Capacity Approaching Signal Constellations for Channels with Memory*

Aleksandar Kavčić, Xiao Ma, Michael Mitzenmacher, and Nedeljko Varnica
Division of Engineering and Applied Sciences
Harvard University
33 Oxford Street
Cambridge, MA 02138
[kavcic,xiaoma,varnica]@hrl.harvard.edu, michaelm@eecs.harvard.edu

Abstract

The i.i.d. rate of a channel is the mutual information rate between the channel input and the channel output when the channel input symbols are independent, identically distributed (i.i.d.) and equally likely. We design a coding strategy that achieves rates higher than the i.i.d. rate of a partial response channel (intersymbol interference channel with inputs constrained to binary values), while still allowing simple encoding and decoding. The strategy breaks the code into an inner trellis code and an outer block code. This paper concentrates on the design of the inner trellis code.

The trellis code is designed according to sub-optimal Markov state transition probabilities which are determined by a rate-optimizing iterative algorithm so that the i.i.d. rate of the trellis code is near the channel capacity. We construct the trellis code using integer approximations to these transition probabilities. This yields a signal constellation that reduces the shaping loss, so that a properly constructed outer code can achieve rates near the channel capacity. We demonstrate that the proposed trellis code, with an appropriately optimized outer irregular low-density parity-check code, achieves rates that surpass the channel i.i.d. information rate.

1 Introduction

The computation of the capacity of a finite-input-alphabet channel with intersymbol interference (ISI) memory is a long-standing open problem [1]. Recent breakthroughs enable the computation of tight upper and lower channel capacity bounds. Arnold and Loeliger [2], and independently Pfister, Soriaga and Siegel [3], devise a Monte-Carlo method to compute the information rates of constrained-input ISI channels if the channel input process forms a Markov chain. This method can be used to compute lower bounds on the channel capacity as well as the i.i.d. information rate of the channel. In [4], Kavčić proposes an iterative method (very much resembling the Arimoto-Blahut algorithm [5, 6]) that optimizes the transition probabilities of the input Markov chain. The method achieves a high information rate, thus tightening the capacity lower bound. The channel output distributions computed using the channel inputs generated in this manner can

*This research is supported in part by NSF Grant No. CCR-0118701

also be used to construct tight upper bounds on the channel capacity, as was shown by Vontobel and Arnold [7].

With these advances in the computation of bounds on the channel capacity, the question arises whether practical codes can be constructed to achieve near-capacity performance. In [8] we have shown that low-density parity-check (LDPC) coset codes constructed over random graphs can at most achieve rates as high as the i.i.d. channel rate, but not higher. In this paper we propose a general coding strategy that can surpass the channel's i.i.d. rate and may ultimately reach the capacity of finite-input-alphabet channels with ISI memory. The basic idea for the method we propose is rather old - we use an inner trellis code and outer block code (say, an LDPC code with an appropriate iterative decoding algorithm).

In this paper we concentrate on the design of the inner code, which we construct as a trellis code. Most previous trellis code constructions are based on either maximizing the minimum free distance [9, 10] or matching the spectral nulls of the channel if the channel has spectral nulls [11], or other lattice-based constructions [12, 13]. We use a new approach to construct the inner trellis code. We design the trellis code using the notion of the *trellis code capacity* C_T. To define the trellis code capacity C_T, we construct a super-channel consisting of the trellis code concatenated onto the channel we are using. The trellis code capacity C_T is defined as the information rate between the super-channel input sequence A_t and the super-channel output sequence Y_t, when the super-channel input symbols A_t are equally likely and i.i.d. See Figure 1 for an illustration.

We construct the inner trellis code such that the trellis code capacity C_T matches the channel capacity lower bound C_L, as computed by the method in [4]. The tighter our computed lower bounds C_L are, the closer we can make the trellis code capacity C_T of our constructed trellis code to the channel capacity. Our approach demonstrates that we can construct an acceptably simple trellis code with a relatively low signal shaping loss, i.e., we construct near-capacity signal constellations on a trellis.

Combined with an appropriate outer code construction, we provide an example of a code that surpasses the dicode $(1-D)$ channel's i.i.d. rate. Asymptotic analysis based on density evolution shows that our code performs within 0.2dB of the channel capacity lower bound and within 0.7dB away from the channel capacity upper bound.

2 An information rate optimizing algorithm

Let r represent the target code rate for the code we want to design. Our strategy assumes an inner code with rate r_{in} and an outer code with rate r_{out}. Obviously, we must have $r_{\text{in}} \cdot r_{\text{out}} = r$. Further, we assume that the outputs of the outer code are equally likely i.i.d. binary random variables. In a strict sense, this assumption cannot be satisfied with any implementable code. However, for turbo codes with large interleaver sizes or for randomly generated LDPC codes with large block lengths, the neighboring bits will appear very nearly independent. Hence this assumption is suitable as a design heuristic for these codes. We focus now on the inner code.

