University of Waterloo
Coding & Signal Transmission Laboratory
Department of Electrical & Computer Engineering
Waterloo, Ontario, Canada, N2L 3G1
Technical Report UW-E&CE#2002-19

Soft Decision Decoding of Fixed-rate Entropy-coded Trellis Coded Quantizer over a Noisy Channel

S. Nikneshan and A. K. Khandani

Soft Decision Decoding of Fixed-rate Entropy-coded Trellis Coded Quantizer over a Noisy Channel *

S. Nikneshan and A. K. Khandani

Dept. of Elec. and Comp. Eng., University of Waterloo, Waterloo, Ont., N2L 3G1

Email: sasan@shannon.uwaterloo.ca, khandani@shannon.uwaterloo.ca

Tel. (519)-885-1211-x5324, Fax. (519)-746-3077

Abstract: This paper presents new techniques to improve the performance of a Fixed-rate Entropy-coded Trellis Coded Quantizer (FE-TCQ) in transmission over a noisy channel. In this respect, we first present the optimal decoder for a Fixed-rate Entropy-coded Vector Quantizer (FEVQ). We show that the optimal decoder of the FEVQ can be a maximum likelihood decoder while a trellis structure is used to model the set of possible codewords in the FEVQ, and the Viterbi algorithm is subsequently applied to select the most likely path through this trellis. In order to add quantization packing gain to the FEVQ, we take advantage of a Trellis Coded Quantization (TCQ) scheme. To prevent the error propagation, it is necessary to use a block structure obtained through a truncation of the corresponding trellis. To perform this task in an efficient manner, we apply the idea of tail-biting to the trellis structure of the underlying TCQ. It is shown that the use of a tail-biting trellis significantly reduces the required block length with respect to some other possible alternatives known for trellis truncation. This results in a smaller delay and also mitigates the effect of the error propagation in signaling over a noisy channel. Finally, we present methods and numerical results for the combination of the proposed FEVQ soft decoder and a tail-biting TCQ. These results show that by an appropriate design of the underlying components, one can obtain a substantial improvement in the overall performance of such a fixed-rate entropy-coded scheme.

keywords Fixed-rate Entropy-coded Vector Quantization, Combined Source and Channel Coding, Viterbi algorithm, Trellis Coded Quantization, Error Propagation

*This work is supported by Communications and Information Technology Ontario (CITO). This work is presented in part in [1].

1 Introduction

Wireless communication, often characterized by narrow-band and noisy channels, is getting a lot of attention in multimedia application. Design principles stemming from Shannon's source and channel separation theorem are being reconsidered, and the attention is moving toward the joint source-channel coding and decoding as viable alternatives for reliable communications across noisy channel. First attempts at joint source-channel decoding considered the fixed-rate encoders. However, the wide use of variable length codes in data compression has motivated recent consideration of their joint source-channel coding. Variable length codes have an inherent problem in the presence of channel error. Due to the sequential decoding and variable length nature of such codes, channel errors can lead to a loss of synchronization resulting in error propagation. In practice, errors may propagate for a considerable period of time before synchronization is re-established. Variable length codes (e.g., Huffman codes) might be improved in respect of synchronization as explored in [2, 3]. The mechanism relies on the existence of universal synchronizing codewords which will obviously result in some overhead in terms of the required bit rate. Conventionally, variable length bit streams are made channel robust through packetization and Forward Error Correction (FEC) which may result in an undesirable overhead in terms of the bandwidth efficiency. Another problem with variable length codes is that they require buffering for transmission over a channel. In practice, such buffer has a finite size, hence undesirable buffer overflow/underflows may occur. Recently, several joint source/channel coding approaches have been introduced to improve the decoding of variable length codes.

One class of source-channel coders use some knowledge of the source or source coder properties to detect channel errors and compensates their effects. In this case, no redundancy is added to the output of the source encoder. Instead, the characteristic of the source or the source code are used to provide protection against possible channel errors. For example, if there is some redundancy remaining at the output of the source encoder, then this residual redundancy can be used to combat the channel noise [10]. The application of such techniques for the case of variable-length encoded data are proposed by [4, 6, 7, 8]. The combination of this technique combined with other error correcting scheme like turbo code are also presented in [11, 12, 13].

In general, the nature of variable-rate systems greatly complicates the estimation problem at the decoder side, and there have been different attempts to reduce the receiver complexity at the cost of being sub-optimum. In [4], assuming the source to be the first order Markovian, a Maximum A Posteriori (MAP) decoding method is presented using an approximate method in

which the receiver operation is independent of the probability of the received codeword. In [5, 6], a computationally complex exact MAP decoding method and an efficient approximation for it are studied. In addition to sequence based MAP decoding, the symbol-by-symbol MAP decoding according to the BCJR-algorithm [9] was also described. In [7], another method is proposed, specifically for memoryless sources. Reference [8] proposes a MAP decoding method which does not include a constraint on the length of the decoded symbol sequences.

To take advantage of the potential gain due to entropy coding, while avoiding the disadvantages associated with conventional methods based on using variable rate codes (including error propagation and buffering problems), one can use a Fixed-rate Entropy-coded Vector Quantizer (FEVQ) which is derived from a set of variable length scalar quantizers. The use of FEVQ confines the error propagation within a block, resulting in a better performance over noisy channels. As we will show in this article, by using residual redundancy in FEVQ, one can further improve the overall performance over a noisy channel. The methods presented in this paper exploit the known characteristics of the output symbols of an FEVQ to provide higher protection against the channel noise resulting in reduction of the error propagation between the symbols within an FEVQ code-vector.

In an FEVQ, every code-vector consists of a fixed number of bits representing N source symbols. Although an FEVQ attempts to remove the redundancy of the source in the first place, due to the imposed constraint on having a fixed number of bits per block of N source symbols, this objective of redundancy removal can not be completely achieved. In this article, the receiver is designed to take advantage of such a leftover redundancy in the encoded source output to improve the overall performance of the decoder in handling possible channel errors. We show that the optimum receiver can be a maximum a posteriori (MAP) or maximum likelihood (ML) receiver. In this case, the decoder estimates the transmitted sequence by applying the Viterbi algorithm through a trellis structure (finding the most probable sequence) where the trellis structure is a model for representing the entire encoder codebook.

It is also well known that the choice of the mapping between the quantizer codewords and channel input symbols may lead to a reduction in the distortion due to channel noise [14, 15]. This is known as the index assignment problem. The idea here is to come up with a distance preserving mapping from the source space to the channel space such that a "small" channel noise results in a "small" source distortion. In this regard, we have proposed a procedure to label the prefix code tree to improve the overall performance.

In order to achieve some quantization packing gain, we make use of a Trellis Coded Quantizer

(TCQ) in conjunction with FEVQ. To prevent the effect of error propagation, it is necessary to use a block structure obtained by a truncation of the trellis. In TCQ, there are two common methods to implement such a block structure; either by transmitting for each block an additional number of bits to specify the starting state, or by starting the TCQ operation in a fixed starting state for each block. Both these methods suffer from an overhead in terms of either the required bit rate or the achievable quantization gain. The disappointing fact is that as the number of states increases, the corresponding loss substantially increases in both methods. To reduce the effect of such a loss, we propose the use of a new form of block structure TCQ by the application of a tail-biting trellis. The use of tail-biting trellis significantly reduces the required block length with respect to conventional methods. This results in a smaller delay and also mitigates the effect of error propagation in signaling over a noisy channel. We will also study the combination of the tail biting TCQ with maximum likelihood (ML) decoder of the FEVQ.

The rest of the paper is organized as follow; Section 2 talks about the soft decoding of the FEVQ over a noisy channel. In this regard, we have a brief review of the FEVQ formulation where the FEVQ is based on a variable-length scalar quantizer with binary, prefix code associated with that. We show the optimum receiver of the FEVQ can be an MAP or an ML decoder, and then a trellis representation of the FEVQ decoder is described. We further show how the idea could be generalized for the case that FEVQ is combined with TCQ. We conclude section 2 by presenting a procedure to label a prefix code tree. Section 3 introduces the tail biting trellis structure for quantization. A sub-optimum algorithm is also presented for the decoding of the proposed tail-biting TCQ. Finally in section 4, we conclude the paper with some numerical results regarding soft decoding of FEVQ, tail biting TCQ, and the combination of the two.

