New convolutional codes from old convolutional codes

Joan-Josep Climent  
Departament de Ciència de la Computació i Intel·ligència Artificial,  
Universitat d’Alacant,  

Victoria Herranz  
Carmen Perea  
Centro de Investigación Operativa,  
Departamento de Estadística y Matemática Aplicada,  
Universidad Miguel Hernández,  
Avenida del Ferrocarril, s/n. E-03202 Elche, Spain.

Abstract

In this paper, starting with a family of convolutional codes, we construct a new convolutional code and we introduce also necessary and sufficient conditions in order that the new convolutional code is MDS.

Key words: MDS convolutional code, free distance, input state output representation, compact code, puncturing convolutional codes, minimal-basic generator matrix.

1 Introduction

There exists only relatively few algebraic constructions of convolutional codes having some good designed distance. In this paper, we construct new convolutional codes from a family of rate $k/n$ convolutional codes.
Let $\mathbb{F}[D]$ be the polynomial ring and let $\mathbb{F}(D)$ be the field of rational functions over the field $\mathbb{F}$. If $G(D)$ is a $k \times n$ matrix over the polynomial ring $\mathbb{F}[D]$, with rank $k$, then the rate $k/n$ convolutional code generated by $G(D)$ is the set

$$
C = \{u(D)G(D) \in \mathbb{F}^n(D) : u(D) \in \mathbb{F}^k(D)\}.
$$

We say that $G(D)$ is a generator matrix or an encoder of the convolutional code $C$.

In the following we will adopt the notation used by McEliece [7] and we call a convolutional code of rate $k/n$ and degree $\delta$ as $(n, k, \delta)$-code, where $\delta$ is the degree of $C$ defined as the number $\delta = \sum_{i=1}^{k} \nu_i$, where $\nu_i$ denotes the $i$th row degree of a minimal-basic generator matrix

$$
G(D) = \sum_{j=0}^{\nu} G_j D^j \in \mathbb{F}[D]^{k \times n}, \quad G_j \in \mathbb{F}^{k \times n}, \quad G_\nu \neq 0
$$

that is, $G(D)$ has a polynomial right inverse and $\sum_{i=1}^{k} \nu_i$ attains the minimal value among all generator matrices of $C$.

**Theorem 1 (Theorem 6 of [5])** Let $G(D)$ be a $k \times n$ basic encoding matrix with degree $\delta$. Then, the following statements are equivalent.

1. $G(D)$ is a minimal-basic encoding matrix.

2. The matrix $[G(D)]_h$ has full rank, where $[G(D)]_h$ is a $(0, 1)$-matrix with 1 in the position $(i, j)$ if $\operatorname{deg}(g_{ij}(D)) = \nu_i$ and 0 otherwise.

On the other hand, Rosenthal and Smarandache [10] establish that the free distance of a $(n, k, \delta)$-convolutional code is always upper-bounded by the generalized Singleton bound

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

and they generalize the concept of MDS block codes.

**Definition 1 (Definition 2.5 of [10])** An $(n, k, \delta)$-convolutional code is called MDS if its free distance is maximal among all rate $k/n$ convolutional codes.
of degree $\delta$, i.e. an $(n, k, \delta)$-convolutional code is MDS if its free distance achieves the generalized Singleton bound

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

The next result, that was derived by Justesen [6], implies that there exists a rate $1/n$ MDS convolutional code for every value of $\delta$.

**Theorem 2 (Theorem 2.9 of [10])** Let $\delta$ and $n$ be fixed and assume that $\mathbb{F}$ is a finite field with $|\mathbb{F}| > 3\delta$. Then there exist a rate $1/n$ MDS convolutional code.

We will use the concept of compact code, done in [7].

**Definition 2** A $(n, k, \delta)$-convolutional code $C$ is a compact code if its Forney indices assume only the two values $\left\lfloor \frac{\delta}{k} \right\rfloor$ and $\left\lfloor \frac{\delta}{k} \right\rfloor + 1$. In fact, in a compact code, there are exactly $\delta \mod k$ Forney indices equal to $\left\lfloor \frac{\delta}{k} \right\rfloor + 1$ and $k - \delta \mod k$ Forney indices equal to $\left\lfloor \frac{\delta}{k} \right\rfloor$.

