

Article

Optimal Construction for Decoding 2D Convolutional Codes over an Erasure Channel

Raquel Pinto ^{1,†}, Marcos Spreafico ^{2,†} and Carlos Vela ^{1,*,†}

¹ CIDMA—Center for Research and Development in Mathematics and Applications, Department of Mathematics, University of Aveiro, 3810-193 Aveiro, Portugal; raquel@ua.pt

² INMA—Institute of Mathematics, Federal University of Mato Grosso do Sul, Campo Grande 79070-900, Brazil; marcos.spreafico@ufms.br

* Correspondence: carlos.vela@ua.pt

† These authors contributed equally to this work.

Abstract: In general, the problem of building optimal convolutional codes under a certain criteria is hard, especially when size field restrictions are applied. In this paper, we confront the challenge of constructing an optimal 2D convolutional code when communicating over an erasure channel. We propose a general construction method for these codes. Specifically, we provide an optimal construction where the decoding method presented in the bibliography is considered.

Keywords: decoding; 2D convolutional codes; complexity; linear decoding; optimal construction

MSC: 94B10; 94B35

1. Introduction

Two-dimensional (2D) convolutional codes are suited for applications where data are organized in a two-dimensional grid, like images. The theory of 2D convolutional codes is a generalization of the theory of one-dimensional (1D) convolutional codes but much more involved. These codes were introduced in [1] and in [2] by Fornasini and Valcher, respectively. In [1], the authors considered 2D convolutional codes that were constituted of sequences indexed on \mathbb{Z}^2 with values in \mathbb{F}^n , where \mathbb{F} was a finite field and established the algebraic properties of these codes. In [2], the authors studied 2D convolutional codes constituted of compact support sequences on \mathbb{Z}^2 . An important property of a code is its distance since that measures its capacity of error correction. In [3], the authors defined the free distance of a 2D convolutional code, established an upper bound for this distance, and then presented some constructions of the 2D convolutional codes with an optimal free distance. A generalization of these codes, called nD convolutional codes, were first introduced in [4,5] and then further developed in [6–8]. However, decoding these kind of codes is a barely explored topic, which we will address in this paper. We will consider transmission over an erasure channel. Unlike the symmetric channel, where errors might occur randomly, over these channels the receiver knows in the transmission which symbols are erased and that the received symbols are correct. These channels are particularly suitable in communication systems like real-time streaming applications or situations where certain packets are more crucial than others. Convolutional codes are particularly convenient for erasure channels due to their ability to consider certain parts of the received data, thereby adapting the correction process to the erasure's location.

There exist only two decoding algorithms for 2D convolutional codes over the erasure channel [9,10]. In [9], the authors made use of the parity-check matrices of the code, while the authors of [10] employed the encoders of the code.

The decoding algorithm presented in [9] considers the specific neighborhoods around erasures that would allow one to decode these erasures using 1D convolutional codes. These



Citation: Pinto, R.; Spreafico, M.; Vela, C. Optimal Construction for Decoding 2D Convolutional Codes over an Erasure Channel. *Axioms* **2024**, *13*, 197. <https://doi.org/10.3390/axioms13030197>

Academic Editors: Jorge Delgado Gracia and Cristina Flaut

Received: 23 December 2023

Revised: 25 February 2024

Accepted: 11 March 2024

Published: 15 March 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1D codes are projections of the 2D convolutional code, and they are obtained by considering 1D parity-check matrices and by taking into account only some of the coefficients of a parity-check matrix that is of 2D code. However, the authors did not provide any such construction in which the corresponding 1D projections produced optimal efficiency for this decoding algorithm. In this paper, we discuss this problem and present several constructions of 2D convolutional codes whose corresponding associated 1D convolutional codes are optimal or quasi-optimal.

2. Preliminaries

In this section, we detail the main backgrounds of 1D and 2D convolutional codes that are required to contextualize the constructions introduced in this paper. We will also recall the decoding algorithm over an erasure channel that was presented in [9].

We denote, by $\mathbb{F}[z]$, the ring of polynomials in the indeterminate z and with coefficients in a finite field \mathbb{F} . By $\mathbb{F}[z_1, z_2]$, we denote the ring of polynomials in the two indeterminates, z_1 and z_2 , with coefficients in \mathbb{F} .

2.1. 1D Convolutional Codes

Definition 1. An (n, k) convolutional code \mathcal{C} is an $\mathbb{F}[z]$ -submodule of $\mathbb{F}[z]^n$ of rank k . A matrix $G(z) \in \mathbb{F}[z]^{n \times k}$ whose columns constitute a basis of \mathcal{C} is called a generator matrix of \mathcal{C} , i.e.,

$$\mathcal{C} = \text{Im}_{\mathbb{F}[z]} G(z) = \{v(z) \in \mathbb{F}[z]^n : v(z) = G(z)u(z) \text{ for some } u(z) \in \mathbb{F}[z]^k\}.$$

The vector $v(z) = G(z)u(z)$ is the codeword corresponding to the information sequence $u(z)$.

Given a convolutional code \mathcal{C} , two generator matrices of this code, $G(z)$ and $\bar{G}(z)$, are said to be equivalent. In addition, they differ by right multiplication with a unimodular matrix $U(z)$ (a $k \times k$ invertible polynomial matrix with a polynomial inverse or, equivalently, a $k \times k$ polynomial matrix with determinant in $\mathbb{F} \setminus \{0\}$), i.e.,

$$G(z) = \bar{G}(z)U(z).$$

A convolutional code is non-catastrophic when it admits a right prime generator matrix $G(z)$, i.e., if $G(z) = \bar{G}(z)X(z)$ with $\bar{G}(z) \in \mathbb{F}_q[z]^{n \times k}$ and $X(z) \in \mathbb{F}_q[z]^{k \times k}$, then $X(z)$ must be a unimodular matrix. A polynomial matrix is said to be left prime if its transpose is a right prime. Note that, since two generator matrices of a convolutional code differ by a right multiplication through a unimodular matrix, if a code admits a right prime generator matrix, then all its generator matrices are right prime and it will be called **non-catastrophic** code.

The **degree** δ of an (n, k) convolutional code \mathcal{C} is the maximum degree of the full-size minors of any generator matrix. Note that the degree can be computed with any generator matrix since two generator matrices of a code differ by the right multiplication of an unimodular matrix, as was previously said. A matrix $G(z) \in \mathbb{F}[z]^{n \times k}$ is said to be **column-reduced** if the sum of its column degrees is equal to the maximum degree of its full-size minors. A polynomial matrix is said to be **row-reduced** if its transpose is column-reduced. An (n, k) convolutional code with degree δ is said to be an (n, k, δ) convolutional code.

Another important matrix associated with some convolutional codes \mathcal{C} is the (full row rank $(n - k) \times n$) **parity-check matrix** noted as $H(z)$. This matrix is the generator matrix of the dual code of \mathcal{C} ; therefore, we can generate \mathcal{C} as its kernel as follows:

$$\begin{aligned} \mathcal{C} &= \ker_{\mathbb{F}[z]} H(z) \\ &= \{v(z) \in \mathbb{F}[z]^n \mid H(z)v(z) = \mathbf{0}\}. \end{aligned}$$

The parity-check matrix plays a central role in the decoding process for block codes and convolutional codes. Although there is always a parity-check matrix for a block code, this is not the case for convolutional codes. In [11], it was shown that a convolutional code

admits a parity-check matrix if and only if it is non-catastrophic. Any non-catastrophic convolutional code admits a left prime and row-reduced parity-check matrix [12].

When transmitting information through erasure channels we may lose some parts of the information. Convolutional codes have been proven to be good for communication over these channels [13].

In [13], a decoding algorithm for these kinds of channels was presented, and it is based on solving linear systems of equations. Consider that we receive the codeword $v(z) = \sum_{i=0}^j v_i z^i \in \mathbb{F}_q[z]$, and that v_{i-t}, \dots, v_t are correct (have no erasures) but the coefficients afterward may have erasures. Let $H(z) = \sum_{j \in \mathbb{N}} H_j z^j \in \mathbb{F}_q[z]^{(n-k) \times n}$ be a parity-check matrix of the code. We define the matrix as follows:

$$\mathfrak{H}_j = \begin{pmatrix} H_\nu & \cdots & H_0 & & \\ & \ddots & & \ddots & \\ & & & & \\ & & & H_\nu & \cdots & H_0 \end{pmatrix} \in \mathbb{F}_q^{(j+1)(n-k) \times (v+j+1)n}. \tag{1}$$

Then, $\mathfrak{H}_j (v_{t-\nu}^T \dots v_t^T \dots v_{t+j}^T)^T = 0$. By reordering these equations we can obtain a linear system of the form

$$H^e v^e = b, \tag{2}$$

where H^e and v^e are the columns of \mathfrak{H}_j and the coordinates of $(v_{t-\nu}^T \dots v_t^T \dots v_{t+j}^T)^T$ that correspond with the erasures, respectively; and $b = -H^c v^c$, where H^c and v^c are analogously defined concerning the correctly received information. Therefore, we can correct the erasures by solving a linear system with as many unknowns as erasures in the received codeword. Note that we have considered only part of the coefficients of $v(z)$, i.e., the coefficients $v_{t-\nu}, \dots, v_t, \dots, v_{t+j}$. We refer to this sequence as a **window**, and we say that it is a window of $j + \nu + 1$ size.

Notice that, in order to correct the erasures produced in a codeword, we need a previous **safe space** where all coefficients are correctly recovered. In the construction of the system (2), the ν vectors, $v_{t-\nu}, \dots, v_{t-1}$ —which were previous to v_t and the first coefficient of $\hat{v}(z)$ with some erasure—conformed to this safe space.

Once we have a method for correcting the erasures produced by a channel, we may want to know the correction capability that a code can achieve. This capacity is described in terms of the distances of the code. For convolutional codes, there exist two different distances: the free distance and the column distance. The **Hamming weight** of a vector $v(z) = \sum_{i \in \mathbb{N}_0} v_i z^i \in \mathbb{F}_q[z]^n$ is given by the expression

$$wt(v(z)) = \sum_{i \in \mathbb{N}_0} wt(v_i),$$

where $wt(v_i)$ is the number of non-zero coordinates of v_i .

