

# Matched Information Rate Codes for Partial Response Channels

(submitted June, 2002; revised December 2004)

Aleksandar Kavčić, Xiao Ma and Nedeljko Varnica

The material in this manuscript was presented in part at the International Symposium on Information Theory, Lausanne, Switzerland, June-July, 2002.

The authors are with the Division of Engineering and Applied Sciences, Harvard University, Cambridge, MA.

This work was supported by the NSF under Grant Number CCR-0118701

### Abstract

In this paper we design capacity-approaching codes for partial response channels. The codes are constructed as concatenations of inner trellis codes and outer low-density parity-check (LDPC) codes. Unlike previous constructions of trellis codes for partial response channels, we disregard any algebraic properties (e.g., the minimum distance or the run-length limit) in our design of the trellis code. Our design is purely *probabilistic* in that we construct the inner trellis code to mimic the transition probabilities of a Markov process that achieves a high (capacity-approaching) information rate. Hence, we name it a *matched information rate* (MIR) design. We provide a set of 5 design rules for constructions of capacity-approaching MIR inner trellis codes. We optimize the outer LDPC code using density evolution tools specially modified to fit the superchannel consisting of the inner MIR trellis code concatenated with the partial response channel. Using this strategy, we design degree sequences of irregular LDPC codes whose noise tolerance thresholds are only fractions of a decibel away from the capacity. Examples of code constructions are shown for channels both with and without spectral nulls.

### Index Terms

channel capacity, concatenated codes, density evolution, low-density parity-check codes, Markov capacity, Markov processes, partial response channel, superchannel, trellis code

## I. INTRODUCTION

A partial response channel is a channel with intersymbol interference (ISI) whose inputs are constrained to a finite alphabet. This paper exposes a methodology for constructing codes that approach the capacity of such channels. Though the methods presented here could apply to general partial response channels (with arbitrary, but finite input alphabets and colored noise), our main topic of interest in this paper are channels whose input alphabets are binary, i.e.,  $\{0, 1\}$  or  $\{1, -1\}$ , with additive white Gaussian noise (AWGN).

The capacity of a power-constrained intersymbol interference channel whose input alphabet is not constrained to be finite, can be computed by the waterfilling theorem [1], [2]. A partial

response channel (with intersymbol interference, binary input alphabet and AWGN) can be represented as a finite-state machine. The definition of the channel capacity  $C$  of a partial response channel (or more precisely, a finite-state machine channel) can be found in [1], p. 109. Often authors refer to a different capacity, the i.u.d. capacity  $C_{\text{i.u.d.}}$ , which is defined as the information rate when the partial response channel inputs are independent and uniformly distributed (i.u.d.) random variables.

The computations of both the capacity  $C$  and the i.u.d. capacity  $C_{\text{i.u.d.}}$  of a partial response channel have been subjects of research for some time, resulting only very recently in arbitrarily tight numeric lower and upper bounds. In [3], Goldsmith and Varaiya formulate the information rate of a finite-state machine channel (without ISI). This formulation has been generalized to ISI channels by Sharma and Singh [4] who also computed a lower bound on the channel capacity of partial response channels. A particularly elegant Monte Carlo method for computing the information rate of a finite-state machine channel whose inputs are Markov processes was proposed independently by Arnold and Loeliger [5] and Pfister et al. [6]. This result can be used to compute the i.u.d. capacity  $C_{\text{i.u.d.}}$  and lower bounds on  $C$ . In [7], Kavčić proposed a Markov process optimization algorithm to tighten the lower bounds, and in [8] Vontobel and Arnold proposed a method to exploit the tight lower bounds to compute tight upper bounds. Another development worthy of mention is the computation of the *feedback* capacities of partial response channels by Yang and Kavčić [9], which is built on Tatikonda's formulation of the feedback capacity [10] of general channels with memory. A summary of the latest results on the computation of the channel capacities of partial response channels can be found in [11]. It should be noted that exact evaluations of the channel capacities are not known, but the methods summarized in [11] give bounds that are so tight that, for all practical purposes, we consider the capacities of partial response channels to be known.

Signal processing and coding for partial response channels have histories that are longer and richer (in terms of literature items) than the history of the channel capacity computation. Here we only mention the contributions that are directly relevant to this paper. In [12], Forney showed that the maximum likelihood sequence detector of symbol sequences transmitted over partial response channels is the trellis-based Viterbi algorithm. For a few special partial response channels (the dicode and the class-4 channels), the maximum likelihood sequence detector was also

formulated by Kobayashi [13]. Kobayashi's detector is essentially equivalent to Forney's trellis-based Viterbi detector. Another important development is the maximum a posteriori symbol detector first formulated by Chang and Hancock [14], but also known as the Baum-Welch algorithm [15], [16], [17] or the BCJR algorithm [18]. In [19], Wiberg et al. showed that both the Viterbi and the BCJR algorithm belong to the class of general min-sum and sum-product algorithms, respectively. In [20], Li et al. demonstrated that both the Viterbi and BCJR algorithms may be executed in a similar fashion utilizing survivor memory paths. All these algorithms may also be viewed as generalizations of the distributive law defined on trellises [21], [22], [23].

Error-correction and modulation codes for partial response channels are numerous and we will not attempt to classify them all. For the purposes of this paper, we take a simplistic approach. We classify the error-correction codes into pre-turbo codes and post-turbo codes, where the boundary is marked by the invention of turbo codes by Berrou et al. [24].

Pre-turbo error-correction codes for partial response channels were dominated by trellis codes. General design rules for constructing trellis codes with large minimum Euclidean distances were formulated by Ungerboeck [25]. Trellis codes for partial response channels can be found in articles by Wolf and Ungerboeck [26], Calderbank et al. [27] and Immink [28]. In these papers, the codes were constructed on trellises (typically as cosets of convolutional codes) to achieve large minimum Euclidean distances between the noiseless outputs of the considered partial response channels. A different strategy was used by Karabed and Siegel [29] who proposed matched spectral null (MSN) trellis codes. These are trellis codes whose binary codeword sequences have nulls in specific frequency locations. MSN codes proved very good for channels that have spectral nulls (i.e., most magnetic and optical recording channels), where the spectral null of the code should match the spectral null of the channel.

The advent of turbo codes [24] has sparked research efforts to construct iteratively decodable codes for partial response channels. A typical scenario is to use turbo equalization [30], which means that the BCJR algorithm [18] is used to perform the maximum a posteriori symbol detection on the channel, while other iterative decoding algorithms are used as constituent decoders in iterative decoding schemes. MacKay's rediscovery of low-density parity-check (LDPC) codes [31], which were originally invented by Gallager [32], spurred research on many other iteratively decodable codes for partial response channels. Irregular LDPC codes were in-

roduced by Luby et al. [33], who also proposed a method to analyze the asymptotic performance of these codes over erasure channels when the block lengths approach  $\infty$ . The analysis was soon thereafter expanded to various other memoryless channels by Richardson and Urbanke [34], who coined the term *density evolution* for the analysis tool. The tool was used for the computation of noise tolerance thresholds [34], and the optimization of irregular code degree sequences [35], that ultimately lead to the construction of codes that approach the capacities of memoryless channels [36]. In [37], Kavčić et al. modified the density evolution tool to fit the partial response channels and computed noise tolerance thresholds of random LDPC codes over partial response channels. They also proved that the thresholds of these codes can at most achieve rates that approach the i.u.d. capacity  $C_{\text{i.u.d.}}$ , but not higher. This was supported by evidence of code constructions that approach  $C_{\text{i.u.d.}}$  very closely [38].

This paper addresses the construction of codes that achieve rates higher than the i.u.d. capacity  $C_{\text{i.u.d.}}$ . The proposed strategy combines algorithms for computing channel capacities, methods for constructing trellis codes and tools for designing iteratively decodable LDPC codes. In short, the approach is to 1) compute a (maximized) channel information rate achievable by an input Markov process (chain) of reasonably low complexity, 2) construct an inner trellis code that essentially mimics this Markov process and therefore has the potential to achieve a comparable information rate over the channel of interest, and 3) construct an outer LDPC code that ensures that the information rate of the inner trellis code is approached. The strategy is therefore to compute an achievable information rate, and then construct a concatenation of two codes whose overall code rate matches the computed information rate. Hence the name *matched information rate* (MIR) code.

The paper is organized into 5 sections. Sections II and III, are tutorial-like expositions of the necessary channel capacity and trellis code terminologies, respectively. In Sections IV, we give the design rules for constructing the inner matched information rate trellis codes, while in Section V we design outer LDPC codes.

## II. THE CHANNEL MODEL AND CAPACITY DEFINITIONS

### A. The channel model

Let  $t \in \mathbb{Z}$  denote a discrete-time variable. Denote the channel input as a sequence of random

variables  $X_t$ , and the channel output as a sequence of random variables  $Y_t$ . The channel law that links the channel input  $X_t$  and the channel output  $Y_t$  is

$$Y_t = \sum_{j=0}^J h_j X_{t-j} + W_t, \quad (1)$$

where the integer  $J > 0$  is the intersymbol interference (ISI) length,  $h_0, h_1, \dots, h_J \in \mathbb{R}$  are the channel coefficients, and  $W_t$  is the noise process. In this paper, we will assume that the noise process  $W_t$  is white and Gaussian, with mean 0 and variance  $\sigma^2$ . We will also assume that the realizations  $x_t$  of the random variables  $X_t$  take values in a finite alphabet  $\mathcal{X}$ . The methods described in this paper could be applied to any finite alphabet  $\mathcal{X}$ . However, for the purpose of a clear presentation and practical relevance in many binary recording and communication channels, we assume that  $\mathcal{X}$  is the bipolar input alphabet  $\mathcal{X} = \{-1, 1\}$ .

We refer to the channel in (1) as the *partial response* channel. The channel may be represented by a partial response polynomial

$$h(D) = \sum_{j=0}^J h_j D^j. \quad (2)$$

In this paper, we will often draw examples using the *dicode* channel

$$Y_t = X_t - X_{t-1} + W_t, \quad (3)$$

whose partial response polynomial is  $h(D) = 1 - D$ . As a general rule, we define the signal-to-noise ratio (in dB) as

$$SNR = 10 \log_{10} \frac{\sum_{j=0}^J h_j^2}{\sigma^2}, \quad (4)$$

where  $\sigma^2$  is the variance of the AWGN.

We will interchangeably use the input alphabet  $\mathcal{X} = \{-1, 1\}$  and the input alphabet  $\mathcal{A} = \{0, 1\}$ . Thereby, we will use the standard conversion

$$X_t = 1 - 2A_t, \quad (5)$$

where  $A_t$  is a discrete-time random process with binary realizations  $a_t \in \mathcal{A} = \{0, 1\}$ . To distinguish between the two types of binary input alphabets, we refer to  $\mathcal{X} = \{-1, 1\}$  as the *bipolar* input alphabet and to  $\mathcal{A} = \{0, 1\}$  as the *binary* input alphabet.

### B. The binary-input channel capacity

Denote by  $X_1^N$  the sequence  $X_1^N = (X_1, X_2, \dots, X_N)$ . The sequence  $Y_1^N$  is similarly defined. Partial response channels are indecomposable finite-state machine channels (see [1], p. 105, for a precise definition of an indecomposable finite-state machine channel), for which the initial state does not affect the information rate. Thus, the capacity is

$$C = \lim_{N \rightarrow \infty} \max_{P_{X_1^N}(x_1^N)} \frac{1}{N} I(X_1^N; Y_1^N), \quad (6)$$

where  $I(X_1^N; Y_1^N)/N$  is the mutual information rate, and the maximization is taken over all valid probability mass functions of the random vector  $X_1^N$ .

Another capacity of interest is the i.u.d. capacity, defined as

$$C_{\text{i.u.d.}} = \lim_{N \rightarrow \infty} \frac{1}{N} \cdot I(X_1^N; Y_1^N) \Big|_{P_{X_1^N}(x_1^N) = 2^{-N}}, \quad (7)$$

i.e., it is the information rate when the input sequence  $X_1^N$  consists of random variables  $X_t$  that are independent and uniformly distributed (i.u.d.). In Subsection III-D, we establish that  $C_{\text{i.u.d.}}$  is the *capacity of random linear (coset) codes*.

### C. Trellis representations

The standard representation of the channel in (1) is through the notion of a *channel trellis* [12]. At time  $t$ , the channel trellis has  $2^L$  states (where  $L \geq J$ ), indexed by  $\{0, 1, 2, \dots, 2^L - 1\}$ . Exactly  $2^n$  branches emanate from each state of the channel trellis, where  $n \geq 1$ . A branch at time  $t$  is determined by a 4-tuple  $b_t = (s_{t-1}, u_t, v_t, s_t)$ . The first and the last symbol of the 4-tuple,  $s_{t-1} \in \{0, 1, 2, \dots, 2^L - 1\}$  and  $s_t \in \{0, 1, 2, \dots, 2^L - 1\}$  are the starting state (at time  $t - 1$ ) and the ending state (at time  $t$ ), respectively. The symbol  $u_t$  is an  $n$ -tuple of input symbols, i.e.,  $u_t \in \mathcal{A}^n = \{0, 1\}^n$  in binary notation or  $u_t \in \mathcal{X}^n = \{-1, 1\}^n$  in bipolar notation. The symbol  $v_t$  denotes an  $n$ -tuple of real-valued *noiseless* channel outputs, i.e.,  $v_t \in \mathbb{R}^n$ . (The noiseless channel outputs can be determined from the channel model (1) by setting  $W_t = 0$ .) Every branch is uniquely determined by its starting state  $s_{t-1}$  and its input  $n$ -tuple  $u_t$ . That is,  $v_t$  and  $s_t$  are (deterministic) functions of  $s_{t-1}$  and  $u_t$ . The random 4-tuple  $B_t = (S_{t-1}, U_t, V_t, S_t)$

stands for a random branch, whose realizations are the 4-tuples  $b_t$ . Figure 1-a shows the channel trellis of the dicode channel in (3).

