EURASIP Journal on Advances in Signal Processing
a SpringerOpen Journal

RESEARCH Open Access

High-rate systematic recursive convolutional encoders: minimal trellis and code search

Isaac Benchimol[1], Cecilio Pimentel[2*], Richard Demo Souza[3] and Bartolomeu F Uchôa-Filho[4]

Abstract

We consider high-rate systematic recursive convolutional encoders to be adopted as constituent encoders in turbo schemes. Douillard and Berrou showed that, despite its complexity, the construction of high-rate turbo codes by means of high-rate constituent encoders is advantageous over the construction based on puncturing rate-1/2 constituent encoders. To reduce the decoding complexity of high-rate codes, we introduce the construction of the minimal trellis for a systematic recursive convolutional encoding matrix. A code search is conducted and examples are provided which indicate that a more finely grained decoding complexity-error performance trade-off is obtained.

1 Introduction

The typical turbo code configuration is the parallel concatenation of two systematic recursive constituent convolutional encoders of rate 1/2 connected via an interleaver, resulting in a code of rate 1/3. However, higher rate turbo codes may be useful in modern wireless, magnetic recording and fiber optics applications [1]. The usual approach to increase the overall rate is to puncture selected bits of the turbo codeword [2]. An alternative is to use high-rate constituent encoders [1,3,4] what, according to [3], offers several advantages, such as better convergence of the iterative process, higher throughput, reduced latency, and robustness of the decoder. If puncturing is still needed to achieve a required rate, fewer bits have to be discarded when compared to the conventional rate-1/2 constituent encoders, resulting in less degradation of the correcting capability of the constituent code [3].

In [1], a class of systematic recursive convolutional encoders restricted to be of rate $k/(k+1)$ is proposed. The codes are optimized in terms of the pairs (d_i, N_i), where d_i is the minimum weight of codewords generated by input sequences of weight i and N_i are their multiplicities, and thus suited to be used as constituent encoders of turbo codes [5]. Good systematic recursive encoding matrices with increasing values of encoder memory sizes for a fixed code rate are listed. Turbo codes constructed with the family of constituent encoders given in [1] are shown to outperform some high-rate turbo codes obtained by puncturing a rate-1/2 constituent encoder.

One major drawback of high-rate constituent encoders is the decoding complexity since it increases exponentially with k and with the constraint length for various decoding algorithms. With the motivation of proposing a class of turbo codes with low complexity, high-rate constituent encoders, Daneshgaran et al. [4] constructed recursive constituent encoders of rate $k/(k+1)$ by puncturing a rate-1/2 recursive mother encoder. However, a reduction in decoding complexity is obtained at the expense of a reduced spectrum (d_i, N_i) when compared to that of the best recursive encoder of rate $k/(k+1)$ [1].

An alternative to reduce the decoding complexity is to consider trellis representations for the constituent codes other than the conventional trellis usually adopted. For nonrecursive convolutional encodes, there is a trellis structure, the minimal trellis [6], which represents the coded sequences minimally under various complexity measures. The interest on the minimal trellis representation comes from its good error performance versus decoding complexity trade-off [7-10] and its potential power consumption and hardware utilization reductions [11]. However, the minimal trellis construction presented in [6] cannot be readily applied to turbo codes. The reason is that the mapping between information bits and coded bits produced by the minimal trellis corresponds to nonrecursive convolutional encoders, while systematic, recursive mappings are required in turbo coding. This article presents a method which can fill this gap.

*Correspondence: cecilio@ufpe.br
[2]UFPE, Recife-PE, Brazil
Full list of author information is available at the end of the article

In this article, we introduce the construction of the minimal trellis for a systematic recursive convolutional encoding matrix, the encoding required for the constituent codes of a turbo code. Our goal is to reduce the decoding complexity of a turbo decoder operating with high-rate constituent encoders. We also conduct a code search to show that a more finely grained decoding complexity-error performance trade-off is achieved with our approach. We tabulate several new encoding matrices with a larger variety of complexities than those in [1], as well as code rates other than $k/(k+1)$. The proposed minimal trellis can be constructed for systematic recursive convolutional encoders of any rate. Thus, our approach is more general than that in [1], while allowing that a distance spectrum (d_i, N_i) better than that of the punctured codes in [4] can be achieved.