The convolutional codes can be represented with the help of a classical systems theory approach. In the coding literature (see [8] and [11]), convolutional codes are usually represented by the *input state output representation*

$$x_{t+1} = Ax_t + Bu_t,$$
$$y_t = Cx_t + Du_t, \quad x_0 = 0 \quad (2)$$

where for each time $t$, $x_t \in \mathbb{F}^\delta$ is the state vector, $u_t \in \mathbb{F}^k$ is the information vector and $y_t \in \mathbb{F}^{n-k}$ is the parity vector. Next theorem shows the relation between representations (1) and (2).

**Theorem 3 (Theorem 6.2 of [8])** Assume that the generator matrix $G(z)$ of a convolutional code has the property that rank $G(0) = k$. Then $G(z^{-1})$ is a proper transfer function, and there exist matrices $(A, B, C, D)$ of sizes $\delta \times \delta$, $\delta \times k, (n - k) \times \delta$ and $(n - k) \times k$ such that

$$G(z^{-1}) = C(zI - A)^{-1}B + D.$$  

The dynamics of (1) are then equivalently described by equations (2).
For applications requiring high data transmission rates powerful high-rate codes are required. Among them, convolutional codes present over block codes the advantage of a simpler soft decoding, and thus yield an inherently higher coding gain. On the other hand, the decoding complexity of such high-rate convolutional codes using a sequence maximum-likelihood decoding algorithm (such as Viterbi algorithm) increases linearly with the number of edges in each trellis section, and this, in turn, increases exponentially with the code rate. The solution, so far, has been to puncture a low-rate mother convolutional code.

A punctured code is obtained by periodically deleting (or puncturing) encoded digits from ordinary encoded sequences (see [1, 2, 3]). Puncturing has the effect of reducing the number of encoded digits corresponding to the information digits, i.e., of increasing the code rate. In general, puncturing can be described as follows. We begin with a “parent” \((n, k, \delta)\) convolutional code, and block it to depth \(M\), i.e., group the input bit stream into blocks of \(M\) bits each. The result is an \((nM, kM, \delta)\) convolutional code with the same free distance as the parent code. Now, let \(P\) be a “puncturing” \((0, 1)\)-matrix of size \(n \times M\), so that the number of ones is \(N\) and the number of zeros is \(nM - N\). If we now delete, or “puncture”, all but \(N\) bits from each \(nM\)-bit output block, using the matrix \(P\) as template, the results is an \((N, kM, \delta)\) convolutional code.

Now, we return of the simple blocking idea, which, as we have seen, is a method of taking a given \((n, k)\) convolutional code \(C\) and block it to depth \(M\), producing then a \((nM, kM)\) convolutional code. We will give results over the relation between this code and the new code defined by us.

**Definition 3 (Definition 8.1 of [7])** If \(f(D) = a_0 + a_1 D + a_2 D^2 + \cdots\) is a power series (or just a polynomial) in the indeterminate \(D\), then for any integer \(M \geq 1\), the \(M\)-th polyphase descomposition of \(f(D)\) is the following list of \(M\) power series:

\[
(f_{0,M}(D), f_{1,M}(D), \ldots, f_{M-1,M}(D)) = \\
\left(\sum_{k \geq 0} a_{kM} D^k, \sum_{k \geq 0} a_{kM+1} D^k, \ldots, \sum_{k \geq 0} a_{kM+(M-1)} D^k\right)
\]  

(3)

The \(j\)-th component of the list in (3) is called the \((j, M)\)-th polyphase component of \(f(D)\).
The components of the \( M \)-th polyphase decomposition can be used to form an important matrix, called the \( M \)-th polycyclic pseudocirculant (of PCPC for short) associated with \( f(D) \).