The main tool used for the construction of the inner code is an iterative algorithm for optimizing the information rates of Markov processes over noisy channels [4]. The algorithm can be represented as an expectation-maximization procedure, where the expectation step is estimated using the sum-product (BCJR, Baum-Welch) algorithm [14] and the maximization step is computable in closed form. No proof is known to show that the algorithm achieves the maximal information rate of the Markov process. However,

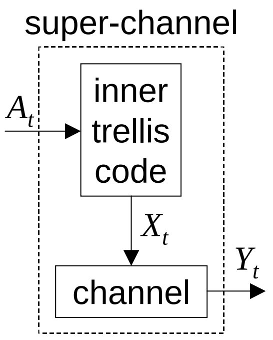

Figure 1: The super-channel consisting of a trellis code and the channel.

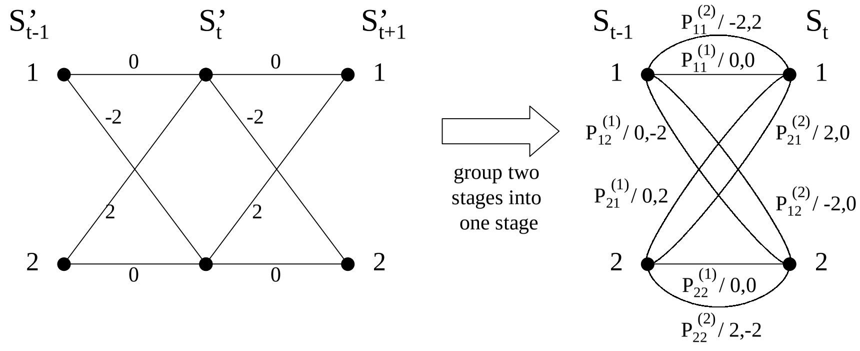

Figure 2: Left-hand side - two sections of the minimal trellis representation of the dicode $(1-D)$ channel. Right-hand side - two trellis sections merged into a single trellis section (the n-th order channel trellis extension; here $n=2$).

numerical tests on low-order Markov processes suggest that it does [4]. Furthermore, in some limiting cases, the algorithm is equivalent to well-known algorithms that are already known to deliver the capacity. Namely, our iterative algorithm becomes the Arimoto-Blahut algorithm [5, 6] if the channel is memoryless, and it becomes Shannon's Markov-chain entropy maximization [15] if the noise vanishes.

We briefly expose (without derivations) the iterative procedure. Unlike [4], the version exposed here allows for multiple branches between a pair of trellis states.

For the iterative algorithm, we assume the channel input is a Markov chain, possibly with multiple branches between a pair of states. For a partial response channel where the input values are confined to a binary alphabet, we can construct such a process by grouping several stages of the trellis into one stage. This is shown on the right-hand side of Figure 2 for the dicode channel (whose channel polynomial equals $H(D) = 1-D$). We denote by $P_{ij}^{(\ell)}$ the conditional probability of going to state j via the ℓ-th branch, given that the process is in state i. In Figure 2 the notation $P_{ij}^{(\ell)}/a,b$ denotes that the branch is taken with probability $P_{ij}^{(\ell)}$ while the noiseless pair of channel output symbols is $[a,b]$. Denote by μ_i the stationary probability of the process being in state $S_t = i$. Obviously, we must have $\sum_{i,\ell} \mu_i P_{ij}^{(\ell)} = \mu_j$ for any state j.

Let the trellis consist of τ segments, where each segment may have parallel branches (as on the right-hand side of Figure 2). Assume that the probabilities $P_{ij}^{(\ell)}$ are given for all branches of the trellis stage. Using the probabilities $P_{ij}^{(\ell)}$, create the channel input realizations corresponding to τ trellis stages, and pass them through the noisy channel to observe a vector of channel output realizations $\underline{y}$. Let B_t stand for the branch at time t. Denote the a posteriori branch and state probabilities at time t as

$$P_t^{(\ell)}(i,j|\underline{y}) = \Pr\left(S_{t-1}=i, S_t=j, B_t=\ell|\underline{y}\right), \quad (1)$$
$$P_t(i|\underline{y}) = \Pr\left(S_{t-1}=i|\underline{y}\right), \quad (2)$$

respectively. The branch and state probabilities in (1) and (2) can be found using the sum-product (BCJR, Baum-Welch) algorithm [14]. Next, define

$$\hat{T}_{ij}^{(\ell)} = \frac{1}{\tau}\sum_{t=1}^{\tau}\left[\frac{P_t^{(\ell)}(i,j|\underline{y})}{\mu_i P_{ij}^{(\ell)}}\log_2 P_t^{(\ell)}(i,j|\underline{y}) - \frac{P_t(i|\underline{y})}{\mu_i}\log_2 P_t(i|\underline{y})\right]. \quad (3)$$

