Nonlinear Turbo Codes For Higher-Order Modulations

Miguel Griot, Andres I. Vila Casado and Richard Wesel
Dept. of Electrical Engineering, University of California
Los Angeles, CA 90095
{mgriot,avila,wesel}@ee.ucla.edu

***Abstract*— In 1982 Ungerboeck presented a set-partitioning design technique for trellis coded modulation (TCM). Although this technique directly assigns constellation points to the branches of the trellis, it has been shown that the codes Ungerboeck designed may be represented by a linear convolutional code with a mapper that assigns a series of coded bits to a constellation point. This notion has remained with the appearance of turbo codes. Therefore, parallel concatenated trellis coded modulation (PC-TCM) has been traditionally designed using parallel concatenated linear convolutional codes with a bits-to-symbol mapper. This paper shows that in the case of PC-TCM the use of convolutional codes and a mapper could be too restrictive, and that designing a nonlinear turbo code directly assigning constellation points to the output branches of the constituent codes can improve the performance. Also, an extension of Benedetto's uniform interleaver for nonlinear constituent codes is presented. Simulation results are shown for a 2 bits/s/Hz 16-state nonlinear turbo code with 8PSK. This code is within 0.5 dB away from capacity with an interleaver length of 10000 bits, and outperforms previous published linear turbo code by around 0.2 dB.**

***Index Terms*— Channel coding, information rates, turbo codes, nonlinear codes, PSK, trellis codes, parallel concatenated trellis codes**

I. Introduction

Trellis Coded Modulation (TCM) was proposed by Ungerboeck in 1982 [1], where he presented a set-partitioning design technique which directly assigns constellation points to the branches of the trellis. However, it has been shown that the codes Ungerboeck designed may be represented by a linear convolutional code with a mapper that assigns a series of coded bits to a constellation point. This notion has remained with the appearance of turbo codes. Therefore, parallel concatenated trellis coded modulation (PC-TCM) has been traditionally designed using parallel concatenated convolutional codes with a bits-to-symbol mapper [2][3][4].

This paper shows that for higher-order modulations the use of parallel concatenated linear convolutional codes and a mapper constrains the performance. Parallel concatenated *nonlinear* trellis coded modulation (PC-NLTCM), which directly assigns constellation points to the output branches of the constituent codes, can improve the performance. As an example, simulation results are shown for a 2 bits/s/Hz 16-state nonlinear turbo code with 8PSK. This code is less than 0.5 dB away from capacity at a BER $= 10^{-5}$ with an interleaver length of 10000 bits, and outperforms the best previous published linear turbo code by around 0.2 dB in the waterfall region. Moreover, this improvement in performance comes at no cost in increased implementation complexity since the decoding algorithm for linear or nonlinear constituent trellis codes is the same.

To facilitate analysis of the new codes, an extension of Benedetto's uniform interleaver for nonlinear constituent codes is presented. It is shown that the same design criteria for linear turbo codes can be applied to nonlinear turbo codes. Namely, we generalize the notion of *effective free distance* for nonlinear codes, and show that this is an important metric to maximize when designing constituent codes for a PC-NLTCM.

This paper is organized as follows. Section II reviews the general structure of the PC-NLTCM. Section III introduces an extension of Benedetto's uniform interleaver analysis for nonlinear codes. Section IV shows an example where PC-NLTCM can outperform the best reported PC-TCM with mapper. Section V delivers the conclusions.

II. Parallel concatenated trellis coded modulation

The structure of parallel concatenated nonlinear trellis codes (PC-NLTCs) was introduced in [5] for binary outputs. It is in essence the well-known turbo-code structure first proposed in [6] for systematic linear encoders, except that the output label is assigned directly to each branch of the trellis by a look-up table rather than a linear function of the state and the input bits. A similar replacement of a linear operation with a look-up table has been successfully proposed for a decision feedback equalizer to equalize channels with trailing nonlinear inter-symbol interference [7]. Looking at Fig. 1, the PC-NLTCM consists of two constituent nonlinear trellis encoders (block NLTC) linked by an interleaver (block Π). The trellis encoder uses k_b bits per trellis section. The NLTC is composed by a 2^ν-state trellis structure (block S), and a look-up table (block LUT). The block S stores the current trellis state, while the look-up table stores the output for each branch of the trellis. Each output consists of n_c constellation points, resulting in a total rate of $k_b/(2n_c)$ bits/symbol. The trellis codes considered in this work are non-systematic. Also, in the application considered in this paper $k_b > 1$, in which case symbol interleaving [4] will be assumed. We will denote the input block length in bits as K_b, and the interleaver length

This work was supported by the Defence Advanced Research Project Agency under SPAWAR Systems Center San Diego Grant N66001-02-1-8938.