2 Soft Decoding of FEVQ over a Noisy Channel

2.1 FEVQ Structure

Consider an N dimensional vector quantizer derived from N variable length scalar quantizers. The ith quantizer consists of M_i partitions with reconstruction levels $\{r_i(1), r_i(2), ..., r_i(M_i)\}$, where $r_i(1) < r_i(2) < \cdots < r_i(M_i)$. There is a variable-length, binary, prefix code $C_i = \{c_i(1), c_i(2), ..., c_i(M_i)\}$ associated with each quantizer, where the codeword corresponding to $c_i(j)$ has a length of $\ell_i(j)$. Using the above notations, the FEVQ operation can be formulated

as,

$$\begin{aligned} &\text{Minimize} \quad \sum_{i=1}^{N}(x_i - r_i(j))^2 \\ &\text{Subject to:} \quad \left\{ \sum_{i=1}^{N} \ell_i(j) \leq L_{Max} \right. \end{aligned} \tag{1}$$

where L_{Max} is the maximum binary length to represent an N-D code-word. The output of FEVQ will be a bit sequence with binary length of L_{Max}. The formulation differs from what was presented in [17] where the self-information of reconstruction levels has been substituted by their lengths (since the N-D codewords is derived by concatenation of its constituent along each dimension). In case the total binary length to represent the N quantized symbols is less than L_{Max}, the rest of the block is filled by zeros. There are a few methods known to solve Equation 1, providing a range of trade off between performance and complexity for different class of sources [16, 17, 18, 19].

2.2 Maximum likelihood decoding of FEVQ

Assume that the FEVQ works at rate of R bits per source dimension, i.e., $L_{Max} = RN$. For each input, the encoder produces a binary vector $\mathbf{X} \in \{0,1\}^{RN}$ for transmission. Each of RN bits of $\mathbf{X}$ is BPSK modulated, and the output $\mathbf{Y} \in \{-1,1\}^{RN}$ is transmitted over an additive white Gaussian noise channel receiving an N-dimensional real vector Z.

To recover the transmitted codeword sequences from the received data Z, the straight forward approach is to decode in a sequential manner (symbol by symbol). This is achieved by hard decision decoding of bits and then reverse substitution after parsing the decoded string into codewords. In the case that the received bits are not enough to specify all the N symbols (error has occurred), the decoder maps the rest of the symbols to a fixed value equal to the statistical average of the reconstruction levels. In this method, the receiver uses neither the fact that each block contains N source symbols nor the information which is embedded in the soft data.

Using a received vector Z, the optimum receiver [20] chooses the most probable transmitted sequences;

$$P(X|Z) = \frac{P(Z|X)P(X)}{P(Z)}$$

For fixed length codes, $P(Z)$ is irrelevant to the receiver operation and the optimal receiver maximizes $P(Z/X)P(X)$. This assumption is not valid for the variable length codes, and

in related research works, in order to simplify the decoder operation the effect of $P(Z)$ is ignored [4, 6, 8]. For an FEVQ encoder, all the transmitted sequences (N-tuples) have the same binary length and almost the same probability (based on the asymptotic equipartition property (AEP) [21]). Therefore, the assumption that $P(Z)$ is approximately the same for all the transmitted code-vectors is valid and the receiver can be simplified to a maximum a posteriori probability (MAP) decoder which maximizes:

$$P(Z/X)P(X).$$

Once again considering the fact that all encoder code-vectors are approximately equiprobable (based on AEP), the receiver can be further simplified to a *maximum likelihood decoder* which maximizes $P(Z|X)$. Thus, the decoder can be a well known Viterbi decoder which uses $P(Z|X)$ as the path metric in a trellis representation of the FEVQ where the corresponding branch metrics depend strictly on the channel statistics. In the following, we further discuss the trellis structure used in the soft decision decoding of the FEVQ.

2.3 Trellis Representation

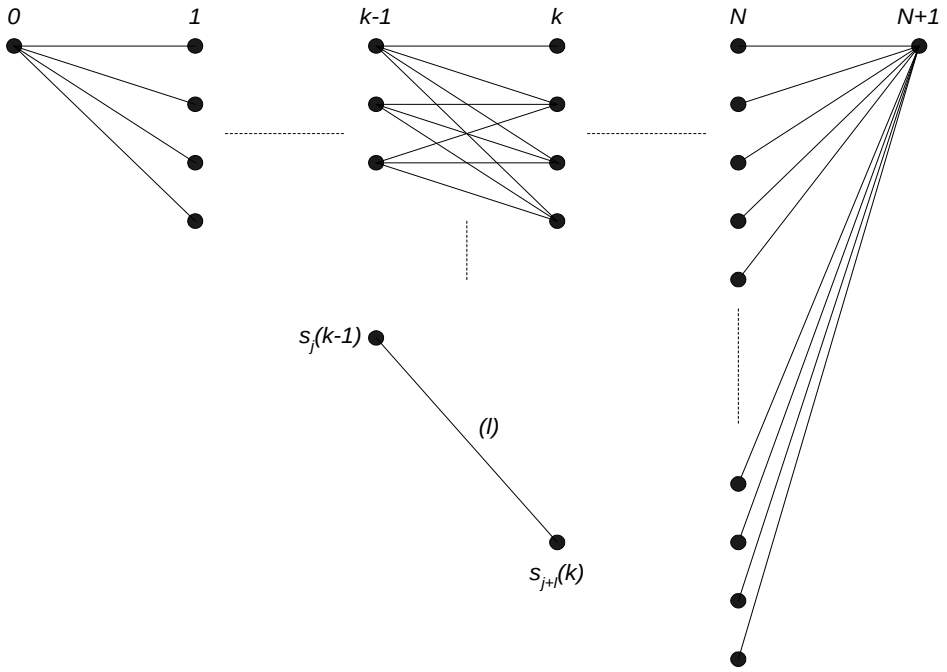

Figure 1: Trellis structure for decoding of prefix codes.

Using the fact that each block of RN bits represents N source symbols, the decoder compares

the received block with all the possible choices of N symbols which has a total binary length of L_{Max} or less. One structured approach is to represent all of the combinations of N symbols by a trellis diagram, with the states corresponding to the accumulative length of the codewords. In this trellis diagram, each transition corresponds to a set of one-D symbols. This results in a trellis composed of $N+1$ stages where the transition(s) from state s_j of stage $k-1$ to state $s_{j+\ell}$ of stage k correspond to the kth symbol(s) of length ℓ (refer to Fig. 1). The states in the kth stage, $k=0,\ldots,N+1$, represent the accumulative codeword length over the set of the first k dimensions. In this case, since the prefix code may consist of codewords of equal length, there may exist parallel branches between some of the states. As we will see later, this is an important point to be noted in selecting the index assignment for increasing robustness in signaling over a noisy channel.

In the final portion of the trellis (from stage N to $N+1$), there is a transition corresponding to the sequence of terminating zero bits, ending to a common final node for all the trellis paths. This ensures that the total length of all the paths is equal to L_{Max}. The final step is to use the Viterbi algorithm to find the most likely path through this trellis.

2.4 Soft Decoding of Fixed-rate Entropy-coded Trellis Coded Quantizer (FE-TCQ)

Consider a Fixed-rate Entropy-coded Trellis Coded Quantizer (FE-TCQ) specified by $\nu = 2^{\mu}$ states, an alphabet (set of quantization levels) $Q = \{q_1, q_2, \cdots, q_{2M}\}$ which is partitioned into four subsets, S_0, S_1, S_2, and S_3 using an Ungerboeck partition chain, a set of binary codeword lengths $L = \{\ell_1, \ell_2, \cdots, \ell_{2M}\}$ where ℓ_i is the binary length corresponding to q_i, a threshold for total binary length of L_{Max}, and an Ungerboeck type trellis structure $T(Q)$ whose branches are labeled with the subsets S_i, $i \in \{0,1,2,3\}$. The codebook set is a collection of all the possible sequences from the trellis $T(Q)$, with an additional constraint that the total binary length is no greater than the threshold L_{Max}.