Algorithm 1 Iterative optimization of Markov chain transition probabilities.

```
Initialization  Pick an arbitrary distribution $P_{ij}^{(\ell)}$ that satisfies the following two constraints: 1) $0 \leq P_{ij}^{(\ell)} \leq 1$ if the process can be taken from state $i$ to state $j$ via branch $\ell$; otherwise $P_{ij}^{(\ell)} = 0$ and 2) for any $i$, require that $\sum_{j,\ell} P_{ij}^{(\ell)} = 1$.
Repeat  until convergence
    1. For $\tau$ large, generate $\tau$ noiseless trellis outputs according to the transition probabilities $P_{ij}^{(\ell)}$ and pass them through the noisy channel to get $\underline{y}$.
    2. Run the forward-backward sum-product (Baum-Welch, BCJR) algorithm and compute $\hat{T}_{ij}^{(\ell)}$ according to (3).
    3. Compute the estimate of the noisy adjacency matrix
       $\hat{A}_{ij} = \begin{cases} 2^{\sum_\ell \hat{T}_{ij}^{(\ell)}} & \text{if states } i \text{ and } j \text{ are connected} \\ 0 & \text{otherwise} \end{cases}$,
       and find its maximal eigenvalue $\hat{W}_{max}$ and the corresponding eigenvector $\left[\hat{b}_1, \hat{b}_2, \cdots, \hat{b}_M\right]^{\mathrm{T}}$.
    4. Compute the new transition probabilities as $P_{ij}^{(\ell)} = \frac{\hat{b}_j}{\hat{b}_i} \cdot \frac{2^{\hat{T}_{ij}^{(\ell)}}}{\hat{W}_{max}}$.
end
```

Table 1: Iterative optimization of Markov chain transition probabilities.

(See [4] for the relation between this quantity and the information rate.)

We use Algorithm 1 (in Table 1) to optimize the transition probabilities to achieve a high information rate. After many iterations, Algorithm 1 typically settles to a fixed point and we obtain the optimized transition probabilities $P_{ij}^{(\ell)}$. Using these optimized transition probabilities $P_{ij}^{(\ell)}$, we estimate the optimized information rate as $C_L = \sum_{i,j,\ell} \mu_i P_{ij}^{(\ell)} \left[\log_2 \frac{1}{P_{ij}^{(\ell)}} + \hat{T}_{ij}^{(\ell)}\right]$. See [4] for a derivation. The index L in C_L denotes that the computed rate is a lower bound on the channel capacity.

3 Choosing the inner code rate

In this section we argue that there should be an upper bound on the inner code rate r_{in}, given our construction strategy. An obvious lower bound is $r_{\text{in}} \geq r$, where r is the target code rate. We argue that we should have $r_{\text{in}} < 1$, for the following reason. By Proposition 1 in [8], when $r_{\text{in}} = 1$, the maximal achievable rate, or *threshold*, of an outer LDPC code is the i.i.d. rate of the channel. But we desire to design an outer LDPC code that surpasses the i.i.d. rate of the channel. We therefore need $r_{\text{in}} < 1$. We next formulate a rule that further tightens the upper bound on r_{in}.

The upper bound that we advocate here is a design principle that eliminates the possibility that two paths of the inner trellis code start and end at the same state and correspond to two different input sequences, which would present ambiguities for decoding. We assume that the inner code is a trellis code with K states that maps an input symbol k-tuple into a noiseless channel output n-tuple. Thus the inner code rate is $r_{\text{in}} = k/n$. We form the n-th order extension of the channel trellis (e.g., Figure 2). We apply Algorithm 1 to the n-th order extension of the trellis to find the optimized transition probabilities $P_{ij}^{(\ell)}$. With these probabilities determined, we require that the inner code rate be

$$r \le r_{\text{in}} \le \frac{1}{n} \min_{i,j,\ell} \left[\log_2 \frac{1}{P_{ij}^{(\ell)}} \right], \tag{4}$$

where the minimum is taken over all branches of the n-th order channel trellis extension.