We distinguish the following 3 trellis types (examples are given in Figures 1 and 2).

- **A minimal** channel trellis is a channel trellis whose trellis section has the smallest possible number of states (i.e.,  $2^J$  states) and whose trellis section corresponds to a single input and a single output (i.e.,  $n = 1$ ). The minimal channel trellis for the dicode channel in (3) is shown in Figure 1-a.
- **An original** channel trellis is a channel trellis (not necessarily minimal, i.e., the number of states is  $2^L \geq 2^J$ ) whose trellis section corresponds to a single input and a single output (i.e.,  $n = 1$ ). The minimal channel trellis in Figure 1-a is automatically an original channel trellis. Another original channel trellis for the dicode channel in (3) is shown in Figure 1-b.
- **An extended** channel trellis is a channel trellis whose trellis section corresponds to  $n$  channel inputs and  $n$  channel outputs, where  $n > 1$ . Thereby, the number of states is  $2^L \geq 2^J$ . In Figure 2 we give two examples of extended channel trellises for the dicode channel in (3).

#### D. The Markov channel capacity

Consider a branch in the trellis section of a channel trellis. Let  $i \in \{0, 1, 2, \dots, 2^L - 1\}$  denote the starting state of the branch in the trellis section, and let  $j \in \{0, 1, 2, \dots, 2^L - 1\}$  be the ending state of the branch. If the trellis is an extended channel trellis, then there may be several branches connecting a pair of states  $i$  and  $j$ . Let  $\mathcal{L}$  be the number of distinct branches that connect states  $i$  and  $j$ . The  $\mathcal{L}$  different branches are denoted by  $b_{ij}^{(\ell)}$ , where  $0 \leq \ell \leq \mathcal{L} - 1$ . Using our previous 4-tuple notation, we may express the branch  $b_{ij}^{(\ell)}$  as  $b_{ij}^{(\ell)} = (i, u_{ij}^{(\ell)}, v_{ij}^{(\ell)}, j)$ , where  $s_{t-1} = i$  is the starting state,  $s_t = j$  is the ending state,  $u_{ij}^{(\ell)} \in \mathcal{A}^n$  is the binary channel input  $n$ -tuple (or  $u_{ij}^{(\ell)} \in \mathcal{X}^n$  in bipolar notation) and  $v_{ij}^{(\ell)} \in \mathbb{R}^n$  is the real-valued noiseless channel output  $n$ -tuple.

To every channel trellis we may assign a Markov source. The source is defined by assigning a probability  $P_{ij}^{(\ell)}$  to every branch  $b_{ij}^{(\ell)}$  of the trellis section. Given that the channel state at time  $t - 1$  is  $S_{t-1} = i$ , then  $P_{ij}^{(\ell)}$  stands for the conditional probability that the branch is  $b_{ij}^{(\ell)}$ , i.e.,

$$P_{ij}^{(\ell)} = \Pr \left( B_t = b_{ij}^{(\ell)} \mid S_{t-1} = i \right). \quad (8)$$



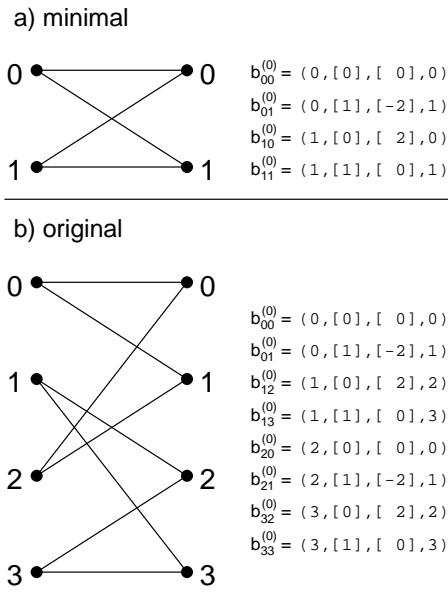


Fig. 1. The minimal and an original channel trellis for the dicode  $(1-D)$  channel. **a)** The minimal channel trellis with  $2^J = 2$  states and  $n = 1$  input (and output) symbol per trellis section. **b)** An original trellis with  $2^L = 4$  states and  $n = 1$  input (and output) symbols per trellis section.

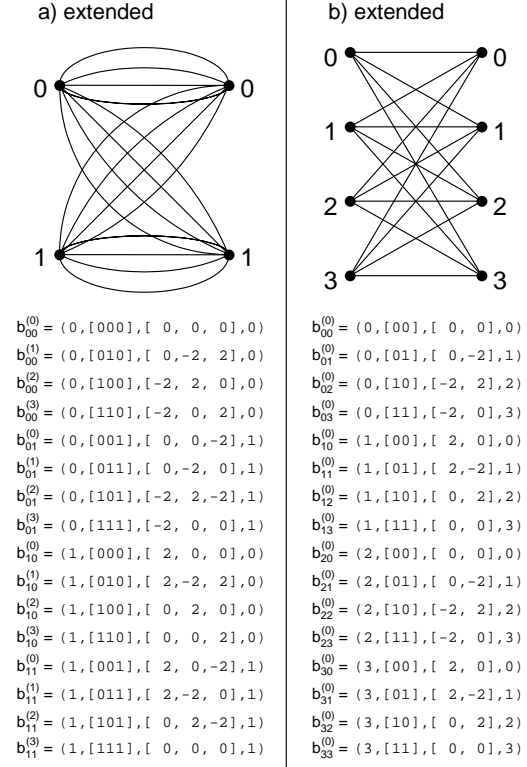


Fig. 2. Two examples of extended channel trellises for the dicode  $(1-D)$  channel. **a)** An extended trellis with  $2^J = 2$  states and  $n = 3$  inputs (and outputs) per trellis section. This trellis is obtained by concatenating  $n = 3$  sections of the minimal trellis in Figure 1-a, hence we call it the 3rd order extension of the minimal trellis. **b)** An extended trellis with  $2^L = 4$  states and  $n = 2$  inputs (and outputs) per trellis section. This trellis is obtained by concatenating  $n = 2$  sections of the original trellis in Figure 1-b, hence we call it the 2nd order extension of the original trellis in Figure 1-b.

Denote a trellis section by  $\tau$ . That is,  $\tau$  is the set of all triples  $(i, \ell, j)$  for which a branch  $b_{ij}^{(\ell)}$  exists in the trellis section. We say that a branch  $b_{ij}^{(\ell)}$  is a valid branch in the channel trellis  $\tau$  if  $(i, \ell, j) \in \tau$ . The probabilities  $P_{ij}^{(\ell)}$  are the transition probabilities of an input Markov process defined on the trellis  $\tau$ . Thereby, we have for any  $i \in \{0, 1, 2, \dots, 2^L - 1\}$

$$\sum_{j, \ell: (i, \ell, j) \in \tau} P_{ij}^{(\ell)} = 1. \quad (9)$$

Denote the invariant probability of state  $i$  by

$$\mu_i = \Pr(S_t = i), \quad (10)$$

whereby  $\mu_j = \sum_{i, \ell: (i, \ell, j) \in \tau} \mu_i P_{ij}^{(\ell)}$ . Denote by  $\mathcal{P}(\tau)$  all Markov transition probabilities  $P_{ij}^{(\ell)}$  for all branches  $(i, \ell, j) \in \tau$ , i.e.,

$$\mathcal{P}(\tau) = \left\{ P_{ij}^{(\ell)} \mid (i, \ell, j) \in \tau \right\}. \quad (11)$$

For this Markov source, the Markov channel information rate is defined as

$$\mathcal{I}_{\mathcal{P}(\tau)} = \lim_{N \rightarrow \infty} \frac{1}{nN} \mathbb{I}(B_1^N; Y_1^{nN}) \Big|_{P_{ij}^{(\ell)} \in \mathcal{P}(\tau)}, \quad (12)$$

where the branch sequence  $B_1^N$  is a random sequence uniquely specified by the Markov transition probabilities  $\mathcal{P}(\tau)$  on a trellis  $\tau$ . Note that in the definition (12), we used the branch sequence  $B_1^N$  instead of the channel input sequence  $X_1^{nN}$ . This is possible because there is a 1-to-1 mapping between input sequences and branch sequences if the initial state  $s_0$  is known. Since the partial response channel is an indecomposable finite-state machine, the choice of the initial state will not change the limit in (12).

We proceed to define the Markov channel capacity for a trellis  $\tau$  as

$$C_\tau = \max_{\mathcal{P}(\tau)} \mathcal{I}_{\mathcal{P}(\tau)}, \quad (13)$$

where the maximization in (13) is conducted over all Markov processes  $\mathcal{P}(\tau)$  supported on the trellis  $\tau$ . Clearly  $C_\tau \leq C$ .

Denote by  $\tau_o(L)$  an *original* channel trellis with  $2^L$  states ( $2^L \geq 2^J$ ). Denote the Markov channel capacity defined on this trellis by  $C_{\tau_o(L)}$ , or simply  $C_L$ . Then an alternative expression for the channel capacity is [39]

$$C = \lim_{L \rightarrow \infty} C_{\tau_o(L)} = \lim_{L \rightarrow \infty} C_L. \quad (14)$$

### E. Markov channel capacity computation

The goal in computing the Markov channel capacity over a trellis  $\tau$  is to perform the maximization in (13) over all Markov probability mass functions  $\mathcal{P}(\tau)$  supported on the trellis  $\tau$ . Table I gives an algorithm to numerically estimate<sup>1</sup>  $C_\tau$  and the capacity-achieving distribution  $\mathcal{P}(\tau)$ . We call the capacity computed by the algorithm in Table I the *numerical Markov capacity* and denote it by  $\hat{C}_\tau$ .

In Figure 3, we show the numerical Markov capacities  $\hat{C}_\tau$  computed by the algorithm in Table I for the decode channel trellises given in Figures 1 and 2. Also shown in Figure 3 is the numerical capacity  $\hat{C}_6$  (which is the tightest known lower bound on the channel capacity  $C$ ) and the best known upper bound on the channel capacity  $C$  (the minimum of the water-filling bound [1], [2], the feedback capacity [9] and the Vontobel-Arnold bound [8]). For comparison, the i.u.d. capacity  $C_{\text{i.u.d.}}$  (as computed in [5], [6]) is also shown in Figure 3.

## III. TRELLIS CODES, SUPERCHANNELS AND THEIR INFORMATION RATES

### A. Trellis codes

Trellises may be used to represent many different types of codes [40]. In this paper, we consider only codes that can be represented by a regular time-invariant trellis. Though trellis codes can in general be non-binary, for the purposes of this paper, a trellis code is a *binary* code that maps  $k$  binary input symbols to  $n$  binary output symbols ( $k \leq n$ ), with the following regular structure. A trellis section of a trellis code has  $\mathcal{S}$  states, indexed by  $\{0, 1, \dots, \mathcal{S} - 1\}$ . There are exactly  $2^k$  branches emanating from each state of the trellis code, i.e., one branch for every binary input  $k$ -tuple. A branch of a trellis code is described by a 4-tuple  $b_t = (s_{t-1}, u_t, v_t, s_t)$ , where  $s_{t-1} \in \{0, 1, \dots, \mathcal{S} - 1\}$  is the starting state,  $s_t \in \{0, 1, \dots, \mathcal{S} - 1\}$  is the ending state,  $u_t$  is the input  $k$ -tuple  $u_t \in \mathcal{A}^k$ , and  $v_t$  is the output  $n$ -tuple  $v_t \in \mathcal{A}^n$  in binary notation (or  $v_t \in \mathcal{X}^n$  in bipolar notation). The code rate is  $r = k/n$ .

Figure 4 shows two known trellis codes, the biphasic code of rate  $1/2$  (which is actually a coset of a simple repetition block code), and a Wolf-Ungerboeck code [26] of rate  $2/3$  (which is

<sup>1</sup>The algorithm in Table I is a modified version of the algorithm in [11], but adapted for *extended* trellises.

**Algorithm 1** Iterative optimization of Markov chain transition probabilities.

**Initialization** Pick a channel trellis  $\tau$  and an arbitrary probability mass function  $P_{ij}^{(\ell)}$  defined over the trellis  $\tau$  that satisfies the following two constraints: 1)  $0 < P_{ij}^{(\ell)} \leq 1$  if  $(i, \ell, j) \in \tau$ ; otherwise  $P_{ij}^{(\ell)} = 0$  and 2) for any  $i$ , require that condition (9) be satisfied.

**Repeat** until convergence

- 1) For  $N$  large (say,  $N > 10^6$ ), generate a realization of a sequence of  $N$  trellis branches  $b_1^N$  according to the Markov probabilities  $P_{ij}^{(\ell)}$ . Determine the channel input sequence  $x_1^N$  that corresponds to the branch sequence  $b_1^N$ . Pass the channel input sequence through the partial response channel to get a realization of the channel output  $y_1^N$ .
- 2) Run the forward-backward sum-product (Baum-Welch, BCJR) algorithm and for all  $1 \leq t \leq N$  compute the a posteriori probabilities  $R_{ij}^{(\ell)}(t, y_1^N) = \Pr(B_t = b_{ij}^{(\ell)} | Y_1^N = y_1^N)$  and  $R_i(t, y_1^N) = \Pr(S_t = i | Y_1^N = y_1^N)$ .

- 3) Compute the estimate of the expectation term  $\hat{T}_{ij}^{(\ell)} = \frac{1}{N} \sum_{t=1}^N \log \frac{R_{ij}^{(\ell)}(t, y_1^N) \frac{R_{ij}^{(\ell)}(t, y_1^N)}{\mu_i P_{ij}^{(\ell)}}}{R_i(t-1, y_1^N) \frac{R_i(t-1, y_1^N)}{\mu_i}}$ .