Definition 2. Let \mathcal{C} be an (n, k) convolutional code, then the **free distance** of \mathcal{C} is

$$d_{free}(\mathcal{C}) = \min\{wt(v(z)) : v(z) \in \mathcal{C} \setminus \{0\}\}.$$

The next theorem establishes an upper bound on the free distance of a convolutional code, and it is called the **generalized Singleton bound**.

Theorem 1 ([14]). Let \mathcal{C} be an (n, k, δ) convolutional code. Then,

$$d_{free}(\mathcal{C}) \leq (n - k) \left(\left\lfloor \frac{\delta}{k} \right\rfloor + 1 \right) + \delta + 1.$$

The free distance of a convolutional code allows us to know the correction capacity for errors and erasures once the whole codeword is received.

An advantage of convolutional codes over block codes is that they permit us to do a partial decoding, i.e., we can start recovering erasures even though we do not have the complete codeword. The measurement of the capability of this kind of correction is given by the **column distances**, which are defined for non-catastrophic convolutional codes [15].

Given a vector $w(z) = \sum_{i \in \mathbb{N}_0} w_i z^i \in \mathbb{F}[z]^n$ and $j_1, j_2 \in \mathbb{N}_0$ with $j_1 < j_2$, we defined the **truncation** of $w(z)$ to the interval $[j_1, j_2]$ as $w_{[j_1, j_2]}(z) = w_{j_1} z^{j_1} + w_{j_1+1} z^{j_1+1} + \dots + w_{j_2} z^{j_2}$.

Definition 3. Let C be an (n, k, δ) non-catastrophic convolutional code, then the j -column distance of C is defined by

$$d_j^c = \min\{v_{[0, j]}(z) : v(z) \in C \text{ with } v(0) \neq 0\}.$$

The following inequalities can be directly deduced from the previous definitions [15]:

$$d_0^c \leq d_1^c \leq \dots \leq \lim_{j \rightarrow \infty} d_j^c = d_{free}(C). \tag{3}$$

In addition to the free distance having a bound, the column distances also have a **Singleton-like** bound, which is given by the following theorem.

Theorem 2 ([15]). Let C be an (n, k, δ) non-catastrophic convolutional code. Then, for $j \in \mathbb{N}_0$, we have

$$d_j^c \leq (n - k)(j + 1) + 1.$$

Moreover, if $d_j^c = (n - k)(j + 1) + 1$ for some $j \in \mathbb{N}_0$, then $d_i^c = (n - k)(i + 1) + 1$ for $i \leq j$.

As seen in (3), the column distances cannot be greater than the free distance. In fact, there exists an integer

$$L = \left\lfloor \frac{\delta}{k} \right\rfloor + \left\lfloor \frac{\delta}{n - k} \right\rfloor$$

for which d_j^c can be equal to $(n - k)(j + 1) + 1$, i.e., the bound can be held for $j \leq L$ and $d_j^c < (n - k)(j + 1) + 1$, as well as for $j > L$ [15]. An (n, k, δ) convolutional code with $d_j^c = (n - k)(j + 1) + 1$ for $j \leq L$ (or equivalently with $d_L^c = (n - k)(L + 1) + 1$) is called a **maximum distance profile (MDP)** code. Next, we detail a characterization of these codes in terms of their generator and parity-check matrices. For that, we need the definition of a superregular matrix.

The determinant of a square matrix $A = [\mu_{i, \ell}]$ over a field \mathbb{F} is given by

$$|A| = \sum_{\sigma \in S_m} (-1)^{\text{sgn}(\sigma)} \mu_{1, \sigma(1)} \mu_{2, \sigma(2)} \dots \mu_{m, \sigma(m)},$$

where S_m is the symmetric group. We will refer to *term* as the addends of $|A|$, that is, to the product of the form $\mu_{1, \sigma(1)} \mu_{2, \sigma(2)} \dots \mu_{m, \sigma(m)}$ and as a *component* of the factor of a term, i.e., the multiplicands of the form $\mu_{i, \mu(i)}$ for $i \in \{1, \dots, m\}$. We will also say that a *term is trivial* if at least one of its components is zero, that is, the term is zero. Now, let A be a square submatrix of a matrix B ; if all the terms of $|A|$ are trivial, then we say that $|A|$ is a trivially zero minor of B or that it is a trivial minor of B .

Definition 4 ([16,17]). Let $n, m \in \mathbb{N}$ and $B \in \mathbb{F}^{n \times m}$, then B is superregular if all of its non-trivial zero minors are non-zero.

Theorem 3 ([15]). Let C be a non-catastrophic convolutional code with a right prime and column-reduced generator matrix $G(z) = \sum_{i=0}^{\mu} G_i z^i \in \mathbb{F}_q[z]^{n \times k}$, as well as a left prime and row-reduced parity-check matrix $H(z) = \sum_{i=0}^{\nu} H_i z^i \in \mathbb{F}_q[z]^{(n-k) \times n}$, then the following statements are equivalent:

1. \mathcal{C} is an MDP code
2. $\mathcal{G} := \begin{bmatrix} G_0 & \cdots & G_L \\ & \ddots & \vdots \\ & & G_0 \end{bmatrix}$, where $G_i = 0$ for $i > \mu$, and it also has the property that every full-size minor that is not trivially zero is non-zero.
3. $\mathcal{H} := \begin{bmatrix} H_0 & & \\ \vdots & \ddots & \\ H_L & \cdots & H_0 \end{bmatrix}$, where $H_i = 0$ for $i > \nu$, and it also has the property that every full-size minor that is not trivially zero is non-zero.

Lemma 1 ([13]). *Let \mathcal{C} be an (n, k, δ) MDP convolutional code. If any sliding window of length $(j_0 + 1)n$ at most $(j_0 + 1)(n - k)$ erasures occur with $j_0 \in \{0, 1, \dots, L\}$, then we can completely recover the transmitted sequence.*

This family of optimal codes requires a great deal of regularity in the sense of the previous theorem, and this causes the construction of the MDP convolutional codes to usually require big base fields. From the applied point of view, this is a concern. In last few years, the efforts of the community have been directed to improving this situation by either giving different constructions [18–21] or finding bounds [22,23]. Research has also led to the development of MDP convolutional codes over finite rings [24].

When MDP codes cannot perform a correction due to the accumulation of erasures, we have to consider some of the packets to be lost and to continue until a safe space is found again. To solve this situation, in [13,25–27], a definition for **Reverse-MDP** convolutional codes was provided. While the usual MDP convolutional codes can only carry out a forward correction, i.e., their decoding direction is from left to right, the reverse-MDP codes will allow a backward correction, i.e., a correction from right to left.

Proposition 1 ([26], Prop 2.9). *Let \mathcal{C} be an (n, k, δ) convolutional code with a right prime and column-reduced generator matrix $G(z)$. Let $\bar{G}(z)$ be the matrix obtained by replacing each entry $g_{ij}(z)$ of $G(z)$ with $\bar{g}_{i,j} := z^{\delta_j} g_{i,j}(z^{-1})$, where δ_j is the i -th column degree of $G(z)$. Then, $\bar{G}(z)$ is a right prime, and column-reduced generator matrix of an (n, k, δ) convolutional code $\bar{\mathcal{C}}$ and*

$$v_0 + z_1z + \cdots + v_s z^s \in \mathcal{C} \Leftrightarrow v_s + z_{s-1}z + \cdots + v_0 z^s \in \bar{\mathcal{C}}.$$

We call $\bar{\mathcal{C}}$ the reverse code of \mathcal{C} . Similarly, we denote the parity-check matrix of $\bar{\mathcal{C}}$ by $\bar{H}(z) = \sum_{i=0}^{\nu} \bar{H}_i z^i$.

Definition 5. *Let \mathcal{C} be an MDP (n, k, δ) convolutional code. We can then say that \mathcal{C} is a reverse-MDP convolutional code if the reverse code $\bar{\mathcal{C}}$ is also an MDP code.*

In [13], it was proven that reverse-MDP codes can perform a correction from right to left as efficiently as an MDP code does in the opposite direction. However, to perform a backward correction, a safe space is required on the right side of the erasures. This can be easily seen in the following example.

Example 1. *Let us assume that we are communicating through an erasure channel. If so, then we have recovered the information correctly up to an instant t and we can later receive the following pattern:*

$$\begin{array}{ccccccc} \overbrace{\star \star \cdots \star}^{(A)14} & \overbrace{\mathbf{v}\mathbf{v} \star \mathbf{v}\mathbf{v} \star \cdots \mathbf{v}\mathbf{v} \star \star}^{(B)108} & \overbrace{|\mathbf{v}\mathbf{v} \cdots \mathbf{v}|}^{(C)122} \\ \overbrace{\star \star \cdots \star}^{(D)50} & \overbrace{\mathbf{v}\mathbf{v} \cdots \mathbf{v}}^{(E)38} & \overbrace{\star \star \cdots \star}^{(F)34} & \overbrace{|\mathbf{v}\mathbf{v} \cdots \mathbf{v}|}^{(G)122} \end{array},$$

where \star indicates that the corresponding component has been erased and \mathbf{v} means that the component has been correctly received. If we use a $(2, 1, 30)$ MDP convolutional code, we cannot perform a correction due to the accumulation of erasures at the beginning of the sequence. Nevertheless, if we consider a $(2, 1, 30)$ reverse-MDP convolutional code and take the first 60 symbols of (C) as a safe space, then we can correct the erasures in (B). We can repeat this method by taking the first 60 symbols in (G) and recovering the section (F), that is, we build the linear system as in (2) by considering the following sets of symbols:

$$\begin{array}{c} \overbrace{\mathbf{v}\mathbf{v}\star\star\mathbf{v}\mathbf{v}\star\star\cdots\mathbf{v}\mathbf{v}\star\star}^{(B)108} \mid \overbrace{\mathbf{v}\mathbf{v}\cdots\mathbf{v}}^{(C)60}, \\ \text{and } \overbrace{\mathbf{v}\mathbf{v}\cdots\mathbf{v}}^{(E)34} \overbrace{\star\star\cdots\star}^{(F)34} \mid \overbrace{\mathbf{v}\mathbf{v}\cdots\mathbf{v}}^{(G)60}. \end{array}$$

This example shows that reverse-MDP convolutional codes can perform better than MDP convolutional codes when transmitting over an erasure channel since we can exploit its backward capability correction. However, this type of approach depends on the fact that over the communication there exists a long enough sequence of symbols that have been correctly received to play the role of a safe space for any of the decoding directions. The following codes are aimed at solving the situation in which a safe space cannot be found.