The upper bound in rule (4) is equivalent to $2^{-k} \ge \max_{i,j,\ell} P_{ij}^{(\ell)}$. To see the reasoning behind this rule, we will assume that it does not hold, i.e., we will assume that $P_{max} = \max_{i,j,\ell} P_{ij}^{(\ell)} > 2^{-k}$. Since the trellis code is a mapping from binary k-tuples onto noiseless channel output n-tuples, each state of the K trellis code states will have 2^k branches leaving the state. Since we assume i.i.d. inputs to the the trellis encoder, the conditional probability of each branch is 2^{-k}. The goal of our trellis code design is to construct a trellis code such that the frequencies of branch occurrences match as closely as possible the probabilities of branch occurrences of the optimal n-th order channel trellis extension as computed by Algorithm 1. That is, if a branch occurs with conditional probability $P_{ij}^{(\ell)}$ in the n-th order trellis extension, the occurrence frequency of the same branch in the trellis code should be as close as possible to $P_{ij}^{(\ell)}$. If $P_{max} > 2^{-k}$, then in our designed trellis code two branches from a trellis code state may be necessary to match a transition in the n-th order channel trellis extension. That is, two branches of the trellis code would have the same starting state and carry the same noiseless channel output n-tuple, but have different input k-tuples. This setup has the potential to violate unique decodability, for example in the case where both of these branches also end in the same state of the trellis decoder. Obeying rule (4) provides a guarantee that this scenario will be avoided.

In order to keep the complexity of the encoder and decoder small, we seek the smallest possible integers k and n for which the inner code rate $r_{\text{in}} = k/n$ satisfies (4). Note that there is a tradeoff here; larger k and n allow for better but more complex trellis codes. It is not clear whether there exist integers k and n that satisfy (4) for any channel. However, for several channels that we tried with $r = 0.5$, condition (4) was easily satisfied with small integers k and n.

We give an example on the 2-state representation of the dicode $(1 - D)$ channel. The channel noise is assumed to be Gaussian and white with variance σ^2. The target code rate was chosen to be $r = 0.5$. For this target code rate, condition (4) is satisfied with $k = 2$ and $n = 3$. The value of the signal-to-noise ratio (SNR) for which the n-th order trellis information rate C_L equals $r = 0.5$ is $10 \log_{10} (2/\sigma^2) = 0.35$dB. The values of the obtained transition probabilities are given in Table 2; they are the transition probabilities for which the optimized information rate C_L equals $r = 0.5$ (for another target rate r, the numbers k and n and the transition probabilities would take different values).

4 Integer approximations of transition probabilities

Our desire is to design a trellis code that maps k input bits into n noiseless channel output symbols. We also desire that the state occurrence probabilities and the conditional branch occurrence probabilities in the trellis code match those computed by Algorithm 1 for the n-th order channel trellis extension (i.e., we want them to match μ_i and $P_{ij}^{(\ell)}$). To keep the complexity of the trellis code in check, we also impose the constraint that the number of states in the trellis code be K, where K is a predetermined integer.

start state	end state	branch #	channel input n-tuple	noiseless channel output n-tuple	transition probability	integer approximations $k=2$, $K=10$ $k_1=5$, $k_2=5$	integer approximation probabilities
1	1	1	0,0,0	0, 0, 0	$P_{11}^{(1)} = 0.005$	$n_{11}^{(1)} = 0$	$n_{11}^{(1)} / \left(k_1 \cdot 2^k\right) = 0.00$
1	1	2	1,0,0	-2, 2, 0	$P_{11}^{(2)} = 0.146$	$n_{11}^{(2)} = 3$	$n_{11}^{(2)} / \left(k_1 \cdot 2^k\right) = 0.15$
1	1	3	0,1,0	0, -2, 2	$P_{11}^{(3)} = 0.146$	$n_{11}^{(3)} = 3$	$n_{11}^{(3)} / \left(k_1 \cdot 2^k\right) = 0.15$
1	1	4	1,1,0	-2, 0, 2	$P_{11}^{(4)} = 0.195$	$n_{11}^{(4)} = 4$	$n_{11}^{(4)} / \left(k_1 \cdot 2^k\right) = 0.20$
1	2	1	0,0,1	0, 0, -2	$P_{12}^{(1)} = 0.066$	$n_{12}^{(1)} = 1$	$n_{12}^{(1)} / \left(k_1 \cdot 2^k\right) = 0.05$
1	2	2	1,0,1	-2, 2, -2	$P_{12}^{(2)} = 0.231$	$n_{12}^{(2)} = 5$	$n_{12}^{(2)} / \left(k_1 \cdot 2^k\right) = 0.25$
1	2	3	0,1,1	0, -2, 0	$P_{12}^{(3)} = 0.145$	$n_{12}^{(3)} = 3$	$n_{12}^{(3)} / \left(k_1 \cdot 2^k\right) = 0.15$
1	2	4	1,1,1	-2, 0, 0	$P_{12}^{(3)} = 0.066$	$n_{12}^{(4)} = 1$	$n_{12}^{(4)} / \left(k_1 \cdot 2^k\right) = 0.05$
2	1	1	0,0,0	2, 0, 0	$P_{21}^{(1)} = 0.066$	$n_{21}^{(1)} = 1$	$n_{21}^{(1)} / \left(k_2 \cdot 2^k\right) = 0.05$
2	1	2	1,0,0	0, 2, 0	$P_{21}^{(2)} = 0.145$	$n_{21}^{(2)} = 3$	$n_{21}^{(2)} / \left(k_2 \cdot 2^k\right) = 0.15$
2	1	3	0,1,0	2, -2, 2	$P_{21}^{(3)} = 0.231$	$n_{21}^{(3)} = 5$	$n_{21}^{(3)} / \left(k_2 \cdot 2^k\right) = 0.25$
2	1	4	1,1,0	0, 0, 2	$P_{21}^{(4)} = 0.066$	$n_{21}^{(4)} = 1$	$n_{21}^{(4)} / \left(k_2 \cdot 2^k\right) = 0.05$
2	2	1	0,0,1	2, 0, -2	$P_{22}^{(1)} = 0.195$	$n_{22}^{(1)} = 4$	$n_{22}^{(1)} / \left(k_2 \cdot 2^k\right) = 0.20$
2	2	2	1,0,1	0, 2, -2	$P_{22}^{(2)} = 0.146$	$n_{22}^{(1)} = 3$	$n_{22}^{(2)} / \left(k_2 \cdot 2^k\right) = 0.15$
2	2	3	0,1,1	2, -2, 0	$P_{22}^{(3)} = 0.146$	$n_{22}^{(3)} = 3$	$n_{22}^{(3)} / \left(k_2 \cdot 2^k\right) = 0.15$
2	2	4	1,1,1	0, 0, 0	$P_{22}^{(3)} = 0.066$	$n_{22}^{(4)} = 0$	$n_{22}^{(4)} / \left(k_2 \cdot 2^k\right) = 0.05$