The rest of this article is organized as follows. In Section 2, we introduce some basic definitions and notations. Section 3 introduces the minimal trellis construction for a systematic recursive convolutional encoding matrix. In Section 4, we present search code results. Section 5 concludes the article.

2 Preliminaries

Consider a convolutional code $C(n, k, \nu)$, where ν, k and n are the overall constraint length, the number of binary inputs and binary outputs, respectively, while the code rate is $R = k/n$. Every convolutional code can be represented by a semi-infinite trellis which (apart from a short transient in its beginning) is periodic, the shortest period being a *trellis module*. The *conventional* trellis module Φ_{conv} consists of a single trellis section with 2^ν initial states and 2^ν final states; each initial state is connected by 2^k directed branches to final states, and each branch is labeled with n bits.

The *minimal* trellis module, Φ_{min}, for nonrecursive convolutional codes was developed in [6]. Such a structure has n sections, $2^{\widetilde{\nu}_t}$ states at depth t, $2^{\widetilde{b}_t}$ branches emanating from each state at depth t, and one bit labeling each branch, for $0 \le t \le n-1$. The trellis complexity of the module Φ, $TC(\Phi)$, defined in [6] captures the complexity of trellis-based decoding algorithms [12]. It is shown in [6] that $TC(\Phi_{conv}) = \frac{n}{k} 2^{\nu+k}$ and $TC(\Phi_{min}) = \frac{1}{k}\sum_{t=0}^{n-1} 2^{\widetilde{\nu}_t+\widetilde{b}_t}$ symbols per bit. The *state* and the *branch complexity profiles* of the minimal trellis are denoted by $\widetilde{\boldsymbol{\nu}} = (\widetilde{\nu}_0, \ldots, \widetilde{\nu}_{n-1})$ and $\widetilde{\boldsymbol{b}} = (\widetilde{b}_0, \ldots, \widetilde{b}_{n-1})$, respectively. It has been shown in [6] that for many nonrecursive convolutional codes the trellis complexity $TC(\Phi_{min})$ of the minimal trellis module is considerably smaller than the trellis complexity $TC(\Phi_{conv})$ of the conventional trellis module.

A *generator matrix* $G(D)$ of a convolutional code $C(n, k, \nu)$ is a full-rank $k \times n$ polynomial (in D) matrix that encodes/generates C, i.e., is realizable by a linear sequential circuit (called an *encoder* for C) [13]. Let $G(0)$ denote the binary matrix obtained when substituting D with 0 in the matrix $G(D)$. If $G(0)$ is full-rank, then $G(D)$ is called an *encoding matrix* and is of particular interest. Two generator (encoding) matrices $G(D)$ and $G'(D)$ are *equivalent* if they generate the same code or, equivalently, if and only if there exists a $k \times k$ nonsingular polynomial matrix $T(D)$ such that $G(D) = T(D)G'(D)$. A generator matrix $G(D)$ is called *basic* if it is polynomial and it has a polynomial right inverse $n \times k$ matrix $G^{-1}(D)$. A basic encoding matrix $G(D)$ has a *Smith form decomposition* ([13], p. 44), ([14], Theorem 4.6)

$$G(D) = A(D)\Gamma'(D)B(D), \tag{1}$$

where the non-zero elements of $\Gamma'(D)$ are called the *invariant factors* of $G(D)$, the $k \times k$ matrix $A(D)$ and the $n \times n$ matrix $B(D)$ are both polynomial with unit determinants. Let ν_i be the *constraint length for the ith input* of a polynomial generator matrix $G(D)$, defined as $\nu_i = \max_{1 \le j \le n}\{\deg G_{i,j}(D)\}$. Then the overall constraint length (already mentioned) is given by $\nu = \nu_1 + \cdots + \nu_k$. Define also the *memory* m of $G(D)$ as $m = \max_{1 \le i \le k}\{\nu_i\}$. A basic generator matrix $G(D)$ is called *minimal-basic* if the overall constraint length ν is minimal over all equivalent basic encoding matrices.