Definition 6 ([13]). Let $H(z) = \sum_{i=0}^v H_i z^i \in \mathbb{F}_q[z]^{(n-k) \times n}$ be a parity-check matrix of an (n, k, δ) convolutional code \mathcal{C} and $L := \lfloor \frac{\delta}{n-k} \rfloor + \lfloor \frac{\delta}{k} \rfloor$, then we obtain the matrix

$$\mathfrak{H} := \begin{pmatrix} H_v & \cdots & H_0 & & \\ & \ddots & & \ddots & \\ & & & & H_v & \cdots & H_0 \end{pmatrix} \in \mathbb{F}^{(L+1)(n-k) \times (v+L+1)n},$$

which is called a partial parity-check matrix of \mathcal{C} . Moreover, \mathcal{C} is said to be a **complete-MDP convolutional code** if, for any of its parity-check matrices $H(z)$, every full-size minor of \mathfrak{H} that is non-trivially zero is non-zero.

Complete-MDP convolutional codes have an extra feature on top of being able to perform a correction as a MDP or as reverse-MDP convolutional codes. If, in the process of decoding the code, we cannot accomplish the correction due to the accumulation of too many erasures, it can compute a new safe space to continue with the process as soon as a correctable sequence of symbols is found.

Theorem 4 ([13], Theorem 6.6). Given a code sequence from a complete-MDP convolutional code. If, in a window of size $(v + L + 1)n$, there are not more than $(L + 1)(n - k)$ erasures, and if they are distributed in such a way that between position 1 and sn and between positions $(v + L + 1)n$ and $(v + L + 1)n - s(n - k)$, for $s = 1, 2, \dots, L + 1$, i.e., there are no more than $s(n - k)$ erasures, then the full correction of all symbols in this interval will be possible. In particular, a new safe space can be computed.

For more constructions, examples, and further content on complete-MDP convolutional codes, we refer the reader to [12,13,17].

Example 2. Again, let us assume that we are using an erasure channel. In this case, we are not able to recover some of the previous symbols and we thus receive the following pattern:

$$\begin{array}{c} \dots \star \star | \overbrace{\star \star \dots \star}^{(A)14} \overbrace{\mathbf{v} \mathbf{v} \dots \mathbf{v}}^{(B)21} \overbrace{\star \star \dots \star}^{(C)12} \overbrace{\mathbf{v} \mathbf{v} \dots \mathbf{v}}^{(D)28} | \\ \overbrace{\mathbf{v} \mathbf{v} \dots \mathbf{v}}^{(E)19} \overbrace{\star \star \dots \star}^{(F)13} \overbrace{\mathbf{v} \mathbf{v} \dots \mathbf{v}}^{(G)30} \overbrace{\star \star \dots \star}^{(H)13} \\ \overbrace{\mathbf{v} \mathbf{v} \dots \mathbf{v}}^{(I)30} \overbrace{\star \star \dots \star}^{(J)6} \overbrace{\mathbf{v} \mathbf{v} \dots \mathbf{v}}^{(K)17} \overbrace{\star \star \dots \star}^{(L)22} | \star \star \dots \end{array}$$

Note that, if we use a (3, 2, 16) MDP or reverse-MDP convolutional code, we require a safe space of 48 symbols to correct the erasures in any of the directions, which we cannot find.

Nevertheless, if we use a (3, 2, 16) complete-MDP convolutional code, we still have one more feature to use. We can compute a new safe space by using Theorem 4, that is, find a window of size $(L + 1 + v)n = (24 + 1 + 16)3 = 123$ where not more than 25 erasures occur. In the received pattern of erasures we can find the following sequence:

$$\overbrace{\mathbf{v} \mathbf{v} \dots \mathbf{v}}^{(B)21} \overbrace{\star \star \dots \star}^{(C)12} \overbrace{\mathbf{v} \mathbf{v} \dots \mathbf{v}}^{(D)28} | \overbrace{\mathbf{v} \mathbf{v} \dots \mathbf{v}}^{(E)19} \overbrace{\star \star \dots \star}^{(F)13} \overbrace{\mathbf{v} \mathbf{v} \dots \mathbf{v}}^{(G)30}.$$

When erasures are recovered, we have a new safe space and we can perform the usual correction.

In [13], an algorithm to recover the information over an erasure channel by performing correction in both directions, i.e., forward and backward, is given.

We provide Algorithm 1, which is a new version of the abovementioned algorithms and includes the case for which the code can decode by using the complete-MDP property. We will maintain the notation given in [13], that is, the value 0 means that a symbol or a sequence of symbols has not been received and that 1 is correctly recovered. The function `findzeros(v)` returns a vector with the positions of the zeros in \mathbf{v} , as well as `forward(C, j, v)`, `backward(C̄, j, v)`, and `complete(C, v)`, which are the forward, backward, and complete recovering functions, respectively. Note that these functions use the parity-check matrices of \mathcal{C} and $\bar{\mathcal{C}}$ to recover the erasures that appear in \mathbf{v} within a window of size $(j + 1)n$ (when necessary).

2.2. 2D Convolutional Codes

Definition 7 ([1]). A 2D finite support convolutional code \mathcal{C} of rate k/n is a free $\mathbb{F}[z_1, z_2]$ -submodule of $\mathbb{F}[z_1, z_2]^n$ with rank k .

As for the 1D case, a full column rank polynomial matrix $\hat{\mathbf{G}}(z_1, z_2)$, whose columns form a basis for the code, is such that we can express it as follows:

$$\begin{aligned} \mathcal{C} &= \text{Im}_{\mathbb{F}[z_1, z_2]} \hat{\mathbf{G}}(z_1, z_2) \\ &= \left\{ \hat{\mathbf{v}}(z_1, z_2) \in \mathbb{F}[z_1, z_2]^n \mid \hat{\mathbf{v}}(z_1, z_2) = \hat{\mathbf{G}}(z_1, z_2) \hat{\mathbf{u}}(z_1, z_2) \text{ with } \hat{\mathbf{u}}(z_1, z_2) \in \mathbb{F}[z_1, z_2]^k \right\}, \end{aligned}$$

which is called a **generator matrix** of \mathcal{C} .

A polynomial matrix $G(z_1, z_2) \in \mathbb{F}[z_1, z_2]^{n \times k}$ is said to be right factor prime if $G(z_1, z_2) = \bar{G}(z_1, z_2)X(z_1, z_2)$, then for some $X(z_1, z_2) \in \mathbb{F}[z_1, z_2]^{k \times k}$ and $\bar{G}(z_1, z_2) \in \mathbb{F}[z_1, z_2]^{n \times k}$, $X(z_1, z_2)$ is unimodular (i.e., $X(z_1, z_2)$ has a polynomial inverse). Again, similarly to the 1D case, if the code admits a right factor prime generator matrix, then it can be defined by using its full rank polynomial parity-check matrix $\hat{\mathbf{H}}(z_1, z_2) \in \mathbb{F}[z_1, z_2]^{(n-k) \times n}$ as follows:

$$\mathcal{C} = \text{Ker}_{\mathbb{F}[z_1, z_2]} \hat{\mathbf{H}}(z_1, z_2) = \left\{ \hat{\mathbf{v}}(z_1, z_2) \in \mathbb{F}[z_1, z_2]^n \mid \hat{\mathbf{H}}(z_1, z_2) \hat{\mathbf{v}}(z_1, z_2) = \mathbf{0} \right\}.$$

Algorithm 1 Decoding algorithm for complete-MDP codes

Input: $[v_0, \dots, v_\ell]$ the received sequence.

Output: $[v_0, \dots, v_\ell]$ the corrected sequence.

```

1:  $i = 0$ 
2: while  $i \leq \ell$  do
3:    $forwardsucces = 0$ 
4:    $backwardsucces = 0$ 
5:    $completesucces = 0$ 
6:   if  $v_i = 0$  then
7:     if  $[v_{i-vn}, \dots, v_{i-1}] = 1$  then
8:        $j = L$ 
9:       while  $forwardsucces = 0$  and  $j \geq 0$  do
10:        if  $\text{length}(\text{findzeros}([v_i, \dots, v_{i+(j+1)n-1}])) \leq (j+1)(n-k)$  then
11:           $[v_{i-vn}, \dots, v_{i+(j+1)n-1}] := \text{forward}(\mathcal{C}, j, [[v_{i-vn}, \dots, v_{i+(j+1)n-1}]])$ 
12:           $forwardsucces = 1$ 
13:           $i = i + (j+1)n - 1$ 
14:           $j = j - 1$ 
15:       if  $forwardsucces \neq 1$  then
16:          $Aux = \text{findzeros}([v_i, \dots, v_{i+(L+1)n-1}])$ 
17:          $k = i + \text{Aux}[\text{length}(Aux)] - 1$ 
18:         while  $backwardsucces = 0$  and  $k \leq \ell$  do
19:           if  $[v_k, \dots, v_{k+vn-1}] = 1$  then
20:              $j = L$ 
21:             while  $backwardsucces = 0$  and  $j \geq 0$  do
22:              if  $\text{length}(\text{findzeros}([v_{k-(j+1)n}, \dots, v_{k-1}])) \leq (j+1)(n-k)$  then
23:                 $[v_{k-(j+1)n}, \dots, v_{k+vn-1}] := \text{backward}(\bar{\mathcal{C}}, j, [v_{k-(j+1)n}, \dots, v_{k+vn-1}])$ 
24:                 $backwardsucces = 1$ 
25:                 $i = k + vn - 1$ 
26:                 $j = j - 1$ 
27:                 $k = k + 1$ 
28:             if  $backwardsucces \neq 1$  then
29:                $h = i$ 
30:               while  $completesucces = 0$  and  $h \leq \ell$  do
31:                  $E1 := \{\text{length}(\text{findzeros}([v_h, \dots, v_{h+(j+1)n-1}])) \text{ for } j = 0, \dots, L\}$ 
32:                  $E2 := \{\text{length}(\text{findzeros}([v_{h+(v+L+1)n-(j+1)(n-k)}, \dots, v_{h+(v+L+1)n}])) \text{ for } j = 0, \dots, L\}$ 
33:                 if  $E1[j] \leq (j+1)(n-k)$  and  $E2[j] \leq (j+1)(n-k)$  for all  $j = 0, \dots, L$  then
34:                    $[v_h, \dots, v_{h+(v+L+1)n-1}] := \text{complete}(\mathcal{C}, [v_h, \dots, v_{h+(v+L+1)n-1}])$ 
35:                    $completesucces = 1$ 
36:                    $i = i + (v+L+1)n$ 
37:                    $h = h + 1$ 
38:                  $i = i + 1$ 