Table 2: Optimized transition probabilities for the 3-rd order extension of the dicode channel ($1-D$ channel) trellis, and good integer approximations.

Assume that the n-th order extension of the channel trellis has M states. We desire to find M integers $k_1, \ldots, k_M$, such that $\sum_{i=1}^{M} k_i = K$ and that the fractions k_i/K are as close as possible to the state probabilities μ_i. Thus state i of the n-th order extension of the channel trellis is replaced by k_i states in the trellis code.

Consider now state i in the n-th order extension of the channel trellis. There is a total of 2^n branches leaving this state, with transition probabilities $P_{ij}^{(\ell)}$ that satisfy $\sum_{j,\ell} P_{ij}^{(\ell)} = 1$. State i will be replaced by k_i states in the trellis code. The group of k_i states that replace state i must have a total of $k_i \cdot 2^k$ branches exiting the group, since the code maps input bit k-tuples to noiseless channel output n-tuples. For each state i of the n-th order channel trellis extension, we desire to find 2^n integers $n_{ij}^{(\ell)}$, such that $\sum_{j,\ell} n_{ij}^{(\ell)} = k_i \cdot 2^k$ and that the fractions $n_{ij}^{(\ell)} / \left(k_i \cdot 2^k\right)$ are as close as possible to the transition probabilities $P_{ij}^{(\ell)}$.

To find the integers k_i and $n_{ij}^{(\ell)}$, we must have an optimization criterion. Here we choose to minimize the Kullback-Leibler distance between two Markov processes: the first process has state probabilities k_i/K and transition probabilities $n_{ij}^{(\ell)} / \left(k_i \cdot 2^k\right)$, while the second process has state probabilities μ_i and transition probabilities $P_{ij}^{(\ell)}$. That is, we desire to find the set of non-negative integers $\left\{k_i, n_{ij}^{(\ell)}\right\}$ such that

$$\left\{k_i, n_{ij}^{(\ell)}\right\} = \arg \min_{\left\{k_i, n_{ij}^{(\ell)}\right\}} \sum_{i,j,\ell} \frac{k_i}{K} \cdot \frac{n_{ij}^{(\ell)}}{k_i \cdot 2^k} \cdot \log_2 \frac{\dfrac{n_{ij}^{(\ell)}}{k_i \cdot 2^k}}{P_{ij}^{(\ell)}}, \tag{5}$$

where the minimum in (5) is taken under the constraints

$$\sum_{i=1}^{M} k_i = K \qquad \text{and} \qquad \sum_{j,\ell} n_{ij}^{(\ell)} = k_i \cdot 2^k. \tag{6}$$

The objective function in (5) is not necessarily the optimal one to minimize for finite K. One could also choose a different objective function, such as the mean squared error. When $K \to \infty$, a good objective function should yield solutions that asymptotically approach $k_i/K \to \mu_i$ and $n_{ij}^{(\ell)} / \left(k_i \cdot 2^k\right) \to P_{ij}^{(\ell)}$. The Kullback-Leibler objective function in (5) certainly exhibits this asymptotic behavior (as does the mean squared error).