A generator matrix $G(D)$ can be decomposed as $G(D) = G_0 + G_1 D + \cdots + G_m D^m$, where G_i, for $i = 0, \ldots, m$ are the $k \times n$ (scalar) generator submatrices. The *scalar* generator matrix G_{scalar} is given by [6]

$$G_{scalar} = \begin{bmatrix} G_0 & G_1 & \cdots & G_m & & \\ & G_0 & G_1 & \cdots & G_m & \\ & & \ddots & & & \ddots \end{bmatrix}. \tag{2}$$

The *"matrix module"* is the matrix $\hat{G}$ defined as [6]

$$\hat{G} = \begin{bmatrix} G_m \\ \vdots \\ G_0 \end{bmatrix}. \tag{3}$$

The generator matrix $G(D)$ is said to be in the *left–right* (LR) (or *minimal span* or *trelis oriented*) form, if no column of G_{scalar} contains more than one underlined entry (the Leftmost nonzero entry in its row), or more than one overlined entry (the Rightmost nonzero entry in its row). If $G(D)$ is in LR form, then it produces the minimal trellis for the code [6].

Next, we introduce a method to construct the minimal trellis for systematic recursive convolutional encoding matrices.

3 Construction of the minimal trellis for systematic recursive encoding matrices

The construction of the "minimal" trellis for the systematic recursive convolutional encoding matrix $G_{\text{sys}}(D)$ involves two main steps. First, we find the minimal trellis of an equivalent nonsystematic nonrecursive minimal-basic generator matrix in LR form. Then, the mapping between the information bits and coded bits in this trellis is changed in order to construct a systematic minimal trellis. The complete algorithm is summarized in the end of this section.

3.1 Minimal trellis for an equivalent encoder

Let $G_{\text{sys}}(D)$ be a systematic recursive encoding matrix for a rate $R = k/n$ convolutional code and let $q(D)$ be the least common multiple of all denominators of the entries in $G_{\text{sys}}(D)$. We construct a nonsystematic nonrecursive basic encoding matrix, denoted by $G_b(D)$, equivalent to $G_{\text{sys}}(D)$, as follows ([13], p. 44), ([14], Theorem 4.6):

- Find the Smith form decomposition of the polynomial matrix $q(D)G_{\text{sys}}(D)$:

$$q(D)G_{\text{sys}}(D) = A(D)\Gamma'(D)B(D), \tag{4}$$

  where the non-zero elements of $\Gamma'(D)$ are called the invariant factors of $q(D)\,G_{\text{sys}}(D)$, the $k \times k$ matrix $A(D)$ and the $n \times n$ matrix $B(D)$ are both polynomial with unit determinants. Thus, the invariant factor decomposition of $G_{\text{sys}}(D)$ is

$$G_{\text{sys}}(D) = A(D)\Gamma(D)B(D), \tag{5}$$

  where $\Gamma(D) = \Gamma'(D)/q(D)$.
- Form the desired encoding matrix $G_b(D)$ as the $k \times n$ submatrix of $B(D)$ in (5) consisting of the first k rows.

We can then perform a sequence of row operations on $G_b(D)$ to construct a nonsystematic nonrecursive minimal-basic encoding matrix $G(D)$ in LR form.

Example 1. *Consider the* $(n, k, \nu) = (4, 3, 3)$ *systematic recursive encoding matrix [1, Table IV]*

$$G_{sys}(D) = \begin{pmatrix} 1 & 0 & 0 & \dfrac{1+D+D^3}{1+D^3} \\ 0 & 1 & 0 & \dfrac{1+D^2+D^3}{1+D^3} \\ 0 & 0 & 1 & \dfrac{1+D+D^2+D^3}{1+D^3} \end{pmatrix}. \tag{6}$$