- 4) Compute the estimate of the noisy adjacency matrix, whose entry on the intersection of the  $i$ -th row and the  $j$ -th column is

$$\hat{A}_{ij} = \begin{cases} \sum_{\ell: (i, \ell, j) \in \tau} 2^{\hat{T}_{ij}^{(\ell)}} & \text{if states } i \text{ and } j \text{ are connected by at least one branch} \\ 0 & \text{otherwise} \end{cases},$$

and find its maximal eigenvalue  $\hat{W}_{max}$  and the corresponding right eigenvector  $[\hat{e}_1, \hat{e}_2, \dots, \hat{e}_M]^T$ .

- 5) For  $(i, \ell, j) \in \tau$ , compute the new transition probabilities as  $P_{ij}^{(\ell)} = \frac{\hat{e}_j}{\hat{e}_i} \cdot \frac{2^{\hat{T}_{ij}^{(\ell)}}}{\hat{W}_{max}}$  and go back to 1).

**Terminate** the algorithm upon convergence and set  $\hat{C}_\tau = \frac{1}{n} \log \hat{W}_{max}$ . The input distribution  $\mathcal{P}(\tau)$  that achieves  $\hat{C}_\tau$  is given by the collection of probabilities  $P_{ij}^{(\ell)}$ .

TABLE I

derived from a coset of a linear convolutional code).

### B. Superchannels

In this paper, we refer to a superchannel as a concatenation of a trellis code and a partial response channel. Since both the trellis code and the partial response channel can be described by their respective trellises, their concatenation (the superchannel) can also be described by

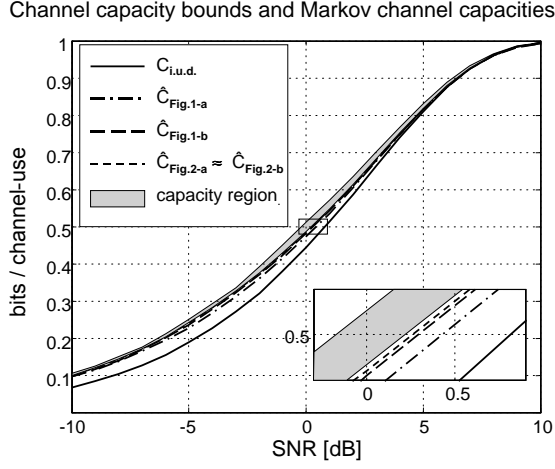
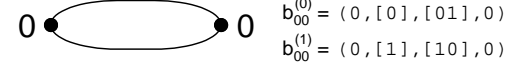


Fig. 3. Different Markov capacities for the dicode  $(1 - D)$  channel:  $\hat{C}_{\text{Fig.1-a}}$  is the Markov capacity of the minimal channel trellis in Figure 1-a computed by the algorithm in Table I.  $\hat{C}_{\text{Fig.1-b}}$ ,  $\hat{C}_{\text{Fig.2-a}}$ ,  $\hat{C}_{\text{Fig.2-b}}$  are similarly computed for their respective channel trellises.  $C_{\text{i.u.d.}}$  is the i.u.d. capacity. The shaded region is the region between the tightest known lower bound  $\hat{C}_6$  and the tightest known upper bound on the channel capacity.

a) biphase code (rate 1/2)



b) Wolf-Ungerboeck code (rate 2/3)

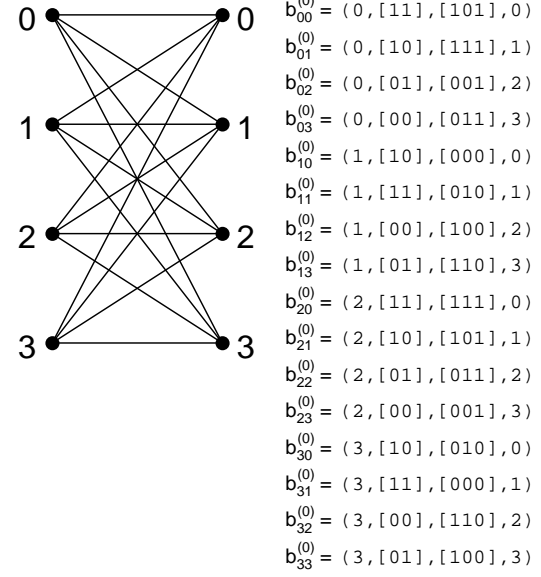


Fig. 4. Two trellis codes. **a)** The biphase block code represented here as a trivial trellis code. **b)** A Wolf-Ungerboeck trellis code of rate  $r = 2/3$ .

a joint trellis. A trellis section of a superchannel has  $\mathcal{S}$  states, where the number of states in the superchannel, may be greater than the number of states in the trellis of the constituent trellis code. Every state of the superchannel trellis has exactly  $2^k$  branches emanating from the state. A branch of the superchannel trellis is described by a 4-tuple  $b_t = (s_{t-1}, u_t, v_t, s_t)$ , where  $s_{t-1} \in \{0, 1, \dots, \mathcal{S} - 1\}$  is the starting state,  $s_t \in \{0, 1, \dots, \mathcal{S} - 1\}$  is the ending state,  $u_t$  is the input  $k$ -tuple  $u_t \in \mathcal{A}^k$ , and  $v_t$  is the noiseless channel output  $n$ -tuple  $v_t \in \mathbb{R}^n$ . Figure 5 gives examples of two superchannel trellises obtained by concatenating the trellis codes in Figure 4 with the dicode partial response channel in Figure 1-a.

### C. Superchannel information rates

Typically, we will assume that the inputs to the trellis codes, i.e., the inputs to the superchannels, are i.u.d. equally likely symbols. This means that the conditional probability  $P_{ij}^{(\ell)} =$

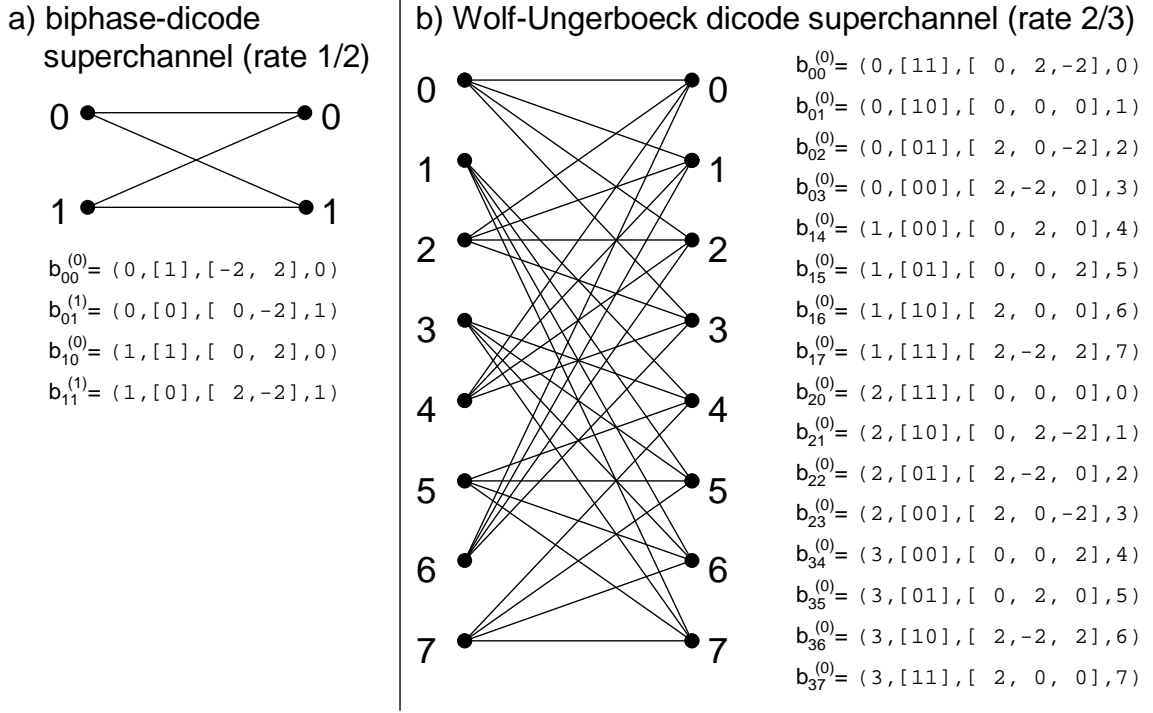


Fig. 5. Two superchannel trellises obtained by concatenating the trellis codes from Figure 4 and the dicode channel in Figure 1-a. **a)** The biphase dicode superchannel trellis (rate  $r = 1/2$ ). **b)** A Wolf-Ungerboeck dicode superchannel trellis (rate  $r = 2/3$ ).

$\Pr(B_t = b_{ij}^{(\ell)} | S_t = i)$  of each branch in the superchannel trellis is equal to  $2^{-k}$ . Under this i.u.d. assumption, we define the *superchannel information rate*  $\mathcal{I}_S$  as the information rate between the superchannel input sequence and the superchannel output sequence, or equivalently as the information rate between the superchannel trellis branch sequence and the superchannel output sequence

$$\mathcal{I}_S = \lim_{N \rightarrow \infty} \frac{1}{nN} \mathcal{I}(B_1^N; Y_1^{nN}) \Big|_{P_{ij}^{(\ell)} = 2^{-k}} = \lim_{N \rightarrow \infty} \frac{1}{nN} \mathcal{I}(U_1^N; Y_1^{nN}) \Big|_{P_{U_1^N}(u_1^N) = 2^{-kN}}. \quad (15)$$

The definition in (15) does not require an optimization, but rather just an evaluation of the information rate for i.u.d. inputs. This evaluation of the superchannel information rate  $\mathcal{I}_S$  can be achieved using the method proposed by Arnold and Loeliger [5], and Pfister et al. [6]. The superchannel information rate represents the maximal rate achievable with a random outer binary block code used on the superchannel (see Section III-D), or by a random LDPC code used on the superchannel ([37], Proposition 1).

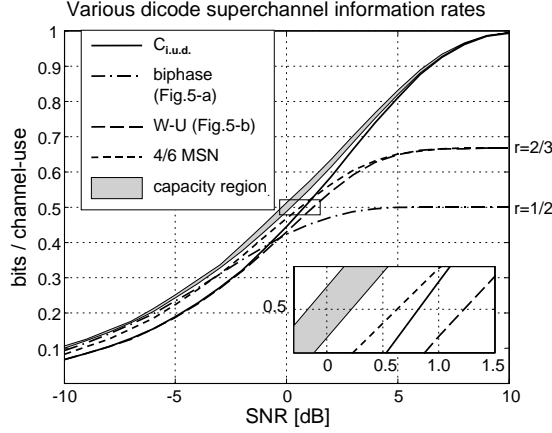


Fig. 6. Different superchannel information rates for the dicode  $(1-D)$  channel: the rate-1/2 biphase superchannel (Fig. 5-a), the rate 2/3 Wolf-Ungerboeck superchannel (Fig. 5-b), and the rate-4/6 Karaded-Siegel matched spectral null (MSN) superchannel.  $C_{i.u.d.}$  is the i.u.d. capacity. The shaded region is the region between the tightest known lower bound  $\hat{C}_6$  and the tightest known upper bound on the channel capacity.

Figure 6 shows the superchannel information rates for the dicode superchannels given in Figure 5 and the superchannel obtained by concatenating a rate-4/6 matched spectral null (MSN) code and a dicode channel [29]. Note that the maximal rate achievable on a superchannel trellis at high SNRs is  $k/n$ , which is the rate of the constituent trellis code. For comparison, Figure 6 also shows the numerical channel capacity  $\hat{C}_6$  of the dicode channel, the upper bound on the channel capacity, and the i.u.d. capacity  $C_{i.u.d.}$ .

From Figure 6, we see that at very low code rates the biphase superchannel rate  $\mathcal{I}_S$  approximately equals the channel capacity. At rate  $r = 1/2$ , the rate-4/6 MSN superchannel information rate is closer to the channel capacity than the Wolf-Ungerboeck rate 2/3 superchannel information rate, but there is still a gap of about 0.4dB between the numerical capacity  $\hat{C}_6$  and the MSN superchannel information rate. Our task is to formulate a strategy to close this gap. The goal is not to outperform the MSN codes on the dicode channel per se, but rather to find a general superchannel construction method that will deliver superchannels that achieve the channel capacity for *any* channel (not just channels with spectral nulls).

#### D. Coding theorems for superchannels

Viewing the concatenation of the encoder/decoder and the channel as a superchannel was originally proposed by Forney [41]. We emphasize two subtle (but essential for practical con-

structions) differences from Forney's point of view:

- 1) Forney [41] tried to construct a superchannel to maximize the error exponent and hence the capacity of the superchannel. Here, we are interested only in the superchannel i.u.d. information rate  $\mathcal{I}_S$ .
- 2) It is difficult and tedious to calculate error exponents for most superchannels. So, Forney [41] proposed an analysis of the concatenation error exponent assuming the inner code is the best code that satisfies the coding theorem. Our inner code is much simpler – it can be represented by a low-complexity time-invariant trellis whose superchannel i.u.d. information rate  $\mathcal{I}_S$  can be evaluated using the Monte Carlo simulation in [5], [6].

**Theorem 1.** [achievability of  $\mathcal{I}_S$  with a linear (coset) code] The i.u.d. superchannel information rate  $\mathcal{I}_S$  in Equation (15) can be achieved by an outer linear (coset) code.  $\square$

The proof may be obtained by modifying the treatment in [1], Section 5.9, so we omit the details.

Theorem 1 suggests that we could use a concatenation scheme to surpass the i.u.d. capacity  $C_{\text{i.u.d.}}$  of the partial response channel if we construct a superchannel whose i.u.d. rate is  $\mathcal{I}_S > C_{\text{i.u.d.}}$ . Consider the block diagram shown in Fig. 7. The binary vector  $D_1^M$  first enters a linear encoder represented by an  $M \times kN$  generator matrix  $\mathbf{G}$ . The output from this encoder is a binary vector of length  $kN$ . Before the codeword enters the trellis encoder (represented by ‘T’ in Figure 7), a coset-defining binary vector  $\gamma_1^{kN}$  is modulo-2-added on. It should be pointed out that different choices of the vector  $\gamma_1^{kN}$  may cause different error rates since the channel has memory. Although we cannot claim that the achievable rate of the best linear (coset) code is upper-bounded by  $\mathcal{I}_S$ , we have the following “random converse” theorem for outer codes of rate  $r_{\text{out}}$ .