```

Since we are going to deal with a situation in which the elements of the codewords are distributed in the plane \mathbb{N}^2 , we will consider the order given by

$$(a, b) \prec (c, d) \text{ if and only if } a + b < c + d, \\ \text{or } a + b = c + d \text{ and } b < d.$$

The codeword $\hat{\mathbf{v}}(z_1, z_2) = \sum_{0 \leq a+b \leq \gamma} \mathbf{v}(a, b) z_1^a z_2^b \in \mathbb{F}[z_1, z_2]^n$ and the matrix $\hat{\mathbf{H}}(z_1, z_2) = \sum_{0 \leq i+j \leq v} \mathbf{H}(i, j) z_1^i z_2^j \in \mathbb{F}[z_1, z_2]^{(n-k) \times n}$ are represented with its coefficients in order, respectively, as follows:

$$\hat{\mathbf{v}}(z_1, z_2) = \mathbf{v}(0, 0) + \mathbf{v}(1, 0)z_1 + \mathbf{v}(0, 1)z_2 + \dots + \mathbf{v}(\gamma, 0)z_1^\gamma + \dots + \mathbf{v}(0, \gamma)z_2^\gamma$$

and

$$\hat{\mathbf{H}}(z_1, z_2) = H(0, 0) + H(1, 0)z_1 + H(0, 1)z_2 + \cdots + H(\delta, 0)z_1^\nu + \cdots + H(0, \delta)z_2^\nu.$$

Note that this depiction allows us to see the kernel representation in a more detailed manner as

$$\sum_{0 \leq a+b \leq \gamma} \left[\sum_{0 \leq i+j \leq \nu} H(i, j) \mathbf{v}(a-i, b-j) \right] z_1^a z_2^b = 0,$$

where $\mathbf{v}(l, k) = 0$ if $l+k > \gamma + \nu$ or $l+k < 0$. It is also possible to denote this product using constant matrices

$$\mathbf{H}\mathbf{v} = \mathbf{0}, \tag{4}$$

where

$$\mathbf{v} = \left[v(0, 0)^T \quad v(1, 0)^T \quad v(0, 1)^T \quad v(2, 0)^T \quad v(1, 1)^T \quad \cdots \quad v(0, \gamma)^T \right]^T$$

is a vector in $\mathbb{F}^{\frac{(\gamma+1)(\gamma+2)}{2}n}$, and \mathbf{H} is a $\frac{(\gamma+\nu+1)(\gamma+\nu+2)}{2}(n-k) \times \frac{(\gamma+1)(\gamma+2)}{2}n$ matrix over \mathbb{F} . An example of this matrix correspondent to a parity-check matrix $\hat{\mathbf{H}}(z_1, z_2) = \sum_{0 \leq i+j \leq 3} \mathbf{H}(i, j)z_1^i z_2^j$ is presented in Figure 1. Is easy to see that it does not follow the same pattern of construction as the partial parity-check matrix in the 1D case. Note that all the matrix coefficients $H(i, j)$ of $\hat{\mathbf{H}}(z_1, z_2)$ appeared in all the columns following the previously established order \prec with the particularity that, for $t = 0, 1, 2, \dots$ in the block columns with indices $\frac{t(t+1)}{2} + 1, \frac{t(t+1)}{2} + 2, \dots, \frac{t(t+1)}{2} + t + 1$, the coefficients $H(i, j)$ with $i+j = d$ for $d = 0, 1, 2, \dots, \delta - 1$ were separated from the matrices $H(i, j)$ with $i+j = d + 1$ by t zero blocks.

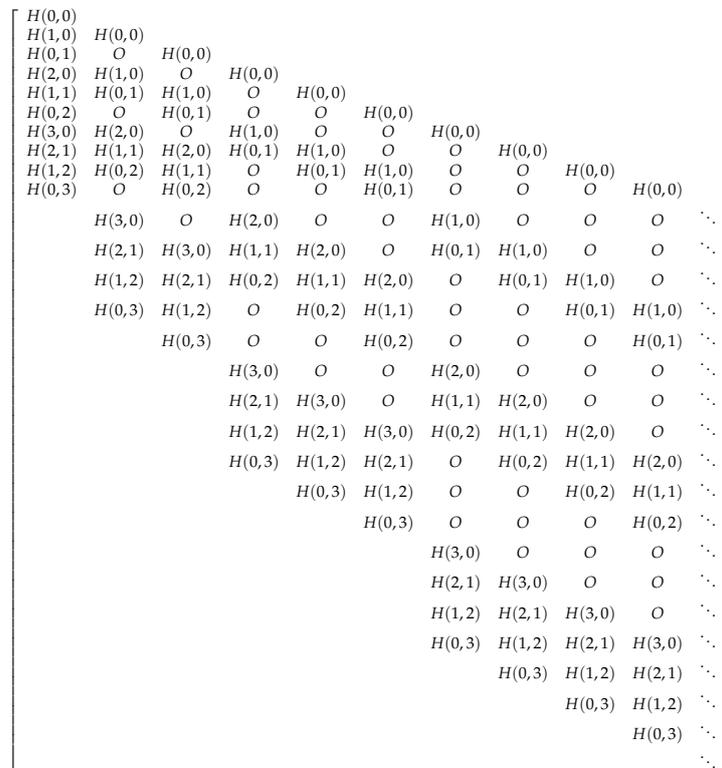


Figure 1. Matrix H.

Consider now that $\hat{v}(z_1, z_2) \in \mathbb{F}[z_1, z_2]$ has been transmitted over an erasure channel. We define the support of $\hat{v}(z_1, z_2) = \sum_{(a,b) \in \mathbb{N}_0^2} v(a,b)z_1^a z_2^b$ as the set of indices of the coefficients of $\hat{v}(z_1, z_2)$ that are non-zero, i. e.,

$$\text{supp}(\hat{v}(z_1, z_2)) = \{(a, b) \in \mathbb{N}_0^2 : v(a, b) \neq 0\}.$$

Let $\mathcal{E}(\hat{v}(z_1, z_2))$ be the set of indices of the support of $\hat{v}(z_1, z_2)$ in which there are erasures in the corresponding coefficients of $\hat{v}(z_1, z_2)$ and $\bar{\mathcal{E}}(\hat{v}(z_1, z_2)) = \text{supp}(\hat{v}(z_1, z_2)) \setminus \mathcal{E}(\hat{v}(z_1, z_2))$, i.e., the set of indices of the support of $\hat{v}(z_1, z_2)$ correspond to the coefficients that were correctly received. For the sake of simplicity, and if the context allows it, we will denote $\mathcal{E}(\hat{v}(z_1, z_2))$ as \mathcal{E} . In Figure 2 an example of erasures distributed in the plane is presented.

Since we have the kernel representation (4) of the code in a block fashion, we can consider the equivalent linear system $\mathbf{H}_{\mathcal{E}} \mathbf{v}_{\mathcal{E}} = -\mathbf{H}_{\bar{\mathcal{E}}} \mathbf{v}_{\bar{\mathcal{E}}}$, where $\mathbf{H}_{\mathcal{E}}$ and $\mathbf{H}_{\bar{\mathcal{E}}}$ denote the submatrices of \mathbf{H} where block columns are indexed by \mathcal{E} and $\bar{\mathcal{E}}$, respectively. Correspondingly, $\mathbf{v}_{\mathcal{E}}$ and $\mathbf{v}_{\bar{\mathcal{E}}}$ refer to the subvectors of \mathbf{v} , whose block rows are indexed by \mathcal{E} and $\bar{\mathcal{E}}$, respectively. In order to recover the erasures, we solved the linear system $\mathbf{H}_{\mathcal{E}} \mathbf{v}_{\mathcal{E}} = -\mathbf{H}_{\bar{\mathcal{E}}} \mathbf{v}_{\bar{\mathcal{E}}}$ by considering the erased components of $\mathbf{v}_{\mathcal{E}}$ as the unknowns. Note that $\mathbf{H}_{\bar{\mathcal{E}}} \mathbf{v}_{\bar{\mathcal{E}}}$ is known.

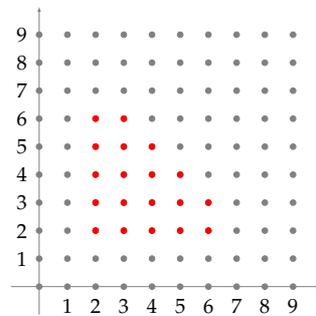


Figure 2. Erasures (red dots) distributed in \mathbb{N}^2 .

As pointed out in [9], this system is massive even when the parameters of the code are small. The authors in this paper developed an algorithm to deal with this situation in being able to recover the information more efficiently. They proposed to decode it by a set of lines, that is, to choose a set of erasures in a line \mathcal{E}^l (i.e., horizontal (\mathcal{E}^h), vertical (\mathcal{E}^v), or diagonal (\mathcal{E}^d)), as well as to define a neighborhood for this set of erasures and then use all the correct information within these related coefficients to construct and solve a linear system.