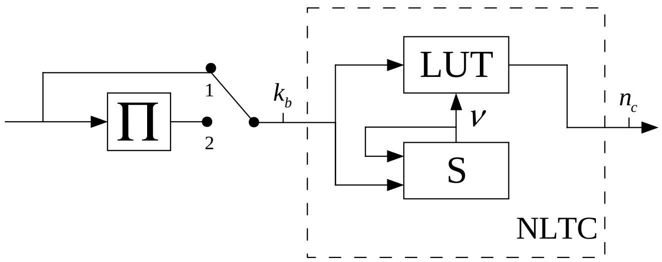

Fig. 1. PC-NLTC structure.

in symbols as $K_s = K_b/k_b$. Also, we will denote the input symbol alphabet size as $q = 2^{k_b}$.

III. Bit Error Rate Bound of Parallel Concatenated Nonlinear Codes

A method to evaluate the bit error probability of a parallel concatenated coding scheme averaged over all interleavers of a certain length has been proposed in [8]. This upper bound is known as the uniform interleaver bound and assumes the use of a Maximum-Likelihood (ML) decoder. However, this bound cannot be applied to PC-NLTCs because it assumes a parallel concatenation of **linear** codes. Hence, an upper bound to the BER is found assuming the all-zero word is transmitted. An extension of the bounding technique proposed in [8] for a parallel concatenation of nonlinear codes was presented in [5]. However, the approach on [5] presented complex equations, making it difficult to draw conclusions and design criteria from them. In this paper, a new probabilistic device will be defined as interleaver, which produces similar equations to the ones presented in [8].

Also, the analysis in [8] assumes a parallel concatenation of systematic codes. Since the codes used in this work are non-systematic, the new error probability upper bounding technique will be derived assuming non-systematic constituent encoders. Nevertheless, it should be clear how to modify the equations in the case of systematic nonlinear codes. Finally, the analysis contemplates constituent encoders with more than one input bit per trellis section. In that case, a symbol-interleaver [4] is assumed, and the symbol error rate (SER), i.e. the average number of k_b-bit symbols that are in error, will be computed. Note that for $k_b = 1$ the symbol error rate is the bit error rate (BER).

The main difference with linear codes, is that for nonlinear codes we can no longer assume that the all-zero codeword is transmitted. We propose a new definition of a uniform interleaver that extends the results, conclusions and design criteria drawn in [8] to nonlinear constituent codes.

Definition 1: A *Uniform Symbol-Interleaver with Re-mapping (USIR)* of length K_s (the number of input symbols) for nonlinear codes is a probabilistic device defined as follows: There are two operations considered in the interleaver. First, the uniform interleaver selects any of the $K_s!$ possible permutations of the symbol positions with equal probability. Second, for each position, the value of the symbol can be re-mapped to any of the $q = 2^{K_b}$ possible values with equal probability. The re-mapping can be different for different positions, but in a fixed position it must be an invertible function over the q-ary symbols, *i.e.* no two different symbols can be re-mapped to a same symbol.

The reason for this extension is that for nonlinear codes we need to consider all the possible input pairs. The uniform interleaver as defined in [8] would maintain the Hamming weight of both input words and their Hamming distance, which would make the equations more complicated and would make it harder to draw conclusions from them (see [5]). With this new definition, any input word can be mapped to any other input word, no matter their Hamming weight. Thus, the only value preserved after the interleaver is the symbol-wise Hamming distance between any two input pairs. This is a generalization of the analysis for linear codes, since the Hamming weight of the erroneous input word, which is the value preserved in [8], is the Hamming distance between the correct input word (the all-zero word) and the erroneous word.