**Theorem 2.** [random linear (coset) code converse] If  $r_{\text{out}} \cdot k/n > \mathcal{I}_S$ , there exists a positive real number  $\delta$  such that for any given  $M$  by  $kN$  generator matrix  $\mathbf{G}$  (of a linear code) with  $M/(kN) \geq r_{\text{out}}$  and any given detection/decoding method, the *average* error rate over all the cosets of  $\mathbf{G}$  is greater than (or equal to)  $\delta$ .  $\square$

The proof may be obtained by modifying standard converse-theorem arguments (see Theorem 4.6.2 in [1]), and the Fano inequality (Theorem 4.3.2 in Gallager [1]). These are by now



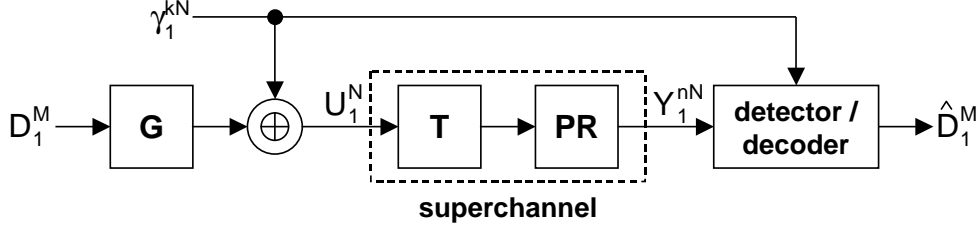


Fig. 7. A concatenation scheme for partial response channels. Note that  $U_t \in \mathcal{A}^k$ , i.e.,  $U_1^N \in \mathcal{A}^{kN}$ .

accepted as standard tools in information theory, so we omit the details.

Theorem 2 exposes the difficulties in finding a linear (coset) code that surpass the superchannel i.u.d. information rate  $\mathcal{I}_S$ . Namely, a randomly chosen linear (coset) code will not achieve a rate higher than  $\mathcal{I}_S$ . Theorem 2 also reveals the importance of the role of the inner trellis code. To see this role, note that any channel can be regarded as a trivial superchannel whose code rate is  $k/n = 1$  and whose i.u.d. information rate is  $\mathcal{I}_S = C_{\text{i.u.d.}}$ . Then Theorem 2 directly implies that without the inner code, the average rate achievable by a linear (coset) code is upper-bounded by the i.u.d. channel capacity  $C_{\text{i.u.d.}}$ .

To surpass  $C_{\text{i.u.d.}}$ , we separate the complex code-construction problem into two smaller problems: designing an inner (generally nonlinear) trellis code, and an outer linear (coset) code. As we will demonstrate, both these design problems are tractable. Thereby, to achieve the capacity  $C$ , we must construct an inner trellis code whose superchannel i.u.d. rate  $\mathcal{I}_S$  satisfies  $C \approx \mathcal{I}_S > C_{\text{i.u.d.}}$  at an SNR of interest. Without a carefully constructed inner code, available linear code optimization methods (such as the degree sequence optimization [35]) will be ineffective in approaching the channel capacity  $C$ .

#### IV. MATCHED INFORMATION RATE TRELLIS CODES

Our aim is to design a trellis code for a partial response channel, such that the i.u.d. information rate  $\mathcal{I}_S$  of the resulting superchannel is as close as possible to the (numerical) channel capacity. This is a very different design methodology from the trellis codes designed in [26], [27], [28] where the design criterion was to maximize the minimum Euclidean distance, or in [29] where the design criterion was to match the spectral null of the channel. Unlike matched

spectral null (MSN) codes, our matched information rate (MIR) design strategy applies to any channel; and not just to channels with special spectral properties.

An interesting feature of the MIR design methodology is that the trellis code is constructed for a specific target code rate  $r$ . Consider the dicode channel whose numerical Markov channel capacities are shown in Figure 3. Suppose our target code rate is  $r = 1/2$ . The numerical channel capacity  $\hat{C}_6$  equals  $1/2$  at  $SNR = 0.22\text{dB}$ , i.e.,  $\hat{C}_6(0.22\text{dB}) = 1/2$ . We shall use this example throughout the paper to illustrate the design method.

The general strategy is as follows. We will first choose an *extended* channel trellis  $\tau$ , and use the algorithm in Table I to find the optimal Markov branching probabilities for this trellis. We will then construct a superchannel to mimic the optimal Markov process on the trellis  $\tau$ . This will be achieved by splitting every state  $i$  of the extended channel trellis  $\tau$  into  $K_i$  states of the superchannel trellis ( $0 \leq K_i \in \mathbb{Z}$ ), and splitting every branch  $b_{ij}^{(\ell)}$  of the extended channel trellis  $\tau$  into  $n_{ij}^{(\ell)}$  branches of the superchannel trellis ( $0 \leq n_{ij}^{(\ell)} \in \mathbb{Z}$ ).

We next specify the design rules to construct matched information rate superchannels. Many of these rules are in fact ad hoc. They are derived to formalize the design methodology, but we make no claim regarding their optimality. We adopt these rules because they satisfy our intuition and because they proved to deliver good codes.

#### A. Choosing the extended channel trellis and the superchannel code rate

We will construct a superchannel trellis with  $n$  output symbols per every  $k$  binary input symbols. Our first task is to pick  $k$  and  $n$ . Let  $r$  be the target code rate of the code we wish to construct. We pick  $k$  and  $n$  according to the following rule.

**Rule 1.** The rate  $r_{\text{in}}$  of the trellis code should satisfy the constraint

$$r < r_{\text{in}} = \frac{k}{n} \leq \frac{1}{n} \min_{(i,\ell,j) \in \tau} \left[ \log \frac{1}{P_{ij}^{(\ell)}} \right] \quad (16)$$

where  $P_{ij}^{(\ell)}$ 's are the optimized transition probabilities of an  $n$ -th order extended channel trellis.

■

Not all extended channel trellises will satisfy Rule 1. The task is to find the smallest possible positive integers  $k$  and  $n$  such that an  $n$ -th order extended trellis  $\tau$  satisfies (16). The search for

these integers can be conducted systematically starting with  $n = 1$  and increasing  $n$  until the extended trellis  $\tau$  is found. We choose the rate of the superchannel (i.e., the rate of the inner trellis code) to be  $r_{\text{in}} = k/n$ .

The reason for obeying the lower bound in Rule 1  $r_{\text{in}} = k/n > r$  is that  $r_{\text{in}} = r$  would mean that we would not have the option of using a powerful outer code. It is arguable whether the upper bound  $r_{\text{in}} \leq n^{-1} \min_{(i,\ell,j) \in \tau} \left[ -\log P_{ij}^{(\ell)} \right]$  is necessary. However, obeying this upper bound definitely avoids non-unique decodability.

As an example, consider the dicode channel in Figure 1-a. For this channel and for the target code rate  $r = 1/2$ , we verified that the simplest extended trellis for which Rule 1 holds is the 3-rd order extension of the minimal trellis, i.e., the extended trellis in Figure 2-a for  $n = 3$ . The corresponding branching probabilities  $P_{ij}^{(\ell)}$  are given in Table II. The corresponding values  $k$  and  $n$  that satisfy Rule 1 are  $k = 2$  and  $n = 3$ . Hence we decide to pick the superchannel code rate (i.e., the inner trellis code rate) to be  $r_{\text{in}} = k/n = 2/3$ .

### B. Choosing the number of states in the superchannel

Let  $P_{ij}^{(\ell)}$  be the branching probabilities of the chosen extended trellis  $\tau$  in Rule 1. Denote by  $\mathcal{S}$  the number of states in the extended channel trellis  $\tau$ . Let  $\mu_i = \Pr(S_t = i)$  denote the stationary probability of each state  $0 \leq i \leq \mathcal{S} - 1$ .

First, for practical reasons, we will impose a limit  $K_{\max}$  on the maximal number of states in the superchannel trellis. Let  $K$  denote the number of states in the pending superchannel trellis ( $K \leq K_{\max}$ ). Our strategy is to split each state  $i$  of the channel trellis  $\tau$  into  $K_i$  states of the superchannel (call them type- $t_i$  states) such that  $\mu_i \approx K_i/K$  and  $K = \sum_{i=0}^{\mathcal{S}-1} K_i \leq K_{\max}$ . This requires specifying an optimization criterion and solving an optimization problem. Define a probability mass function (pmf)  $\kappa = (\kappa_0, \kappa_1, \dots, \kappa_{\mathcal{S}-1})$ , where  $\kappa_i = K_i/K$ . Denote by  $\mu$  the pmf  $\mu = (\mu_0, \mu_1, \dots, \mu_{\mathcal{S}-1})$ .

**Rule 2.** Pick the pmf  $\kappa$  such that  $D(\kappa||\mu)$  is minimized under the constraints  $\mathcal{S} \leq K = \sum_{i=0}^{\mathcal{S}-1} K_i \leq K_{\max}$ , where  $D(\cdot||\cdot)$  denotes the divergence (Kullback-Leibler distance) [2]. ■

By solving the optimization problem in Rule 2, we determine the number of states  $K$  of the

OPTIMIZED TRANSITION PROBABILITIES FOR THE DICODE CHANNEL  $(1 - D)$  AT RATE  $r = 1/2$  USING THE EXTENDED CHANNEL TRELLIS IN FIGURE 2-A. THE INTEGER VALUES  $k$ ,  $n$ ,  $K_i$  AND  $n_{ij}^{(\ell)}$  ARE DETERMINED USING RULES 1-3.

branch $b_{ij}^{(\ell)}$	branch label $i, u_{ij}^{(\ell)}, v_{ij}^{(\ell)}, j$	transition probability (Rule 1)	optimized integers $k=2, K_0=5, K_1=5$	integer probability (Rule 3)
$b_{00}^{(0)}$	$(0, [000], [0, 0, 0], 0)$	$P_{00}^{(0)}=0.005$	$n_{00}^{(0)}=0$	$n_{00}^{(0)} / (K_0 \cdot 2^k)=0.00$
$b_{00}^{(1)}$	$(0, [010], [0, -2, 2], 0)$	$P_{00}^{(1)}=0.146$	$n_{00}^{(1)}=3$	$n_{00}^{(1)} / (K_0 \cdot 2^k)=0.15$
$b_{00}^{(2)}$	$(0, [100], [-2, 2, 0], 0)$	$P_{00}^{(2)}=0.146$	$n_{00}^{(2)}=3$	$n_{00}^{(2)} / (K_0 \cdot 2^k)=0.15$
$b_{00}^{(3)}$	$(0, [110], [-2, 0, 2], 0)$	$P_{00}^{(3)}=0.195$	$n_{00}^{(3)}=4$	$n_{00}^{(3)} / (K_0 \cdot 2^k)=0.20$
$b_{01}^{(0)}$	$(0, [001], [0, 0, -2], 1)$	$P_{01}^{(0)}=0.066$	$n_{01}^{(0)}=1$	$n_{01}^{(0)} / (K_0 \cdot 2^k)=0.05$
$b_{01}^{(1)}$	$(0, [011], [0, -2, 0], 1)$	$P_{01}^{(1)}=0.145$	$n_{01}^{(1)}=3$	$n_{01}^{(1)} / (K_0 \cdot 2^k)=0.15$
$b_{01}^{(2)}$	$(0, [101], [-2, 2, -2], 1)$	$P_{01}^{(2)}=0.231$	$n_{01}^{(2)}=5$	$n_{01}^{(2)} / (K_0 \cdot 2^k)=0.25$
$b_{01}^{(3)}$	$(0, [111], [-2, 0, 0], 1)$	$P_{01}^{(3)}=0.066$	$n_{01}^{(3)}=1$	$n_{01}^{(3)} / (K_0 \cdot 2^k)=0.05$
$b_{10}^{(0)}$	$(1, [000], [2, 0, 0], 0)$	$P_{10}^{(0)}=0.066$	$n_{10}^{(0)}=1$	$n_{10}^{(0)} / (K_1 \cdot 2^k)=0.05$
$b_{10}^{(1)}$	$(1, [010], [2, -2, 2], 0)$	$P_{10}^{(1)}=0.231$	$n_{10}^{(1)}=5$	$n_{10}^{(1)} / (K_1 \cdot 2^k)=0.25$
$b_{10}^{(2)}$	$(1, [100], [0, 2, 0], 0)$	$P_{10}^{(2)}=0.145$	$n_{10}^{(2)}=3$	$n_{10}^{(2)} / (K_1 \cdot 2^k)=0.15$
$b_{10}^{(3)}$	$(1, [110], [0, 0, 2], 0)$	$P_{10}^{(3)}=0.066$	$n_{10}^{(3)}=1$	$n_{10}^{(3)} / (K_1 \cdot 2^k)=0.05$
$b_{11}^{(0)}$	$(1, [001], [2, 0, -2], 1)$	$P_{11}^{(0)}=0.195$	$n_{11}^{(0)}=4$	$n_{11}^{(0)} / (K_1 \cdot 2^k)=0.20$
$b_{11}^{(1)}$	$(1, [011], [2, -2, 0], 1)$	$P_{11}^{(1)}=0.146$	$n_{11}^{(1)}=3$	$n_{11}^{(1)} / (K_1 \cdot 2^k)=0.15$
$b_{11}^{(2)}$	$(1, [101], [0, 2, -2], 1)$	$P_{11}^{(2)}=0.146$	$n_{11}^{(2)}=3$	$n_{11}^{(2)} / (K_1 \cdot 2^k)=0.15$
$b_{11}^{(3)}$	$(1, [111], [0, 0, 0], 1)$	$P_{11}^{(3)}=0.005$	$n_{11}^{(3)}=0$	$n_{11}^{(3)} / (K_1 \cdot 2^k)=0.00$

TABLE II

superchannel trellis, and the number of states  $K_i$  of each state type  $t_i$ .