The invariant factor decomposition of $G_{sys}(D)$ *is given by*

$$G_{sys}(D) = \begin{pmatrix} 1 & 0 & 0 \\ D^2+D^4+D^5 & 1 & 1 \\ D^2+D^3+D^4+D^5 & 1 & 0 \end{pmatrix} \begin{pmatrix} \frac{1}{1+D^3} & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 1+D^3 & 0 & 0 & 1+D+D^3 \\ D^2+D^3+D^4+D^5 & 0 & 1 & 1+D+D^4+D^5 \\ D^3 & 1 & 1 & D+D^3 \\ D^2 & 0 & 0 & 1+D^2 \end{pmatrix}. \tag{7}$$

The basic encoding matrix $G_b(D)$ *equivalent to* $G_{sys}(D)$ *is readily obtained from* (7). *Using the greedy-algorithm [15] we turn* $G_b(D)$ *into the LR form, or equivalently, into the minimal-span form [15, Theorem 6.11], resulting in the following generator matrix*

$$G(D) = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 1+D & D & 1 \\ D+D^2 & D+D^2 & 1 & 1+D \end{pmatrix} \tag{8}$$

with overall constraint length $\nu = 3$. *The trellis complexity of the conventional module for* $G_{sys}(D)$ *is* $TC(\Phi_{conv}) = 85.33$ *symbols per bit.*

The minimal trellis module for the convolutional code with $G(D)$ given in (8), constructed with the method presented in [6], is shown in Figure 1. It has state and branch complexity profiles given by $\tilde{\boldsymbol{\nu}} = (3, 4, 4, 4)$ and $\tilde{\boldsymbol{b}} = (1, 1, 1, 0)$, respectively. Thus, the trellis complexity

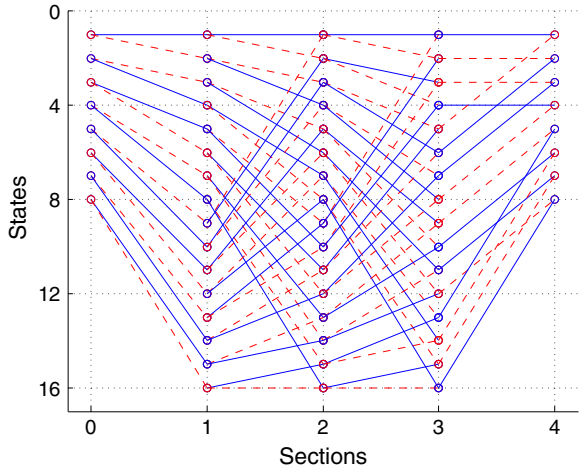

Figure 1 Minimal trellis module for the $(n, k, \nu) = (4, 3, 3)$ convolutional code with $G(D)$ given in (8). The solid/blue branches represent "0" codeword bits and the dashed/red branches represent "1" codeword bits. In the first three transitions, the upper (resp. lower) branches correspond to information bit "0" (resp. "1"), *i.e.*, the standard convention.

of the minimal trellis is $TC(\Phi_{min}) = 32$ symbols per bit. In the first three trellis sections, the upper (resp. lower) branches correspond to information bit "0" (resp. "1"), i.e., the standard convention [6]. The solid branches represent "0" codeword bits and the dashed branches represent "1" codeword bits. The mapping between information bits and coded bits in Figure 1 is not systematic, so this trellis does not represent the same encoder as $G_{\text{sys}}(D)$. The construction of a minimal trellis for systematic recursive encoders is discussed in the next section.

3.2 Minimal trellis for systematic recursive encoders

Originally, minimal trellises have been constructed for codes, not for matrices (or encoders). However, the convention that the upper branches refer to the information bit 0 and the lower branches refer to the information bit 1 yields a particular encoding which, in general, is not systematic. We note that the association of solid/dashed branches with codeword bits can not be changed, as any change in this regard would result in a trellis which would no longer represent the convolutional code. However, by enforcing a different convention on the association of the branches to the information bits, only a different encoding for the same code is obtained.

Since in the systematic part the information bit and the coded bit must have the same value, all we need to do is to change, in the information sections of the minimal trellis, the standard convention to that where solid branches refer to the information bit 0 and the dashed branches refer to the information bit 1. Let us refer to this convention as the *systematic* convention. Getting back to the minimal trellis in Figure 1, for the nonsystematic nonrecursive convolutional encoding matrix $G(D)$ given in (8), we only need to adopt the systematic convention in the first three sections to turn this minimal trellis into a systematic trellis.