The maximum likelihood decoder explained in the previous section can be applied for soft decision decoding of the suggested FE-TCQ with some differences caused by the method used to design the corresponding prefix code. We consider two possible cases to design such a prefix code:

(I) A variable-length, binary, prefix code is assigned to the points of each of the Ungerboeck type subsets S_0, S_1, S_2, S_3 [22]. In this case, every bit sequence corresponding to a thresh-

old point consists of two parts: one bit which represents the subset[1] (TCQ path indicator) and the rest of the bits which represent the prefix codeword. Since in this method of labeling these two parts are separable, the decoder could consist of a maximum likelihood decoder (with the same trellis structure explained in previous section) for the fixed-rate entropy-coded part plus a standard Viterbi decoder for the TCQ part (the issue of TCQ decoding will be explained later).

(II) A variable-length, binary, prefix code is assigned to $S_0 \cup S_2$ and $S_1 \cup S_3$ [23]. In this case, the path indicator bits of TCQ and the prefix codewords bits are not separable. Therefore, the maximum likelihood decoder for the fixed-rate entropy-coded part and the TCQ decoder are combined. In other words, the two trellis structures are merged, as a result the decoder complexity increases substantially.

2.5 Index Assignment

As discussed earlier, the FEVQ under consideration relies on a variable-length scalar quantizer with prefix codes. As prefix codes can be represented by a tree structure, the number of possible choices for different index assignments are limited. This makes the task of optimizing the index assignment much simpler in comparison with that of other vector quantizers. To obtain the best performance in assigning the binary codewords to the quantization levels, we follow a rule that the quantizer points which have the "*same binary length*" and are far from each other should differ in as many bit positions as possible, and the ones which are close together should differ in as less bit positions as possible. In other words, among the codewords with equal binary lengths, the largest hamming distance between the codewords (indexes) should correspond to the largest Euclidean distance between the corresponding threshold levels, and vice versa. This will reduce the chance of error between parallel branches in the trellis.

In a given prefix code (Huffman code), the tree can be labeled in different ways resulting in different binary prefix codes, all with the same set of code-word lengths. Satisfying the abovementioned rule is not possible for some choices of labeling. Our goal is to devise a method to label a binary tree such that it allows us to map indexes to reconstruction levels in accordance with their Euclidean distances from each other. The labeling of the tree is done such that the leaves of the tree from the left to the right represents the reconstruction levels of

[1]Note that following an Ungerboeck type trellis structure for TCQ, at each node of the trellis two of the possible four subsets are allowed.

the scalar quantizer from the left to the right respectively.

Assume there is a prefix code assigned to a set of reconstruction levels, and the lengths of the codewords and the binary tree representing the codewords are known. For most class of sources e.g., Gaussian, Laplacian ..., the probability density function is a symmetric function respect to the y axis ($x = 0$). These types of sources are also either monotonically decreasing or increasing in each side of the y axis. To reflect the source *pdf* symmetry in prefix code tree and its label, we choose the labels of one side of the tree complement of the other side. Obviously, the tree has a symmetric structure. This guarantees that codewords of equal length get the maximum hamming distance when representing two points at both sides of the y axis. Codewords of equal lengths may happen at one side of the y axis. To satisfy being a distance preserving mapping, we choose the labels of one side of the tree to form a gray code at any depth.

For example, Figure 2 shows a symmetric binary tree representing Code(I) from Table 1. The labeling in half of the tree (the right side) is done such that the codewords form a gray code at any depth and the labels of one half of the tree are complement of that of the other half. In Figure 3, at depth 3 codewords 000, 001, 011 and 010 form a gray code. In order to have a gray code at any depth, we alternately map the $\{0, 1\}$ and $\{1, 0\}$ to pair of leaves. In other words, if we have mapped the $\{0, 1\}$ to the first pair of leaves the next pair will have the label of $\{1, 0\}$ and vice versa (obviously, this method does not result in a unique way of labeling). In Figure 3, in right side of the tree at the depth of 3, there are two pairs of leaves which have the label of $0, 1, 1, 0$ from the left to the right.

The following procedure describes how to label a given tree to satisfy the index assignment conditions.

1. Index the right (left) side of symmetric tree. The indexing is such that at any depth, the codewords form a gray code.
2. Index of each branch from the left (right) side of symmetric tree is obtained by flipping the index of its image from the right (left) side.

Table 1 shows three different possible index assignments for an 8-point scalar quantizer with a specific set of codeword lengths. Code (I) and (II) are the two choices obtained by above mentioned procedure, and Code (III) is not satisfying the rules.

Table 2 shows different index assignments for a 16-points scalar quantizer to be used in our FE-TCQ with two different codeword length sets [23]. Note that each index is repeated two

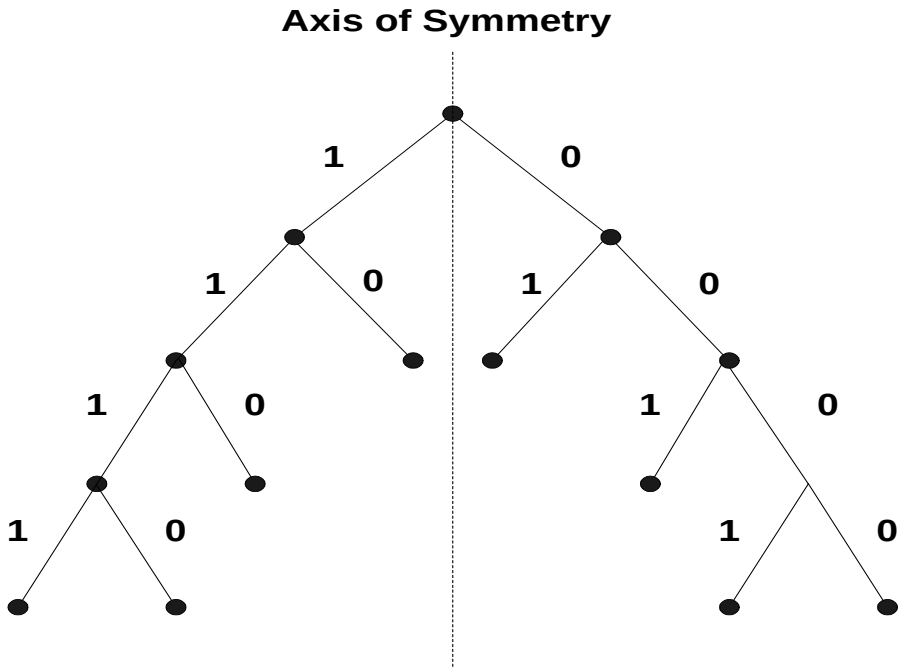

Figure 2: An example of index assignment for a symmetric tree (Code(I) from Table 1).