The integer optimization in (5) can be solved either by integer programming techniques or in the case of the dicode channel here by a simple greedy algorithm. We optimized the integers when the channel is the dicode $(1-D)$ channel, the trellis code rate is $r_{\text{in}} = k/n = 2/3$, and the chosen number of trellis code states is $K = 10$. The optimal values are $k_1 = k_2 = 5$, and the values $n_{ij}^{(\ell)}$ are given in Table 2.

5 Permutations increase the trellis code capacity

The integer optimization from the previous section gives us the optimal number of branches of each type to use in the formation of the trellis code. For example, by reading the integers in Table 2, we conclude that 4 branches carrying the noiseless output n-tuple $[-2, 0, 2]$ should be used in the trellis code. However, we still do not know how to connect these branches to the states of the trellis code. In fact, there are many possible ways of creating a trellis code with the integer assignment in Table 2. The question arises as to which connection assignment is the best.

Our goal is to design a trellis code whose trellis code capacity C_T is maximized. The maximum possible trellis code capacity is the rate C_L computed by Algorithm 1 (although C_T will actually be lower than C_L because of the integer approximations). For any trellis code concatenated to a finite-state machine channel, the trellis code capacity C_T can be determined by the Arnold-Loeliger Monte-Carlo method [2]. For the dicode $(1-D)$ channel example in Table 2, the maximal trellis code capacity of $k/n = 2/3$ is achievable if the signal-to-noise ratio (SNR) approaches infinity. However, recall that our specific goal in designing this inner code is to find a trellis with capacity close to $r = 0.5$ at SNR $= 0.35$dB (which is the SNR at which the n-th order channel trellis extension rate C_L reaches rate 0.5; see Figure 3 in Section 6 for further clarification). It is not clear which connection of branches delivers the best trellis for this situation.

There is no known solution to this problem. However, a random search over all branch connection assignments (which we call *permutations*) typically delivers results that are acceptable. We simply pick a random assignment that has the determined integer coefficients k_i and $n_{ij}^{(\ell)}$ and compute the trellis code capacity C_T of the resulting trellis code using the Arnold-Loeliger method [2] with i.i.d. input k-tuples. If the trellis code capacity C_T is close to the desired code rate C_L (at the point where $C_L = r$), we stop; otherwise, we continue the random search. To avoid creating a problem for the decoder, we ensure that two branches leaving the same state never carry the same noiseless channel output n-tuple. This random search strategy delivered the trellis code given in Table 3. The trellis code in Table 3 has $K = 10$ states, and the integer occurrences of branches match the values $n_{ij}^{(\ell)}$ in Table 2.

The trellis code capacity of the trellis code in Table 3 is plotted in Figure 3 - curve C_T. Also plotted in Figure 3 are: C_L - the lower bound on the channel capacity (computed

start state	input k-tuple of bits	output n-tuple of bits	end state	noiseless channel output n-tuple
1	0,0	0,1,0	1	0, -2, 2
1	0,1	1,0,0	5	-2, 2, 0
1	1,0	0,0,1	6	0, 0, -2
1	1,1	1,0,1	9	-2, 2, -2
2	0,0	1,1,0	2	-2, 0, 2
2	0,1	1,0,0	4	-2, 2, 0
2	1,0	0,1,1	9	0, -2, 0
2	1,1	1,0,1	8	-2, 2, -2
3	0,0	1,1,0	3	-2, 0, 2
3	0,1	1,0,0	2	-2, 2, 0
3	1,0	0,1,1	10	0, -2, 0
3	1,1	1,0,1	7	-2, 2, -2
4	0,0	0,1,0	4	0, -2, 2
4	0,1	1,1,0	1	-2, 0, 2
4	1,0	1,1,1	7	-2, 0, 0
4	1,1	1,0,1	6	-2, 2, -2
5	0,0	0,1,0	3	0, -2, 2
5	0,1	1,1,0	5	-2, 0, 2
5	1,0	0,1,1	8	0, -2, 0
5	1,1	1,0,1	10	-2, 2, -2
6	0,0	0,1,0	1	2, -2, 2
6	0,1	1,0,0	3	0, 2, 0
6	1,0	0,0,1	6	2, 0, -2
6	1,1	1,0,1	8	0, 2, -2
7	0,0	0,1,0	5	2, -2, 2
7	0,1	0,0,0	4	2, 0, 0
7	1,0	0,0,1	10	2, 0, -2
7	1,1	1,0,1	7	0, 2, -2
8	0,0	0,1,0	4	2, -2, 2
8	0,1	1,0,0	1	0, 2, 0
8	1,0	0,1,1	9	2, -2, 0
8	1,1	0,0,1	8	2, 0, -2
9	0,0	0,1,0	3	2, -2, 2
9	0,1	1,0,0	2	0, 2, 0
9	1,0	0,1,1	7	2, -2, 0
9	1,1	0,0,1	9	2, 0, -2
10	0,0	0,1,0	2	2, -2, 2
10	0,1	1,1,0	5	0, 0, 2
10	1,0	0,1,1	6	2, -2, 0
10	1,1	1,0,1	10	0, 2, -2