It remains to show that the minimal trellis in Figure 1 with the systematic convention is the minimal trellis for the systematic recursive convolutional encoding matrix $G_{\text{sys}}(D)$ in (6). We note that the generator matrices $G(D)$ given in (8) and $G_{\text{sys}}(D)$ in (6) are equivalent in the sense that both generate the same code. Therefore, except for the edge convention, the two trellises are exactly the same. Assuming that the information bits at the input of the encoder associated with $G_{\text{sys}}(D)$ and the information bits associated with the minimal trellis in Figure 1 occupy the same positions, the systematic convention is unique. Consequently, the trellis in Figure 1 with the systematic convention in the first three sections is the minimal trellis for the systematic recursive convolutional encoding matrix $G_{\text{sys}}(D)$ in (6).

The complete algorithm for the construction of the minimal trellis for a systematic recursive encoding matrix is summarized as:

1. From $G_{\text{sys}}(D)$, use the Smith form decomposition procedure to obtain the basic nonsystematic nonrecursive encoding matrix $G_b(D)$;
2. If $G_{\text{b}}(D)$ does not have the LR property, then apply the greedy algorithm described in [15] to turn it into a LR form. Denote the new generator matrix by $G(D)$. Else set $G(D) \leftarrow G_{\text{b}}(D)$;
3. Construct the minimal trellis module for $G(D)$ with the method presented in [6];
4. Adopt the systematic convention on the minimal trellis module obtained in the previous step. The resulting trellis is the desired systematic trellis.

Example 2. *Consider the following systematic recursive* $(n, k, \nu) = (5, 4, 3)$ *encoder matrix from [1, Table IV] with* $TC(\Phi_{conv}) = 160$ *symbols per bit*

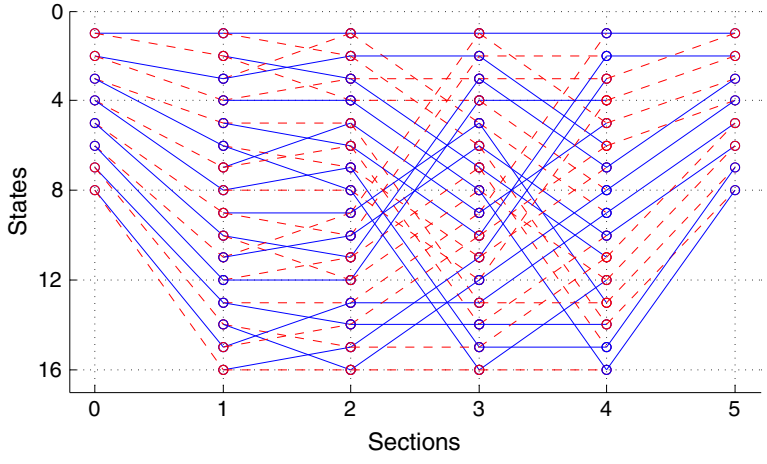

Figure 2 Minimal trellis module for the $(n, k, \nu) = (5, 4, 3)$ systematic recursive encoder. Solid/blue branches represent "0" codeword bits while dashed/red branches represent "1" codeword bits. The same convention (*i.e.*, the systematic convention) applies to the first four trellis sections.

$$G_{sys}(D) = \begin{pmatrix} 1 & 0 & 0 & 0 & \frac{1+D+D^2+D^3}{1+D+D^3} \\ 0 & 1 & 0 & 0 & \frac{1+D+D^2}{1+D+D^3} \\ 0 & 0 & 1 & 0 & \frac{1+D^2+D^3}{1+D+D^3} \\ 0 & 0 & 0 & 1 & \frac{1+D^3}{1+D+D^3} \end{pmatrix}.$$

Using the procedure described in Section 3 we construct the "minimal" trellis shown in Figure 2. This module has state and branch complexity profiles given by $\tilde{\boldsymbol{\nu}} = (3,4,4,4,4)$ *and* $\tilde{\boldsymbol{b}} = (1,1,1,1,0)$*, respectively. Thus,* $TC(\Phi_{min}) = 32$ *symbols per bit.*

