O^{k+1}_{s+1-j-k} \\ &= (-1)^{s+2-j-(i-1)} \sum_{k=i}^{s+2-j} A_{i-1,k}(-1)^{i-k}O^{k}_{s+2-j-k} \\ &= -\sum_{k=i}^{s+2-j} A_{i-1,k}(-1)^{s+2-j-k}O^{k}_{s+2-j-k} \\ &= -\left(A_{i-1,s+2-j} + \sum_{k=i}^{s+1-j} A_{i-1,k}(-1)^{s+2-j-k}O^{k}_{s+2-j-k}\right) \\ &= -\left(A_{i-1,s+2-j} + \sum_{k=i}^{s+1-j} A_{i-1,k}H_{k,j}\right) \\ &= H_{i-1,j}. \end{aligned}$$

The first equality is by definition, the second is by Corollary 2, the third is a rearrangement of the indices, the sixth is by the induction hypothesis, and the last one is by definition. ■

Recall that $\mathbb{Z}_{p^s}$ is a chain ring, so $\mathbb{Z}_{p^s}$-additive codes are included in the family of linear codes over chain rings. Let $\mathcal{C}$ be a linear code over a finite commutative chain ring R with maximal ideal $\langle\gamma\rangle$ and nilpotency index s. It is well-known that $\mathcal{C}$ is permutation equivalent to a code generated by a matrix in standard form as in (4) [17], just by replacing p by γ. Therefore, all the results given in this paper can be applied, exactly in the same way, to linear codes over chain rings since we only use the general properties of rings and the form of the generator matrix in standard form.

III. Performance comparison

In this section, we describe two algorithms that implement the computation of a parity-check matrix for $\mathbb{Z}_{p^s}$-additive codes (or, more generally, linear codes over a chain ring), from a generator matrix in standard form. They are based on Theorems 1 and Theorem 2, respectively. First, we show a naive implementation, which is based on computing each submatrix $H_{i,j}$ in (12) by using the expression given in (13) and Corollary 2. Afterwards, we present an iterative construction that reduces the calculations considerably by using the expression given in (15). Then, the performance of these algorithms implemented in Magma is compared with the performance if we use the current available function in Magma for codes over finite rings in general. A time computation and time complexity analysis are also given.

A. Algorithms description

The first procedure corresponds to the one presented in Theorem 1, considering that each one of the blocks $H_{i,j}$ in (12) is computed independently, by using the expression given in (13), that is, $H_{i,j} = (-1)^{s+2-i-j}O^i_{s+2-i-j}$, and Corollary 2 to compute each one of the block-minors $O^i_{s+2-i-j}$ from the computation of different minors of lower order. This implementation is shown in Algorithm 1.

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in IEEE Transactions on Information Theory. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TIT.2024.3370410

Algorithm 1 Parity-check matrix in standard form. Minors construction.

Require: A $\mathbb{Z}_{p^s}$-additive code $\mathcal{C}$ of type $(n; t_1, \ldots, t_s)$.

Compute a generator matrix G in standard form of $\mathcal{C}$.
Define $t := t_1 + \cdots + t_s$.
Define a zero matrix H^T with n rows and $n-t$ columns.
Define $numCol := 1$.
for $j := 1, \ldots, s$ **do**
  Define $numRow := 1$.
  for $i := 1, \ldots, s-j+1$ **do**
    Compute $O^i_{s+2-i-j}$ using the block-matrix G and Corollary 2.
    Define $H_{i,j} := (-1)^{s+2-i-j} O^i_{s+2-i-j}$.
    Insert $p^{j-1}H_{i,j}$ at position $(numRow, numCol)$ in H^T.
    $numRow := numRow + t_i$.
  end for
  if $j = 1$ **then**
    Insert Id_{n-t} at position $(numRow, numCol)$ in H^T.
    $numCol := numCol + n - t$.
  else
    Insert $p^{j-1}\mathrm{Id}_{t_{s-j+2}}$ at position $(numRow, numCol)$ in H^T.
    $numCol := numCol + t_{s-j+2}$.
  end if
end for
return The parity-check matrix H.