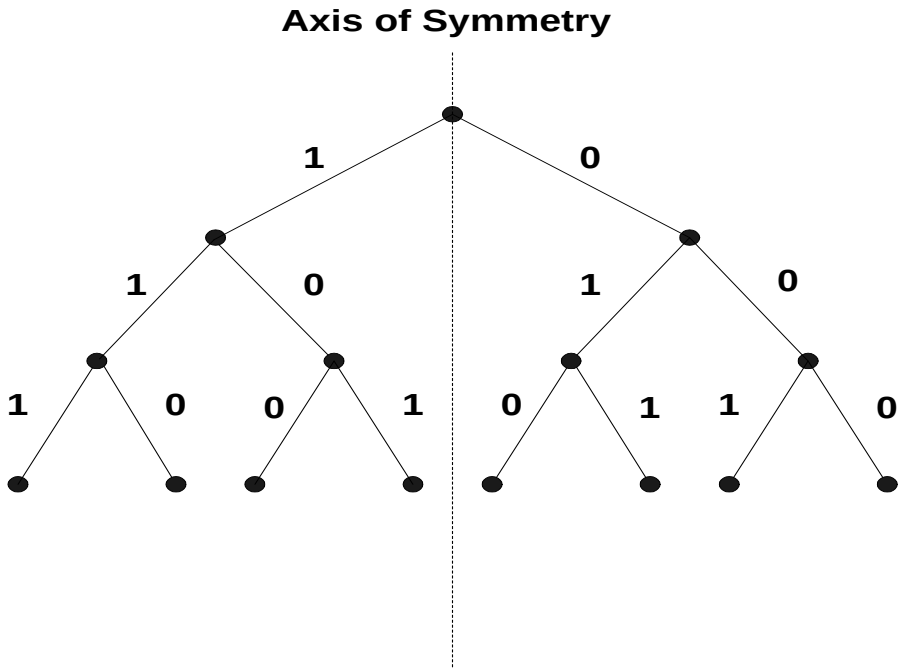

Figure 3: An example of index assignment for a symmetric tree.

Quantizer level	Code(I)	Code(II)	Code(III)
-2.17	0000	0100	0111
-1.36	0001	0101	0110
-0.77	001	011	010
-0.25	01	00	00
$+0.25$	10	11	10
$+0.77$	110	100	110
$+1.36$	1110	1010	1110
$+2.17$	1111	1011	1111

Table 1: Different index assignment for prefix code structures.

times due to the natural redundancy in the trellis structure of the underlying TCQ. In Code (I), a prefix code with codewords of $\{111, 110, 10, 0\}$ is designed for each subset S_i, $i \in \{0, 1, 2, 3\}$. A different bit is added as an index to differentiate between S_0 and S_2 or S_1 and S_3 (there is 4 elements in each subset and the index assignment presented before does not have any impact on labeling). In all sets of Code (II), a prefix code is designed for $S_0 \cup S_2$ and $S_1 \cup S_3$. Thus, no bit is required as an indicator for each subset. Code (II,a) and (II,b) are designed for a specific set of lengths which is different from Code (II,c). Therefore, the differences between (II,a) and (II,b) with Code (II,c) is the matter of a different prefix code rather than a different index assignment. In designing Code (II,a) the index assignment rules are not satisfied while in Code (II,b) they are.

3 Tail Biting Trellis Coded Quantization (TB-TCQ)

Consider an N-dimensional TCQ (N is the block length) with $\nu = 2^\mu$ states. The bit sequence at the quantizer output specifies the choice among the possible branches at each state plus additional μ bits which specify the starting state. These additional μ bits will have a strong impact on the effective bit rate for small values of block length.

Two modifications can be made to avoid sending the extra μ bits. First, one can use a fixed starting state TCQ to encode each block of source samples. As we see later, one will loose part of the achievable granular gain by imposing this constraint.

As an alternative, we propose to use a tail biting trellis structure in which the start and the end state on the trellis paths are the same. Using this structure, one does not need any

	Code (I)	Code (II)		
Quantizer level		(a)	(b)	(c)
-5.12	0 111	0111	0000	111111
-3.09	0 111	0111	0000	111110
-2.14	1 111	1001	0001	11110
-1.51	1 111	1001	0001	1110
-1.06	0 10	010	001	110
-0.69	0 10	010	001	10
-0.39	1 0	11	01	01
-0.12	1 0	11	01	00
0.12	0 0	00	10	00
0.39	0 0	00	10	01
0.69	1 10	101	110	10
1.06	1 10	101	110	110
1.51	0 110	0110	1110	1110
2.14	0 110	0110	1110	11110
3.09	1 110	1000	1111	111110
5.12	1 110	1000	1111	111111

Table 2: Different index assignment of prefix code for FE-TCQ.

extra bits to specify the starting state. The idea is similar to tail biting trellis structures used in convolutional codes to construct a block code from a convolutional code [27, 28, 29]. To search through a tail biting trellis, one straight forward procedure is to run the Viterbi algorithm ν times for different starting states. The main disadvantage of this method is the increase in the computational complexity (since the Viterbi algorithm has to be run ν times the encoding computational complexity of tail biting TCQ is ν times of the encoding computational complexity of TCQ [24]). To reduce the complexity, several sub-optimum and optimum search algorithms have been proposed in the context of convolutional codes [27, 28].

In order to overcome the complexity issue of the tail biting trellis structure, we present a possible sub-optimum search methods and a new trellis structure. The new structure which we call "Modified Tail Biting Trellis" (MTB-TCQ), consist of all paths with a fixed starting and ending state (the starting and ending states may be different), The encoding complexity of this modified tail biting trellis is the same as a conventional trellis coded quantization. Although this structure may loose some parts of the packing gain, it can achieve more boundary gain when it is combined with an FEVQ. This gain is noticeable when the FEVQ has not achieved most of the boundary gain by itself. We explain this issue in more detail when we present our

numerical results.

The main idea for applying a sub-optimum search method is to reduce the number of independent TCQ search iterations. In the following, we present a sub-optimum search algorithm which requires just one search iteration.

Algorithm :

Step 1. Choose all states as the starting state.

Step 2. At stage $N-\mu$, check the winning path. Extract the starting state of the winning path (S_i).

Step 3. At stage $N-\mu$, set the metric of all the path which does not start from state S_i to infinity (forcing all state at stage $N-\mu$to end at state S_i at stage N).

Step 4. Continue the encoding until the last stage.

Step 5. Path starting from S_i and ending to S_i is the final answer.

4 Numerical Results

In this section, we present numerical results for the performance of the proposed methods. In all cases, a sequence of 600000 source samples is used to design the quantizer (using an LBG type iterative design procedure), and a different sequence of the same length is used to measure the corresponding performance. Input samples are from a memoryless Gaussian or Laplacian source. The underlying scalar quantizer is composed of 8 points, and for the case with TCQ structure the number is 16. Space dimension is $N = 32$, and $L_{\max} = 80$ resulting in an effective rate of 2.5 bits/sample (for FEVQ and FE-TCQ). The quantization performance is measured in terms of the mean square distance. The channel model is Additive White Gaussian Noise (AWGN) with Binary Phase Shift Keying (BPSK) modulation.

In the first set of the results we present the numerical results of proposed ML decoder for FEVQ and FE-TCQ. Figure 4 shows the received SNR vs E_b/N_0 for ML decoder of FEVQ (in this case a priori information is used) and sample by sample decoding (in this case the priori information is not used). Generally, when ML decoder of the FEVQ is used, the transmitter power can be reduced between 1 to 2 dB. For example, at SNR= 4 dB the difference in required E_b/N_0 for the two systems is about 1.4 dB which means the ML decoder reduces the required transmitted power by 1.4 dB. The proposed method can be considered as a zero redundancy source/channel coding. It means that no redundancy from channel is added to data

stream and only the residual redundancy existing in encoders output is used. Since our FEVQ is based on variable-length code (prefix code) we compare the performance of the proposed method with that of other source-channel coder which is using variable-length code. However, because of a different objective and a different amount of residual redundancy, the comparison of two different zero redundancy source-channel coding schemes may be difficult and sometimes inappropriate. One of the methods presented in reference [13] is based on a variable-length code where a packeting technique is used to reduce the BER. Their improvement in transmitter power is 0.3 dB at BER equal to 10^{-2}.

Figure 5 shows the comparison between three different decoding methods, maximum likelihood decoding approach using hard and soft decision, and sample by sample decoding. The structure of the codewords is the same as code (I) in Table 1. It is observed that the maximum likelihood decoding using soft decision results in about 1-3 dB improvement in comparison with the other two alternatives. Figure 6 shows a comparison of soft decision decoding for prefix code with three different labeling methods given in Table 1. As it is expected, codes (I) and (II) show almost the same performance while they are both superior in performance in comparison with code (III).