4 New codes

Graell i Amat et al. [1] tabulated good $k \times (k+1)$ encoding matrices $G_{\text{sys}}(D)$ to be used as component encoders of parallel concatenated turbo codes. We search for good (with respect to the pair (d_i, N_i) constituent systematic recursive convolutional encoder matrices with a larger

Table 1 Minimal trellis complexity-performance trade-off for $R = 2/4, 3/4, 3/5, 4/5$

R	TC	d_2, N_2	d_3, N_3	d_4, N_4	d_5, N_5	d_6, N_6	$G(D)$
$\frac{2}{4}$	12.00	4,2	4,2	6,3	8,6	10,9	[2 3 0 1; 3 0 1 1]
	24.00	6,2	5,4	6,4	7,2	8,3	[2 3 1 1; 5 1 0 3]
	28.00	7,2	5,1	6,1	7,2	8,2	[2 3 3 0; 7 1 2 3]
	32.00	8,2	6,4	6,3	8,6	8,2	[3 2 2 3; 4 3 1 3]
	48.00	10,2	6,1	6,1	6,1	8,2	[7 2 3 3; 4 5 1 3]
	56.00	11,2	6,2	7,2	7,1	8,1	[4 3 1 3; 7 4 4 3]
	64.00	12,2	7,3	6,1	7,2	10,20	[7 3 1 2; 2 7 7 7]
	80.00	14,2	7,3	8,12	9,31	6,1	[7 4 3 3; 4 3 5 5]
$\frac{3}{4}$	10.67[a]	3,3	3,3	4,7	5,15	6,36	[1 1 1 1; 2 0 1 1; 2 3 1 0]
	18.67	3,2	3,1	4,4	5,14	6,40	[2 0 1 1; 0 1 2 1; 3 3 1 1]
	21.33	4,4	4,15	4,10	6,215	6,86	[3 1 0 1; 2 1 3 3; 2 2 1 1]
	32.00[a]	4,1	4,3	4,1	5,5	6,24	[1 1 1 1; 0 3 2 1; 6 6 1 3]
	42.67	5,2	4,2	4,1	5,9	6,23	[1 1 1 1; 6 0 1 3; 4 5 2 3]
	53.33	6,3	4,3	5,12	5,7	6,23	[2 3 2 1; 2 0 3 3; 7 1 1 1]
	64.00[a]	6,2	4,1	5,8	5,4	6,11	[1 1 1 1; 4 4 3 1; 2 5 4 3]
	74.67	7,2	4,3	4,1	5,5	6,17	[2 1 1 1; 5 7 1 0; 6 0 7 0]
$\frac{3}{5}$	10.67	3,2	3,1	5,2	7,10	8,5	[1 1 1 0 0; 0 3 0 0 1; 2 0 1 1 0]
	21.33	4,1	4,3	5,4	5,1	6,1	[0 2 1 1 1; 3 0 1 0 1; 2 1 3 1 0]
	29.33	6,3	5,8	4,1	7,33	6,1	[2 2 3 1 1; 2 3 0 3 0; 3 1 1 1 1]
	32.00	6,1	5,4	4,1	6,3	6,1	[0 3 2 1 1; 2 2 1 2 3; 3 1 1 0 1]
	37.33	7,2	4,1	5,2	6,2	7,3	[0 3 2 1 0; 2 2 1 1 1; 1 3 1 2 2]
	48.00	8,3	5,1	5,2	6,2	6,1	[4 0 3 1 3; 3 2 3 2 1; 2 3 1 1 0]
	58.66	8,2	5,2	5,1	6,1	7,2	[6 3 2 3 3; 1 3 3 0 1; 2 2 1 3 1]
	64.00	10,3	5,2	5,2	6,3	7,2	[6 2 3 3 3; 1 0 3 2 3; 0 3 2 1 1]
	74.67	10,2	5,1	5,2	6,2	8,10	[2 2 3 2 3; 7 6 2 3 1; 2 1 2 1 1]
$\frac{4}{5}$	14.00[a]	2,1	3,5	4,17	5,65	6,236	[1 1 0 0 1; 2 0 1 1 0; 0 1 0 1 0; 2 0 2 1 1]
	18.00	3,4	3,6	4,23	5,80	6,284	[3 1 0 0 1; 0 2 1 1 1; 0 1 1 1 0; 2 0 2 1 1]
	20.00	3,2	3,4	4,11	5,45	6,220	[2 0 1 1 0; 2 2 0 1 1; 0 1 1 1 1; 3 1 3 1 1]
	32.00[a]	4,2	3,1	4,8	5,42	6,179	[2 0 3 1 0; 2 2 0 3 1; 1 0 1 1 1; 0 3 0 1 0]
	48.00	4,1	3,1	4,4	5,22	6,105	[2 1 1 1 1; 1 1 3 3 0; 2 2 0 1 1; 0 2 1 2 3]
	56.00	5,2	3,1	4,3	5,24	6,129	[2 1 1 1 1; 3 2 3 1 1; 0 2 1 3 0; 2 0 0 3 3]
	64.00[a]	5,2	4,4	4,3	5,14	6,86	[1 0 1 1 1; 0 2 0 3 1; 2 1 3 1 1; 6 6 3 3 2]
	72.00	6,4	4,10	4,8	6,392	6,227	[2 0 1 1 1; 1 0 3 0 1; 2 3 1 3 0; 0 4 2 3 3]