With the result given by Theorem 2, we can easily define a new implementation, which reduces the number of operations compared to Algorithm 1. In particular, for each block-column j, we can compute each $H_{i,j}$ starting from $H_{s-j+1,j} = -A_{s-j+1,s-j+2}$ and using (15) to obtain $H_{i,j}$ for $1 \leq i < s-j+1$, in decreasing order. Since all $H_{k,j}$, for $k \geq i$, have been already determined when $H_{i,j}$ is computed, no additional operations are performed apart from the sums and products of matrices represented in (15). This new implementation is shown in Algorithm 2.

Algorithm 2 Parity-check matrix in standard form. Iterative construction.

Require: A $\mathbb{Z}_{p^s}$-additive code $\mathcal{C}$ of type $(n; t_1, \ldots, t_s)$.

Compute a generator matrix G in standard form of $\mathcal{C}$.
Define $t := t_1 + \cdots + t_s$.
Define a zero matrix H^T with n rows and $n-t$ columns.
Define $numCol := 1$
for $j := 1, \ldots, s$ **do**
  Define $numRow := t_1 + \cdots + t_{s-j+1} + 1$.
  if $j = 1$ **then**
    Insert Id_{n-t} at position $(numRow, numCol)$ in H^T.
  else
    Insert $p^{j-1}\mathrm{Id}_{t_{s-j+2}}$ at position $(numRow, numCol)$ in H^T.
  end if
  $numRow := numRow - t_{s-j+1}$
  Define $H_{s-j+1,j} := -A_{s-j+1,s-j+2}$.
  Insert $p^{j-1}H_{s-j+1,j}$ at position $(numRow, numCol)$ in H^T.
  for $i := s-j, \ldots, 1$ by -1 **do**
    $numRow := numRow - t_i$.
    Compute $H_{i,j}$ using (15) and previously computed matrices $H_{k,j}$, for $k := i+1, \ldots, s-j+1$.
    Insert $p^{j-1}H_{i,j}$ at position $(numRow, numCol)$ in H^T.
  end for
  if $j = 1$ **then**
    $numCol := numCol + n - t$.
  else
    $numCol := numCol + t_{s-j+2}$.
  end if
end for
return The parity-check matrix H.

B. Performance comparison

In this subsection, we compare three different methods for computing the parity-check matrix of a $\mathbb{Z}_{p^s}$-additive code. Two of them are the different versions of the method that can be obtained from Theorem 1 and Theorem 2, which are described in Algorithm 1 and Algorithm 2, respectively. The third method consists on using the Magma function `ParityCheckMatrix()`, included in the current official distribution [3], which computes a parity-check matrix for a linear code defined over any finite ring. First, we make an experimental comparison by using Magma and present the results through some graphs, thereafter we calculate the complexity of the methods introduced in this paper.

1) Computation time analysis: In order to compare the performance of Algorithms 1 and 2 and the Magma function, we consider a random $\mathbb{Z}_{p^s}$-additive code $\mathcal{C}$ of type $(n; \ell, \ldots, \ell)$, that is, with $t_i = \ell$ for any $1 \leq i \leq s$. We study the effect of changing the parameters n, s, and ℓ on the three different methods. The first method (Algorithm 1), which computes each minor independently, is labeled as *Minors*. The second method (Algorithm 2), which computes the minors iteratively, is labeled as *Iterative*. Finally, the third method, which uses the Magma function `ParityCheckMatrix()`, is labeled as *Generic*.

Figure 1 shows the computation times of all methods for random $\mathbb{Z}_{3^s}$-additive codes of type $(1000; 2, \ldots, 2)$, where s takes values between 2 and 16. Similarly, Figure 2 shows the computation times of all methods for random $\mathbb{Z}_{3^4}$-additive codes of type $(1000; \ell, \ldots, \ell)$, where ℓ takes values between 2 and 20. Finally, Figures 3 and 4 show the computation times of all methods for random $\mathbb{Z}_{3^{10}}$-additive codes of type $(n; 2, \ldots, 2)$, where $n \in \{2^i \cdot 100 \mid 0 \leq i \leq 8\}$.