Using the USIR, any pair of input words U and $\tilde{U}$ such that $d_H(U,\tilde{U}) = i$, can be mapped to any pair of input words satisfying $d_H(\Pi(U),\Pi\tilde{U}) = i$ with probability:

$$P\Big((U,\mathrm{U}) \to (\Pi(U),\Pi(\tilde{U}))\Big| \; d_H(U,\tilde{U}) = d_H(\Pi(U),\Pi(\tilde{U})) = i\Big) \;=\; \frac{1}{q^{K_s}\cdot(q-1)^i\cdot\binom{K_s}{i}}. \tag{1}$$

Consider any two words of length n, $X = \{x_1,\cdots,x_n\}$ and $\tilde{X} = \{\tilde{x}_1,\cdots,\tilde{x}_n\}$, where the x_i's are constellation points. For the AWGN, the probability of transmitting X, and receiving a word which makes $\tilde{X}$ more likely to have been transmitted than X is:

$$P_e(X \to \tilde{X}) = Q\left(\sqrt{d_E^2(X,\tilde{X})\frac{E_s}{2N_0}}\right) \le \frac{1}{2}e^{-\frac{E_s}{4N_0}d_E^2(X,\tilde{X})}, \tag{2}$$

where E_s/N_0 is the signal-to-noise ratio and $d_E^2(X,\tilde{X})$ is the squared Euclidean distance assuming unit-norm transmission. Although we will be focusing in the AWGN channel the technique presented in this work is valid for any channel for which an additive distance can be defined and for which an upper bound to the pair-wise error probability can be upper-bounded by:

$$P(X \to \tilde{X}) \le \nu\lambda^{d(X,\tilde{X})} \tag{3}$$

where the distance metric $d(\tilde{X},X)$, and the parameters ν, λ depend on the channel. Note that (2) is equivalent to (3) with $d(X,\tilde{X}) = d_E^2(X,\tilde{X})$, $\nu = 1/2$ and $\lambda = e^{-\frac{E_s}{4N_0}}$. We will use this last general notation throughout the rest of the derivations.

Define the *Input-Output Distance Enumerating Function* (IODEF) of a given (n,K_s) code C as

$$A^C(I,D) = \sum_{i,d} A^C_{i,d} I^i D^d, \tag{4}$$

where $A^C_{i,d}$ is the number data-word pairs $(U,\tilde{U})$ that satisfy $d_H(U,\tilde{U}) = i$, and the distance between the corresponding codewords $d(X,\tilde{X}) = d$. I and D are placeholders.

Also define the *Conditional IODEF* (IODEF) as:

$$A_i^C(D) = \sum_d A_{i,d}^C D^d. \quad (5)$$

Inserting Eq. (5) in Eq. (4), the expression for the IODEF can be rewritten as:

$$A^C(I,D) = \sum_i A_i^C(D) I^i. \quad (6)$$

Using (3) and (4) the symbol error rate (SER) or bit error rate (BER) in case $k_b = 1$ can be upper bounded by:

$$\text{SER} \leq \frac{\nu}{K_s} \cdot (1/q)^{K_s} \frac{\partial A^C(I,D)}{\partial I}\bigg|_{D=\lambda, I=1}. \quad (7)$$

A. Parallel concatenation of block codes

Denote C_P as the $(n_1 + n_2, K_s)$ block code resulting from the parallel concatenation of two codes, an (n_1, K_s) block code C_1 and an (n_2, K_s) block code C_2. We will assume an interleaver of length K_s symbols, equal to the input word length, in order to simplify the analysis (An extension can easily be made for the case when l consecutive codewords of the constituent codes are used for one operation of the interleaver, as explained in [8]). The directional distance is additive, so the directional distance of the concatenated codeword is the sum of the directional distances between the corresponding constituent codewords.

Hence, the conditional IODEF of C_P can be expressed (using (1)) as:

$$A_i^{C_P}(D) = \frac{A_i^{C_1}(D) \cdot A_i^{C_2}(D)}{q^{K_s} \cdot (q-1)^i \cdot \binom{K_s}{i}}. \quad (8)$$

Notice that the USIR as defined in Sec. III can map any input word to any other input word. Now, using (8) and (4) in (7), it can be observed that there are two terms of the form $(1/q)^{K_s}$, corresponding to the probability of the correct input word and the probability of that input word being mapped to any other word after the interleaver. Define the *Normalized Input-Output Distance Enumerating Function* (NIODEF) of a given (n, K_s) code C as

$$\tilde{A}^C(I,D) = \sum_{i,d} \tilde{A}_{i,d}^C I^i D^d, \quad (9)$$

where $\tilde{A}_{i,d}^C = A_{i,d}^C / q^{K_s}$. Hence, the symbol error probability can be upper bounded by:

$$\text{SER} \leq \frac{\nu}{K_s} \frac{\partial \tilde{A}^C(I,D)}{\partial I}\bigg|_{D=\lambda, I=1}. \quad (10)$$