Next, we describe such a method for the correction of erasures in horizontal lines, wherein the method for vertical and diagonal will be analogous. Let us consider a subset of \mathcal{E} , \mathcal{E}_s^h , where its subindices lie in an horizontal line with the vertical coordinate s , that is,

$$\mathcal{E}_s^h = \{(r, s) \mid r \in \mathbb{N}_0\} \cap \mathcal{E}. \tag{5}$$

With this in mind we can “rewrite” (4) as

$$\mathbf{H}_{\mathcal{E}_s^h} \mathbf{v}_{\mathcal{E}_s^h} = -\mathbf{H}_{\bar{\mathcal{E}}_s^h} \mathbf{v}_{\bar{\mathcal{E}}_s^h}, \tag{6}$$

where—similar to the above— $\mathbf{H}_{\mathcal{E}_s^h}$ and $\mathbf{H}_{\bar{\mathcal{E}}_s^h}$ indicate the submatrices of \mathbf{H} that are indexed by \mathcal{E}_s^h and $\bar{\mathcal{E}}_s^h = \text{supp}(\hat{v}(z_1, z_2)) \setminus \mathcal{E}_s^h$ block-wise, respectively, as well as where $\mathbf{v}_{\mathcal{E}_s^h}$ and $\mathbf{v}_{\bar{\mathcal{E}}_s^h}$ are defined analogously. (Note that $\mathbf{v}_{\bar{\mathcal{E}}_s^h}$ may contain erasures.)

Definition 8. Let W_t be a horizontal window of length $t + 1$. i.e.,

$$W_t = \{(r, s), (r + 1, s), \dots, (r + t, s)\},$$

for some $r \in \mathbb{N}_0$. We then define the following neighborhood of W_t as

$$\Omega_v(W_t) = \{(a,b) \mid r - v \leq a \leq r + t + v, s - v \leq b \leq s, a + b \geq r + s - v, r \geq v, s \geq v\}.$$

Example 3. Let $W_3 = \{(3,4), (4,4), (5,4), (6,4)\}$, which is represented in Figure 3 as big red dots. In addition, $\Omega_2(W_3)$ is depicted in Figure 3.

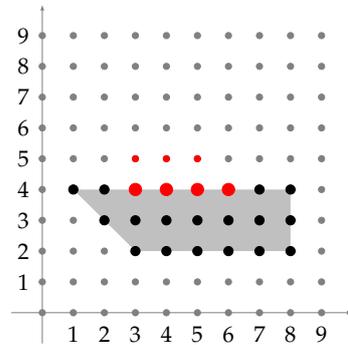


Figure 3. $\Omega_2(W_3)$, neighborhood of W_3 .

Note that the neighborhood is constituted of an aligned set of triangles. The main role of the above defined neighbors in the decoding process is described in the next set of results from [9].

Lemma 2 ([9]). Let $\mathcal{C} = \text{Ker}_{\mathbb{F}[z_1, z_2]} \hat{H}(z_1, z_2)$ with $\hat{H}(z_1, z_2) = \sum_{0 \leq i+j \leq v} H(i, j) z_1^i z_2^j$. Suppose that $\hat{\mathbf{v}}(z_1, z_2) \in \mathcal{C}$ is a transmitted codeword, then \mathcal{E} is the support of its coefficients with erasures and $\mathcal{E}^h \subseteq \mathcal{E}$ is the support of the coefficients with erasures that are distributed on a horizontal line in \mathbb{N}_0^2 in a window W of length $t + 1$ such that $\mathcal{E}^h \subset W$. Consider \mathbf{H} and \mathbf{v} as in (4), as well as \mathbf{H}_W and \mathbf{v}_W as in (6). Then, define the vector \mathbf{v}_Ω by selecting the coefficients $v(c, d)$ of \mathbf{v} with $(c, d) \in \Omega_v(W) \setminus W$. Define \mathbf{H}_{Ω_v} as a submatrix of \mathbf{H} accordingly. Then, it holds that

$$\mathbf{H}_W \mathbf{v}_W = -\mathbf{H}_{\Omega_v} \mathbf{v}_{\Omega_v}, \tag{7}$$

where

$$\mathbf{H}_W = \begin{bmatrix} H(0,0) \\ H(1,0) & H(0,0) \\ H(2,0) & H(1,0) & H(0,0) \\ \vdots & \vdots & \vdots & \ddots \\ H(v,0) & H(v-1,0) & H(v-2,0) \\ & H(v,0) & H(v-1,0) & \dots & \dots & H(0,0) \\ & & \ddots & \ddots & \ddots & \ddots \\ & & & \ddots & \ddots & & H(0,0) \\ & & & & \ddots & & \vdots \\ & & & & & H(v,0) & H(v-1,0) \\ & & & & & & H(v,0) \end{bmatrix}, \tag{8}$$

is an $(n - k)(t + 1 + v - j) \times n(t + 1)$ matrix.

can be made. Finally, we followed the same procedure with diagonal lines of erasures, and we then repeated this cycle until any of them were able to recover further information.

3. Construction of 2D Convolutional Codes from 1D Convolutional Codes

In this section, we propose a construction of 2D convolutional codes from a 1D convolutional code.

Let \hat{C} be an (n, k) non-catastrophic 1D convolutional code with a parity-check matrix $\hat{H}(z) = \sum_{i=0}^{2v} H_i z^i \in \mathbb{F}_q[z]^{(n-k) \times n}$. Consider the 2D convolutional code \mathcal{C} , which is defined by the parity-check matrix

$$\begin{aligned} H(z_1, z_2) &= \sum_{i=0}^v \sum_{j=i}^{2i} H_j z_1^{2i-j} z_2^{j-i} \\ &= H_0 + H_1 z_1 + H_2 z_2 + H_2 z_1^2 + H_3 z_1 z_2 \\ &\quad + H_4 z_2^2 + \dots + H_v z_1^v + \dots + H_{2v} z_2^v. \end{aligned} \tag{12}$$

We consider the 1D convolutional codes associated with \mathcal{C} (named horizontal, vertical, and diagonal), which are constructed by its line positions as follows:

- Horizontal: $\mathcal{C}^h = \text{Ker}H^h(z)$;
- Vertical: $\mathcal{C}^v = \text{Ker}H^v(z)$;
- Diagonal: $\mathcal{C}^d = \text{Ker}H^d(z)$,

where the matrices H^h, H^v , and $H^d \in \mathbb{F}_q[z]^{(n-k) \times n}$ are defined by

$$\begin{aligned} H^h(z) &= \sum_{i=0}^v H_i z^i \\ &= H_0 + H_1 z + H_2 z^2 + \dots + H_v z^v. \\ H^v(z) &= \sum_{i=0}^v H_{2i} z^i \\ &= H_0 + H_2 z + H_4 z^2 + \dots + H_{2v} z^v. \\ H^d(z) &= \sum_{i=0}^v H_{v+i} z^i \\ &= H_v + H_{v+1} z + H_{v+2} z^2 + \dots + H_{2v} z^v. \end{aligned}$$

Naturally, at this point, one wants to know if the optimal properties of \hat{C} are inherited by the line associated codes $\mathcal{C}^h, \mathcal{C}^v$, and \mathcal{C}^d . In particular, we were interested in the complete-MDP property. In general, these codes did not inherit this property as the next examples show.

Example 6. In [28], Almeida and Lieb presented a description of all $(2, 1, 2)$ complete-MDP convolutional codes over the smallest possible fields. Let \hat{C} be the one of these $(2, 1, 2)$ convolutional code over \mathbb{F}_{13} , which is defined by the parity-check matrix $\hat{H}(z) = H_0 + H_1 z + H_2 z^2$, where $H_2 = [1, 1]$, $H_1 = [2, 9]$ and $H_0 = [8, 8]$, which is a complete-MDP (as stated in Theorem 13 [28]). Note that the partial parity-check matrices for the vertical, horizontal, and diagonal associated codes are

$$\mathfrak{H}^v = \begin{pmatrix} H_2 & H_0 \\ H_2 & H_0 \end{pmatrix} \mathfrak{H}^h = \begin{pmatrix} H_1 & H_0 \\ H_1 & H_0 \end{pmatrix} \mathfrak{H}^d = \begin{pmatrix} H_2 & H_1 \\ H_2 & H_1 \end{pmatrix},$$

respectively, and the parameters of these codes are $(2, 1, 1)$. For \mathfrak{H}^v , the non-trivial zero minor, which is zero, can be obtained by considering columns 1, 5, and 6:

$$\begin{vmatrix} 1 & 0 & 0 \\ 0 & 8 & 8 \\ 0 & 1 & 1 \end{vmatrix},$$

which means that C^v is not a complete-MDP. With a similar reasoning, we see that C^h and C^d are also not complete-MDPs. As such, in this case, no associated code maintained the complete-MDP property.

Next, we considered a 1D complete-MDP, which is defined in [17], as follows:

Lemma 4 ([17]). For $a, b \in \mathbb{N}$, with $b < a$, let

$$X = \begin{bmatrix} 1 & & & 0 \\ 1 & \ddots & & \\ & \ddots & \ddots & \\ 0 & & 1 & 1 \end{bmatrix} \in \mathbb{Z}^{a \times a}$$

and let $\hat{X} := (X^b)_{i_1, \dots, i_r}^{j_1, \dots, j_r}$ be constructed out of rows $1 \leq i_1 \leq \dots \leq i_r \leq a$ and columns $1 \leq j_1 \leq \dots \leq j_r \leq a$ of

$$X^b = \begin{bmatrix} 1 & & & & & & 0 \\ \binom{b}{1} & \ddots & & & & & \\ \vdots & \ddots & \ddots & & & & \\ \binom{b}{b-1} & & \ddots & \ddots & & & \\ 1 & \ddots & & \ddots & \ddots & & \\ & \ddots & \ddots & & \ddots & \ddots & \\ 0 & & 1 & \binom{b}{b-1} & \cdots & \binom{b}{1} & 1 \end{bmatrix}.$$

Then, we have $\det(\hat{X}) \geq 0$ and $\det(\hat{X}) > 0 \Leftrightarrow j_l \in \{i_l - b, \dots, i_l\}$ for $l = 1, \dots, r$.