2) Time complexity analysis: Let us consider Algorithms 1 and 2, which are based on the following expressions, respectively:

$$\hat{H}_{i,j} = (-1)^{s+2-i-j} O^i_{s+2-i-j} \quad \text{and} \tag{17}$$

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in IEEE Transactions on Information Theory. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TIT.2024.3370410

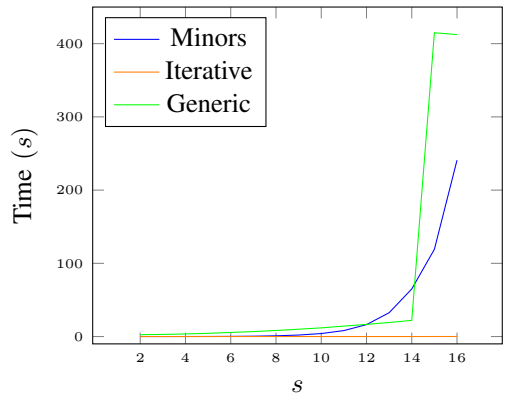


Figure 1. Codes of type $(1000; 2, \ldots, 2)$, with $p = 3$ and $2 \leq s \leq 16$.

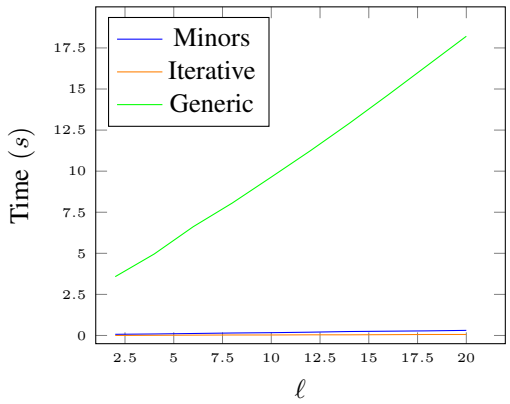


Figure 2. Codes of type $(1000; \ell, \ldots, \ell)$, with $p = 3$, $s = 4$ and $2 \leq \ell \leq 20$.

$$H_{i,j} = -\left(A_{i,s-j+2} + \sum_{k=i+1}^{s-j+1} A_{i,k} H_{k,j} \right), \qquad (18)$$

for $1 \leq j \leq s$ and $1 \leq i \leq s - j + 1$. Note that (17) and (18) coincide with the equations given in Theorem 1 and Theorem 2, respectively. In the first case, we denote the submatrices as $\hat{H}_{i,j}$ instead of $H_{i,j}$ in order to distinguish between both methods.

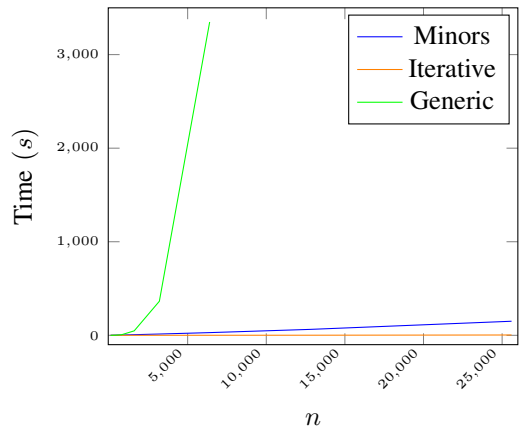


Figure 3. Codes of type $(n; 2, \ldots, 2)$, with $p = 3$, $s = 10$ and $100 \leq n \leq 25600$.

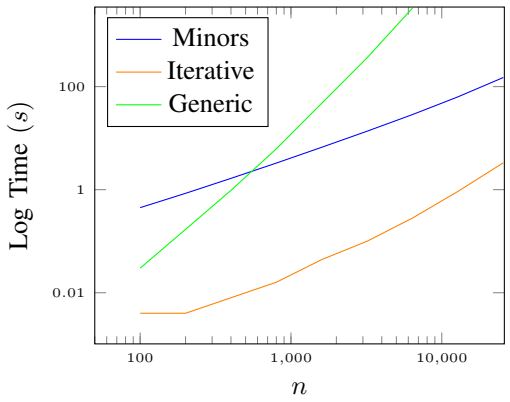


Figure 4. Codes of type $(n; 2, \ldots, 2)$, with $p = 3$, $s = 10$ and $100 \leq n \leq 25600$ (logarithmic scale).