Now, using (5) and (9):

$$\tilde{A}_i^{C_P}(D) = \frac{\tilde{A}_i^{C_1}(D) \cdot \tilde{A}_i^{C_2}(D)}{(q-1)^i \cdot \binom{K_s}{i}}. \quad (11)$$

Note that except for the term $1/q^{K_s}$ in $\tilde{A}_{i,d}$, and the term $1/(q-1)^i$ the equations (9)-(11) for a parallel concatenation of nonlinear codes are the same as for the linear case [8]. As it turns out, all the conclusions and design criteria derived in [8] apply to nonlinear constituent codes, as we will prove in Section III-B. In particular, it is shown that feed-forward encoders are not suitable for parallel concatenation, and that recursive convolutional codes are required. Moreover, just as in [8] an important parameter to maximize is the effective free distance, which we generalize for nonlinear constituent codes as:

Definition 2: *Effective free distance* of a constituent nonlinear code is the minimum distance $d(X, \tilde{X})$ between the two outputs corresponding to any two possible input words U and $\tilde{U}$ with input Hamming distance $d_H(U, \tilde{U}) = 2$.

B. Parallel Concatenation of Nonlinear Trellis Codes

Biglieri et al. presented a union bound in [9][10] for general trellis codes, using a $2^{2\nu}$-state trellis diagram. This concept can be used to find $A^{C_P}(I,D)$ for the case of parallel concatenated nonlinear trellis codes.

As in [9], the product state diagram consists of state pairs, (s_e, s_r), where s_e is the encoder state and s_r the receiver state. Following Biglieri's notation, the product states can be divided into two sets, the good states denoted by S_G and the bad states denoted by S_B defined as

$$S_G = \{(s_e, s_r) \mid s_e = s_r\}, \ S_B = \{(s_e, s_r) \mid s_e \neq s_r\}. \quad (12)$$

By suitably renumbering the product states, we get the transition matrix

$$S(I,D) = \left[\begin{array}{c|c} S_{GG}(I,D) & S_{GB}(I,D) \\ \hline S_{BG}(I,D) & S_{BB}(I,D) \end{array} \right], \quad (13)$$

where the $N \times N$ matrix $S_{GG}(I,D)$ accounts for the transitions between good product states, the $N \times (N^2 - N)$ matrix $S_{GB}(I,D)$ accounts for the transition from good product states to bad product states, and so forth. N is the number of encoder states 2^ν. For each transition in the product state diagram from product state S_1 to S_2, the branch label is:

$$(1/q) I^{d_H(u_e, u_r)} D^{d(x_e, x_r)}, \quad (14)$$

where u_e and x_e denote the input and output word for the encoder states respectively, and u_r and x_r denote the input and output word for the receiver. Note that since, there are $q = 2^{k_b}$ possible inputs per trellis branch, $(1/q)$ is the the probability of each branch transition given a certain current state.

Although $\tilde{A}^C(I,D)$ can be computed using $S(I,D)$, it becomes very complex. To reduce complexity, two approximations can be made. *Approximation 1*: Use the same idea presented in [8]: every path in the trellis representation starts and ends in the same state. Any possible incorrect word departs from a good state to a bad state at some trellis section a certain number of times m, and returns to a good state the same number of times m. *Approximation 2*: In the encoding process, at any trellis section, the encoder state can be any of the possible $N = 2^v$ states with equal probability.