Assume the prefix code of FEVQ has the set of $\{\ell_1, \ell_2, \ldots, \ell_M\}$ as the codeword length, ℓ_{min} as the minimum codeword length, and M' as the distinct number of codeword lengths. The total allowed path or the total states at stage N (refer to Figure 1) will be $L_{Max} - N\ell_{min}$. To keep the information of these paths in all the stages, a total of $(L_{Max} - N\ell_{min})N$ units of memory is required. We also require $2L_{Max}$ units of memory to keep the Euclidean distance of every received packet of L_{Max} bits to BPSK level of "1" and "-1". There is no multiplication required to calculate the Euclidean distance (the two terms of $(x-1)^2 = (x^2+1) - 2x$ and $(x+1)^2 = (x^2+1) + 2x$ can be substituted by $-x$ and x respectively without affecting any decisions at any stages). Since there are M branches leaving each state of the trellis in Figure 1, there are $\sum_{i=1}^{M}(\ell_i - 1)$ additions required to calculate the metric of the M branches. There are also M' additions required to find the total metric of the M' paths ending to the next stage. Therefore, the total number of additions will be $\sum_{i=1}^{M}(\ell_i - 1) + M'$ times the total number of states in the trellis.

FE-TCQ presents a different complexity based on the way prefix coding is done for the subsets (designing a prefix code for each subset or for the union of pair of them). For type (i), the TCQ decoder and ML decoder of FEVQ can be separated into two modules that can work in parallel. Therefore, the complexity of the FE-TCQ at rate R will be the sum of:

TCQ complexity at rate 1 and ML decoder complexity at rate $R-1$. For type (ii), the TCQ decoder and ML decoder of FEVQ act as one module and they can not be separated any more. In this case, the total allowed paths or the total number of the states at stage N will be $(L_{Max} - N\ell_{min})\mu$, where μ is the number of the states in TCQ trellis. To keep the information of these paths at all stages, a total of $(L_{Max} - N\ell_{min})N\mu$ units of memory is required. The number of multiplications remains zero the same as that of the FEVQ, and the number of additions is multiplied with the number of TCQ states (μ).

Table 3 presents the computational complexity of a maximum likelihood decoder for FEVQ and FE-TCQ at the rate of 2.5 bits/dimension. As it is anticipated, the FE-TCQ using code (II) has higher complexity than code (I). It is shown that the achieved gain due to ML decoder is at the cost of a negligible complexity.

Table 4 shows the comparison of the soft decision decoding of the FE-TCQ over a noisy channel for the code (I) and (II,c) from Table 2. Although code (II,c) shows better performance in an error free channel, code (I) is more robust in the presence of channel noise. This is because the prefix code with the least expected length is more sensitive to channel noise. This means that the synchronization happens faster in the code with the larger expected length. Another way of explaining the results is; the less the redundancy the less immunity from the channel noise.

Figure 7 compares the performance of the FE-TCQ using four different index assignments given in Table 2. Because of the proper index assignment, code (II,b) shows better performance than code (I) and code (II,a). Code (II,c) has the worse performance in a noisy channel (against its best performance in noiseless channel) because of less redundancy in comparison with code (I), (II,a) and (II,b).

We have included some numerical results for the methods proposed in [25, 26]. The results correspond to the performance of modified trellis-based scalar-vector quantizer (TB-SVQ) in a noisy channel (refer to Table 5). This is a fixed-rate trellis coded quantizer which uses an enumerating algorithm to label the codeword sequences. By comparing with Table 4, TB-SVQ outperforms FE-TCQ in a noiseless channel but in a noisy channel the proposed maximum likelihood decoder substantially improves the FE-TCQ performance resulting in a better performance than TB-SVQ. more robust.

In the second set of numerical results, we compare the performance of the tail biting trellis structure with other fixed block length trellises as well as TCQ and TCQ with an early decision in decoding. Table 6 and 7 provide comparison between TB-TCQ and TCQ for both Gaussian

and Laplacian data. We present numerical results for three cases of the TCQ. In the first case, TCQ has a block length of 1000 samples[2], and in the second case, the starting state is fixed which we call it Fixed Starting state TCQ (FS-TCQ). In the third case, $N = 600000$ block length is used with an early decision after $N = 16$ stages (decoding depth of the Viterbi algorithm is 16). It is observed that TB-TCQ has a performance similar to TCQ with a much smaller delay. It is also observed that TB-TCQ offers a slight improvement over truncated TCQ. Table 8 and 9 compare the performance of the full search algorithm with that of the sub-optimum search algorithm. The results show a small gap between their performance. For the Gaussian data, this gap becomes smaller as the number of states increases, but for Laplacian data, as the number of states increases the gap becomes larger.

Figure 8 show the end-to-end performance of TB-TCQ and TCQ ($N = 1000$) as a function of the resulting bit error probability. These curves demonstrate a significant improvement for TB-TCQ in comparison with TCQ. Similar improvements are observed for trellis diagrams with a larger number of states.

Table 10 and 11 present the comparison of FE-TCQ for the different trellis structures. These comparisons clarify more the advantage of tail biting trellis structure over the other methods for trellis termination. All the results correspond to $R = 2.5$ bits/sample (the total binary length of FEVQ is $L_{Max} = 80$). In some cases, a few more bits are required to specify the starting state depending on the type of trellis structure (in which case the total binary length will be modified accordingly). For the regular trellis diagram (TCQ), the numbers of those bits are 2 or 3 (for 4 or 8 state trellis diagrams, respectively), and there are no bits required to specify the starting state for the "Fixed-Starting state" (FS-TCQ) and "Tail Biting" (TB-TCQ) trellis structures. For the "Modified Tail Biting" (MTB-TCQ) trellis structure, the encoder not only does not need any extra bits for the starting state but also saves another 2 or 3 bits. Those bits correspond to the last 2 or 3 stages of the trellis since the path must end to the same state as the starting state. trellis sections connecting to the same ending state as the starting state. In other words, since the starting and ending states for this quantizer is the same, the decoder does not require to transmit the last 2 or 3 path indicator bits. In summary, if the trellis structure has $\nu = 2^{\mu}$ states, the actual total binary length for the regular trellis is $L_{max} - \mu$, for the modified tail biting it is $L_{max} + \mu$, and for the tail biting and fixed starting state it is

[2]This is the same block length as used in [24]. It should be mentioned that the numerical results presented for $N = 1000$ are computed independent of [24] (following the same set up as used for all our numerical results) and are slightly different from the values reported in [24].

L_{max}. Therefore, if FEVQ has not already achieved most part of boundary gain the FE-TCQ using modified tail biting trellis may show better performance than when a tail biting trellis is used (as example for Gaussian data, FE-TCQ with Code(I) labeling and modified tail biting trellis performs slightly better than the case with a tail biting trellis, refer to table 10).

The numerical results of this comparison for the four states trellis structure and two different prefix code assignments indicate prefix code (II,c) produces better results for all cases. The same trend is also observed for 8-state trellis, except for the fixed-starting state trellis. The same results is observed in Table 6 and 7, the fixed-starting state trellis performs even worse when the number of states increases.

Table 12 and 13 show the comparison of soft decision decoding of the FE-TCQ using the four different trellis structures. The prefix codes (II,c) and (II,b) are chosen from Table 2 for this comparison. In summary, fixed-rate entropy-coded quantizer using a tail biting trellis structure shows a better performance in comparison with other trellis structures. Prefix code (II,c) has the best performance in error free channel where as prefix code (II,b) shows the most robust performance in a noisy channel.

References

[1] S. Nikneshan and A. K. Khandani, "Soft Decision Decoding of Fixed Rate Entropy Constrained Quantizer over a Noisy Channel," *20th Biennial Symposium on Communications,* pp.116—118, (Queen's Univ., Kingston, ON), May 28–May 31, 2000.

[2] T. J. Ferguson and J. J. Rabinowitz, "Self-synchronizing Huffman codes," *IEEE Trans. Inform. Theory,* vol. 30, pp. 687–693, July 1984.