**Definition 4 (Definition 8.2 of [7])** If \( f(D) = a_0 + a_1D + a_2D^2 + \cdots \) is a power series (or just a polynomial) in the indeterminate \( D \), then for any integer \( M \geq 1 \), if \((f_0, f_1, \ldots, f_{M-1})\) is the \( M \)-th polyphase decomposition of \( f(D) \), the \( M \)-th polycyclic pseudocirculant associated with \( f(D) \) is the \( M \times M \) polynomial matrix \( f^{[M]}(D) \) defined as follows:

\[
f^{[M]}(D) = \begin{pmatrix}
    f_0 & f_1 & \cdots & f_{M-1} \\
    Df_{M-1} & f_0 & \cdots & f_{M-2} \\
    Df_{M-2} & Df_{M-1} & \cdots & f_{M-3} \\
    \vdots & \vdots & \ddots & \vdots \\
    Df_1 & Df_2 & \cdots & f_0
\end{pmatrix}
\]

Next theorem gives an algebraic description of the generator matrix \( G^{[M]}(D) \) for the \((nM, kM)\)-code that results blocking the \((n, k)\)-convolutional code (with generator matrix \( G(D) \)) to depth \( M \).

**Theorem 4 (Theorem 8.3 of [7])** If \( C \) is an \((n, k)\)-convolutional code, then the \( M \)-th blocking of \( C \), denoted by \( C^{[M]} \), is an \((nM, kM)\)-convolutional code. If \( G(D) = (g_{i,j}(D)) \) is a \( k \times n \) polynomial generator matrix for the original code \( C \), then a generator matrix for \( C^{[M]} \), say \( G^{[M]}(D) \), can be obtained from \( G(D) \) by replacing each entry \( g_{i,j}(D) \) from \( G(D) \) with the corresponding \( M \)-th PCPC, and then interleaving the columns to depth \( M \).

One consequence of the above theorem is the next corollary.

**Corollary 1 (Corollary 8.4 of [7])** If the row degrees of the generator matrix \( G(D) \) are \((e_1, \ldots, e_k)\), then the row degrees of \( G^{[M]}(D) \) are

\[
    ([e_i/M], [e_i + 1/M], \ldots, [(e_i + M - 1)/M]) \quad \text{for } i = 1, \ldots, k.
\]

In other words, a single row degree of \( e \) in \( G(D) \) is replaced with \( M \) row degrees in \( G^{[M]}(D) \). If \( e = e'M + r \), with \( 0 \leq r < M \), then \( M - r \) of these degrees will be equal to \( e' \) and \( r \) will be equal to \( e' + 1 \). Note that, in fact, \( e' = \lfloor e/M \rfloor \). The sum of the row degrees \( G^{[M]}(D) \), i.e., the external degree of \( G^{[M]}(D) \), is thus the same as for \( G(D) \).
Another important result is that if the $k \times n$ polynomial generator matrix $G(D)$ is minimal-basic, then the general generator matrix $G^{[M]}(D)$ of the $M$-th blocking of $G(D)$ is also minimal-basic.

**Theorem 5 (Theorem 8.5 of [7])** If $G(D)$ is a minimal-basic generator matrix for $C$, then $G^{[M]}(D)$ will be a minimal-basic generator matrix for $C^{[M]}$. Thus the degree of the blocked code $C^{[M]}$ is the same as that for the original code.

Not only the degree is the same for the codes $C$ and $C^{[M]}$. It is easy to see the fact that $d_{free}$ is the same for all of the blocked codes of the parent code $C$, since the set of codewords is the same for all values of $M$.

Using these facts, together with Theorem 4 and Corollary 1, we can obtain the following result.