Define the approximated single-error event function as:

$$E(I, D) = p_s\{S_{GB}(I - S_{BB})^{-1}S_{BG}\}\mathbf{1}, \qquad (15)$$

where $p_s = [\frac{1}{N}\frac{1}{N}\cdots\frac{1}{N}]$ is the probability distribution of the encoder states and $\mathbf{1} = [11\cdots1]^T$. Then, $E(I, D)$ can be written as:

$$E(I, D) = \sum_{i,d} e_{i,d}D^dI^i. \qquad (16)$$

Now define:

$$E_j(I, D) = \Big[E(I, D)\Big]^j = \sum_{i,d} e_{i,d,j}I^iD^d, \qquad (17)$$

which counts every concatenation of j single-error events, without leaving any trellis section between them, using Approximation 2. Every error event can be represented as a concatenation of single-error events. Using Approximation 2, a concatenation of j single-error events, with a total length l can be positioned in

$$K[l, j] \leq \binom{K_s - l + j}{j} \approx \frac{K_s^j}{j!}, \qquad (18)$$

ways in the trellis. Note that the two sides of the inequality in (18) are not exactly equal, since the error events start at a particular state, and there might be positions where the concatenation of two error events is not possible. However, for K_s large the upper bound becomes very tight. Also, the symbols of the rest of the $K_s - l$ positions of both input words are equal and could be almost any of the possible q^{K_s-l} combinations, which divided by the term q^{K_s} appearing in $\tilde{A_{i,d}}$ gives q^{-l} which is already counted by the terms $(1/q)$ appearing in the branch labels in $S(I, D)$ (see (14)). The approximation in (18) follows from the fact that $K_s >> l$, $K_s >> j$ and the Stirling approximation $\binom{k}{i} \approx k^i/i!$ for $k >> i$. Therefore, for each constituent code,

$$\tilde{A^C} \approx \sum_j \frac{K_s^j}{j!}E_j(I, D). \qquad (19)$$

Using (10), (11) and (19), and using again Stirling for $K_s >> j$, we get:

$$SER \approx \sum_{i,j_1,j_2,d_1,d_2} \nu \frac{ii!}{j_1!j_2!}\frac{K_s^{(j_1+j_2-i-1)}}{(q-1)^i} e^{C_1}_{i,d_1,j_1} e^{C_1}_{i,d_2,j_2} \lambda^{d_1+d_2}. \qquad (20)$$

Therefore, as K_s increases, the performance of the code will be driven by the terms with the largest possible value of $(j_1 + j_2 - i - 1)$. For recursive encoders, that happens for a concatenation of error events with $i = 2$. Therefore, an important parameter to maximize is the effective free distance as defined in Sec. III-A. As for linear encoders, feed-forward encoders lead to poor performance since i can be equal to 1 in which case $j_1 + j_2 - i - 1 = 0$.

IV. Design Example, 2-bits/s/Hz 16-state PC-NLTCM with 8PSK

In this section we will show that directly assigning constellation points to the trellis branches of each constituent code can produce codes that outperform linear codes with mapping. As an example, we will design a 2-bits/s/Hz 16-state PC-NLTCM with 8PSK and compare its performance against the 16-state turbo code presented in [4]. In order to make a fair comparison, we will use the same spread-interleaver technique used presented in that work and the same interleaver length $K_b = 10000$ bits, and therefore $K_s = 2500$ symbols with $k_b = 4$. Each output branch of each constituent encoder consists of one 8PSK constellation point, which produces a code rate of 2 bits/s/Hz. This is an interesting comparison since there hasn't been any published work that shows a turbo code with symbol interleaving that outperforms the code presented in [4] under same conditions. The code in [4] presents an effective free distance of $d_{\text{eff,free}} = 1.171573$.

We present a 16-state PC-NLTM that has $d_{\text{eff,free}} = 2$. For the design, we make the following observations. Since $k_b = 4$, there are 16 branches leaving each state with each of the 16 possible inputs. It is clear that parallel branches should be avoided, so the trellis structure is fully connected, *i.e.* there is one (and only one) branch connecting each of the 16 origin states with each of the 16 destination states. The design consists of assigning each branch and input symbol and an 8PSK constellation point. These assignments are constrained by the following conditions:

- Branches starting at a same state cannot be produced by the same input symbol.
- Branches merging to a same state cannot be produced by the same input symbol. This constraint avoids error events with input Hamming distance equal to 1 and can be satisfied by using recursive encoders.

Note that since the trellis is fully connected, any two branches leaving a same state at a certain trellis section will produce 16 error events with input Hamming distance equal to 2 in the following trellis section. In other words, there are 16 length-two error events starting at each of the 16 states, which have an input Hamming distance of 2. Thus the effective free distance of the code is upper bounded by these length-two error events. A first step in the design is to assign output labels to each branch so that the minimum distance produced by a length-two error event is maximized. Given the constraints stated above, there is no need to consider the input symbols at this stage. Table I shows the output branch label assignment. The constellation labeling for 8PSK used in this work is shown in Fig. 2. Each row represents the starting state (S_s) of the branch, and each column represents the ending state (S_e). This output labeling produces a minimum length-two error-event distance of 2, assuming a unit-norm constellation.

