Trellis Coding for High Signal-to-Noise Ratio Gaussian Noise Channels

Erdal Arıkan
Bilkent University, Ankara, Turkey

Abstract

It is known that under energy constraints it is best to have each codeword of a code satisfy the constraint with equality, rather than having the constraint satisfied only in an average sense over all codewords. This suggests the use of fixed-composition codes on additive gaussian noise channels, for which the coding gains achievable by this method are significant, especially in the high SNR case. Here, we examine the possibility of achieving these gains by using fixed-composition trellis codes.

1. The Problem

Consider a discrete-time, memoryless, additive gaussian noise channel $y = x + z$ where y is the channel output; x, an arbitrary real number, is the channel input; and z is a gaussian random variable with mean 0 and variance σ^2. A block code with M codewords and blocklength n is said to satisfy an *average power constraint* if

$$\sum_{m=1}^{M}(1/M)\sum_{i=1}^{n}x_{m,i}^2 \leq nP, \tag{1}$$

where $x_{m,i}$ is the i'th letter of the m'th codeword, and P is a given constant.

Shannon [1] showed that if one wishes to minimize the probability of maximum-likelihood decoding error under an average power constraint, one may restrict attention to codes satisfying the (more restrictive) *shell constraint*, i.e. codes for which each codeword satisfies

$$x_{m,1}^2 + \cdots + x_{m,n}^2 = nP. \tag{2}$$

Shannon's results indicate that a block code selected at random from the ensemble of shell-constrained codes is near-optimal with probability approaching one in the limit as the blocklength goes to infinity. Unfortunately, no practical method is known for decoding near-optimal block codes.

Shell-constrained *trellis* codes are more promising in this regard, since they can be decoded by sequential decoding at least at rates below the computational cutoff rate. The purpose of this paper is to examine such trellis codes.

2. Shell-Constrained Trellis Codes

A trellis code is said to satisfy a shell constraint if each path $x_1, x_2, \ldots$ through the trellis satisfies

$$x_{kn+1}^2 + \cdots + x_{(k+1)n}^2 = nP \tag{3}$$

for every integer $k \geq 0$, where n and P are given constants.

Since we consider using a sequential decoder, the channel parameter of primary interest here is the cutoff rate. For the ensemble of shell-constrained trellis codes, the cutoff rate (in bits) is given by [1], [2]

$$R_0^* = \frac{\log e}{2}\left(1 + A/2 - \sqrt{1 + A^2/4}\,\right) + \frac{1}{2}\log\left[\frac{1}{2}\left(1 + \sqrt{1 + A^2/4}\,\right)\right] \tag{4}$$

where $A = P/\sigma^2$ is the SNR, e is the base of natural logarithm, and the logarithms are to base 2 throughout the paper.

Table 1 lists the capacity $C = (1/2)\log(1 + A)$ and the cutoff rate R_0^* for various values of A. Also listed in Table 1 is $R_0' = (1/2)\log(1 + A/2)$, which

0196 CH2681-5/89/0000-0196 $1.00 © 1989 IEEE

A	0.1	1	5	10	20	100
C	0.069	0.50	1.29	1.73	2.20	3.33
R_0^*	0.036	0.32	1.02	1.45	1.92	3.05
R_0'	0.035	0.29	0.90	1.29	1.73	2.84

Table 1: Capacity and cutoff rates.

is the cutoff rate for the *independent-letters* ensemble in which the trellis symbols are selected independently from a gaussian distribution with mean 0 and variance P. (The choice of gaussian distribution is somewhat arbitrary here; it does not maximize the cutoff rate of independent-letters ensembles.)

While the shell-constrained ensemble yields the largest possible cutoff rate for a given SNR, encoding and decoding a shell-constrained trellis code can be difficult. On the other hand, for independent-letters ensembles, such difficulties are minimal. So, in deciding whether it is worth using a shell-constrained code, a table listing the cutoff rates for various candidate ensembles proves helpful. This is the point of view originally put forward in [4].

Table 1 shows that the performance advantage of the shell-constrained ensemble over the gaussian ensemble increases with the SNR, though it saturates beyond some point. The table also shows that at very low SNR's the superiority of the shell-constrained ensemble becomes insignificant.

In a paper [3], which motivated the present work, Gallager gave the following asymptotic results, which clarify these observations. At high SNR's ($A > 20$)

$$R_0^* - R_0' \approx \frac{1}{2}(\log e - 1) = 0.22 \text{ bits} \tag{5}$$

and

$$C - R_0 \approx 1 - (\log e)/2 = 0.28 \text{ bits} \tag{6}$$

At low SNR's ($A < 0.1$),

$$R_0 \approx R_0' \approx C/2 \approx A/4 \tag{7}$$

So, the payoff for the extra complications of shell-constrained trellis coding may be significant only at moderate to high SNR's.