**Corollary 2 (Corollary 8.6 of [7])** The $M$-th blocking of an $(n, k, \delta, d)$ convolutional code is an $(nM, kM, \delta, d)$-convolutional code. If the Forney indices of the original code are $(e_1, \ldots, e_k)$, then the Forney indices of $C^{[M]}$ are

$$(\lfloor e_i/M \rfloor, \lceil (e_i + 1)/M \rceil, \ldots, \lceil (e_i + M - 1)/M \rceil) \quad \text{for} \quad i = 1, \ldots, k.$$ 

## 2 Main Results

For $i = 1, 2, \ldots, s$ we consider an $(n, k, \delta_i)$-convolutional code $C^{(i)}$ of memory $\nu_i$ generated by the minimal-basic generator matrix

$$G^{(i)}(D) = \sum_{j=0}^{\nu_i} G^{(i)}_j D^j.$$ 

Now, we define a new convolutional code $\tilde{C}$ of rate $sk/sn$ from the above convolutional codes as the convolutional code whose generator matrix is

$$\tilde{G}(D) = \begin{pmatrix}
G^{(1)}(D) \\
G^{(2)}(D) \\
\vdots \\
G^{(s)}(D)
\end{pmatrix}$$

(4)
that is,
\[
\tilde{C} = \{ u(D)\tilde{G}(D) \in \mathbb{F}^{sn}(D) : u(D) \in \mathbb{F}^{sk}(D) \}.
\] (5)

From (4) we deduce that
\[
d_{\text{free}}(\tilde{C}) = \min_{1 \leq i \leq s} d_{\text{free}}(C^{(i)}).
\]

Letting \( s = M \), the difference between the new convolutional code and
the “blocked” code \( C^{[M]} \) is that while here we get \( M \) encoders (and they
can be different or just the same encoder; we will take the same encoder,
\( M \) times, in order to compare with the “blocked” code), the time in \( C^{[M]} \) is
divided by \( M \). That is, for every tick of a clock, the encoder accepts \( kM \)
input bits, and in response produces \( nM \) output bits.

Next lemma shows that the generator matrix \( \tilde{G}(D) \) inherits the properties
of the generator matrices \( G^{(i)}(D) \) for \( i = 1, 2, \ldots, s \). Recall that from
Theorem 5, the matrix \( G^{[M]}(D) \) verifies also this property.

**Lemma 1** For \( i = 1, 2, \ldots, s \), let \( G^{(i)}(D) \) be a minimal-basic generator
matrix of the convolutional code \( C^{(i)} \). Then, the generator matrix (4) of \( \tilde{C} \) is also
a minimal-basic generator matrix and the degree of the convolutional code \( \tilde{C} \)
is \( \tilde{\delta} = \sum_{i=1}^{s} \delta_i \).

**Proof.** If \( G^{(i)}(D) \) for \( i = 1, \ldots, s \) is basic then it has a polynomial right
inverse \( (G^{(i)})^{-}(D) \). Then, it is easy to see that the polynomial matrix
\[
\tilde{H}(D) = \begin{pmatrix}
(G^{(1)})^{-}(D) \\
(G^{(2)})^{-}(D) \\
\vdots \\
(G^{(s)})^{-}(D)
\end{pmatrix}
\]
is a right inverse of \( \tilde{G}(D) \), so \( \tilde{G}(D) \) is a basic generator matrix. Furthermore,
from Theorem 1 the matrix \( [G^{(i)}(D)]_h \) is of full rank, for each \( i = 1, \ldots, s \)
so the matrix \( [	ilde{G}(D)]_h \) is also of full rank, and then, again from Theorem 1,
\( \tilde{G}(D) \) is a minimal-basic generator matrix of \( \tilde{C} \). So we obtain that the degree
of \( \tilde{C} \) is \( \tilde{\delta} \). \( \Box \)

In the next lemma we provide a first-order representation associated with
the convolutional code which we have defined, based on first-order representa-
tions of the codes \( C^{(i)} \).
Lemma 2 For \( i = 1, 2, \ldots, s, \) suppose that the convolutional code \( C(i) \) is represented by the input state output representation
\[
\begin{align*}
x_{t+1} &= A(i)x_t + B(i)u_t, \\
y_t &= C(i)x_t + D(i)u_t, \quad x_0 = 0.
\end{align*}
\]
If
\[
\tilde{A} = \begin{pmatrix} A^{(1)} & \cdots & A^{(s)} \end{pmatrix}, \quad \tilde{B} = \begin{pmatrix} B^{(1)} & \cdots & B^{(s)} \end{pmatrix},
\]
\[
\tilde{C} = \begin{pmatrix} C^{(1)} & \cdots & C^{(s)} \end{pmatrix}, \quad \tilde{D} = \begin{pmatrix} D^{(1)} & \cdots & D^{(s)} \end{pmatrix},
\]
then the realization
\[
\begin{align*}
X_{t+1} &= \tilde{A}X_t + \tilde{B}U_t, \\
Y_t &= \tilde{C}X_t + \tilde{D}U_t, \quad X_0 = 0
\end{align*}
\]
is an input state output representation of the new convolutional code (5).