For simplicity, let us assume that $\mathcal{C}$ is a $\mathbb{Z}_{p^s}$-additive code of type $(n; \ell, \ldots, \ell)$. Then, $t = s\ell$, $H_{i,1}$ is a $\ell \times (n - t)$ matrix for any $1 \leq i \leq s$, and $H_{i,j}$ is a $\ell \times \ell$ matrix for any $2 \leq j \leq s$ and $1 \leq i \leq s - j + 1$. We denote by $\hat{T}_{i,j}(s, n, \ell)$ and $T_{i,j}(s, n, \ell)$ the runtime needed to compute $\hat{H}_{i,j}$ and $H_{i,j}$, respectively. We also denote by $S(a, b)$ the computation time of the addition of two $a \times b$ matrices over $\mathbb{Z}_{p^s}$ and $P(a, b, c)$ the computation time of the product of an $a \times b$ matrix by a $b \times c$ matrix.

With the aim of computing $\hat{H}_{i,j}$, we first estimate the complexity of determining any block-minor O^i_j. Due to the structure of G^{RA}, by Corollary 2, we can compute O^i_j by calculating block-determinants of one dimension less and the same structure. Then, by induction, it easy to show that the runtime needed to compute O^i_j is $(2^{s-i} - 1)\left(P(\ell, \ell, n - t) + S(\ell, n - t)\right)$ for $j = s + 1 - i$ and $(2^{j-1} - 1)\left(P(\ell, \ell, \ell) + S(\ell, \ell)\right)$ for $j < s + 1 - i$. Thus, by using (17), we have that

$$\begin{aligned} \hat{T}_{i,1}(s, n, \ell) &= (2^{s-i} - 1)\left(P(\ell, \ell, n - t) + S(\ell, n - t)\right), \\ \hat{T}_{i,j}(s, n, \ell) &= (2^{s+1-i-j} - 1)\left(P(\ell, \ell, \ell) + S(\ell, \ell)\right) \text{ for } j > 1. \end{aligned} \qquad (19)$$

In order to obtain $H_{i,j}$, we need to compute $H_{i',j}$ for all $i \leq i' \leq s - j + 1$. Thus, for each $1 \leq j \leq s$, we start with $H_{s-j+1,j} = -A_{s-j+1,s-j+2}$ and then compute the sequence of matrices $H_{s-j,j}, H_{s-j-1,j}, \ldots, H_{1,j}$ by using (18). In this case, we have that

$$\begin{aligned} T_{i,1}(s, n, \ell) &= (s - i)P(\ell, \ell, n - t) + (S(\ell, n - t)), \\ T_{i,j}(s, n, \ell) &= (s - j - i + 1)(P(\ell, \ell, \ell) + S(\ell, \ell)) \text{ for } j > 1. \end{aligned} \qquad (20)$$

Therefore, the total runtime of computing the parity-check matrix of $\mathcal{C}$ by using Algorithms 1 and 2 is given by the following results:

Proposition 4: Let $\mathcal{C}$ be a $\mathbb{Z}_{p^s}$-additive code of type $(n; \ell, \ldots, \ell)$, and $t = s\ell$. The total runtime of computing the parity-check matrix of $\mathcal{C}$ by using Algorithm 1 is

$$\begin{aligned} \hat{T}(s, n, \ell) = (2^s - 1 - s)\left(P(\ell, \ell, n - t) + S(\ell, n - t)\right) + \\ + \left(2^s - 1 - \frac{s^2}{2} - \frac{s}{2}\right)\left(P(\ell, \ell, \ell) + S(\ell, \ell)\right). \end{aligned}$$

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in IEEE Transactions on Information Theory. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TIT.2024.3370410

Proof: We have $\hat{T}(s,n,\ell) = \sum_{i=1}^{s} \hat{T}_{i,1}(s,n,\ell) + \sum_{j=2}^{s}\sum_{i=1}^{s-j+1} \hat{T}_{i,j}(s,n,\ell)$. Recall that $\sum_{k=0}^{n} 2^k = 2^{n+1}-1$. By using (19), since $\sum_{i=1}^{s}(2^{s-i}-1) = 2^s - 1 - s$ and $\sum_{j=2}^{s}\sum_{i=1}^{s-j+1}(2^{s+1-i-j}-1) = \sum_{j=2}^{s}(2^{s+1-j}-s+j-2) = 2^s - 1 - \frac{s^2}{2} - \frac{s}{2}$, the result follows. ∎