[a]Code listed in [1].

variety of complexities compared to those listed in [1]. The main idea is to propose templates with polynomial generator matrix $G(D)$ in trellis-oriented form [6,7] with fixed $TC(\Phi_{min})$. This can be done by placing the leading (underlined) and trailing (overlined) 1's of each row of the "matrix module" in specific positions, leaving the other positions free to assume any binary value.

Example 3. *The following "matrix module"*

$$\begin{pmatrix} \overline{1} & 0 & 0 & 0 \\ * & * & * & \overline{1} \\ * & \overline{1} & 0 & 0 \\ \underline{1} & * & * & * \\ 0 & \underline{1} & * & * \\ 0 & 0 & \underline{1} & * \end{pmatrix}$$

is associated with an ensemble of nonsystematic nonrecursive convolutional codes of rate $3/4$ *and trellis complexity of the minimal trellis module* $TC(\Phi_{min}) = 21.33$ *symbols per bit.*

Remark. In our code search we enforce that the positions of the underlined 1's are in the first k columns of the matrix module in order to assure that the information bits of the corresponding minimal trellis are in the first k sections. □

By varying the free positions (marked with an "*") in this matrix, several polynomial generator matrices $G(D)$ are produced. For each matrix $G(D)$ in this ensemble, we apply Steps (3) and (4) of the algorithm, i.e., we construct the minimal trellis module for $G(D)$ and adopt the systematic convention to the information sections, and then calculate the pairs (d_i, N_i), for $i = 2, \ldots, 6$. The dominant term, d_2, is called the *effective free distance* of the turbo code [16].

Codes of rate $R = 2/4, 3/4, 3/5, 4/5$ are listed in Table 1, which indicates the relationship between $TC(\Phi_{min})$ and the error performance expressed in terms of the pairs (d_i,N_i). The matrix $G(D)$ shown in the table together with the systematic convention is used to construct the minimal trellis that attains the corresponding (d_i, N_i), $i = 2, \ldots, 6$. The existing codes taken from [1] are also indicated in Table 1. Their (minimal) trellis complexity, shown in the table, was obtained by applying the complete algorithm of Section 3.2 to their respective systematic recursive encoding matrices. For example, the matrices $G_{sys}(D)$ listed in [1, Table IV] for $R = 3/4$ yield $TC(\Phi_{min}) = 10.67$ (for $\nu = 2$), $TC(\Phi_{min}) = 32$ (for $\nu = 3$) and $TC(\Phi_{min}) = 64$ (for $\nu = 4$). New codes with a variety of trellis complexities are found with our code search procedure. The effective free distance and/or multiplicity of the code are gradually improved as more complex codes are sought.