[3] B. L. Montgomery and J. Abrahams, "Synchronization of binary source codes," *IEEE Trans. Inform. Theory,* vol. 32, pp. 849–854, Nov. 1986.

[4] N. Demir and K. Sayood, "Joint source-channel decoding for variable-length codes," in *Proc. Data Comp. Conf.,* Snowbird, UT, pp. 139-148,1998.

[5] M. Park and D. Miller, "Improved joint source-channel decoding for variable-length encoded data by using soft decisions and MMSE estimation," *IEEE Data Compression Conference,* Snowbird, Utah, March 1999.

[6] M. Park and D. Miller, "Joint source-channel decoding for variable-length encoded data by exact and approximate MAP sequence estimation," *IEEE Trans. Inform. Theory,* vol. 48, pp. 1–6, Jan. 2000.

[7] K. P. Subbalakshmi and J. Vaisey "Optimal decoding of entropy coded memoryless sources over binary symmetric channel," *Data Compression Conference,* pp. 573, March 1998.

[8] A. H. Murad and T. E. Fuja, "Joint source-channel decoding of variable-length encoded sources," in *Proc. Information Theory Workshop*, Killarney, Ireland, pp. 94–95, June 1998.

[9] 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, March 1974.

[10] K. Sayood and J. C. Borkenhagen, "Use of residual redundancy in the design of joint source/channel coders," *IEEE Trans. Commun.,* vol. 39, pp. 838–846, June 1991.

[11] R. Bauer and J. Hagenauer, "Iterative source/channel-decoding using reversible variable-length codes," in *Proc. Data Comp. Conf.,* DCC 2000, pp. 93-102, 2000.

[12] A. Guyaderi, E. Fabre, C. Guillemot, and M. Robert, "Joint source-channel turbo decoding of entropy-coded sources," *IEEE Journal of Selected Area in Communications,* vol. 19, pp. 1680–1696, Sept. 2001.

[13] M. Jeanne, J. C. Carlach, P. Sihon and L. Guivarch, "Source and joint source-channel decoding of variable length codes," *ICC2002, IEEE International Conference on Communications,* vol. 2, pp. 768–772, 2002.

[14] K. A. Zeger and A. Gersho, "Zero redundancy channel coding in vector quantization," *IEEE Trans. Inform. Theory,* vol. 33, pp. 654–655, June. 1987.

[15] N. Farvardin and V. Vaishampayan, "Optimal quantizer design for noisy channel: An approach to combined source-channel coding," *IEEE Trans. Inform. Theory,* vol. 33, pp. 827–837, Nov. 1987.

[16] R. Laroia and N. Farvardin, "A structured fixed-rate vector quantizer derived from variable-length scalar quantizer—Part I: Memoryless sources," *IEEE Trans. Inform. Theory,* vol. 39, pp. 851–867, May 1993.

[17] A. K. Khandani, "A Hierarchical Dynamic Programming Approach to Fixed-rate, Entropy-Coded Quantization," *IEEE Trans. Inform. Theory,* vol. 42, pp. 1298–1303, July 1996.

[18] A. K. Khandani, "A linear (zero-one) programming approach to fixed-rate entropy-coded vector quantization," *IEE Proceedings Communications*, volume 146, No. 5, pp.275—282, Nov. 99.

[19] S. Nikneshan, "Lattice/Trellis based Fixed-rate Entropy-coded Vector Quantization," Ph.D. Thesis, Elec. and Comp. Eng. Dept., Univ. of Waterloo, Sept. 2001.

[20] J. M. Wozencraft and I. M. Jacobs, "Principles of Communication engineering," *John Wiley and Sons,* New York, 1965.

[21] T. M. Cover and J. A. Thomas, "Elements of Information Theory," *John Wiley and Sons,* New York, 1991.

[22] T. R. Fischer and M. Wang, "Entropy-constrained trellis-coded quantization," *IEEE Trans. Inform. Theory,* vol. 38, pp. 415–426, March 1992.

[23] M. W. Marcellin, "On entropy constrained TCQ," *IEEE Trans. Commun.,* vol. 42, pp. 14–16, Jan. 1994.

[24] M. W. Marcellin and T. R. Fischer, "Trellis-coded quantization of memoryless and Gauss-Markov sources," *IEEE Trans. Commun.* vol. 38, pp. 82–93, Nov. 1990.

[25] R. Laroia and N. Farvardin, "Trellis-based scalar-vector quantizer for memoryless sources," *IEEE Trans. Inform. Theory,* vol. 40, pp. 860–870, May 1994.

[26] L. Yang and T. R. Fischer "A new trellis source code for memoryless sources," *IEEE Trans. Inform. Theory,* vol. 44, pp. 3056–3063, Nov. 1998.

[27] H. H. Ma and J. K. Wolf, "On tail biting convolutional codes," *IEEE Trans. Commun.,* vol. 34, pp. 104–111, Feb. 1986.

[28] Q. Wang and V. K. Bhargava, "An efficient Maximum Likelihood decoding algorithm for generalized tail biting convolutional codes including quasi-cyclic codes," *IEEE Trans. Commun.,* vol. 37, pp. 875–879, Aug. 1989.

[29] G. Solomon and H. C. A. van Tilborg, "A connection between block and convolutional codes," *SIAM J. Appl. Math.,* vol. 37, pp. 358–369, Aug. 1989.

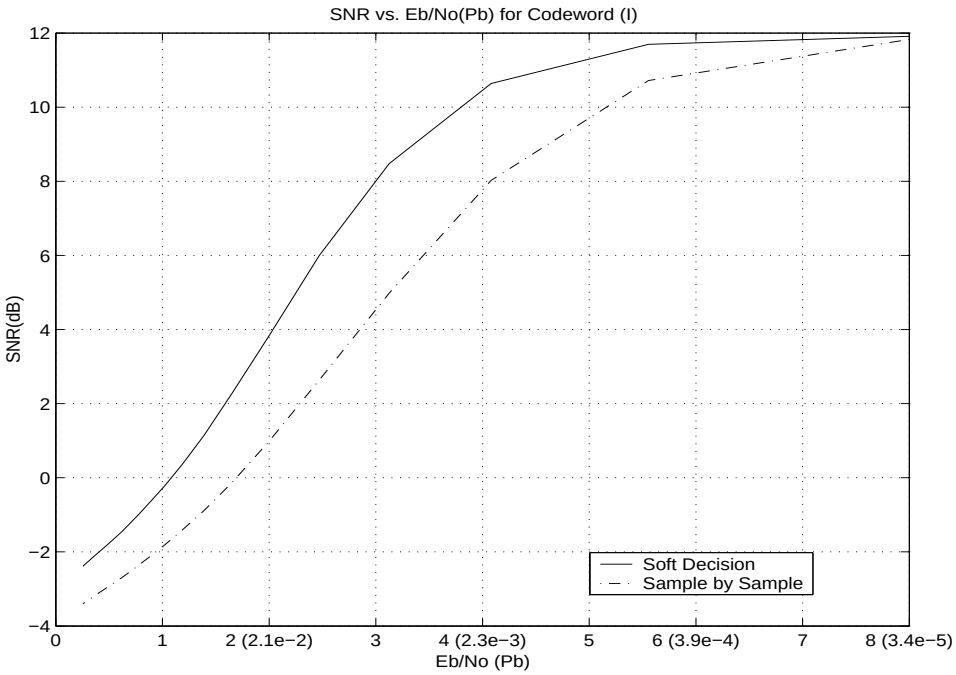

Figure 4: Comparison of received SNR for FEVQ using ML decoder (with a priori) and sample by sample decoder (without a priori) for different $E_b/N_0(P_b)$ (Code(I) of Table 1 is used).

	Addition	Multiplication	Memory
FEVQ	1632	0	0.7 k
FE-TCQ (Code (I))	180	0	0.6 k
FE-TCQ (Code (II,a))	2037	0	2.4 k

Table 3: The complexity of FEVQ at rate 2.5 bits/dimension. In all calculations, the memory size is in byte per N dimensions and the number of additions/multiplications are counted per dimension.