Table 3: A 10-state trellis code for the dicode $(1-D)$ channel, obtained by randomly searching for the permutation that has the largest trellis capacity (the trellis capacity of this trellis code is plotted in Figure 2). Note that the random permutations preserve the integers in Table 2. Consequently, the integer occurrences of the noiseless channel output n-tuples match the values $n_{ij}^{(\ell)}$ in Table 2.

by Algorithm 1 on the 3rd order extension of the channel trellis), C_U - the Vontobel-Arnold upper bound on the channel capacity [7], and C_0 - the i.i.d. rate of the dicode channel. Notice that at the target SNR of 0.35dB, the trellis code capacity C_T is only 0.05dB away from the bound C_L in the region $r = 1/2$. The bound C_L can be further increased by about 0.2dB by taking channel trellis extensions with more states and larger n. Correspondingly, the trellis code capacity C_T can also be increased by about 0.2dB by increasing k and n to be larger than the minimal values that satisfy condition (4) and by increasing the number of states K in the trellis code. The price paid is a higher complexity of the trellis code encoder/decoder.

The significance of the trellis code capacity C_T is two-fold. First, if the trellis code is designed properly, the trellis code capacity C_T can be higher than the channel i.i.d. rate C_0 in the region of interest (here, the region of interest was around $r = 0.5$), as seen in Figure 3. Second, the trellis code capacity C_T can be achieved by a random linear outer code (see [8], Theorem 3), and it is an upper bound for the thresholds of random LDPC coset codes (see [8], Proposition 1). Approaching C_T arbitrarily closely with an outer LDPC code requires optimizing the edge degree distributions [16, 17]. This task lies beyond the scope of this paper, but the threshold of an optimized LDPC code is plotted in Figure 3 to demonstrate the feasibility of the design.

6 Code construction examples

Figure 3 shows the trellis code capacity curve (C_T) for the inner trellis code with rate $r_{\text{in}} = 2/3$ (Table 3) constructed for the dicode $(1-D)$ channel and a target rate $r = 0.5$. The point marked by 'o' shows the position of a the threshold for an optimized outer irregular LDPC code with rate $r_{\text{out}} = 0.75$ designed to approach the trellis code capacity C_T at rate $r = r_{\text{in}} \cdot r_{\text{out}} = 0.5$. The design of this outer LDPC code falls beyond the scope

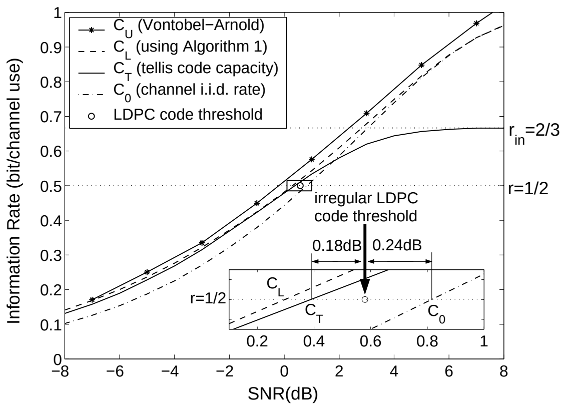

Figure 3: Shown are upper (C_U) and lower (C_L) bounds on the channel capacity of the dicode $(1 - D)$ channel. Also shown are the channel's i.i.d. information rate (C_0) and the trellis code capacity C_T) of the trellis code in Table 3. The point marked by 'o' shows the threshold location of an optimized outer irregular LDPC code concatenated onto the inner trellis code of Table 3.

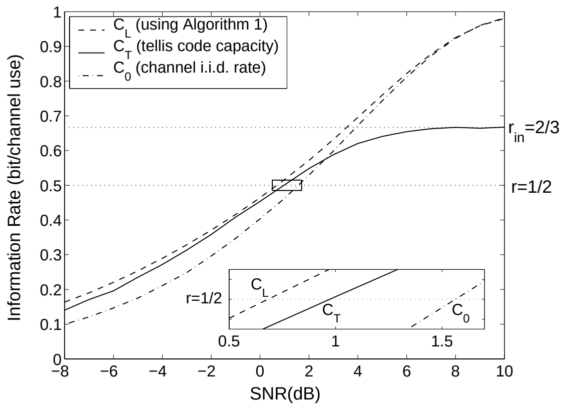

Figure 4: Shown is a tight lower bound (C_L) on the channel capacity of the $1 - D + 0.8D^2$ channel, the channel i.i.d. information rate (C_0) and the trellis code capacity (C_T) of a constructed trellis code with 12 states. Though we did not construct an outer LDPC code for this channel, we conjecture that a properly optimized LDPC code could have a threshold in the gap between C_0 and C_T.