Proposition 5: Let $\mathcal{C}$ be a $\mathbb{Z}_{p^s}$-additive code of type $(n;\ell,\ldots,\ell)$, and $t = s\ell$. The total runime of computing the parity-check matrix of $\mathcal{C}$ by using Algorithm 2 is

$$T(s,n,\ell) = \frac{s(s-1)}{2}(P(\ell,\ell,n-t) + S(\ell,n-t)) + \frac{1}{6}(s^3 - 3s^2 + 2s)(P(\ell,\ell,\ell) + S(\ell,\ell)).$$

Proof: We have $T(s,n,\ell) = \sum_{i=1}^{s} T_{i,1}(s,n,\ell) + \sum_{j=2}^{s}\sum_{i=1}^{s-j+1} T_{i,j}(s,n,\ell)$. We also have that $\sum_{i=1}^{s}(s-i) = \sum_{j=0}^{s-1} j = s(s-1)/2$ and

$$\begin{aligned}
&\sum_{j=2}^{s}\sum_{i=1}^{s-j+1}(s-j-i+1) = \sum_{j=2}^{s}(s-j)(s-j+1)/2\\
&= \frac{1}{2}\Big(\sum_{j=2}^{s} s^2 + \sum_{j=2}^{s} j^2 - \sum_{j=2}^{s} j(2s+1)\Big)\\
&= \frac{1}{2}(s^3 - s + (2s^3+3s^2+s-6)/6 - (2s^3+3s^2-3s-2)/2)\\
&= \frac{1}{6}(s^3 - 3s^2 + 2s).
\end{aligned}$$

Finally, by (20), the result follows. ∎

Regarding the asymptotic complexity of the algorithms, since $S(a,b)$ is $O(ab)$ and $P(a,b,c)$ is $O(abc)$, we obtain $S(\ell,n-t) + P(\ell,\ell,n-t) = O((n-t)\ell^2)$ and $S(\ell,\ell) + P(\ell,\ell,\ell) = O(\ell^3)$. Therefore,

$$\hat{T}(s,n,\ell) = O(2^s\ell^2\,(n+s\ell))$$
$$T(s,n,\ell) = O(s^2\ell^2 n).$$

If we only consider the variable n, we obtain that the algorithms are $O(n)$. Otherwise, if we take into account the variable s, we can see that while the first algorithm is exponential, the second proposal has square polynomial complexity, which adjust with the data shown in Figures 1, 2, 3, and 4.

IV. Conclusion

Two different methods to compute a parity-check matrix for $\mathbb{Z}_{p^s}$-additive codes have been introduced. Even though they are very similar methods, and their performance are comparable under some conditions, we have showed that they perform very different when the parameters of the code changes. We have also established experimentally that both are better than the current algorithm included in Magma for any linear code over a finite ring. These methods may also be used to compute a parity-check matrix for codes over chain rings in general.

A Magma function to compute the dual of $\mathbb{Z}_{p^s}$-linear codes has been included in a new Magma package to deal with linear codes over $\mathbb{Z}_{p^s}$ [13]. This function is based on the construction of a parity check matrix using Algorithm 2, and it is more efficient than the current available function in Magma for codes over finite rings in general. This new package also allows the construction of some families of $\mathbb{Z}_{p^s}$-linear codes, and includes functions related to generalized Gray maps, information sets, the process of encoding and decoding using permutation decoding, among others. Indeed, this package generalizes some of the functions for codes over $\mathbb{Z}_4$, which are already included in the standard Magma distribution [3]. It has been developed mainly by the authors of this paper and the collaboration of some undergraduate students. The first version of this new package and a manual describing all functions is available in a GitHub repository (https://github.com/merce-github/ZpAdditiveCodes) and in the CCSG web site (https://ccsg.uab.cat).

Further research is possible in different directions. A natural generalisation would be to adapt these algorithms to compute a parity-check matrix for codes over mixed alphabets like $\mathbb{Z}_p\mathbb{Z}_{p^s}$-additive codes or even the more generic $\mathbb{Z}_p\mathbb{Z}_{p^2}\ldots\mathbb{Z}_{p^s}$-additive codes.