Consider again the example  $r = 1/2$  and the dicode channel. We have determined by Rule 1 that  $k = 2$  and  $n = 3$ , and that the probabilities  $P_{ij}^{(\ell)}$  are as in Table II. Solving for  $\mu_i$ , we get  $\mu_0 = \mu_1 = 0.5$ . With  $K_{max} = 12$ , we get six solutions that satisfy Rule 2. They are  $1 \leq K_0 = K_1 \leq 6$ . This illustrates that there may not be a unique solution to the optimization problem in Rule 2. If this happens, then further refinement using Rule 3 (presented in the next subsection) is needed.

### C. Choosing the branch type counts in the superchannel

We say that a branch of the superchannel trellis is of type  $t_{ij}^{(\ell)}$  if

- 1) its starting state is one of the  $K_i$  states of type  $t_i$ .
- 2) its ending state is one of the  $K_j$  states of type  $t_j$ .
- 3) its noiseless channel output  $n$ -tuple matches the noiseless output  $n$ -tuple of the branch  $b_{ij}^{(\ell)}$  in the extended channel trellis  $\tau$ .

Denote by  $n_{ij}^{(\ell)}$  the number of branches in the superchannel trellis of type  $t_{ij}^{(\ell)}$ . For a fixed state  $i \in \{0, 1, \dots, \mathcal{S} - 1\}$  of the channel trellis  $\tau$ , denote by  $\nu_i$  the pmf whose individual probabilities are  $n_{ij}^{(\ell)} / (K_i \cdot 2^k)$ , where  $i$  is fixed and  $\ell$  and  $j$  are varied under the constraint  $(i, \ell, j) \in \tau$ . Obviously, we have

$$\sum_{\ell, j: (i, \ell, j) \in \tau} \frac{n_{ij}^{(\ell)}}{K_i \cdot 2^k} = 1. \quad (17)$$

Similarly, for a fixed  $i \in \{0, 1, \dots, \mathcal{S} - 1\}$ , denote by  $\pi_i$  the pmf whose individual probabilities are  $P_{ij}^{(\ell)}$ , where  $i$  is fixed and  $\ell$  and  $j$  are varied under the constraint  $(i, \ell, j) \in \tau$ . We choose  $n_{ij}^{(\ell)}$  according to the following rule.

**Rule 3.** Determine  $0 \leq n_{ij}^{(\ell)} \in \mathbb{Z}$  such that  $\sum_{i=0}^{\mathcal{S}-1} \kappa_i \cdot D(\nu_i \| \pi_i)$  is minimized under the constraints  $\sum_{j, \ell: (i, \ell, j) \in \tau} n_{ij}^{(\ell)} = K_i \cdot 2^k$ , where  $\kappa_i = K_i / K$  is determined by Rule 2. ■

We have already established that Rule 2 may not deliver a unique solution. If this is the case, then among all solution candidates for Rule 2, we pick the solution that also minimizes the objective function in Rule 3.

We return to the example of the dicode channel and target rate  $r = 1/2$ . Rule 1 delivered  $k = 2$ ,  $n = 3$ , the extended channel trellis  $\tau$  in Figure 2-a, and the branching probabilities  $P_{ij}^{(\ell)}$  in Table II. Rule 2 delivered  $K_0 = K_1 \leq K_{\max}/2 = 6$ , but was unable to more narrowly specify  $K_0$  and  $K_1$ . Applying now Rule 3, we get the integers  $n_{ij}^{(\ell)}$  in Table II and  $K_0 = K_1 = 5$ , i.e., a superchannel trellis with  $K = K_0 + K_1 = 10$  states.

### D. Choosing the branch connections

We establish the following three requirements for the pending superchannel trellis:

- 1) Exactly  $n_{ij}^{(\ell)}$  branches should be of type  $t_{ij}^{(\ell)}$ .
- 2) A branch of type  $t_{ij}^{(\ell)}$  must start at a state of type  $t_i$  and end at a state of type  $t_j$ .
- 3) All branches emanating from a given state in the superchannel trellis must have distinct types, i.e., there cannot be two (or more) branches of the same type  $t_{ij}^{(\ell)}$  emanating from a given state of type  $t_i$ .

Fulfilling requirements 1)-3) will guarantee that the marginal probability of a superchannel branch being of type  $t_{ij}^{(\ell)}$  is very close to the value  $\mu_i P_{ij}^{(\ell)}$ . However, this does not guarantee that the resulting output process of the superchannel will mimic the hidden Markov output process  $\mathcal{P}(\tau)$  over the channel trellis  $\tau$ . Among all branch connections that match the marginal probabilities, we pick the one that satisfies the following rule:

**Rule 4.** Pick the superchannel branch connections that satisfy requirements 1)-3) and deliver a superchannel with the maximal information rate  $\mathcal{I}_S$  evaluated at  $SNR_{(\tau,r)}$ , where  $SNR_{(\tau,r)}$  is the signal-to-noise ratio at which  $\hat{C}_\tau = r$ . ■

If the integers  $K_i$  and  $n_{ij}^{(\ell)}$  are reasonably small, Rule 4 can be satisfied by an exhaustive search. Otherwise, we may conduct a randomized search. That is, we randomly pick a superchannel trellis connection that satisfies requirements 1)-3), we compute  $\mathcal{I}_S$ , and we repeat the procedure until we find a superchannel trellis with an acceptably high information rate  $\mathcal{I}_S$ . An acceptably high superchannel information rate  $\mathcal{I}_S$  is a rate that is close to  $r$ . Alternatively, we may conduct the randomized search by using ordinal optimization [42].

Applying Rule 4 to the decode channel example with the target rate  $r = 1/2$  delivered the branch connections assignment presented in Table III. This branch assignment characterizes a matched information rate (MIR) superchannel, whose information rate is plotted in Figure 8. Notice that the information rate of the constructed MIR superchannel is only 0.1 dB away from the Markov capacity  $\hat{C}_\tau$  of the extended channel trellis in Figure 2-a at the target rate  $r = 1/2$ . This was to be expected because we designed the MIR superchannel to mimic the optimal Markov process for the trellis in Figure 2-a. Also notice that at the target rate  $r = 1/2$ , the designed MIR superchannel has a higher superchannel information rate than other known trellis codes (the rate-2/3 Wolf-Ungerboeck code [26] in Figure 5-b and the rate-4/6 Karabed-Siegel MSN code [29]). This is because the previously known codes were not designed to match the information rate, but rather to optimize algebraic properties such as the minimum Euclidean

SUPERCHANNEL TRELLIS FOR THE DICODE  $(1 - D)$  CHANNEL. THE DESIGN RATE IS  $r = 0.5$ . THE SUPER-CHANNEL RATE IS  $r_{\text{in}} = 2/3$ . THE BRANCH TYPE  $t_{ij}^{(\ell)}$  PROVIDES A LINK TO TABLE II.

branch type	type $t_0$ start state	super- channel input $k$ -tuple (Rule 5)	trellis- code output $n$ -tuple	noiseless superchannel output $n$ -tuple (Rule 4)	end state
$t_{01}^{(2)}$	0	0,0	1,0,1	-2, 2, -2	9
$t_{00}^{(2)}$	0	0,1	1,0,0	-2, 2, 0	0
$t_{01}^{(1)}$	0	1,0	0,1,1	0, -2, 0	5
$t_{00}^{(3)}$	0	1,1	1,1,0	-2, 0, 2	1
$t_{01}^{(3)}$	1	0,0	1,1,1	-2, 0, 0	9
$t_{01}^{(2)}$	1	0,1	1,0,1	-2, 2, -2	6
$t_{01}^{(1)}$	1	1,0	0,1,1	0, -2, 0	8
$t_{00}^{(3)}$	1	1,1	1,1,0	-2, 0, 2	2
$t_{01}^{(1)}$	2	0,0	0,1,1	0, -2, 0	6
$t_{01}^{(2)}$	2	0,1	1,0,1	-2, 2, -2	8
$t_{00}^{(1)}$	2	1,0	0,1,0	0, -2, 2	3
$t_{00}^{(3)}$	2	1,1	1,1,0	-2, 0, 2	0
$t_{01}^{(2)}$	3	0,0	1,0,1	-2, 2, -2	7
$t_{00}^{(2)}$	3	0,1	1,0,0	-2, 2, 0	2
$t_{00}^{(1)}$	3	1,0	0,1,0	0, -2, 2	4
$t_{00}^{(3)}$	3	1,1	1,1,0	-2, 0, 2	4
$t_{01}^{(0)}$	4	0,0	0,0,1	0, 0, -2	7
$t_{01}^{(2)}$	4	0,1	1,0,1	-2, 2, -2	5
$t_{00}^{(1)}$	4	1,0	0,1,0	0, -2, 2	1
$t_{00}^{(2)}$	4	1,1	1,0,0	-2, 2, 0	3

branch type	type $t_1$ start state	super- channel input $k$ -tuple (Rule 5)	trellis- code output $n$ -tuple	noiseless superchannel output $n$ -tuple (Rule 4)	end state
$t_{11}^{(1)}$	5	0,0	0,1,1	2, -2, 0	6
$t_{11}^{(2)}$	5	0,1	1,0,1	0, 2, -2	8
$t_{10}^{(1)}$	5	1,0	0,1,0	2, -2, 2	4
$t_{10}^{(3)}$	5	1,1	1,1,0	0, 0, 2	2
$t_{11}^{(0)}$	6	0,0	0,0,1	2, 0, -2	5
$t_{11}^{(2)}$	6	0,1	1,0,1	0, 2, -2	5
$t_{11}^{(1)}$	6	1,0	0,1,1	2, -2, 0	7
$t_{10}^{(1)}$	6	1,1	0,1,0	2, -2, 2	2
$t_{11}^{(0)}$	7	0,0	0,0,1	2, 0, -2	9
$t_{11}^{(2)}$	7	0,1	1,0,1	0, 2, -2	6
$t_{10}^{(1)}$	7	1,0	0,1,0	2, -2, 2	1
$t_{10}^{(2)}$	7	1,1	1,0,0	0, 2, 0	3
$t_{11}^{(0)}$	8	0,0	0,0,1	2, 0, -2	7
$t_{10}^{(2)}$	8	0,1	1,0,0	0, 2, 0	1
$t_{10}^{(1)}$	8	1,0	0,1,0	2, -2, 2	3
$t_{10}^{(0)}$	8	1,1	0,0,0	2, 0, 0	0
$t_{11}^{(0)}$	9	0,0	0,0,1	2, 0, -2	8
$t_{10}^{(2)}$	9	0,1	1,0,0	0, 2, 0	4
$t_{11}^{(1)}$	9	1,0	0,1,1	2, -2, 0	9
$t_{10}^{(1)}$	9	1,1	0,1,0	2, -2, 2	0

TABLE III

distance [26] or to match the channel spectral null [29]. The main point here is not that we have a code that surpasses the information rate of the MSN code, but rather that we now have a *method* to construct superchannel trellises for *any* channel, even if the channel does not have spectral nulls.

**Complexity:** Before leaving this section, we briefly discuss the complexity of constructing the inner code for a channel with memory length  $J$  and a target code rate  $r$ . First, we note that

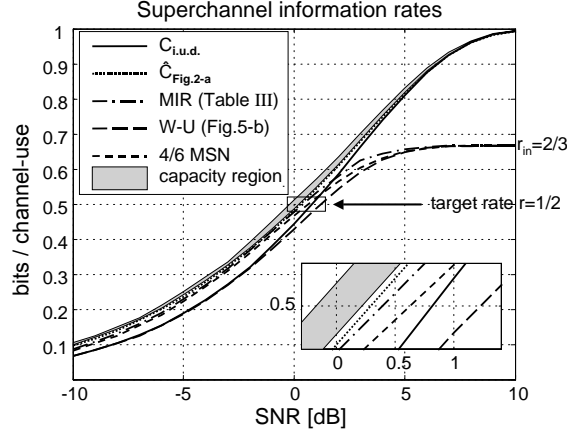


Fig. 8. Comparison of the information rate curve of the constructed MIR superchannel in Table III, to previously known decode  $(1 - D)$  superchannels. Shown are the information rates of the Wolf-Ungerboeck (Fig. 5-b) and Karaded-Siegel MSN superchannels with rate  $r_{in} = 2/3$ , compared at the target rate  $r = 1/2$ .  $C_{i.u.d.}$  is the i.u.d. capacity.  $\hat{C}_{Fig.2-a}$  is the numerical Markov capacity of the extended channel trellis in Figure 2-a, whose optimal Markov probabilities were used to construct the MIR superchannel. The shaded region is the region between the tightest known lower bound  $\hat{C}_6$  and the tightest known upper bound on the channel capacity.

we use an extended  $n$ -th order channel trellis with  $2^L \geq 2^J$  states to perform the optimization of transition probabilities in the algorithm in Table I. The complexity of this algorithm is  $O(N \cdot 2^L \cdot 2^n)$ , where  $N$  is the number of trellis sections used for the Monte Carlo simulation. Next, note that the pending inner trellis code has at most  $K_{max}$  states, where  $K_{max}$  is an a-priori chosen parameter. The rate of the inner code is  $r_{in} = k/n \geq r$ , and the number of branches emanating from each state is  $2^k$ , which makes the decoding complexity at most  $O(K_{max} 2^k)$ . Since  $K_{max} \geq 2^J$ , this complexity is at least  $O(2^{J+rn})$ . Hence, we see that both the code construction complexity and the decoding complexity grow at least exponentially with the channel memory length  $J$  and the extension parameter  $n$ . Thus, one realizes that this coding strategy is practical only for reasonably small values of  $n$  and  $J$ .