5 Conclusions

We present a method to construct the minimal trellis for a recursive systematic convolutional encoding matrix. Such a trellis minimizes the trellis complexity measure introduced by McEliece and Lin [6], which applies to trellis-based decoding algorithms. As a contribution of this work, several new convolutional encoding matrices having an equivalent systematic recursive encoding matrix, optimized for turbo codes, are tabulated. They provide a wide range of performance-complexity trade-offs, to serve several practical applications.

Endnotes

[a]This work was presented in part at the IEEE International Symposium on Information Theory (ISIT 2011), Saint Petersburg, Russia, July 2011.

Competing interests
The authors declare that they have no competing interests.

Acknowledgements
This work was partially supported by the CAPES and CNPq (Brazil).[a]

Author details
[1]IFAM, Manaus-AM, Brazil. [2]UFPE, Recife-PE, Brazil. [3]UTFPR, Curitiba-PR, Brazil. [4]UFSC, Florianópolis-SC, Brazil.

Received: 25 May 2012 Accepted: 6 October 2012
Published: 21 November 2012

References

1. Graell i, A Amat, G Montorsi, S Benedetto, Design and decoding of optimal high-rate convolutional codes. IEEE Trans. Inf. Theory. **50**, 867–881 (2004)
2. MA Kousa, AH Mugaibel, Puncturing effects on turbo codes. IEEE Proc. Commun. **149**, 132–138 (2002)
3. C Douillard, C Berrou, Turbo codes with rate-$m/(m+1)$ constituent convolutional codes. IEEE Trans. Commun. **53**, 1630–1638 (2005)
4. F Daneshgaran, M Laddomada, M Mondin, High-rate recursive convolutional codes for concatenated channel codes. IEEE Trans. Commun. **53**, 1846–1850 (2004)
5. S Benedetto, G Montorsi, Design of parallel concatenated convolutional codes. IEEE Trans. Commun. **44**, 591–600 (1996)
6. RJ McEliece, W Lin, The trellis complexity of convolutional codes. IEEE Trans. Inf. Theory. **42**, 1855–1864 (1996)
7. BF Uchôa-Filho, RD Souza, C Pimentel, M Jar, Convolutional codes under a minimal trellis complexity measure. IEEE Trans. Commun. **57**, 1–5 (2009)
8. A Katsiotis, P Rizomiliotis, N Kalouptsidis, New constructions of high-performance low-complexity convolutional codes. IEEE Trans. Commun. **58**, 1950–1961 (2010)
9. F Hug, IE Bocharova, R Johannesson, B Kudryashov, in *Proc. IEEE Int. Symp. Inform. Theory*. Searching for high-rate convolutional codes via binary syndrome trellises (Seoul, Korea, 2009), pp. 1358–1362
10. A Katsiotis, N Kalouptsidis, On (n,n-1) punctured convolutional codes and their trellis modules. IEEE Trans. Commun. **59**, 1213–1217 (2011)
11. BU Pedroni, VA Pedroni, RD Souza, in *Proc. of the 4th Inter. Symp. on Commun., Control and Signal Processing (ISCCSP'2010)*. Hardware implementation of a Viterbi decoder using the minimal trellis (Limassol, Cyprus, 2010), pp. 1–4
12. B Vucetic, J Yuan, *Turbo Codes: Principles and Applications*. (Kluwer Academic Publishers, Boston/Dordrecht/London, 2000)
13. R Johannesson, KS Zigangirov, *Fundamentals of Convolutional Coding*. (Wiley-IEEE Press, New York, 1999)

14. C Schlegel, L Pérez, *Trellis and Turbo Coding*. (Wiley-IEEE Press, New York, 2004)
15. RJ McEliece, On the BCJR trellis for linear block codes. IEEE Trans. Inf. Theory. **42**, 1070–1092 (1996)
16. D Divsalar, RJ McEliece, Effective free distance of turbo codes. IEEE Electron. Lett. **32**, 445–446 (1996)

doi:10.1186/1687-6180-2012-243
Cite this article as: Benchimol *et al.*: **High-rate systematic recursive convolutional encoders: minimal trellis and code search.** *EURASIP Journal on Advances in Signal Processing* 2012 **2012**:243.