The generator matrix in standard form as in (4) has a very similar structure to the partial generator matrix of convolutional codes (also called expanded partial generator matrix). Due to this similarity, the methods presented in this paper are adaptable to compute a parity-check matrix for these codes as long as it exists.

Acknowledgment

We would like to thank the referees for carefully reading our manuscript and pointing out that exactly the same results can be applied to any linear code over a finite commutative chain ring.

References

[1] I. Aydogdu and I. Siap, "The structure of $\mathbb{Z}_2\mathbb{Z}_{2^s}$-additive codes: bounds on the minimum distance," Appl. Math. Inf. Sci. vol. 7(6), pp. 2271-2278, 2013.

[2] M. Bilal, J. Borges, S. T. Dougherty, and C. Fernández-Córdoba, "Maximum distance separable codes over $\mathbb{Z}_4$ and $\mathbb{Z}_2 \times \mathbb{Z}_4$," Designs, Codes and Cryptography, vol. 61, pp. 31-40, 2011.

[3] W. Bosma, J. J. Cannon, C. Fieker, and A. Steel, Handbook of Magma functions, Edition 2.25, 2020. https://magma.maths.usyd.edu.au/magma/

[4] J. Borges, C. Fernández-Córdoba, J. Pujol, J. Rifà, and M. Villanueva, "$\mathbb{Z}_2\mathbb{Z}_4$-linear codes: generator matrices and duality," Designs, Codes and Cryptography, vol. 54(2), pp. 167-179, 2010.

[5] J. Borges, C. Fernández, and J. Rifà, "Propelinear structure of $\mathbb{Z}_{2k}$-linear codes," preprint, arXiv 0907.5287

[6] D. K. Bhunia, C. Fernández-Córdoba, M. Villanueva, "On the linearity and classification of $\mathbb{Z}_{p^s}$-linear generalized Hadamard codes," Designs, Codes and Cryptography, vol. 90, pp. 1037-1058, 2022.

[7] D. K. Bhunia, C. Fernández-Córdoba, C. Vela, M. Villanueva, "On the equivalence of $\mathbb{Z}_{p^s}$-linear generalized Hadamard codes," Designs, Codes and Cryptography, 2023. DOI:10.1007/s10623-023-01325-2

[8] A. R. Calderbank and J. A. Sloane, "Modular and p-adic cyclic codes," Designs, Codes and Cryptography, vol. 6, pp. 21-35, 1995.

[9] C. Carlet, "$\mathbb{Z}_{2^k}$-linear codes," IEEE Trans. Inform. Theory, vol. 44, pp. 1543-1547, 1998.

[10] S. T. Dougherty and C. Fernández-Córdoba, "Codes over $\mathbb{Z}_{2^k}$, Gray map and self-dual codes," Advances in Mathematics of Communications, vol. 5(4), pp. 571-588, 2011.

[11] C. Fernández, J. Rifà, and J. Borges, "Every $\mathbb{Z}_{2k}$-code is a binary propelinear code," Electronic Notes in Discrete Mathematics, vol. 10, pp. 100-102, 2001.

[12] C. Fernández-Córdoba, A. Torres-Martín, C. Vela, and M. Villanueva, "Parity-check matrix for $\mathbb{Z}_{p^s}$-additive codes: efficient computation," submitted to 2024 IEEE International Symposium on Information Theory (ISIT), 7-12 July, 2024.

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

This article has been accepted for publication in IEEE Transactions on Information Theory. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TIT.2024.3370410