## V. OUTER LDPC CODES

To motivate the design strategy for the outer code, we again consider the decode channel example. For the decode channel, Figure 8 reveals that the i.u.d. capacity  $C_{i.u.d.}$  equals the design rate  $r = 1/2$  bits/channel-use at  $SNR = 0.82$ dB. By applying Rules 1-4 to the decode channel, we have constructed a trellis code whose superchannel trellis is shown in Table III, and whose



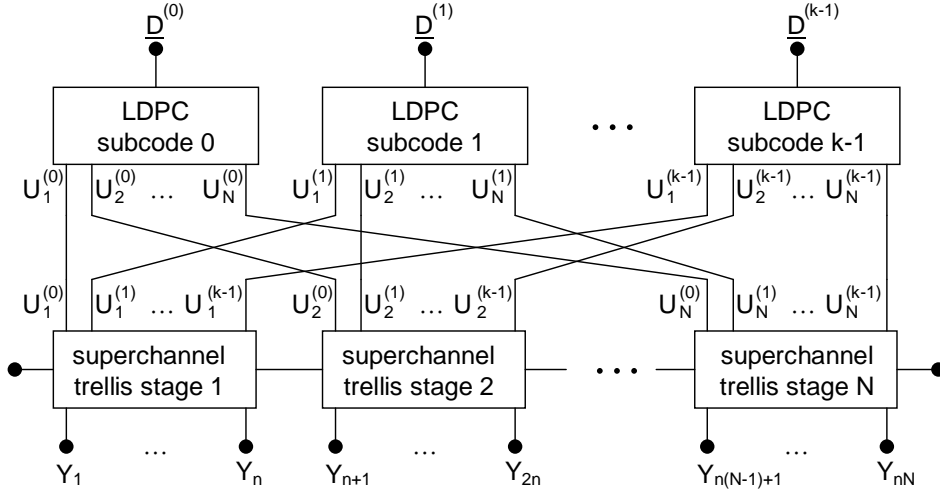


Fig. 9. The normal graph of an outer LDPC code (consisting of  $k$  subcodes) and the inner trellis superchannel.

superchannel i.u.d information rate  $\mathcal{I}_S$  is plotted in Figure 8. The superchannel i.u.d. information rate  $\mathcal{I}_S$  equals  $r = 1/2$  bits/channel-use at  $\text{SNR} = 0.40\text{dB}$ . Theorem 1 asserts that there exists at least one linear (coset) code such that, if we apply this code to the superchannel and utilize maximum likelihood decoding, we may gain  $0.42\text{dB}$  over the  $C_{\text{i.u.d.}}$ . Since it is impractical to perform the maximum likelihood decoding for a general linear code, we show how to construct practical low-density parity-check (LDPC) codes as outer codes.

#### A. Encoding/decoding system

The normal graph [43] of the entire concatenated coding system is shown in Figure 9. In our design we utilize  $k$  separate subcodes as the outer code (we explain the reason below), where  $k$  is the number of input bits at each trellis stage of the MIR trellis code. Let the rate of the  $i$ -th subcode be  $r^{(i)} = K^{(i)}/N$ , where  $0 \leq i \leq k-1$ . Let  $N$  and  $K^{(total)}$  be two positive integers such that  $K^{(total)}/(kN) = r_{\text{out}}$  and  $N$  is large enough. Let  $\underline{D}$  be an i.u.d. binary sequence of length  $K^{(total)} = \sum_{i=0}^{k-1} K^{(i)}$  to be transmitted over the channel. The encoding algorithm can be described by the following three steps (see also Figure 9 for the explanation of the notation):

- 1) The sequence  $\underline{D}$  is separated into  $k$  subsequences

$$\underline{D} = [\underline{D}^{(0)}, \underline{D}^{(1)}, \dots, \underline{D}^{(k-1)}],$$

where the  $i$ -th subsequence  $\underline{D}^{(i)}$  is of length  $K^{(i)}$ .

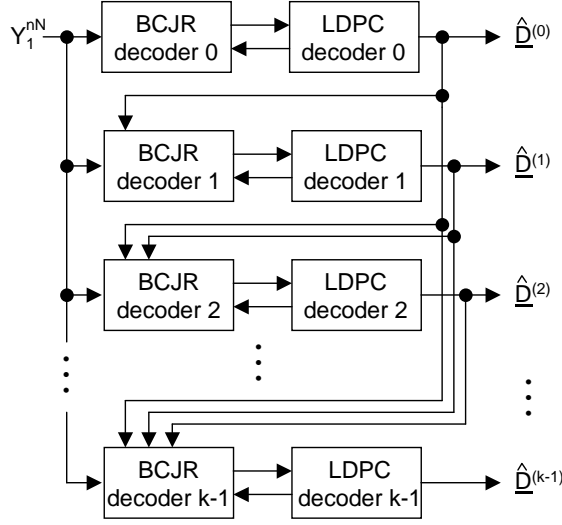


Fig. 10. The iterative decoder. The  $k$  codes are successively decoded, each iteratively.

- 2) The  $i$ -th subsequence  $\underline{D}^{(i)}$  enters the  $i$ -th LDPC encoder whose code-rate is  $r^{(i)} = K^{(i)}/N$ . The output sequence from the  $i$ -th (sub)encoder is denoted by  $[U^{(i)}]_1^N = [U_1^{(i)}, \dots, U_N^{(i)}]$ .
- 3) The whole sequence  $U_1^N = [U_1, \dots, U_t, \dots, U_N]$  enters the MIR trellis encoder, where  $U_t = (U_t^{(0)}, \dots, U_t^{(k-1)})$ . The sequence  $Y_1^{nN}$  is observed at the channel output.

To recover the input bits to such a system, we may perform the iterative sum-product algorithm [19] on the normal graph shown in Figure 9. However, for the purpose of systematically *designing* good outer (sub)codes, we specify a serial multi-stage decoding schedule [44] depicted in Figure 10. This algorithm is very similar to the scheme proposed in [6], with the only difference being that we do not use the BCJR detector only once, but rather we iterate between the BCJR decoder and the LDPC decoder.

To guarantee that the proposed decoding algorithm of Figure 10 works well, we have to solve the following three problems:

- A trellis section can be viewed as a collection of branches  $\{b_t = (s_{t-1}, u_t, v_t, s_t)\}$ . In terms of optimizing the superchannel i.u.d. information rate  $\mathcal{I}_S$ , it is irrelevant how the input vector  $u_t$  of each branch is chosen. In other words, the input-bit assignment does not affect the superchannel i.u.d. information rate  $\mathcal{I}_S$ . However, to construct a good and practically decodable LDPC code we need to choose the branch input-bit assignment judiciously. We have empirically observed that different bit assignments on the branches cause the cor-

responding LDPC message-passing (belief propagation) decoders to perform drastically differently. Hence, we made the conclusion that a good bit assignment must be found in order for the message-passing decoder to achieve the desired performance. We develop the input-bit assignment design rule (Rule 5) in Subsection V-B.

- Consider the soft-output random variables  $L_t^{(i)} \stackrel{\text{def}}{=} \Pr(U_t^{(i)} = 0 | Y_1^{nN})$ . For a general  $k/n$  inner trellis code, when  $N \rightarrow \infty$ , the statistical properties of  $L_{t_1}^{(i)}$  and  $L_{t_2}^{(j)}$  are the same if and only if  $i = j$ . In other words, different superchannel bit positions have different soft-output statistics. This is why we utilize  $k$  different (and separated) subcodes in our design. Now the question is: how do we determine the rates  $r^{(i)} = K^{(i)}/N$  of the constituent subcodes? Subsection V-C gives the answer.
- For large  $N$ , we need to optimize each of the  $k$  subcodes. In Subsection V-D we develop an optimization method along the lines of [35], [38].

### B. Choosing the branch input-bit assignment

Every branch  $b_t = (s_{t-1}, u_t, v_t, s_t)$  of the superchannel trellis needs to have the input  $k$ -tuple  $u_t = [u_t^{(0)}, u_t^{(1)}, \dots, u_t^{(k-1)}]$  specified, where  $u_t^{(i)} \in \mathcal{A}$ . Considering the decoder in Figure 10, we observe that if the bit assignments for  $u_t^{(0)}, u_t^{(1)}, \dots, u_t^{(i-1)}$  are given, then only the bit assignment for  $u_t^{(i)}$  determines the performance of decoder  $i$ , irrespective of the bit assignment for  $u_t^{(i+1)}, u_t^{(i+2)}, \dots, u_t^{(k-1)}$ .

**Rule 5.** Determine the branch bit assignment using the following greedy bit assignment algorithm:

- 0) Find the bit assignment for the 0-th location  $u_t^{(0)}$  such that the 0-th BCJR decoder in Figure 10 delivers the lowest probability of bit error *in the first iteration*.
- $\vdots$
- $i$ ) Assume that Decoders 0 through  $i - 1$  decode without errors. Find the bit assignment for the  $i$ -th location  $u_t^{(i)}$  such that the  $i$ -th BCJR decoder in Figure 10 delivers the lowest probability of bit error *in the first iteration*.
- $\vdots$
- $k-1$ ) Assume that Decoders 0 through  $k - 2$  decode without errors. Find the bit assignment for the  $(k - 1)$ -th location  $u_t^{(k-1)}$  such that the  $(k - 1)$ -th BCJR decoder in Figure 10

delivers the lowest probability of bit error *in the first iteration*. ■

This bit assignment algorithm guarantees a good decoding start. That is, this algorithm ensures that the first iteration of each BCJR decoder in Figure 10 delivers a good bit error rate, which is typically sufficient to ensure good decoding properties for the concatenated LDPC decoder.

For low-complexity superchannel trellises, we may perform an exhaustive search among all bit assignments to satisfy Rule 5, but for even moderately complex superchannel trellises, the randomized search using ordinal optimization [42] is the best known method. By applying Rule 5, we have selected one good input-bit assignment for the trellis code shown in Table III. The input bit assignment is shown in the third column of Table III.

### C. Determining the subcode rates

Now we show how to determine the code rates for the outer subcodes. Ideally, the superchannel i.u.d. rate  $\mathcal{I}_S$  should equal the target rate  $r$

$$r = r_{\text{in}} \cdot r_{\text{out}} = \frac{k}{n} \cdot r_{\text{out}} = \lim_{N \rightarrow \infty} \frac{1}{nN} \cdot I(U_1^N; Y_1^{nN}) \Big|_{P_{U_1^N}(u_1^N) = 2^{-kN}} = \mathcal{I}_S, \quad (18)$$

which amounts to

$$r_{\text{out}} = \lim_{N \rightarrow \infty} \frac{1}{kN} \cdot I(U_1^N; Y_1^{nN}) \Big|_{P_{U_1^N}(u_1^N) = 2^{-kN}}. \quad (19)$$

From the chain rule [2], we have

$$\begin{aligned} I(U_1^N; Y_1^{nN}) &= I([U^{(0)}]_1^N; Y_1^{nN}) \\ &\quad + I([U^{(1)}]_1^N; Y_1^{nN} \mid [U^{(0)}]_1^N) \\ &\quad + \dots \\ &\quad + I([U^{(k-1)}]_1^N; Y_1^{nN} \mid [U^{(0)}]_1^N, [U^{(1)}]_1^N, \dots, [U^{(k-2)}]_1^N), \end{aligned} \quad (20)$$

where  $[U^{(i)}]_1^N = [U_1^{(i)}, U_2^{(i)}, \dots, U_N^{(i)}]$ . The decoding algorithm given in Figure 10 assumes that Decoders 0 through  $i - 1$  perform errorless decoding before Decoder  $i$  starts the decoding process. From (20) we see that this is possible only if the rate of the  $i$ -th code is

$$r^{(i)} \leq \lim_{N \rightarrow \infty} \frac{1}{N} \cdot I([U^{(i)}]_1^N; Y_1^{nN} \mid [U^{(0)}]_1^N, [U^{(1)}]_1^N, \dots, [U^{(i-1)}]_1^N). \quad (21)$$

Therefore, a reasonable rate-assignment is

$$r^{(0)} = \lim_{N \rightarrow \infty} \frac{1}{N} \cdot I \left( [U^{(0)}]_1^N ; Y_1^{nN} \right) \quad (22)$$

$$r^{(1)} = \lim_{N \rightarrow \infty} \frac{1}{N} \cdot I \left( [U^{(1)}]_1^N ; Y_1^{nN} \middle| [U^{(0)}]_1^N \right) \quad (23)$$

$\vdots$

$$r^{(k-1)} = \lim_{N \rightarrow \infty} \frac{1}{N} \cdot I \left( [U^{(k-1)}]_1^N ; Y_1^{nN} \middle| [U^{(0)}]_1^N, [U^{(1)}]_1^N, \dots, [U^{(k-2)}]_1^N \right), \quad (24)$$

where the sequences  $[U^{(i)}]_1^N$  are i.u.d. for all  $0 \leq i \leq k-1$ . The rates in (22)-(24) can be computed by Monte Carlo simulation [5], [6]. Consequently, we get  $r_{\text{out}} = \frac{1}{k} \sum_{i=0}^{k-1} r^{(i)}$ .

To summarize, first we choose two integers  $(K^{(total)}, N)$ , where  $N$  is large enough, such that  $K^{(total)}/(kN)$  is not greater than (and is as close as possible to)  $r_{\text{out}}$  given in (19). Then we choose  $K^{(i)}$ , for  $0 \leq i \leq k-1$  such that  $\sum_{i=0}^{k-1} K^{(i)} = K^{(total)}$  and  $K^{(i)}/N \approx r^{(i)}$ , where  $r^{(i)}$  is computed by (22)-(24).

#### D. Subcode optimization

Using the subcode rates chosen in Section V-C we optimize the LDPC subcodes. Each of the  $k$  subcodes is optimized in a very similar fashion as in [38].

##### a) Density evolution and the noise tolerance threshold

For the optimization, we use density evolution tools for channels with ISI memory [37]. In this setting, an LDPC code is typically represented by a bipartite Tanner graph with two types of nodes: 1) *variable nodes* corresponding to the encoded binary symbols, and 2) *check nodes* corresponding to the parity-check constraints that the encoded symbols have to satisfy [45].