The next step is to search over all the possible input symbol assignments in order to avoid error events of length three or more that have input Hamming distance of two, and output distance of less than 2. From the conclusions drawn from

TABLE I

OUTPUT LABELS FOR 8PSK. THE ROWS INDICATE THE STARTING STATES S_s, AND THE COLUMNS THE ENDING STATES S_e. s_1/s_2 INDICATES THAT OUTPUT LABEL IS THE SAME FOR BOTH ENDING STATES.

$S_s : S_e$	0/8	1/9	2/10	3/11	4/12	5/13	6/14	7/15
0	0	1	2	3	4	5	6	7
1	4	5	6	7	0	1	2	3
2	0	3	2	5	4	7	6	1
3	4	7	6	1	0	3	2	5
4	0	1	2	3	4	5	6	7
5	4	5	6	7	0	1	2	3
6	0	3	2	5	4	7	6	1
7	4	7	6	1	0	3	2	5
8	2	3	4	5	6	7	0	1
9	6	7	0	1	2	3	4	5
10	2	1	4	3	6	5	0	7
11	6	5	0	7	2	1	4	3
12	2	3	4	5	6	7	0	1
13	6	7	0	1	2	3	4	5
14	2	1	4	3	6	5	0	7
15	6	5	0	7	2	1	4	3

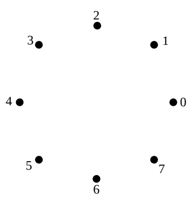

Fig. 2. Labeling for 8PSK.

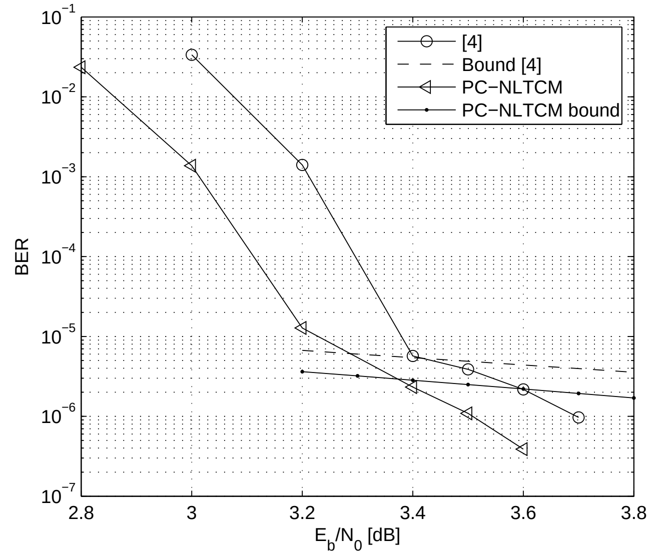

Fig. 3. BER vs. E_b/N_0 comparison, for 2 bits/s/Hz 16-state parallel concatenated codes with 8PSK.

the USIR analysis (Section III-A and Appendix I), this search must be constrained to recursive trellis structures. We searched over trellis structures of the form:

$$S = A \cdot S + B \cdot u \mod 2, \tag{21}$$

where $S = [s_1, s_2, s_3, s_4]^T$ represents the state, and $u = [u_1, u_2, u_3, u_4]$ represents the input symbol. The trellis structure selected is given by:

$$A = \begin{bmatrix} 0\,0\,0\,1 \\ 0\,1\,1\,1 \\ 1\,0\,1\,1 \\ 1\,0\,0\,0 \end{bmatrix}, B = \begin{bmatrix} 1\,0\,0\,0 \\ 0\,1\,0\,0 \\ 0\,0\,1\,0 \\ 0\,0\,0\,1 \end{bmatrix} \tag{22}$$

This trellis structure together with the output labeling of Table I produces a $d_{\text{eff,free}} = 2$.