[13] C. Fernández-Córdoba, A. Torres-Martín, and M. Villanueva, "Linear codes over the integer residue ring $\mathbb{Z}_{p^s}$. A MAGMA package", version 1.0, Universitat Autònoma de Barcelona, 2024. https://ccsg.uab.cat
[14] M. K. Gupta, M. C. Bhandari, and A. K. Lal. "On some linear codes over $\mathbb{Z}_{2^s}$". Designs, Codes and Cryptography, vol. 36(3), pp. 227– 244, 2005.
[15] A. R. Hammons, P. V. Kumar, A. R. Calderbank, N. J. A. Sloane and P. Solé, "The $\mathbb{Z}_4$-linearity of Kerdock, Preparata, Goethals and related codes," IEEE Trans. Inform. Theory, vol. 40, pp. 301-319, 1994.
[16] D. S. Krotov, "On $\mathbb{Z}_{2^k}$-dual binary codes," IEEE Transactions on Information Theory, vol. 53(4), pp. 1532–1537, 2007.
[17] G. H. Norton and A. Sălăgean, "On the structure of linear and cyclic codes over a finite chain ring," Applicable Algebra in Engineering, Communications and Computing, vol. 10, pp. 489–506, 2000.
[18] M. Shi, Z. Sepasdar, A. Alahmadi, and P. Solé, "On two-weight $\mathbb{Z}_{2^k}$ - codes", Designs, Codes and Cryptography, vol. 86(6), pp. 1201– 1209, 2018.
[19] M. Shi, T. Honold, P. Solé, Y. Qiu, R. Wu, and Z. Sepasdar, "The geometry of two-weight codes over $\mathbb{Z}_{p^m}$," IEEE Trans. Inform. Theory, vol. 67(12), pp. 7769-7781, 2021.
[20] H. Tapia-Recillas and G. Vega, "On $\mathbb{Z}_{2^k}$-linear and quaternary codes," SIAM Journal on Discrete Mathematics, vol. 17(1), pp. 103–113, 2003.
[21] A. Torres-Martín and M. Villanueva, "Systematic encoding and permutation decoding for $\mathbb{Z}_{p^s}$-linear codes," IEEE Trans. Inform. Theory, vol. 68(7), pp. 4435–4443, 2022.
[22] Z.-X. Wan, Quaternary Codes, Series on Applied Mathematics, vol. 8, World Scientific Publishing Co. Pte. Ltd., 1997.

Cristina Fernández-Córdoba was born in Sabadell, Catalonia (Spain) in 1977. She received the B.Sc. degree in Mathematics in 2000 from the Universitat Autònoma de Barcelona and the Ph.D. degree in Science (Computer Science Section) in 2005 from the same university. In 2000 she joined the Department of Computer Science at the Universitat Autònoma de Barcelona, and in 2005 the Department of Information and Communications Engineering at the same university. In 2008, she joined the Fundación Española para la Ciencia y la Tecnología and she did a one year research stay at Auburn University under a Fulbright grant. From 2009, she is within the Department of Information and Communications Engineering at the Universitat Autònoma de Barcelona where currently is an Associate Professor. Her research interests include subjects related to combinatorics, coding theory, and graph theory.

Adrián Torres-Martín was born in Sabadell, Catalonia, in December 1996. He received the B.Sc. degree in Mathematics and the B.Sc. degree in Physics in 2019 from the Universitat Autònoma de Barcelona, and the M.Sc. degree in Fundamental Principles of Data Science in 2021 from the Universitat de Barcelona. In 2021, he joined the Department of Information and Communications Engineering, at the Universitat Autònoma de Barcelona, as Research Support Technician and is currently a Ph.D. student in the Computer Science programme. His research interests include subjects related to algebra, coding theory, machine learning, and artificial intelligence.

Carlos Vela was born in Bilbao, Spain, in November 1992. He received the B.Sc. degree in Mathematics and the M.Sc. degree in Computer Science and AI from the University of Sevilla, Spain, in 2014 and 2015, respectively, and the Ph.D. degree in Computer Science from the Universitat Autònoma de Barcelona, Spain, in 2018. He worked at the Department of Information and Communications Engineering, from the Universitat Autònoma de Barcelona while pursuing his Ph.D. degree. In September 2021, he joined the Department of Mathematics, at the University of Aveiro, Portugal, as a postdoctoral researcher. Since February 2024, he is a researcher at the School of Computer Science from the University of St. Gallen, Switzerland. His research interests include subjects related to coding theory and cryptography.

Mercè Villanueva was born in Roses, Catalonia, in January 1972. She received the B.Sc. degree in Mathematics in 1994 from the Universitat Autònoma de Barcelona, the M.Sc. degree in Computer Science in 1996, and the Ph.D. degree in Science (Computer Science Section) in 2001 from the same university. In 1994 she joined the Department of Information and Communications Engineering, at the Universitat Autònoma de Barcelona, as an Assistant Professor. She was promoted to Associate Professor in 2002 and became Full Professor in 2023. Her research interests include subjects related to combinatorics, algebra, coding theory, and graph theory.

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/