The *degree* of a node is the number of edges that are connected to the node. The fraction of edges connected to a variable node of degree  $i \geq 2$  is denoted by  $\lambda_i$ , whereas the fraction of edges connected to a check node of degree  $j \geq 2$  is denoted by  $\rho_j$ . Denoting by  $L$  the maximal variable node degree and by  $R$  the maximal check node degree, we have that the check and variable *degree sequences* are  $\underline{\lambda} = [\lambda_2, \lambda_3, \dots, \lambda_L]$  and  $\underline{\rho} = [\rho_2, \rho_3, \dots, \rho_R]$ , respectively. The code rate of the LDPC code is

$$r \geq 1 - \frac{\sum_{j=2}^R (\rho_j/j)}{\sum_{i=2}^L (\lambda_i/i)}. \quad (25)$$

We say that an LDPC code is *regular* if  $\lambda_L = \rho_R = 1$ . Otherwise, the code is *irregular*. For details regarding the regular and irregular LDPC codes, see [32], [33], [46].

The noise tolerance *threshold*  $\sigma^*$  of an LDPC code on a given channel is defined as the supremum of the noise standard deviation  $\sigma$  such that the probability of the decoding error can be made arbitrarily small given that the code block length and the number of iterations of the message passing decoding on a graph are sufficiently large [34]. For any degree sequence pair  $(\underline{\lambda}, \underline{\rho})$  the noise tolerance threshold may be computed using *density evolution* (for memoryless channels see [34]; for channels with memory see [37]). The corresponding SNR threshold for a partial response channel is

$$SNR^* = 10 \log_{10} \frac{\sum_{j=0}^J h_j^2}{(\sigma^*)^2}, \quad (26)$$

where  $h_j$  are the channel response coefficients.

#### b) Outer code optimization

Optimizing an LDPC code of rate  $r$  means finding sequences  $\underline{\lambda}$  and  $\underline{\rho}$  that satisfy (25) and maximize the noise threshold  $\sigma^*$ . To optimize the outer code for a given superchannel we generalize the *single* code optimization method for a partial response channel (with no inner code)<sup>2</sup> to a *joint* optimization of  $k$  different subcodes. This generalization is summarized as follows:

- 1) Given a superchannel trellis of rate  $r_{\text{in}} = k/n$ , we need to optimize  $k$  different constituent subcodes shown in Figure 9 according to the decoding scenario shown in Figure 10. That is, while optimizing the degree sequences of the LDPC Subcode  $i$ , we assume that the symbols  $[U^{(0)}]_1^N, [U^{(1)}]_1^N, \dots, [U^{(i-1)}]_1^N$  are decoded without errors and that no prior information on the symbols  $[U^{(i+1)}]_1^N, \dots, [U^{(k-1)}]_1^N$  is available.
- 2) The  $i$ -th constituent subcode of rate  $r^{(i)}$  is optimized using the optimization tools presented in [38]. Thereby, we make a change in the density evolution algorithm to reflect the fact that Decoders 0 through  $i-1$  have decoded symbols  $[U^{(0)}]_1^N$  through  $[U^{(i-1)}]_1^N$  without errors. This is done by generating i.u.d. realizations of symbols  $[u^{(0)}]_1^N$  through  $[u^{(i-1)}]_1^N$ , and running the BCJR algorithm (in the density evolution step) with prior knowledge of these symbols; see [37] for density evolution details.
- 3) We obtain  $k$  pairs  $(\underline{\rho}^{(i)} \text{ and } \underline{\lambda}^{(i)})$  of the optimized degree sequences, each pair correspond-

<sup>2</sup>For details about the single code optimization and some ad-hoc rules to speed up the optimization see [35][36] and [38].

ing to one constituent subcode;  $0 \leq i \leq k - 1$ . Using the density evolution algorithm, we compute the noise tolerance thresholds  $\sigma_i^*$  for all subcodes ( $0 \leq i \leq k - 1$ ). The noise threshold of the entire code is then given by  $\sigma^* = \min\{\sigma_0^*, \sigma_1^*, \dots, \sigma_{k-1}^*\}$ .

### c) LDPC decoding

While we used Figure 10 to systematically construct the outer code, we note that we only need *one* BCJR detector and *one* LDPC decoder once the constituent subcodes are obtained. The *single* LDPC code is constructed by interleaving the subcodes.

Denote the parity check matrix of Subcode  $i$  (constructed from the degree sequences  $\underline{\lambda}^{(i)}$  and  $\underline{\rho}^{(i)}$ ) by

$$\mathbf{H}^{(i)} = \begin{bmatrix} \underline{h}_1^{(i)} & \underline{h}_2^{(i)} & \dots & \underline{h}_N^{(i)} \end{bmatrix},$$

where  $\underline{h}_j^{(i)}$  represents the  $j$ -th column of the matrix  $\mathbf{H}^{(i)}$ . The parity check matrix  $\mathbf{H}$  of the entire code is obtained by interleaving the columns of the subcode parity check matrices

$$\mathbf{H} = \left[ \begin{array}{cccc|cccc| \dots | \dots | \dots |} \begin{array}{cccc} \underline{h}_1^{(0)} & \underline{0} & \dots & \underline{0} \\ \underline{0} & \underline{h}_1^{(1)} & \dots & \underline{0} \\ \vdots & \vdots & \ddots & \vdots \\ \underline{0} & \underline{0} & \dots & \underline{h}_1^{(k-1)} \end{array} & \begin{array}{cccc} \underline{h}_2^{(0)} & \underline{0} & \dots & \underline{0} \\ \underline{0} & \underline{h}_2^{(1)} & \dots & \underline{0} \\ \vdots & \vdots & \ddots & \vdots \\ \underline{0} & \underline{0} & \dots & \underline{h}_2^{(k-1)} \end{array} & \dots & \begin{array}{cccc} \underline{h}_N^{(0)} & \underline{0} & \dots & \underline{0} \\ \underline{0} & \underline{h}_N^{(1)} & \dots & \underline{0} \\ \vdots & \vdots & \ddots & \vdots \\ \underline{0} & \underline{0} & \dots & \underline{h}_N^{(k-1)} \end{array} \end{array} \right]. \quad (27)$$

The size of the matrix  $\mathbf{H}$  is  $(kN - K^{(total)}) \times (kN)$ , where  $K^{(total)} = \sum_{i=0}^{k-1} K^{(i)}$ .

The resulting LDPC code is decoded using the message-passing decoder for the code represented by the matrix  $\mathbf{H}$  over the superchannel with a known trellis structure [37]. An interesting feature of this decoder is that decoding occurs in stages. That is, even though the message-passing is performed on a single graph, the initial iterations contribute mostly to decoding Subcode 0. After Subcode 0 is decoded, the following iterations contribute mostly to decoding Subcode 1, and so on. This is not surprising given that Figure 10 was used to optimize the code.

### E. Optimization results

We demonstrate that our code construction method applies to both channels with and without spectral nulls. We perform the code optimization on the dicode  $(1 - D)$  channel which has a spectral null at frequency  $\omega = 0$ , and on the  $1 + 3D + D^2$  channel which has no spectral nulls.

#### a) Dicode channel

We revert to the dicode  $(1 - D)$  channel example with the target rate  $r = 1/2$ . The inner trellis code with code rate  $r_{\text{in}} = k/n = 2/3$  for the dicode channel is given in Table III. Since

GOOD DEGREE SEQUENCES AND THE NOISE THRESHOLDS FOR SEPARATELY CODED EVEN AND ODD BITS OF THE OUTER LDPC CODE ON THE SUPERCANNEL FROM TABLE III;  $r^{(0)} = 0.660$ ,  $r^{(1)} = 0.840$ ,  $r_{\text{out}} = \frac{r^{(0)} + r^{(1)}}{2} = 0.75$ ;  $r = r_{\text{in}} \cdot r_{\text{out}} = 0.5$ .

Subcode 0 (even bits)			Subcode 1 (odd bits)		
$r^{(0)} = 0.660$			$r^{(1)} = 0.840$		
$x$	$\lambda_x^{(0)}$	$\rho_x^{(0)}$	$x$	$\lambda_x^{(1)}$	$\rho_x^{(1)}$
2	0.2225		2	0.2191	
3	0.1611		3	0.1526	
5	0.1627		4	0.1057	
6	0.0164		15	0.0017	
10	0.0024		16	0.0003	
11			17		
13			23		
16			49	0.1205	
24			50	0.4001	
48	0.0035		51		
49	0.4314	59			
threshold - Subcode 0			threshold - Subcode 1		
$\sigma_0^* = 1.322$			$\sigma_1^* = 1.326$		
threshold $\sigma^* = 1.322$					
$SNR^* = 10 \log_{10} \frac{2}{(\sigma^*)^2} = 0.59\text{dB}$					

TABLE IV

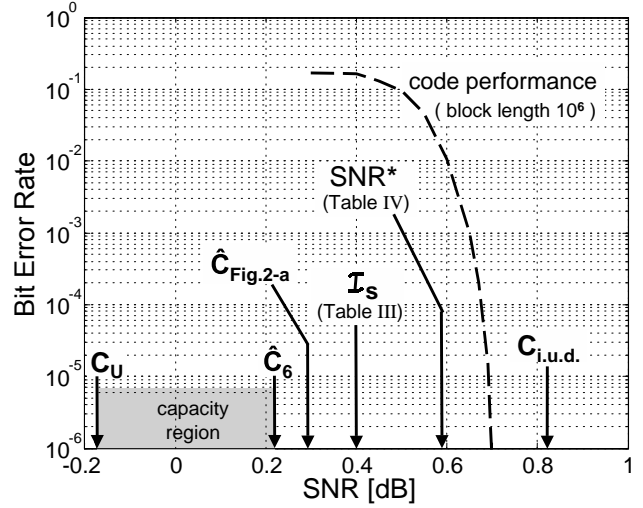


Fig. 11. Bit error rate vs. SNR for the Dicode  $(1-D)$  channel. Code rate  $r = 1/2$ ; code block length  $10^6$  bits. Also shown are the i.u.d. channel capacity ( $C_{\text{i.u.d.}}$ ), the superchannel i.u.d. rate ( $\mathcal{I}_s$ ), an upper bound ( $C_U$ ) and a lower bound ( $\hat{C}_6$ ) on the capacity and the noise threshold ( $\sigma^*$ ).

$k = 2$ , we optimize 2 outer LDPC subcodes. Using equations (22)-(24), we found their rates to be  $r^{(0)} = 0.66$  and  $r^{(1)} = 0.84$ , respectively. The rate of the outer code is thus  $r_{\text{out}} = \frac{1}{2} \cdot (r^{(0)} + r^{(1)}) = 3/4$ . The resulting overall code rate is  $r = r_{\text{in}} \cdot r_{\text{out}} = \frac{2}{3} \cdot \frac{3}{4} = 1/2$ , which is exactly our target code rate.

The optimized degree sequences together with their respective thresholds are given in Table IV. We constructed an outer LDPC code by interleaving the parity check matrices obtained from the optimized degree sequences, see (27). The code length was set to  $10^6$  binary symbols. The bit error rate simulation curve is shown in Figure 11. For comparison, Figure 11 also shows a tight lower bound ( $\hat{C}_6$ ) and an upper bound ( $C_U$ ) on the capacity, the superchannel i.u.d. rate ( $\mathcal{I}_s$ ), the i.u.d. capacity ( $C_{\text{i.u.d.}}$ ) and the noise threshold ( $SNR^*$ ) of the code in Ta-



SUPERCHANNEL TRELLIS FOR THE  $1+3D+D^2$  CHANNEL AND DESIGN RATE  $r = 1/3$ . THE SUPERCHANNEL RATE IS  $r_{\text{in}} = 1/2$ .

start state	super- channel input $k$ -tuple	trellis- code output $n$ -tuple	noiseless superchannel output $n$ -tuple	end state
0	0	0,0	5, 5	1
0	1	0,1	5, 3	3
1	0	0,0	5, 5	2
1	1	1,1	3, -3	6
2	0	0,0	5, 5	0
2	1	1,1	3, -3	5
3	0	1,0	-3, -3	4
3	1	1,1	-3, -5	7
4	0	0,0	3, 5	0
4	1	0,1	3, 3	3
5	0	0,0	-3, 3	2
5	1	1,1	-5, -5	7
6	0	0,0	-3, 3	1
6	1	1,1	-5, -5	5
7	0	1,0	-5, -3	4
7	1	1,1	-5, -5	6

TABLE V

GOOD DEGREE SEQUENCES AND THE NOISE THRESHOLD FOR THE OUTER LDPC CODE DESIGNED FOR THE SUPERCHANNEL IN TABLE V;  $r_{\text{out}} = r^{(0)} = 2/3$ .

$r_{out} = r^{(0)} = 0.6667$		
$i$	$\lambda_i$	$\rho_i$
2	0.2031	0.2974
3	0.2195	
4	0.0022	
6	0.1553	
10		
13		
15		
27	0.1616	
28	0.1399	
38		
50	0.1184	0.2056
threshold		
$\sigma_0^* = \sigma^* = 4.475$		
$SNR^* = 10 \log_{10} \frac{11}{(\sigma^*)^2}$		
$= -2.60\text{dB}$		

TABLE VI

ble IV. We see from Figure 11 that the threshold  $SNR^*$  is 0.19dB away from the superchannel i.u.d. rate  $\mathcal{I}_S$  and that this threshold surpasses the i.u.d. capacity  $C_{\text{i.u.d.}}$  by 0.23dB. The code simulation shows that a bit-error-rate (BER) of  $10^{-6}$  is achieved at an SNR that surpasses  $C_{\text{i.u.d.}}$  by 0.14dB.