3. Fixed-Composition Trellis Codes

In this section, we shall consider *fixed-composition* trellis codes, which is a class of shell-constrained codes over a finite channel input alphabet. There will be a degradation in the cutoff rate due to quantization of the input alphabet, but this will be compensated by a reduction in complexity.

n	10	20	30	40
$R_{0,n}(Q)$	0.746	0.768	0.774	0.777

Table 2: Cutoff rate improvement.

A trellis code is said to be of fixed-composition with parameter (n, Q), for an integer $n \geq 1$ and a probability distribution Q on a finite set of real numbers $\mathcal{A} = \{a_1, \ldots, a_I\}$, if each trellis path $x_1, x_2, \ldots$ satisfies

$$Q(a) = (1/n) \sum_{j=kn+1}^{(k+1)n} \mathbf{1}\{x_j = a\} \tag{8}$$

for all $a \in \mathcal{A}$ and all $k \geq 0$, where the function $\mathbf{1}$ takes the value 1 or 0 according as the indicated event is true or false. Thus, $nQ(a)$ is the number of times the letter $a \in \mathcal{A}$ occurs in successive blocks of n symbols along each path in the trellis.

A fixed-composition code satisfies the constraint (3) with $P = \sum_{a \in \mathcal{A}} Q(a)a^2$. The cutoff rate for the ensemble (n, Q) can be shown to be

$$R_{0,n}(Q) = -\frac{1}{n} \log \sum_{\mathbf{x} \in T_Q} \sum_{\mathbf{x}' \in T_Q} \frac{1}{|T_Q|^2} \exp -\frac{d(\mathbf{x}, \mathbf{x}')^2}{8\sigma^2} \tag{9}$$

where T_Q is the subset of elements in $\mathcal{A}^n$ with composition Q, $|T_Q|$ denotes the cardinality of T_Q, and $d(\mathbf{x}, \mathbf{x}')$ is the euclidean distance between $\mathbf{x}$ and $\mathbf{x}'$.

It should be clear that $R_{0,n}(Q)$ approaches R_0^* as the parameter (n, Q) represents a finer quantization (subject to the power constraint) of the real line. The number of quantization levels (hence the constraint length n) necessary to achieve a given fraction of R_0^* increases with the SNR.

To obtain the largest cutoff rate for a given composition Q, we should consider the sequence of ensembles (nm, Q) for $m = 1, 2, \ldots$ It can be shown that $R_{0,nm}(Q)$ increases monotonically with m. To illustrate these points, consider the composition $Q(1/2) = Q(-1/2) = 4/10$, $Q(3/2) = Q(-3/2) = 1/10$, and suppose that $\sigma^2 = 0.2$. Then, $P = 0.65$, $A = 3.25$, and we have the cutoff rates listed in Table 2 for various values of n.

For comparison, we have $R_0^* = 0.787$ for $A = 3.25$. Thus, even with a four-letter alphabet, it is possible to operate within 1.3% of R_0^* by using fixed-composition codes.

We also note that, for the independent-letters ensemble with distribution Q, $R_0 = 0.713$. (This is the optimum R_0 over all independent-letters ensembles for the four-letter input alphabet here. For the gaussian ensemble, $R_0' = 0.696$.) We thus observe a 9% improvement in the cutoff rate in going from an independent-letters ensemble to a fixed-composition ensemble.

4. Code Construction

A straightforward approach to constructing a trellis code with a fixed composition (n, Q) is to consider trellises for which the number of symbols per branch equals n, and to choose each branch independently from composition Q. Unfortunately, the degree (number of branches emerging from a trellis state) for a trellis code constructed by this method equals 2^{nR}, which may be too large for decoding purposes.

For example, with $n = 10$ and $R = 0.7$ bits (numbers suggested by the example in the preceding section), we get a trellis with degree 128, which is impractical to decode using either a Viterbi or a sequential decoder. (In sequential decoding, the average number of metric computations per elementary step is lower-bounded by one half the degree of the code.)

In the remainder of this section, we describe a method for constructing fixed-composition trellis codes with smallest possible degrees.

Let the input alphabet $\mathcal{A} = \{a_1, \ldots, a_I\}$ and the parameter (n, Q) be specified. The method is based on a state diagram in which there is one state for every n-tuple $\mathbf{n} = (n_1, \ldots, n_I)$ where n_i is an integer between 0 and $nQ(a_i)$ (inclusive). The state $(nQ(a_1), \ldots, nQ(a_I))$ is called the initial state, and the state $(0, \ldots, 0)$ the final state. There is a transition from state $\mathbf{n}$ to state $\mathbf{n}'$ if and only if $\mathbf{n}'$ is less than $\mathbf{n}$ in one coordinate, but equal in all others. Fig. 1 illustrates the state diagram for a two-letter alphabet with $n = 7$ and $Q = (3/7, 4/7)$.

It will be noted that there is a natural one-to-one correspondence between blocks of n channel inputs with composition Q and paths from the initial to the final state of the state diagram.

What is needed next is a *path selector*, whose function will be to map source sequences into paths through the state diagram (and thereby to channel input sequences satisfying the fixed-composition constraint). We now describe a possible implementation for such a path selector.