The encoding equations for the convolutional code \( \tilde{C} \) can be written as
\[
v(D) = u(D)\tilde{G}(D)
\]
where
\[
u(D) = (u_1(D), u_2(D), \ldots, u_k(D), u_{k+1}(D), \ldots, u_{sk}(D))
\]
is the sequence of information digits and
\[
v(D) = (v_1(D), v_2(D), \ldots, v_n(D), v_{n+1}(D), \ldots, v_{sn}(D))
\]
is the sequence of transmitted digits or codeword. Let us denote, for \( i = 1, 2, \ldots, s, \) the \( i \)th block of \( k \) components of \( u(D) \) by
\[
u^{(i)}(D) = (u_{(i-1)k+1}(D), \ldots, u_{ik}(D))
\]
as well as the \( i \)th block of \( n \) components of \( v(D) \) by
\[
v^{(i)}(D) = (v_{(i-1)n+1}(D), \ldots, v_{in}(D)).
\]
Then, from equation (5), we obtain the encoding equations for each convolutional code \( C^{(i)} \),
\[
v^{(i)}(D) = u^{(i)}(D)G^{(i)}(D), \text{ for } i = 1, 2, \ldots, s
\]
Theorem 6 Let $\tilde{C}$ be the convolutional code of rate $sk/sn$ generated by the generator matrix (4). Assume that for $i = 1, \ldots, s$, the matrix $G^{(i)}(D)$ generates a $(n, k, \delta_i)$-compact convolutional code. Then the $(sk, sn, \tilde{\delta})$-convolutional code $\tilde{C}$ is compact if and only if

$$\left\lfloor \frac{\delta_i}{k} \right\rfloor = \left\lfloor \frac{\delta_j}{k} \right\rfloor \quad \text{for } i \neq j. \quad (6)$$

Proof. If $\tilde{C}$ is a compact convolutional code, then it has at most two different Forney indices, $\left\lfloor \frac{\tilde{\delta}}{sk} \right\rfloor$ and $\left\lfloor \frac{\tilde{\delta}}{sk} \right\rfloor + 1$. Now, taking into account the form of $\tilde{G}(D)$ we obtain that

$$\left\lfloor \frac{\tilde{\delta}}{sk} \right\rfloor = \left\lfloor \frac{\delta_i}{k} \right\rfloor \quad \text{for } i = 1, 2, \ldots, s$$

and therefore (6) holds.

Conversely, assume that (6) holds, and let $\left\lfloor \frac{\delta_i}{k} \right\rfloor = \lambda$ for $i = 1, 2, \ldots, s$.

Since $C^{(i)}$ is a compact convolutional code for $i = 1, 2, \ldots, s$, we have that $C^{(i)}$ has

- $\delta_i - k\lambda$ Forney indices equals to $\lambda + 1$
- $k - (\delta_i - k\lambda)$ Forney indices equals to $\lambda$.

Observe that if $k$ divides $\delta_i$, then $C^{(i)}$ has only $k$ Forney indices equals to $\lambda$.

Now, taking into account the construction of $\tilde{C}$ it is easy to see that $\left\lfloor \frac{\tilde{\delta}}{sk} \right\rfloor = \lambda$. Furthermore, $\tilde{C}$ has

- $\tilde{\delta} - sk\lambda$ Forney indices equals to $\lambda + 1$
- $sk - (\tilde{\delta} - sk\lambda)$ Forney indices equals to $\lambda$.