#### b) A channel without spectral nulls

For the  $1 + 3D + D^2$  channel we chose the design rate  $r = 1/3$ . Using Rules 1-5, we constructed an inner trellis code of rate  $r_{\text{in}} = k/n = 1/2$  for this channel; the superchannel trellis is given in Table V. Since  $k = 1$ , we optimize only one outer LDPC subcode of rate  $r_{\text{out}} = 2/3$ , which gives the desired target code rate  $r = \frac{1}{2} \cdot \frac{2}{3} = 1/3$ . The optimization of

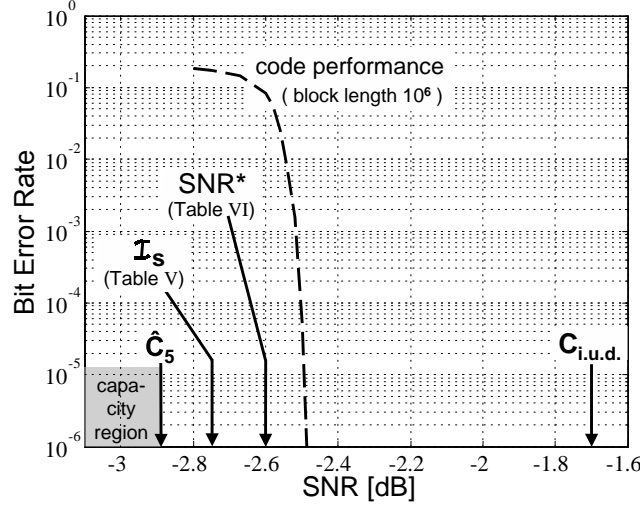


Fig. 12. Bit error rate vs. SNR for the  $1 + 3D + D^2$  channel. Code rate  $r = 1/3$ ; code block length  $10^6$  bits. Also shown are the i.u.d. channel capacity ( $C_{i.u.d.}$ ), the superchannel i.u.d. rate ( $\mathcal{I}_S$ ), a lower bound on the capacity ( $\hat{C}_5$ ) and the noise threshold ( $\sigma^*$ ).

the outer LDPC code delivered the degree sequences and threshold shown in Table VI. Using these degree sequences we constructed an LDPC code of block length  $10^6$  binary symbols. The code's bit error rate performance curve is shown in Figure 12. We observe from Figure 12 that the superchannel i.u.d. rate  $\mathcal{I}_S$  is 0.14dB away from the lower bound on the channel capacity  $\hat{C}_5$ . The noise threshold  $SNR^*$  is 0.15dB away from the superchannel i.u.d. rate  $\mathcal{I}_S$ . The threshold  $SNR^*$  surpasses the i.u.d. capacity  $C_{i.u.d.}$  by 0.90dB. The code simulation reveals that our design method yields a code that achieves a bit error rate of  $10^{-6}$  at an SNR that surpasses the i.u.d. capacity  $C_{i.u.d.}$  by 0.77dB.

## VI. CONCLUSION

In this paper, we developed a methodology for designing capacity approaching codes for partial response channels. Since the partial response channel is a channel with memory, its capacity  $C$  is greater than the i.u.d. capacity  $C_{i.u.d.}$ . We showed that rates above  $C_{i.u.d.}$  cannot be achieved by random linear codes. Given that our goal was to construct a code that surpasses the i.u.d. capacity  $C_{i.u.d.}$ , we chose a concatenated coding strategy where the inner code is a trellis code, and the outer code is an LDPC code.

The role of the inner trellis code is to transform the i.u.d. input symbols to a correlated binary sequence that matches the characteristics of the channel. The key step in the design of the inner trellis is to identify a Markov input process that achieves a high (capacity-approaching) information rate over the partial response channel of interest. Then, we construct a trellis code that mimics the identified Markov process whose information rate is high. Hence, we name it a *matched information rate* (MIR) trellis code.

MIR trellis code constructions are different from any previously known trellis code construction methods in that we do *not* base the code construction on an algebraic criterion (e.g., maximizing the minimum Euclidean distance or matching a spectral null). Instead, the code construction is purely *probabilistic*. We provided a set of 5 rules to construct the inner MIR trellis codes. These rules apply to *any* partial response channel.

We chose the outer code to be an LDPC code. We proposed to design the outer code by interleaving  $k$  different LDPC subcodes, where  $k/n$  is the rate of the inner MIR code. We provided rules to pick the code rates of the constituent LDPC subcodes, and devised modified LDPC subcode optimization methods to fit our specific inner code and the partial response channel. Explicit code constructions that follow these rules were given for both channels with and without spectral nulls. In both examples, the codes achieved rates that surpassed their respective i.u.d. capacities  $C_{\text{i.u.d.}}$ , and approached the numerical capacities to within 0.3 – 0.5dB.

We should note that capacity approaching codes for *certain* (special) partial response channels and/or *certain* (typically very low or very high) code rates were known prior to this work. One example is the biphasic matched spectral null code over the dicode channel which can achieve the channel capacity at low rates when an optimal outer code is applied. Another example is a regular LDPC code over any partial response channel in the region where  $C \approx C_{\text{i.u.d.}}$ , i.e., at very high rates ( $r \geq 0.9$ ), see [37]. However, apart from these rather special examples, constructions of capacity approaching codes for general partial response channels were not known prior to this work. The main contribution of this paper is that we now have a method to *systematically* construct codes that surpass the i.u.d. capacity  $C_{\text{i.u.d.}}$  (and ultimately approach the capacity  $C$ ) on *any* partial response channel at *any* design code rate of interest. To the best of our knowledge, the MIR codes presented in this paper are the first to achieve this goal.

## ACKNOWLEDGMENTS

The authors would like to thank Michael Mitzenmacher for very helpful discussions over the course of the research that lead to this paper.

## REFERENCES

- [1] R. G. Gallager, *Information Theory and Reliable Communication*. New York: John Wiley and Sons, Inc., 1968.
- [2] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. New York: John Wiley and Sons, 1991.
- [3] A. J. Goldsmith and P. P. Varaiya, "Capacity, mutual information and coding for finite state Markov channels," *IEEE Transactions on Information Theory*, vol. 42, pp. 868–886, May 1996.
- [4] V. Sharma and S. K. Singh, "Maximum likelihood sequence estimation in channels with intersymbol interference and noise memory," in *Proc. IEEE Int. Symp. Inform. Theory*, (Cambridge, MA), p. 366, Aug. 1998.
- [5] D. Arnold and H.-A. Loeliger, "On the information rate of binary-input channels with memory," in *Proceedings IEEE International Conference on Communications 2001*, (Helsinki, Finland), June 2001.
- [6] H. D. Pfister, J. B. Soriaga, and P. H. Siegel, "On the achievable information rates of finite state ISI channels," in *Proceedings IEEE Global communications Conference 2001*, (San Antonio, Texas), pp. 2992–2996, November 2001.
- [7] A. Kavčić, "On the capacity of Markov sources over noisy channels," in *Proceedings IEEE Global Communications Conference 2001*, (San Antonio, Texas), pp. 2997–3001, November 2001.
- [8] P. Vontobel and D. M. Arnold, "An upper bound on the capacity of channels with memory and constraint input," in *Proc. IEEE Information Theory Workshop*, (Cairns, Australia), September 2001.
- [9] S. Yang and A. Kavčić, "Markov sources achieve the feedback capacity of finite-state machine channels," in *Proc. IEEE Int. Symp. Inform. Theory*, (Lausanne, Switzerland), p. 361, July 2002.
- [10] S. C. Tatikonda, *Control Under Communications Constraints*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, September 2000.
- [11] S. Yang and A. Kavčić, "Capacity of partial response channels," in *Coding and Signal Processing for Magnetic Recording Systems* (B. Vasic and M. E. Kurtas, eds.), ch. 13, pp. 13.1–13.19, CRC Press, 2004.
- [12] G. D. Forney Jr., "Maximum-likelihood sequence estimation of digital sequences in the presence of intersymbol interference," *IEEE Transactions on Information Theory*, vol. 18, pp. 363–378, March 1972.
- [13] H. Kobayashi, "Application of probabilistic decoding to magnetic recording systems," *IBM J. Res. Dev.*, vol. 16, pp. 64–74, Jan. 1971.
- [14] R. W. Chang and J. C. Hancock, "On receiver structures for channels having memory," *IEEE Transactions on Information Theory*, vol. 12, pp. 463–468, October 1966.
- [15] L. E. Baum and T. Petrie, "Statistical inference for probabilistic functions of finite state Markov chains," *Ann. Math. Statist.*, vol. 37, pp. 1559–1563, 1966.
- [16] P. L. McAdam, L. R. Welch, and C. L. Weber, "M.A.P. bit decoding of convolutional codes," in *IEEE Int. Symp. Inform. Theory*, (Asilomar, CA), 1972.
- [17] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*. Englewood Cliffs: Prentice Hall, 1993.
- [18] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inform. Theory*, vol. 20, pp. 284–287, Sept. 1974.
- [19] N. Wiberg, H.-A. Loeliger, and R. Kötter, "Codes and iterative decoding on general graphs," *European Trans. on Commun.*, vol. 6, pp. 513–526, September 1995.

- [20] Y. Li, B. Vucetic, and Y. Sato, "Optimum soft-output detection for channels with intersymbol interference," *IEEE Trans. Inform. Theory*, vol. 41, pp. 704–713, May 1995.
- [21] S. M. Aji and R. J. McEliece, "The generalized distributive law," *IEEE Transactions on Information Theory*, vol. 46, pp. 325–343, March 2000.
- [22] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Transactions on Information Theory*, vol. 47, pp. 498–519, February 2001.
- [23] X. Ma and A. Kavčić, "Path partitions and forward-only trellis algorithms," *IEEE Trans. Inform. Theory*, pp. 38–52, January 2003.
- [24] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo-codes," in *Proc. IEEE Int. Conf. on Communications*, (Geneva, Switzerland), pp. 1064–1070, May 1993.
- [25] G. Ungerboeck, "Channel coding with multilevel/phase signals," *IEEE Transactions on Information Theory*, vol. 28, pp. 55–67, January 1982.
- [26] J. K. Wolf and G. Ungerboeck, "Trellis coding for partial-response channels," *IEEE Trans. Commun.*, vol. 34, pp. 744–765, Mar. 1986.
- [27] A. R. Calderbank, C. Heegard, and T. A. Lee, "Binary convolutional codes with application to magnetic recording," *IEEE Journal on Selected Areas in Communications*, vol. 32, pp. 797–815, November 1986.
- [28] K. A. S. Immink, "Coding techniques for the noisy magnetic recording channel: A state-of-the-art report," *IEEE Trans. Commun.*, vol. 35, pp. 413–419, May 1987.
- [29] R. Karabed and P. H. Siegel, "Matched spectral-null codes for partial response channels," *IEEE Trans. Inform. Theory*, vol. 37, pp. 818–855, May 1991.
- [30] J. Douillard, M. Jezequel, C. Berrou, A. Picart, P. Didier, and A. Glavieux, "Iterative correction of intersymbol interference: Turbo-equalization," *European Trans. on Commun.*, vol. 6, pp. 507–511, September 1995.
- [31] D. J. C. MacKay and R. M. Neal, "Near Shannon limit performance of low-density parity-check codes," *Electronic Letters*, vol. 32, pp. 1645–1646, 1996.
- [32] R. G. Gallager, *Low-Density Parity-Check Codes*. Cambridge, MA: MIT Press, 1962.
- [33] M. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. Spielman, "Improved low-density parity-check codes using irregular graphs," *IEEE Trans. Inform. Theory*, vol. 47, pp. 585–598, February 2001. also in *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, pages 249–258, 1998.
- [34] T. Richardson and R. Urbanke, "The capacity of low-density parity check codes under message-passing decoding," *IEEE Trans. Inform. Theory*, vol. 47, pp. 599–618, February 2001.
- [35] T. Richardson, A. Shokrollahi, and R. Urbanke, "Design of capacity-approaching low-density parity-check codes," *IEEE Trans. Inform. Theory*, vol. 47, pp. 619–637, February 2001.
- [36] S.-Y. Chung, G. D. Forney, T. Richardson, and R. Urbanke, "On the design of low-density parity-check codes within 0.0045 dB of the Shannon limit," *IEEE Communications Letters*, Feb. 2001.
- [37] A. Kavčić, X. Ma, and M. Mitzenmacher, "Binary intersymbol interference channels: Gallager codes, density evolution and code performance bounds," *IEEE Trans. Inform. Theory*, pp. 1636–1652, July 2003.
- [38] N. Varnica and A. Kavčić, "Optimized low-density parity-check codes for partial response channels," *IEEE Communications Letters*, vol. 7, pp. 168–170, April 2003.
- [39] J. Chen and P. H. Siegel, "Markov processes asymptotically achieve the capacity of finite-state intersymbol interference channels," in *Proc. IEEE International Symposium on Information Theory*, (Chicago, IL), p. 349, July 2004.
- [40] A. Vardy, "Trellis structure of codes," in *Handbook of Coding Theory* (V. S. Pless, W. C. Huffman, and R. A. Brualdi, eds.), Amsterdam, Netherlands: Elsevier Science Publishers, 1998.

- [41] G. D. Forney Jr., *Concatenated Codes*. Cambridge: M. I. T. Press, 1966.
- [42] Y.-C. Ho, "An explanation of ordinal optimization: Soft computing for hard problems," *Information Sciences*, vol. 113, pp. 169–192, Feb. 1999.
- [43] G. D. Forney Jr., "Codes on graphs: Normal realizations," *IEEE Trans. Inform. Theory*, vol. 47, pp. 520–548, February 2001.
- [44] H. Imai and S. Hirakawa, "A new multilevel coding method using error correcting codes," *IEEE Trans. Inform. Theory*, vol. 23, pp. 371–377, May 1977.
- [45] B. M. Tanner, "A recursive approach to low complexity codes," *IEEE Trans. Inform. Theory*, vol. 27, pp. 533–547, September 1981.
- [46] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Trans. Inform. Theory*, vol. 45, pp. 399–431, March 1999.