Theorem 5 ([17]). With the notation from the preceding lemma, choose $X \in \mathbb{F}^{(v+L+1)n \times (v+L+1)n}$, i.e., $a := (v + L + 1)n$, as well as $b := vn + k$.

For $j = 0, \dots, L$, set $I_j = \{(v + j)n + k + 1, \dots, (v + j + 1)n\}$ and $I = \cup_{j=0}^L I_j$. Then, those rows of X^b , whose indices lie in I , form the partial parity-check matrix of an (n, k, δ) complete-MDP convolutional code if the characteristic of the field is greater than $\binom{vn+k}{\lfloor 1/2(vn+k) \rfloor}^{(n-k)(L+1)}$. $((n - k)(L + 1))^{1/2(n-k)(L+1)}$.

Let \hat{C} be the 1D complete-MDP convolutional code defined in Theorem 5 with the degree 2δ . In the construction, it was assumed that $2\delta = 2v(n - k)$. The associated line codes defined from \hat{C} had the parameters (n, k, δ) .

The next theorem shows that the associated vertical 1D convolutional code was also a complete-MDP.

Theorem 6. Let C be a 2D convolutional code that was constructed as in (12) by using a $(n, k, 2\delta)$ 1D complete-MDP convolutional code \hat{C} , as shown in Theorem 5 with the parity-check matrix $H(z) = \sum_{i=0}^{2v} H_i z^i$, where $2v = \frac{2\delta}{n-k}$. Then, the associated line code C^v with the parity-check matrix

$$H^v(z) = \sum_{i=0}^v H_{2i} z^i$$

is an (n, k, δ) 1D complete-MDP convolutional code.

Proof. The proof follows since every full-size minor of \mathfrak{H}^v that is not trivially zero is a non-trivial minor of X^b , which is a super-regular matrix by Lemma 4. In fact, Lieb showed Theorem 5 [17]. This proof involves demonstrating that the conditions of the indices of the columns in the partial parity-check matrix \mathfrak{H} , which describes non-trivial zero full-size minors, are equivalent to the conditions outlined in Lemma 4. Moreover, both sets of conditions were found to be mutually equivalent to $j_l \in \{l + kx, \dots, l + kx + vn + k\}$, where $x \in \{0, \dots, L\}$ is chosen such that $l \in \{x(n - k) + 1, \dots, (x + 1)(n - k)\}$. Now, to prove the theorem, it is sufficient to observe that the columns of \mathfrak{H}^v still satisfy $j_l \in \{l + 2kx, \dots, l + 2kx + 2vn + k\} \setminus \{(x + 1)n + 1, \dots, (x + 2)n\}$, where $x \in \{0, \dots, L/2\}$ is chosen such that $l \in \{2x(n - k) + 1, \dots, (2x + 1)(n - k)\}$. It is worth remembering that \mathfrak{H}^v is obtained from \mathfrak{H} by removing all the block-rows (of size $n - k$) and block-columns (of size n) with odd indices. \square

Although the horizontal and diagonal associated line codes do not maintain the optimality as is the case in the vertical one, these codes still satisfy a weaker optimality property.

Theorem 7. Let \mathcal{C} be a 2D convolutional code, constructed as in (12), by using a $(n, k, 2\delta)$ 1D complete-MDP convolutional code $\hat{\mathcal{C}}$, as represented in Theorem 5 with the parity-check matrix $H(z) = \sum_{i=0}^{2v} H_i z^i$, where $2v = \frac{2\delta}{n-k}$. Then, the horizontal \mathcal{C}^h and the reverse diagonal $\overline{\mathcal{C}}^d$ associated line codes are expressed with the parity-check matrices

$$\begin{aligned} H^h(z) &= \sum_{i=0}^v H_i z^i \\ &= H_0 + H_1 z + H_2 z^2 + \dots + H_v z^v. \\ \overline{H}^d(z) &= \sum_{i=0}^v H_{2v-i} z^i \\ &= H_v + H_{v+1} z + H_{v+2} z^2 + \dots + H_{2v} z^v. \end{aligned}$$

respectively, which are (n, k, δ) 1D convolutional codes with the column distance $d_v^c = (n - k)(v + 1) + 1$.

Proof. By ([15], Prop 2.1), we need to show that the matrices

$$\mathcal{H}^h := \begin{bmatrix} H_0 & & & \\ \vdots & \ddots & & \\ H_v & \dots & H_0 & \end{bmatrix} \quad \overline{\mathcal{H}}^d := \begin{bmatrix} H_{2v} & & & \\ \vdots & \ddots & & \\ H_v & \dots & H_{2v} & \end{bmatrix}$$

have the property that every full-size minor that is non-trivially zero is non-zero. Note that $\overline{\mathcal{H}}^d$ can be viewed as the submatrix of the partial parity-check matrix of $\hat{\mathcal{C}}$, \mathfrak{H} by considering it as the first $(v + 1)(n - k)$ rows and $(v + 1)n$ columns. In this context, considering that \mathfrak{H} is superregular, we can deduce that, within $\overline{\mathcal{H}}^d$, every non-trivial full-size minor is non-zero.

For \mathcal{H}^h , note that it is exactly the submatrix of \mathfrak{H} that considers the first $(v + 1)(n - k)$ rows and the columns with indices in the set $\{2vn + 1, \dots, (3v + 1)n + 1\}$, as well as the conditions over the columns of the nontrivial minors that imply the condition of Lemma 4. \square

While it is demonstrated in this section that the associated line 1D convolutional codes do not necessarily inherit the complete-MDP property of the 1D convolutional code $\hat{\mathcal{C}}$, there exist codes in which this phenomenon occurs. The next section presents a construction of a 2D convolutional code from a complete-MDP 1D convolutional code, whose associated line of 1D convolutional codes are also complete-MDPs. These types of codes are the ones that perform better with the decoding procedure that we are considering, which we will also see later in Section 5.

4. An Optimal Construction

In this section, we present the construction of an optimal 2D convolutional code in the sense that all its associated line codes hold the complete-MDP property. In this case, we will use a complete-MDP 1D convolutional code that was introduced in [17].

Theorem 8 ([17]). Let $n, k, \delta \in \mathbb{N}$ with $k < n$ and $(n - k) | \delta$, and let α be a primitive element of a finite field \mathbb{F}_{p^N} with $N > (L + 1)2^{(v+2)n-k-1}$. Then, $H(z) = \sum_{i=0}^v H_i z^i$ with

$$H_i = \begin{bmatrix} \alpha^{2^{in}} & \dots & \alpha^{2^{(i+1)n-1}} \\ \vdots & & \vdots \\ \alpha^{2^{(i+1)n-k-1}} & \dots & \alpha^{2^{(i+2)n-k-2}} \end{bmatrix}$$

for $i = 0, \dots, v = \frac{\delta}{n-k}$ is the parity-check matrix of an (n, k, δ) complete-MDP convolutional code.

In order to prove the optimality of our constructions, we will need the next result, which states that the matrices considered in Theorem 8 are superregular.

Proposition 2 ([16]). Let α be a primitive element of a finite field $\mathbb{F} = \mathbb{F}_{p^N}$, and let $B = [b_{i\ell}]$ be a matrix over \mathbb{F} with the following properties:

1. If $b_{i\ell} \neq 0$, then $b_{i\ell} = \alpha^{\beta_{i\ell}}$ for a positive integer $\beta_{i\ell}$;
2. If $b_{i\ell} = 0$, then $b_{i'\ell} = 0$ for any $i' > i$, or $b_{i\ell'} = 0$ for any $\ell' < \ell$;
3. If $\ell < \ell'$, $b_{i\ell} \neq 0$, and $b_{i\ell'} \neq 0$, then $2\beta_{i\ell} \leq \beta_{i\ell'}$;
4. If $i < i'$, $b_{i\ell} \neq 0$, and $b_{i'\ell} \neq 0$, then $2\beta_{i\ell} \leq \beta_{i'\ell}$.

Suppose N is greater than any exponent of α appearing as a non-trivial term of any minor of B , then B is superregular.

It is worth mentioning that if we have two matrices A and B that are equivalent (i.e., that we can obtain A from B by using row and column linear transformations), and if one of them holds the hypothesis of the theorem, then both of them are superregular.

Theorem 9. Let C be a 2D convolutional code, which was constructed as in (12) by using a $(n, k, 2\delta)$ 1D complete-MDP convolutional code \hat{C} , as expressed in Theorem 8, with the parity-check matrix $H(z) = \sum_{i=0}^{2v} H_i z^i$, where $v = \frac{\delta}{n-k}$. Then, the associated line codes, C^h , C^v , and C^d are (n, k, δ) 1D complete-MDP convolutional codes.