Consequently, $\tilde{C}$ is a compact convolutional code. \qed

As a direct consequence of Theorem 2 we have the existence of compact convolutional codes based on old ones without increasing the number of elements of the field.
Corollary 3 Let $\delta, n$ and $s$ be fixed and assume that $\mathbb{F}$ is a finite field with $|\mathbb{F}| > 3\delta$. Then there exists a $(sn, s, s\delta)$-compact convolutional code.

Proof. By Theorem 2 we know that there exist a rate $1/n$ MDS convolutional code $\tilde{C}$, and then it is a compact convolutional code. So it is sufficient to take $C^{(i)} = \tilde{C}$ for $i = 1, \ldots, s$ in Theorem 6. \qed

We get an analogous result of Theorem 6 for the code $C^{[M]}$.

Theorem 7 Let $C$ an $(n, k, \delta)$-convolutional code and $C^{[M]}$ the $(nM, kM, \delta)$-convolutional code obtained from blocking $C$ to depth $M$. If $C$ is a compact code, then $C^{[M]}$ is compact if and only if

$$\left\lfloor \frac{\delta}{kM} \right\rfloor = \left\lfloor \frac{\left\lfloor \frac{\delta}{M} \right\rfloor}{M} \right\rfloor$$

(7)

Proof. Assume that $C$ is a compact convolutional code, then $C$ has

- $\delta - k \left\lfloor \frac{\delta}{k} \right\rfloor$ Forney indices equals to $\left\lfloor \frac{\delta}{k} \right\rfloor + 1$
- $k - (\delta - k \left\lfloor \frac{\delta}{k} \right\rfloor)$ Forney indices equals to $\left\lfloor \frac{\delta}{k} \right\rfloor$

Assume that

$$\left\lfloor \frac{\delta}{k} \right\rfloor = \left\lfloor \frac{\left\lfloor \frac{\delta}{M} \right\rfloor}{M} \right\rfloor M + r_1 \quad \text{with} \quad 0 \leq r_1 < M$$

$$\left\lfloor \frac{\delta}{k} \right\rfloor + 1 = \left\lfloor \frac{\left\lfloor \frac{\delta}{M} \right\rfloor}{M} + 1 \right\rfloor M + r_2 \quad \text{with} \quad 0 \leq r_2 < M$$

Observe that if $k$ divides $\delta$, then $C$ has only $k$ Forney indices equals to $\frac{\delta}{k}$ and then we do not consider the above expression for $\left\lfloor \frac{\delta}{k} \right\rfloor + 1$.

By Corollary 1, we know that each Forney index $\left\lfloor \frac{\delta}{k} \right\rfloor$ produces in $C^{[M]}$

- $r_1$ Forney indices equals to $\left\lfloor \frac{\left\lfloor \frac{\delta}{M} \right\rfloor}{M} \right\rfloor + 1$
- $M - r_1$ Forney indices equals to $\left\lfloor \frac{\left\lfloor \frac{\delta}{M} \right\rfloor}{M} \right\rfloor$
Similarly, each Forney index \( \left\lfloor \frac{\delta}{kM} \right\rfloor + 1 \) produces in \( C^{[M]} \):

- \( r_2 \) Forney indices equals to \( \left\lfloor \frac{\delta}{M} + 1 \right\rfloor \)
- \( M - r_2 \) Forney indices equals to \( \left\lfloor \frac{\delta}{M} + 1 \right\rfloor \)

Now, taking into account that

\[
\left\lfloor \left\lfloor \frac{\delta}{kM} \right\rfloor \right\rfloor M = \left\lfloor \frac{\delta}{M} + 1 \right\rfloor \]

and the above considerations, we obtain that \( C^{[M]} \) has

- \( \delta - \left\lfloor \frac{\delta}{kM} \right\rfloor kM \) Forney indices equals to \( \left\lfloor \frac{\delta}{kM} \right\rfloor + 1 \)
- \( kM - \left( \delta - \left\lfloor \frac{\delta}{M} \right\rfloor kM \right) \) Forney indices equals to \( \left\lfloor \frac{\delta}{M} \right\rfloor \)