Fig. 3 shows a performance comparison between the 2 bits/s/Hz 16-state turbo code for proposed in [4], and the PC-NLTCM presented in this work. The same symbol interleaver has been used for both codes. The interleaver length is $K_b = 10000$ bits, or $K_s = 2500$ symbols from the symbol interleaving perspective. It can be observed that the nonlinear code outperforms the linear code by a little less than 0.2 dB. At $BER = 10^{-5}$, the PC-NLTCM is within 0.5 dB from the constrained capacity 2.8 dB.

Fig. 3 also shows the uniform-interleaver BER bounds for each code. In order to plot the BER bound and not the SER bound, we assumed that any symbol error is equally likely, the symbol in error is equally likely to have any of the $2^{k_b} - 1$ possible values (leaving out the correct symbol), and therefore we used a correction factor on the error bound of

$$BER^{\text{bound}} \approx \frac{k_b \cdot 2^{k_b - 1}}{2^{k_b} - 1} SER^{\text{bound}}. \tag{23}$$

The reason why the BER bound is not tight in the error floor is that the interleaver design plays an important role in these high-rate applications, as shown in [4], and therefore an average interleaver would perform much worse than the carefully designed one used here. However, at the constituent code design stage, it gives a good prediction of which constituent code would perform better than the other.

It is worth mentioning that this is merely one example where constraining the design to a linear code with a mapper could be too restrictive, and directly assigning constellation points to each branch could produce a larger effective free distance and a better parallel concatenated code. General nonlinear turbo code design is a rich area for continued research.

V. CONCLUSIONS

Parallel concatenated nonlinear trellis codes can be beneficial for higher-order modulations. Although trellis coded modulation can achieve optimal performance using convolutional codes with a proper labeling, we showed with an example that for parallel concatenated trellis coded modulation using convolutional codes with labeling may be suboptimal under certain scenarios. As an example, we have designed a rate 2 bits/s/Hz 16-state parallel concatenated nonlinear trellis code for 8PSK, which outperforms the best previously reported linear turbo code with labeling by 0.2 dB over AWGN under same conditions. This code is within 0.5 dB away from capacity at a $BER = 10^{-5}$. Moreover, this improvement comes with the same decoding complexity as with convolutional codes as constituent codes.

To facilitate analysis of the new codes, an extension of Benedetto's uniform interleaver analysis for nonlinear constituent codes was derived. It was shown that the design criteria for linear codes can be generalized to nonlinear codes. In particular, we generalize the notion of effective free distance for nonlinear constituent codes, and conclude that this is an important parameter to maximize at the design stage.

REFERENCES

[1] Gottfried Ungerboeck. Channel coding with multilevel/phase signal. *IEEE Trans. on Info. Theory*, pages 55–67, Jan 1982.

[2] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara. Parallel concatenated trellis coded modulation. In *IEEE International Conference on Communications*, June 1996.

[3] D. Divsalar and F. Pollara. Turbo trellis coded modulation with iterative decoding for mobile satellite communications. In *International Mobile Satellite Conference*, June 1997.

[4] Christina Fragouli and Richard Wesel. Turbo-encoder design for symbol-interleaved parallel concatenated trellis-coded modulation. *IEEE Trans. on Info. Theory*, 47:873–890, Mar 2001.

[5] M. Griot, A.I. Vila Casado, and R.D. Wesel. Non-linear turbo codes for interleaver-division multiple access on the OR channel. In *GLOBECOM '06. IEEE Global Telecomm. Conf.*, 27 Nov. - 1 Dec. 2006.

[6] C. Berrou and A. Glavieux and P. Thitimajshima. Near shannon limit error-correcting coding and decoding: turbo-codes. In *Proc. ICC'93*, pages 873–890, May 1993.

[7] K.D. Fisher, J.M. Cioffi, W.L. Abbott, P.S. Bednarz, and C.M. Melas. An adaptive RAM-DFE for storage channels. *IEEE Trans. on Comm.*, 39:1559–1568, Nov 1991.

[8] S. Benedetto and G. Montorsi. Unveiling turbo codes: Some results on parallel concatenated coding schemes. 42, Mar 1996.

[9] E. Biglieri. High-Level Modulation and Coding for Nonlinear Satellite Channels. *IEEE Trans. on Comm.*, 32, May 1984.

[10] Y. J. Liu, I. Oka, and E. Biglieri. Error probability for digital transmission over non-linear channels with application to TCM. *IEEE Trans. on Info. Theory*, 36, Sep 1990.