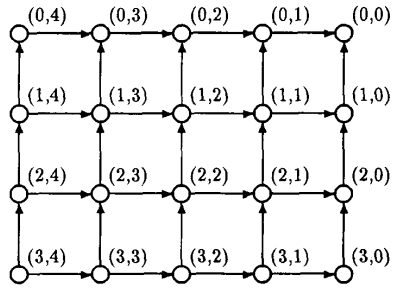

Figure 1: Example of a state diagram.

First, we assign certain probabilities to the transitions in the state diagram. We assign the probability $n_i/(n_1 + \cdots + n_I)$ to the transition from state $(n_1, \ldots, n_i, \ldots, n_I)$ to state $(n_1, \ldots, n_i - 1, \ldots, n_I)$. The probability assigned to the transition from state $\mathbf{n}$ to state $\mathbf{n}'$ has the significance of being equal to the fraction of paths, among all paths from state $\mathbf{n}$ to the final state, that go through state $\mathbf{n}'$. The product of transition probabilities along a path define a probability for that path. It can be verified easily that the path probabilities are equal for all paths from the initial to the final state.

The mapping from source sequences to paths is implemented by using the circuitry in Fig. 2 in conjunction with the transition probabilities just defined. The idea here is to use the source sequence as a random number generator and select the path step-by-step in accordance with the transition probabilities. The details are as follows.

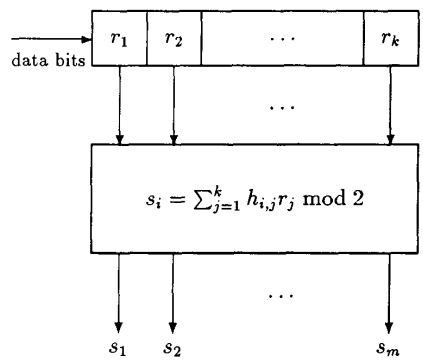

Figure 2: Path selector.

Initially, the path selector is positioned at the initial state, and the shift-register in Fig. 2 contains only 0's. Each encoding step begins with a shift of R data bits into the shift-register. (Nonintegral values of R can be handled by an extension of the method.) Then, a set of numbers $s_1, \ldots, s_m$ are computed:

$$s_i = \sum_{j=1}^{k} h_{i,j} r_j \bmod 2 \tag{10}$$

where $h_{i,j}$ are 0-1 valued connection coefficients, k denotes the number of stages of the shift register, and r_j denotes the content of the j'th stage. We regard these m bits as forming the binary expansion of the number

$$s = \sum_{i=1}^{m} s_i 2^{-i} \tag{11}$$

The value of s together with the transition probabilities determine the next state in the following manner. The interval [0,1] is thought of as partitioned into as many subintervals as there are transitions out of the present state, where the length of each subinterval equals the corresponding transition probability. The path selector moves to the state whose subinterval contains the number s. The process continues until the path selector reaches the final state, whereupon it returns to the initial state, initiating a new round of path selection. The contents of the shift register are not altered during this transition from the final to the initial state. Retaining the contents of the shift register from one round to next introduces memory into the encoding process, and is beneficial for error correction purposes.

The encoder mapping that results can be represented (for conceptual purposes) by a trellis by augmenting the states of the shift register with those of the state diagram. In this trellis, there are

$$2^{k-R} \prod_{a \in \mathcal{A}} (nQ(a) + 1)$$

states altogether, where the first factor is the number of states for the shift register and the second is that in the state diagram. The degree of the trellis code obtained by this method equals 2^R, independently of (n, Q).

We conclude with a few remarks on some computational difficulties relating to the metric to be used in sequential decoding of fixed-composition codes. As with ordinary codes, there is a Fano-type metric that achieves the cutoff rate for fixed-composition ensembles. Unfortunately, the fixed-composition property introduces memory into the input ensemble and the metric just mentioned is no longer letterwise-additive, making it hard to compute. More work is needed to understand how much degradation results from using suboptimal but more easily computable metrics.

Acknowledgement. I am grateful to N. C. Oğuz for providing the entries in Table 2.

References

[1] Shannon, C. E., 'Probability of error for optimal codes in a gaussian channel,' *Bell Syst. Tech. J.*, vol. 38, no. 3, pp. 611-656, May 1959.

[2] Gallager, R. G., 'A simple derivation of the coding theorem and some applications,' *IEEE Trans. Inform. Theory*, vol. IT-11, pp. 3-18, Jan. 1965.

[3] Gallager, R. G., 'Coding bounds for high signal-to-noise ratio gaussian noise channels,' *Abstracts of Papers, IEEE Int. Symp. Inform. Theory,* pp. 66-67, Ann Arbor, Michigan, Oct. 1986.

[4] J. M. Wozencraft and R. S. Kennedy, 'Modulation and demodulation for probabilistic coding,' *IEEE Trans. Inform. Theory,* vol. IT-12, pp. 291-297, July 1966.