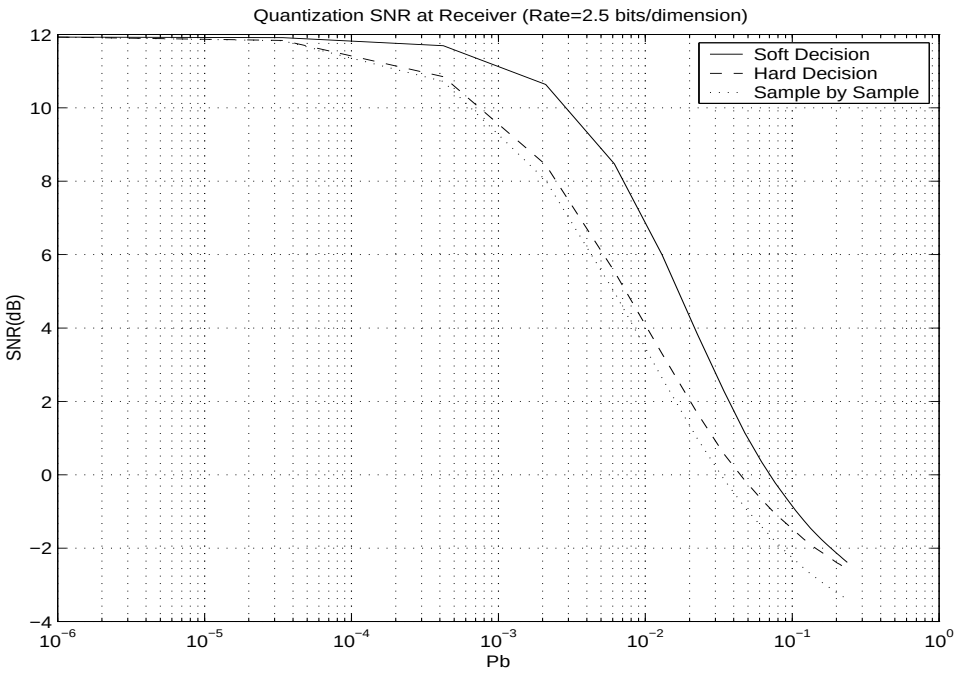

Figure 5: Comparison of received SNR for soft, hard decision decoding using trellis structure and sample by sample decoding for different bit error probabilities of BPSK channel, P_b (Code(I) of Table 1 is used).

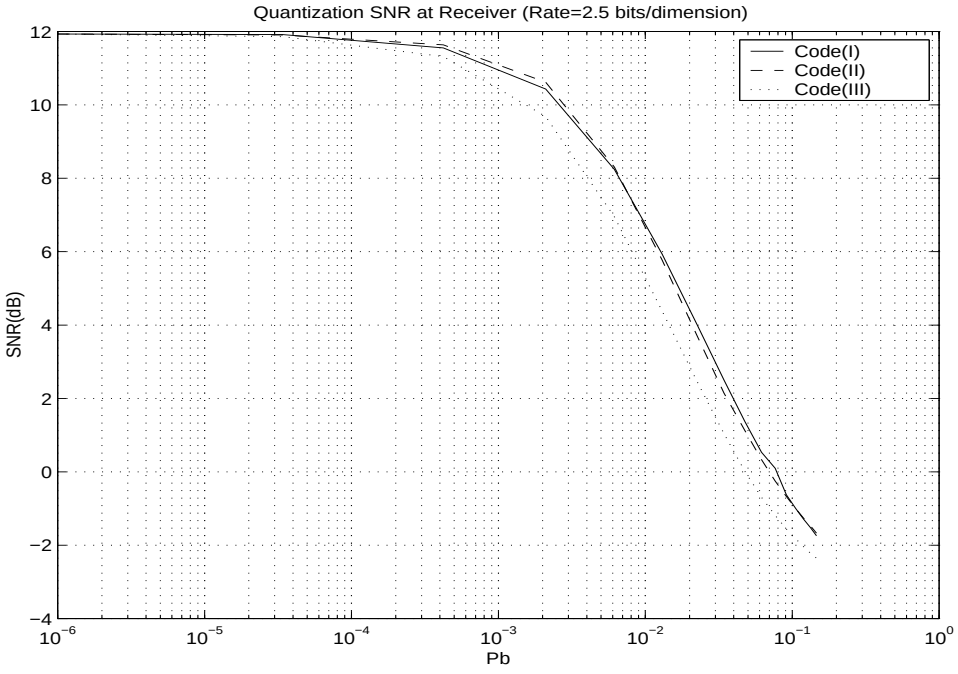

Figure 6: Comparison of soft decision decoding for different binary codes of Table 1 as a function of bit error probability of BPSK channel, P_b.

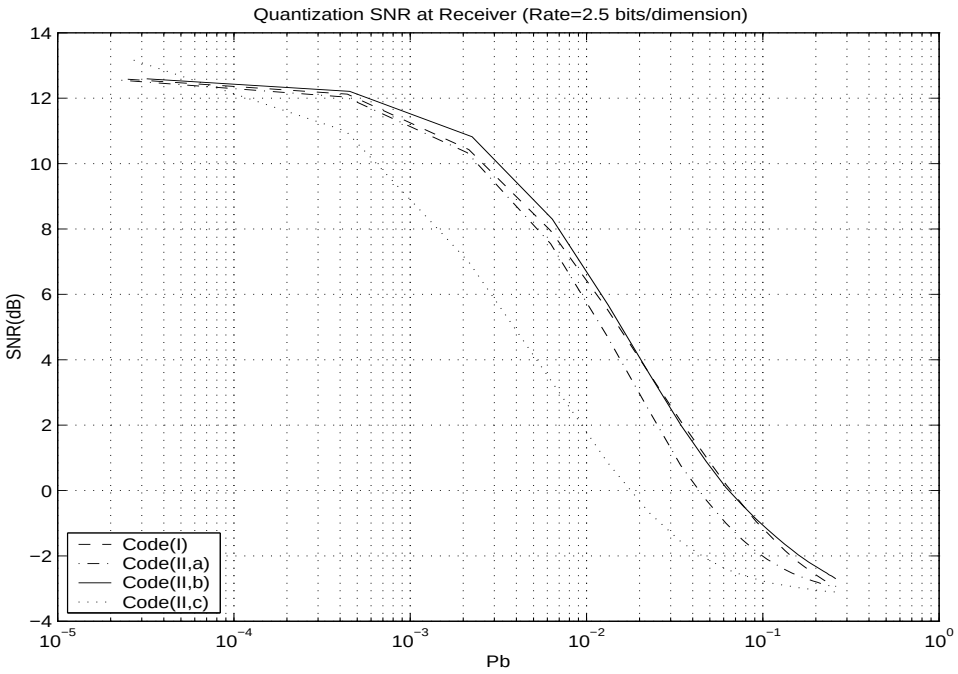

Figure 7: Comparison of soft decision decoding for different binary codes of Table 2 as a function of bit error probability of BPSK channel, P_b.

	$P_b = 0.01$	$P_b = 0.005$	$P_b = 0.001$	$P_b = 0$
SNR, Code I	6.51	8.54	11.24	12.61
SNR Code II,c	1.65	3.96	8.91	13.35

Table 4: Soft decoding of FE-TCQ using the code (I) and (II,c) of table 2 for different bit error probability for a four state trellis and $2M = 16$ points scalar quantizer, dimension is $N = 32$, rate is 2.5 bits/sample.

Bit rate	$P_b = 0.01$	$P_b = 0.005$	$P_b = 0.001$	$P_b = 0$
2	1.53	3.69	8.19	11.15
3	0.5	2.74	8.61	16.71

Table 5: Performance of the fixed-rate TCQ scheme proposed in [26] using a four state trellis.