of this paper. We just note that the code consists of two constituent irregular LDPC codes (because $k = 2$). Note that the code threshold is well above the i.i.d. information rate, but below the trellis code capacity C_T as predicted by Proposition 1 in [8]. This demonstrates that the trellis code design method proposed in this paper can indeed be used to construct codes that perform above the i.i.d. information rate of partial response channels (and perhaps ultimately reach the capacity by constructing more complex trellis codes and matching outer LDPC codes to them).

Figure 4 shows the curves for a similarly designed trellis code for the partial response channel with the channel response polynomial $G(D) = 1 - D + 0.8D^2$. This channel (unlike the $1 - D$ dicode channel) does not have a spectral null, showing that the trellis code construction method applies equally well to channels with and without spectral nulls. Though we did not construct an outer LDPC code for the inner trellis code of Figure 4, we conjecture that a properly optimized LDPC code could be constructed to have the threshold in the gap between C_0 and C_T.

7 Conclusion

We have proposed a strategy for code design for finite-input-alphabet channels with intersymbol interference memory. The strategy consists of splitting the capacity approaching code into an inner trellis code and an outer block code. The main contribution of this paper is a novel method for constructing inner trellis codes. We construct the inner trellis code such that its trellis code capacity C_T matches the channel capacity (or a close lower bound) at the design rate r of interest. The main tool used for the trellis code construction is an iterative Monte Carlo algorithm for computing trellis transition probabilities. We disclosed a trellis code for the dicode $(1 - D)$ channel whose trellis code capacity C_T surpasses the channel i.i.d. information rate C_0 by 0.42dB. We also demonstrated that with a properly optimized irregular LDPC code we can achieve within 0.18dB of the code capacity C_T at rate $r = 0.5$. (Degree coefficient distributions and the methodology for finding them will be the subject of a future paper.) Although we cannot compute the channel capacity exactly, we can guarantee that this code performs within 0.7dB of the channel capacity since its noise tolerance threshold is about 0.7dB away from the tightest known upper bound on the channel capacity.

References

[1] K. A. S. Immink, P. H. Siegel, and J. K. Wolf, "Codes for digital recorders," *IEEE Trans. Inform. Theory*, vol. 44, pp. 2260–2299, October 1998.

[2] 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.

[3] 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), November 2001.

[4] A. Kavčić, "On the capacity of Markov sources over noisy channels," in *Proceedings IEEE Global Communications Conference 2001*, (San Antonio, Texas), November 2001. available at http://hrl.harvard.edu/˜kavcic/publications.html.

[5] S. Arimoto, "An algorithm for computing the capacity of arbitrary discrete memoryless channels," *IEEE Transactions on Information Theory*, vol. 18, pp. 14–20, January 1972.

[6] R. E. Blahut, "Computation of channel capacity and rate-distortion functions," *IEEE Transactions on Information Theory*, vol. 18, pp. 460–473, July 1972.

[7] P. Vontobel and D. M. Arnold, "An upper bound on the capacity of channels with memory and contraint input," in presented at *IEEE Information Theory Workshop*, (Cairns, Australia), September 2001.

[8] A. Kavčić, X. Ma, and M. Mitzenmacher, "Binary intersymbol interference channels: Gallager codes, density evolution and code performance bounds." 41 pages, submitted for publication in *IEEE Trans. Inform. Theory*;
available at http://hrl.harvard.edu/˜kavcic/publications.html, February 2001.

[9] G. Ungerboeck, "Channel coding with multilevel/phase signals," *IEEE Transactions on Information Theory*, vol. 28, pp. 55–67, January 1982.

[10] J. K. Wolf and G. Ungerboeck, "Trellis coding for partial-response channels," *IEEE Trans. Commun.*, vol. 34, pp. 765–773, August 1986.

[11] 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.

[12] G. D. Forney, "Trellis shaping," *IEEE Transactions on Information Theory*, vol. 28, pp. 281–300, March 1992.

[13] M. V. Eyuboğlu and G. D. Forney, "Trellis precoding: Combined coding, precoding and shaping for intersymbol interference channels," *IEEE Transactions on Information Theory*, vol. 28, pp. 301–314, March 1992.

[14] 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.

[15] C. E. Shannon, "A mathematical theory of communications," *Bell Systems Technical Journal*, vol. 27, pp. 379–423 (part I) and 623–656 (part II), 1948.

[16] 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.

[17] 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." accepted for publication in *IEEE Communications Letters*; available at http://lcavwww.epfl.ch/publications.html., 2000.