Proof. Recall that the associated line 1D convolutional codes C^h , C^v , and C^d have the parity-check matrices

$$\begin{aligned} H^h(z) &= \sum_{i=0}^v H_i z^i \\ &= H_0 + H_1 z + H_2 z^2 + \dots + H_v z^v. \\ H^v(z) &= \sum_{i=0}^v H_{2i} z^i \\ &= H_0 + H_2 z + H_4 z^2 + \dots + H_{2v} z^v. \\ H^d(z) &= \sum_{i=0}^v H_{v+i} z^i \\ &= H_v + H_{v+1} z + H_{v+2} z^2 + \dots + H_{2v} z^v. \end{aligned}$$

To see that \mathcal{C}^h , \mathcal{C}^v , and \mathcal{C}^d are 1D complete-MDP codes, we need to prove that the non-trivial zero full-size minors of the matrices

$$\begin{aligned} \mathfrak{H}^h &= \begin{bmatrix} H_v & H_{v-1} & \dots & H_0 & & & \\ & H_v & & H_1 & H_0 & & \\ & & \ddots & & \ddots & \ddots & \\ & & & H_v & \dots & H_1 & H_0 \end{bmatrix} \in \mathbb{F}^{(L'+1)(n-k) \times (v+L'+1)n}, \\ \mathfrak{H}^v &= \begin{bmatrix} H_{2v} & H_{2v-2} & \dots & H_0 & & & \\ & H_{2v} & & H_2 & H_0 & & \\ & & \ddots & & \ddots & \ddots & \\ & & & H_{2v} & \dots & H_2 & H_0 \end{bmatrix} \in \mathbb{F}^{(L'+1)(n-k) \times (v+L'+1)n}, \\ \mathfrak{H}^d &= \begin{bmatrix} H_{2v} & H_{v-1} & \dots & H_v & & & \\ & H_{2v} & & H_{v+1} & H_v & & \\ & & \ddots & & \ddots & \ddots & \\ & & & H_{2v} & \dots & H_{v+1} & H_v \end{bmatrix} \in \mathbb{F}^{(L'+1)(n-k) \times (v+L'+1)n}. \end{aligned} \tag{13}$$

are non-zero, where $L' = \frac{L}{2}$. To this end, we will see that they satisfy the conditions of Proposition 2, i.e., that they are superregular.

First, since $\hat{\mathcal{C}}$ is a complete-MDP code, we have

$$\mathfrak{H} = \begin{bmatrix} H_{2v} & H_{v-1} & \dots & H_0 & & & \\ & H_{2v} & & H_1 & H_0 & & \\ & & \ddots & & \ddots & \ddots & \\ & & & H_{2v} & \dots & H_1 & H_0 \end{bmatrix},$$

which has the property that all its full-size minors that are not trivially zero are non-zero. Note that we can see \mathfrak{H}^v as a submatrix of \mathfrak{H} when removing all the block-rows and block-columns with odd indices. Now, is easy to see that if all of the non-trivial zero full-size minors of \mathfrak{H}^v are not trivially minors of \mathfrak{H} , then they are non-zero.

Finally, through following the same argument as in the proof of Theorem 8 in [17], if we permute the columns of \mathfrak{H}^h and \mathfrak{H}^d in such a way that they have reverse order, then we have

$$\begin{bmatrix} & & & H_0 & \dots & H_{v-1} & H_v \\ & & \ddots & & \ddots & \ddots & \\ & H_0 & & H_{v-1} & H_v & & \\ H_0 & H_1 & \dots & H_v & & & \end{bmatrix} \text{ and } \begin{bmatrix} & & & & & H_v & \dots & H_{2v-1} & H_{2v} \\ & & \ddots & & & \ddots & \ddots & \ddots & \\ & H_v & & H_{2v-1} & H_{2v} & & & & \\ H_v & H_{v+1} & \dots & H_{2v} & & & & & \end{bmatrix},$$

which keeps the same terms for the non-trivial zero full-size minors. It is easy to see that these matrices fulfill the conditions of Proposition 2, thus making them super regular matrices. \square

Example 7. Consider the (3, 1, 4) 1D convolutional code \mathcal{C}' over \mathbb{F}_{p^N} , which was constructed as in Theorem 8, having the parity-check matrix $H(z) = H_0 + H_1z + H_2z^2$ with $H_0 = \begin{bmatrix} \alpha & \alpha^2 & \alpha^4 \\ \alpha^2 & \alpha^4 & \alpha^8 \end{bmatrix}$, $H_1 = \begin{bmatrix} \alpha^8 & \alpha^{16} & \alpha^{32} \\ \alpha^{16} & \alpha^{32} & \alpha^{64} \end{bmatrix}$, and $H_2 = \begin{bmatrix} \alpha^{64} & \alpha^{128} & \alpha^{256} \\ \alpha^{128} & \alpha^{256} & \alpha^{512} \end{bmatrix}$. Note that in order to obtain a complete-MDP code, $N > 7168$. The 2D convolutional code obtained by following the proposed construction was a (3, 1, 1) 2D convolutional code with the parity-check matrix

$$H(z_1, z_2) = H_0 + H_1z_1 + H_2z_2,$$

with

$$v_j^{(2)}(z_2) = \sum_{l+k=j} G_l^{(2)}(z_2) u_k^{(2)}(z_2).$$

Note that, for this description, the coefficients of $v_j^{(2)}(z_2)$ form the j -th horizontal line, and the analogously coefficient $v_j^{(1)}(z_1)$ determine the j -th vertical line.

Algorithm 2 Decoding 2D convolutional codes

Input: $\mathbf{v}(z_1, z_2)$, received codeword

Output: $\mathbf{v}(z_1, z_2)$, corrected codeword

```

1: fail = 0
2: med = 0
3: ori = 0
4:  $\mathcal{E} = \{v(a, b) \mid v(a, b) \in \mathbf{v}(z_1, z_2) \text{ and } v(a, b) = 0\}$ 
5: while fail < 3 do
6:   if med = 0 then {Decides orientation depending on med}
7:     ori = h
8:     ext = (0, L)
9:   else if med = 1 then
10:    ori = v
11:    ext = (L, 0)
12:   else if med = 2 then
13:    ori = d
14:    ext = (L, -L)
15:    $\mathcal{E}^{ori} = \text{Algorithm2}(\mathcal{E}, ori)$ 
16:    $e_{\min} = \min_{\prec} (a, b) \in \mathcal{E}^{ori}$ 
17:    $e_{\max} = \max_{\prec} (a, b) \in \mathcal{E}^{ori}$ 
18:    $[v_0, \dots, v_\ell] := [v(e_{\min}), \dots, v(e_{\max} + ext)]$ 
19:    $[v_0, \dots, v_\ell] := \text{Algorithm1}(\mathcal{C}_{j_0}, [v_0, \dots, v_\ell])$ 
20:    $\mathcal{R}^{ori} = \{(a, b) \mid v_i = v(a, b) \text{ and } v_i = 1\}$  {correct coordinates}
21:    $\mathcal{E} = \mathcal{E} \setminus \mathcal{R}^{ori}$ 
22:   if length( $\mathcal{R}^{ori}$ ) =  $\ell + 1$  then {All corrected}
23:     fail = 0
24:     med = 0
25:   else if length( $\mathcal{R}^{ori}$ ) >  $\ell + 1 - \text{length}(\mathcal{E}^{ori})$  then {Some corrected}
26:     fail = 0
27:     med = med + 1
28:   else if length( $\mathcal{R}^{ori}$ ) =  $\ell + 1 - \text{length}(\mathcal{E}^{ori})$  then {None corrected}
29:     fail = fail + 1
30:     med = med + 1

```

Algorithm 3 Choosing the set of coefficients \mathcal{E}^i to be corrected

Input: $\mathbf{v}(z_1, z_2)$, \mathcal{E} and an orientation h, v or d

Output: \mathcal{E}^i with $i \in \{h, v, d\}$.

```

1: if i=h then
2:    $c^h = \min_{(a,b) \in \mathcal{E}} \{b\}$  {We pick the index of the first row}
3:    $\mathcal{E}^h := \text{mergesort}(\{(a, b) \mid (a, b) \in \mathcal{E} \text{ and } b = c^h\})$ 
4: else if i=v then
5:    $c^v = \min_{(a,b) \in \mathcal{E}} \{a\}$  {We pick the index of the first row}
6:    $\mathcal{E}^v := \text{mergesort}(\{(a, b) \mid (a, b) \in \mathcal{E} \text{ and } a = c^v\})$ 
7: else if i=d then
8:    $c^d = \max_{(a,b) \in \mathcal{E}} \{a + b\}$  {We pick the index of the last diagonal}
9:    $\mathcal{E}^d := \text{mergesort}(\{(a, b) \mid (a, b) \in \mathcal{E} \text{ and } a + b = c^d\})$ 

```

There are two main differences with respect to the algorithm described in this paper, the first of which is the decoding performance in a given line. While Algorithm 1 tries forward, backward, and complete-MDP (if possible) decoding for a given line, the method from [10] performs partial corrections that may leave gaps of erasures, whereby when it later it tries to fill those gaps it is always in the forward direction. This last-mentioned process of “filling the gaps” was performed by correcting one point at a time; however, this may make this algorithm less efficient in terms of time.

The second difference was the requirement of a neighborhood. Whereas our algorithm needs a safe neighborhood to perform a correction, this new approach does not require it. This allows it to have more plasticity, that is, it can correct a line when the adjacent ones still have erasures. This feature makes this new approach more flexible against different erasure patterns.

Example 8. Let us consider Example 3 in [10]. In Figure 5, the pattern of erasures (*) considered in that example is shown. We will use this example to understand how different the behavior of the algorithms is. We assumed that we received a codeword (with erasures) $\hat{v}(z_1, z_2) = \sum_{0 \leq i, j \leq 6} \hat{v}_{ij} z_1^i z_2^j$, where * denotes an erasure and $v_{ij} = \begin{pmatrix} v_{ij,1} \\ v_{ij,2} \end{pmatrix}$ for $i, j = 1, \dots, 6$. By applying the algorithm in [10], the order of decoding was as follows: First—by horizontal decoding— v_{01}, v_{03} , and v_{04} were recovered. Then—by vertical decoding— $v_{20}, v_{30}, v_{11}, v_{31}, v_{12}$, and v_{32} were recovered. By horizontal decoding, we then obtained the rest of $v_{13}, v_{14}, v_{23}, v_{24}, v_{33}$, and $v_{36} \dots$

If our decoding method is employed, it can be observed that we also can take back the information but in the following different order:

1. We started with horizontal decoding and found v_{01}, v_{03} , and v_{04} .
2. Since we could not decode the second line due to an excessive accumulation of erasures, we changed to vertical decoding. Here, we could compute $v_{20}, v_{30}, v_{11}, v_{31}, v_{12}$, and v_{32} .
3. Again, we could not continue due to the amount of erasures; as such, we moved to diagonal decoding, and—through this approach—we recovered v_{46} .
4. Finally, we switched to horizontal decoding. Through this approach, we could decode the rest of the erasures one line at a time.