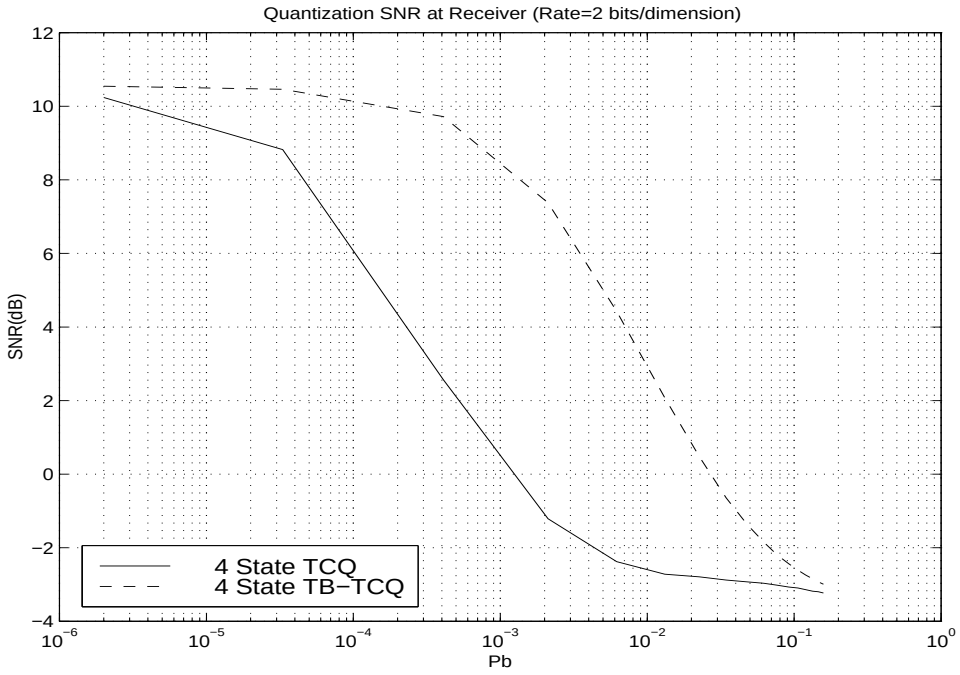

Figure 8: Comparison of the end-to-end quantization SNR (dB) for 4 state TCQ ($N = 1000$) and TB-TCQ($N = 32$) for different probability of bit error (P_b).

	4 State		8 State		16 State		32 State	
Block Length	TB-TCQ	FS-TCQ	TB-TCQ	FS-TCQ	TB-TCQ	FS-TCQ	TB-TCQ	FS-TCQ
16	10.47	10.16	10.52	10.18	10.53	10.11	10.52	9.96
20	10.50	10.23	10.58	10.27	10.60	10.21	10.61	10.08
24	10.52	10.28	10.62	10.32	10.64	10.29	10.68	10.19
28	10.52	10.32	10.63	10.38	10.67	10.34	10.71	10.26
32	10.53	10.34	10.65	10.41	10.70	10.39	10.74	10.33
64	10.53	10.43	10.66	10.51	10.74	10.56	10.80	10.56
128	10.53	10.48	10.66	10.59	10.74	10.64	10.80	10.68
256	10.53	10.51	10.66	10.62	10.74	10.69	10.80	10.74
512	10.53	10.53	10.66	10.64	10.74	10.71	10.80	10.77
1024	10.53	10.53	10.66	10.65	10.74	10.72	10.80	10.79
TCQ ($N = 1000$, [24])	10.54		10.67		10.75		10.82	
TCQ (Truncated,$N = 16$)	10.42		10.44		10.46		10.44	

Table 6: Quantization SNR (dB) for the 4, 8, 16 and 32 states trellis diagrams using a Gaussian source at rate of 2 bits/sample.

	4 State		8 State		16 State		32 State	
Block Length	TB-TCQ	FS-TCQ	TB-TCQ	FS-TCQ	TB-TCQ	FS-TCQ	TB-TCQ	FS-TCQ
16	9.38	9.02	9.48	9.06	9.49	8.98	9.50	8.72
20	9.39	9.09	9.51	9.15	9.56	9.09	9.58	8.88
24	9.39	9.15	9.53	9.22	9.60	9.17	9.62	8.99
28	9.40	9.19	9.54	9.28	9.62	9.24	9.65	9.10
32	9.41	9.21	9.54	9.31	9.63	9.29	9.67	9.14
64	9.41	9.31	9.55	9.44	9.65	9.48	9.71	9.41
128	9.41	9.37	9.56	9.51	9.65	9.56	9.71	9.54
256	9.41	9.40	9.56	9.54	9.65	9.61	9.71	9.62
512	9.41	9.41	9.56	9.56	9.65	9.63	9.71	9.65
1024	9.41	9.41	9.56	9.57	9.65	9.64	9.71	9.68
TCQ ($N = 1000$ [24])	9.41		9.56		9.67		9.73	
TCQ (Truncated, $N = 16$)	9.34		9.40		9.44		9.43	

Table 7: Quantization SNR (dB) for the 4, 8, 16 and 32 states trellis diagrams using a Laplacian source at rate of 2 bits/sample.

	4-state	8-state	16-state	32-state
Full search	10.53	10.64	10.70	10.74
Sub-optimum search	10.50	10.60	10.65	10.70

Table 8: Quantization SNR (dB) for full search TB-TCQ and the sub-optimum search TB-TCQ for different trellis structure using a Gaussian source. The block length is $N = 128$, and the rate is 2 bits/sample.

	4-state	8-state	16-state	32-state
Full search	9.40	9.56	9.65	9.71
Sub-optimum search	9.37	9.48	9.56	9.59

Table 9: Quantization SNR (dB) for full search TB-TCQ and the sub-optimum search TB-TCQ for different trellis structure using a Laplacian source. The block length is $N = 128$, and the rate is 2 bits/sample.

	4-state trellis			
	Regular	Fixed-starting state	Tail biting	Modified tail biting
Code (I)	12.61	12.78	12.94	12.97
Code (II,c)	13.35	13.31	13.45	13.27
	8-state trellis			
	Regular	Fixed-starting state	Tail biting	Modified tail biting
Code (I)	12.55	12.83	13.04	13.10
Code (II,c)	13.39	13.33	13.50	13.41

Table 10: Quantization SNR (dB) for FE-TCQ for different trellis structures. Trellises have four and eight states. There are 16 threshold points along each dimension the rate is 2.5 bits/sample and data is taken from a memoryless Gaussian source.

	4-state trellis			
	Regular	Fixed-starting state	Tail biting	Modified tail biting
Code (I)	12.8	12.89	13.07	12.90
Code (II,c)	13.23	13.13	13.33	13.13
	8-state trellis			
	Regular	Fixed-starting state	Tail biting	Modified tail biting
Code (I)	12.75	12.90	13.16	12.95
Code (II,c)	13.31	13.15	13.45	13.05

Table 11: Quantization SNR (dB) for FE-TCQ for different trellis structures. Trellises have four and eight states. There are 16 threshold points along each dimension the rate is 2.5 bits/sample and data is taken from a memoryless Laplacian source.

	$P_b = 0.01$	$P_b = 0.005$	$P_b = 0.001$	$P_b = 0$
TCQ	6.67	9.00	11.66	12.61
TB-TCQ	8.14	10.65	12.20	12.94
FS-TCQ	6.82	8.98	11.87	12.80
MTB-TCQ	7.97	10.51	12.78	12.97

Table 12: Received SNR (dB) for soft decoding of fixed-rate entropy-coded quantizer using different trellis structure for different bit probability of error (P_b) at rate of 2.5 bits/sample. The codewords (II,b) is used.

	$P_b = 0.01$	$P_b = 0.005$	$P_b = 0.001$	$P_b = 0$
TCQ	1.75	4.03	9.04	13.36
TB-TCQ	2.71	5.56	10.49	13.46
FS-TCQ	1.86	4.21	9.34	13.31
MTB-TCQ	3.08	5.78	10.57	13.27

Table 13: Received SNR (dB) for soft decoding of fixed-rate entropy-coded quantizer using different trellis structure for different bit probability of error (P_b) at rate of 2.5 bits/sample. The codewords (II,c) is used.