Finally, since \( C^{[M]} \) is a compact convolutional code if and only if \( C^{[M]} \) has

- \( \delta - \left\lfloor \frac{\delta}{kM} \right\rfloor kM \) Forney indices equals to \( \left\lfloor \frac{\delta}{kM} \right\rfloor + 1 \)
- \( kM - (\delta - \left\lfloor \frac{\delta}{kM} \right\rfloor kM) \) Forney indices equals to \( \left\lfloor \frac{\delta}{kM} \right\rfloor \)

we conclude that \( C^{[M]} \) is a compact convolutional code if and only if equality (7) holds.

The construction of \( G^{[M]}(D) \), as we mention in Section 1, is used for constructing new convolutional codes from old ones, via puncturing. The standard puncturing process applied to the matrix \( G^{[M]}(D) \) can be applied to the generator matrix \( \tilde{G}(D) \). We summarize this remark in a simple example.

**Example 1** Consider the \((3, 1, 3)\) codes with generator matrices

\[
G^1(D) = (1 + D, 1 + D + D^2, 1 + D + D^3)
\]

and

\[
G^2(D) = (D, 1 + D^2 + D^3, 1 + D^3),
\]
respectively. Then
\[
\tilde{G}(D) = \begin{pmatrix}
1 + D & 1 + D + D^2 & 1 + D + D^3 & 0 & 0 & 0 \\
0 & 0 & 0 & D & 1 + D^2 + D^3 & 1 + D^3
\end{pmatrix}
\]

Now, puncturing according to the pattern \( P = \begin{pmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 0 \end{pmatrix} \) we obtain the \( P \)-puncturing
\[
\tilde{G}_P(D) = \begin{pmatrix}
1 + D & 1 + D + D^2 & 1 + D + D^3 & 0 & 0 \\
0 & 0 & 0 & D & 1 + D^2 + D^3
\end{pmatrix}
\]
that correspond to an \((5, 2, 6)\) convolutional code.

In general, the \( P \)-puncturing of \( \tilde{C} \), as a consequence of the construction of matrix \( \tilde{G}(D) \), is an \((N, k\bar{M}, \bar{\nu}, \bar{d})\)-code, where \( N \) is the Hamming weight of \( P \), \( \bar{\nu} \leq \max_{1 \leq i \leq \bar{M}} \{ \nu_i \} \) is the memory and \( \bar{d} \leq \min_{1 \leq i \leq \bar{M}} \{ d_i \} \) is the free distance. Unfortunately, like in the standard puncturing process, little more about \( \bar{\nu} \) and \( \bar{d} \) can be said, in general. Although, the determination of \( \bar{\nu} \) is usually quite easy; if the generator matrices \( G^i(D) \) for \( i = 1, 2, \ldots, m \) are minimal-basic, by Lemma 1, the generator matrix \( \tilde{G}(D) \) is also minimal-basic, and then \( \tilde{G}_P(D) \) is likely to be minimal-basic, in which case \( \bar{\nu} = \max_{1 \leq i \leq \bar{M}} \{ \nu_i \} \), or very close to minimal-basic, in which case a few row operation on \( \tilde{G}_P(D) \) will produce a minimal-basic generator matrix.

On the other hand, the construction of \( \tilde{C} \) allow us consider for \( i = 1, 2, \ldots, M \) an \((n_i, k, \delta_i)\)-convolutional code \( C^{(i)} \) with memory \( \nu_i \) instead of an \((n, k, \delta)\)-convolutional code \( C^{(\iota)} \). In this case the construction of matrix \( \tilde{G}(D) \), is an \( \sum_{i=1}^{M} n_i, kM, \nu, \bar{d} \) code with the same properties of \( \tilde{G}_P(D) \). We present now a simple example.

**Example 2** Consider the \((3, 1, 3)\) code with generator matrix \( G^1(D) \) of Example 1 and \( G^2(D) = (D, 1 + D^2 + D^3) \), then the matrix \( \tilde{G}(D) \) is the matrix \( \tilde{G}_P(D) \) of Example 1.
References