$i = 6$	$v_{60,1}$ $v_{60,2}$	$v_{61,1}$ $v_{61,2}$	$v_{62,1}$ $v_{62,2}$	*	*	$v_{65,1}$ $v_{65,2}$	$v_{66,1}$ $v_{66,2}$
$i = 5$	$v_{50,1}$ $v_{50,2}$	$v_{51,1}$ $v_{51,2}$	$v_{52,1}$ $v_{52,2}$	*	*	$v_{55,1}$ $v_{55,2}$	$v_{56,1}$ $v_{56,2}$
$i = 4$	$v_{40,1}$ $v_{40,2}$	$v_{41,1}$ $v_{41,2}$	$v_{42,1}$ $v_{42,2}$	*	*	$v_{45,1}$ $v_{45,2}$	$v_{46,1}$ $v_{46,2}$
$i = 3$	*	*	*	*	$v_{34,1}$ $v_{34,2}$	$v_{35,1}$ $v_{35,2}$	*
$i = 2$	*	$v_{21,1}$ $v_{21,2}$	$v_{22,1}$ $v_{22,2}$	*	*	$v_{25,1}$ $v_{25,2}$	$v_{26,1}$ $v_{26,2}$
$i = 1$	$v_{10,1}$ $v_{10,2}$	*	*	*	*	$v_{15,1}$ $v_{15,2}$	$v_{16,1}$ $v_{16,2}$
$i = 0$	$v_{00,1}$ $v_{00,2}$	*	$v_{02,1}$ $v_{02,2}$	*	$v_{04,1}$ $v_{04,2}$	$v_{05,1}$ $v_{05,2}$	$v_{06,1}$ $v_{06,2}$
$\hat{v}_{i,j}$	$j = 0$	$j = 1$	$j = 2$	$j = 3$	$j = 4$	$j = 5$	$j = 6$

Figure 5. Pattern of the erasures proposed in [10], Example 3.

In this particular example, in which neither the partial decoding of [10] nor the complete-MDP property of our algorithm were used, it is easy to see that both of them produced similar behavior. When the more complex pattern of erasures happens and these features are needed, then the performance for these two algorithms may differ. With this in mind, we conjectured that whenever a correction is possible from both routines, then the efficiency differs whether this “special feature” is needed for each of the cases. That is, if

one algorithm needs its own special property and the other does not, then the second one will be more efficient when establishing the time computation. To decide which of these algorithms works more efficiently and under which circumstances, a deeper and more complex analysis is required that is beyond the aim of this paper.

6. Conclusions and Further Research

Throughout this paper, we introduced a general construction of 2D convolutional codes by using 1D convolutional codes. By following this scheme, we proposed two constructions based on 1D complete-MDP convolutional codes from which one is optimal for the decoding procedure defined in [9]. We also described and complemented the decoding algorithm proposed in [9] by providing its pseudo-code, as well as by adding the possibility of performing the decoding by using the complete-MDP property.

Further research is possible in different directions. Through proposing a general construction of 2D convolutional codes based on 1D convolutional codes, we have a starting point to see if an n dimensional (n -D) convolutional code can be constructed by using $n - 1$ dimensional convolutional codes. In the same direction, we could address the problem of decoding such n -D codes by using lower dimensional decoding methods. Moreover, constructions of decoding algorithms for 2D convolutional codes over an erasure channel is still an interesting topic for future research since there currently exist only two of these algorithms in the literature.

Author Contributions: All authors contributed equally to this work. All authors have read and agreed to the published version of the manuscript.

Funding: The authors Raquel Pinto and Carlos Vela were supported by The Center for Research and Development in Mathematics and Applications (CIDMA) through the Portuguese Foundation for Science and Technology (FCT—Fundação para a Ciência e a Tecnologia) (references UIDB/04106/2020 and UIDP/04106/2020): <https://doi.org/10.54499/UIDB/04106/2020> and <https://doi.org/10.54499/UIDP/04106/2020>.

Data Availability Statement: No data required.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Fornasini, E.; Valcher, M.E. Algebraic aspects of two-dimensional convolutional codes. *IEEE Trans. Inf. Theory* **1994**, *40*, 1068–1082. [[CrossRef](#)]
2. Valcher, M.E.; Fornasini, E. On 2D finite support convolutional codes: An algebraic approach. *Multidim. Syst. Signal Proc.* **1994**, *5*, 231–243. [[CrossRef](#)]
3. Climent, J.-J.; Napp, D.; Perea, C.; Pinto, R. Maximum Distance Separable 2D Convolutional Codes. *IEEE Trans. Inf. Theory* **2016**, *62*, 669–680. [[CrossRef](#)]
4. Gluesing-Luerssen, H.; Rosenthal, J.; Weiner, P. Duality between multidimensional convolutional codes and systems. In *Advances in Mathematical Systems Theory*; Colonius, F., Helmke, U., Wirth, F., Praetzel-Wolters, D., Eds.; Birkhäuser: Boston, MA, USA, 2000; pp. 135–150.
5. Weiner, P.A. Multidimensional Convolutional Codes. Ph.D. Thesis, University of Notre Dame, Notre Dame, IN, USA, 1998.
6. Charoenlarnopparut, C. Applications of Grobner bases to the structural description and realization of multidimensional convolutional code. *Sci. Asia* **2009**, *35*, 95–105. [[CrossRef](#)]
7. Charoenlarnopparut, C.; Tantaratana, S. Algebraic approach to reduce the number of delay elements in the realization of multidimensional convolutional code. In Proceedings of the 47th IEEE International Midwest Symposium Circuits and Systems (MWSCAS 2004), Hiroshima, Japan, 25–28 July 2004; pp. 529–532.
8. Kitchens, B. Multidimensional convolutional codes. *SIAM J. Discret. Math.* **2002**, *15*, 367–381. [[CrossRef](#)]
9. Climent, J.-J.; Napp, D.; Pinto, R.; Simões, R. Decoding of 2D convolutional codes over an erasure channel. *Adv. Math. Commun.* **2016**, *10*, 179–193. [[CrossRef](#)]
10. Lieb, J.; Pinto, R. A decoding algorithm for 2D convolutional codes over the erasure channel. *Adv. Math. Commun.* **2023**, *17*, 935–959. [[CrossRef](#)]
11. York, E.V. Algebraic Description and Construction of Error Correcting Codes: A Linear Systems Point of View. Ph.D. Thesis, University of Notre Dame, Notre Dame, IN, USA, 1997.
12. Lieb, J.; Pinto, R.; Rosenthal, J. Convolutional Codes. In *A Concise Encyclopedia of Coding Theory*; Huffman, W.C., Kim, J.-L., Solé, P., Eds.; CRC Press: Boca Raton, FL, USA, 2021; pp. 197–225.

13. Tomas, V.; Rosenthal, R.; Smarandache, R. Decoding of Convolutional Codes over the Erasure Channel. *IEEE Trans. Inf. Theory* **2012**, *58*, 90–108. [[CrossRef](#)]
14. Rosenthal, J.; Smarandache, R. Maximum distance separable convolutional codes. *Appl. Algebra Eng. Commun. Comput.* **1999**, *10*, 15–32. [[CrossRef](#)]
15. Gluesing-Luerssen, H.; Rosenthal, J.; Smarandache, R. Strongly-MDS convolutional codes. *IEEE Trans. Inf. Theory* **2006**, *52*, 584–598. [[CrossRef](#)]
16. Almeida, P.J.; Napp, D.; Pinto, R. Superregular matrices and applications to convolutional codes. *Linear Algebra Its Appl.* **2016**, *499*, 1–25. [[CrossRef](#)]
17. Lieb, J. Complete MDP convolutional codes. *J. Algebra Its Appl.* **2019**, *18*, 1950105. [[CrossRef](#)]
18. Alfarano, G.N.; Napp, D.; Neri, A.; Requena, V. Weighted Reed-Solomon codes. *Linear Multilinear Algebra* **2023**, *72*, 841–874. [[CrossRef](#)]
19. Chen, Z. Convolutional Codes with a Maximum Distance Profile Based on Skew Polynomials. *IEEE Trans. Inf. Theory* **2023**, *68*, 5178–5184. [[CrossRef](#)]
20. Luo, G.; Cao, X.; Ezerman, M.F.; Ling, S. A Construction of Maximum Distance Profile Convolutional Codes with Small Alphabet Sizes. *IEEE Trans. Inf. Theory* **2023**, *69*, 2983–2990. [[CrossRef](#)]
21. Napp, D.; Smarandache, R. Constructing strongly-MDS convolutional codes with maximum distance profile. *Adv. Math. Commun.* **2016**, *10*, 275–290. [[CrossRef](#)]
22. Lieb, J. Necessary field size and probability for MDP and complete MDP convolutional codes. *Des. Codes Cryptogr.* **2019**, *87*, 3019–3043. [[CrossRef](#)]
23. Chen, Z. A lower bound on the field size of convolutional codes with a maximum distance profile and an improved construction. *IEEE Trans. Inf. Theory* **2023**. [[CrossRef](#)]
24. Alfarano, G.N.; Gruica, A.; Lieb, J.; Rosenthal, R. Convolutional codes over finite chain rings, MDP codes and their characterization. *Adv. Math. Commun.* **2022**, *17*, 1–22. [[CrossRef](#)]
25. Tomás, V.; Rosenthal, J.; Smarandache, R. Reverse-maximum distance profile convolutional codes over the erasure channel. In Proceedings of the 19th International Symposium on Mathematical Theory of Networks and Systems, MTNS 2010, Budapest, Hungary, 5–9 July 2010; pp. 1212–2127.
26. Hutchinson, R. The existence of strongly MDS convolutional codes. *SIAM J. Control Optim.* **2008**, *47*, 2812–2826. [[CrossRef](#)]
27. Massey, J.L. Reversible codes. *Inf. Control* **1964**, *77*, 369–380. [[CrossRef](#)]
28. Almeida, P.J.; Lieb, J. Complete j-MDP Convolutional Codes. *IEEE Trans. Inf. Theory* **2020**, *66*, 7348–7359. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.