*Supported by the National Science Foundation Engineering Research Center Program (NSFD CD 8803012), Industry and the University*

# Trellis-Based Scalar-Vector Quantizer for Memoryless Sources

*by R. Laroia and N. Farvardin*

TR 92-82

# Trellis-Based Scalar-Vector Quantizer for Memoryless Sources [†]

by

*Rajiv Laroia* and *Nariman Farvardin*

Electrical Engineering Department

and Systems Research Center

University of Maryland

College Park, Maryland 20742

## Abstract

This paper describes a structured vector quantization approach for stationary memoryless sources that combines the scalar-vector quantizer (SVQ) ideas (Laroia and Farvardin) with trellis coded quantization (Marcellin and Fischer). The resulting quantizer is called the trellis-based scalar-vector quantizer (TB-SVQ). The SVQ structure allows the TB-SVQ to realize a large boundary gain while the underlying trellis code enables it to achieve a significant portion of the total granular gain. For large block-lengths and powerful (possibly complex) trellis codes the TB-SVQ can, in principle, achieve the rate-distortion bound. As indicated by the results obtained here, even for reasonable block-lengths and relatively simple trellis codes, the TB-SVQ outperforms all other reasonable complexity fixed-rate quantizers.

[†] This work was supported in part by National Science Foundation grants NSFD MIP-9109109 and NSFD CDR-85-00108.

## I. Introduction

Quantizers can be broadly classified into two different types – scalar quantizers and vector quantizers. Scalar quantizers quantize one source sample at a time and are very simple to implement. The simplest kind of scalar quantizer is the fixed-rate uniform scalar quantizer [1], [2]. This quantizer has quantization (or reconstruction) levels uniformly spaced in a certain interval $I \equiv (b, a]$, $b < a$ and $a, b \in \mathbb{R}$; the quantization thresholds are midway between the levels. The rate of this quantizer, in bits/sample, is the logarithm (base 2) of the number of levels. When the source sample to be quantized is inside the interval $I$, the resulting quantization distortion is called granular noise; when the source sample is outside $I$, this event is referred to as an overload event and the resulting distortion is called overload noise. For a given source and rate, the spacing between the levels of this quantizer is determined to minimize the average distortion which is a weighted sum of the granular noise and the overload noise. Let us take the uniform scalar quantizer as the baseline quantization system and study the various gains obtained by more complex quantization schemes against it.

Optimal fixed-rate scalar quantizers introduced by Max [2] and Lloyd [3], minimize the average (squared-error) distortion for a given number of quantization levels (or rate) and are known as Lloyd-Max quantizers (LMQs). As expected intuitively, the quantization levels of these quantizers are relatively closely spaced in regions where the source probability density $p(x)$ is large and widely spaced where it is small, resulting in better performance than uniform quantizers. This type of gain over uniform quantizers will be called the *non-uniform density gain* and is dependent on the source probability density. In spite of this gain, there exists a big gap between the LMQ performance and the optimum achievable performance given by the rate-distortion theory. To explain this gap, consider a block $\mathbf{x}$ of $m$ source samples ($m$-vector) from a stationary memoryless source $\mathcal{X}$. According to the asymptotic equipartition property (AEP), as $m$ becomes large, the $m$-dimensional probability density function $p(\mathbf{x})$ becomes increasingly localized to a region $\mathcal{R}_m \in \mathbb{R}^m$ inside which the density $p(\mathbf{x})$ is almost uniform. For a Gaussian source the region $\mathcal{R}_m$ is an $m$-sphere shell while for a Laplacian source it is an $m$-pyramid shell [112], [13]. Scalar quantizers quantize the source samples based on the one-dimensional density $p(x)$ and fail to capitalize on this localization of density that occurs in higher dimensional space, thus resulting in poor performance.

To improve the performance of scalar quantizers one could use variable-length encoding of the quantizer output. Optimal entropy-constrained scalar quantizers (ECSQs) [4], [27] minimize the average distortion for a given output entropy and are known to asymptotically (at high rates) perform within 1.53 dB of the rate-distortion bound [5]. This increased performance comes at the cost of a variable-rate output with its concomitant difficulties. In this paper we focus our attention only on fixed-rate quantization of memoryless sources.

Vector quantizers ($m$-dimensional) quantize an $m$-vector of source samples at a time, enabling the placement of codevectors (reconstruction vectors) in high probability density regions of the $m$-space. For memoryless sources there are three different gains that vector quantizers can realize over uniform scalar quantizers. The first is the *boundary gain* [6], [7], [29] (called the shape advantage in [29]) and, as the name suggests, is realized by selecting an appropriate codebook boundary which ensures that most of the codevectors are placed in the high probability region $\mathcal{R}_m$ as dictated by the AEP. In contrast, the uniform scalar quantizer output (in $m$-dimensions) lies on a cubic-lattice bounded by an $m$-cube. A quantizer that has all its codevectors uniformly distributed in $\mathcal{R}_m$ asymptotically achieves the optimal boundary gain as $m \to \infty$. The second type of gain over uniform scalar quantization is the *granular gain* [6], [7], [29] (called the space filling advantage in [29]) that is achieved by having more spherical (assuming squared-error distortion measure) Voronoi regions of codevectors than the $m$-cubes for uniform scalar quantizers. The boundary and granular gains are realized only by vector quantizers. As noted in [6], quantization and transmission are dual problems. The granular gain in quantization is analogous to the shaping gain [8] in transmission and the boundary gain is analogous to the coding gain in transmission. Finally, there is the non-uniform density gain and just as in the scalar case, it results from having the codevectors closely spaced in higher probability density regions and farther apart in lower probability density regions of $\mathbb{R}^m$. Note that due to the uniform density nature of the region $\mathcal{R}_m$, this gain disappears in optimally shaped codebooks for a large $m$. The non-uniform density gain has no analog in transmission.

For 'uniform' (lattice or trellis code based) vector quantizers the definitions of boundary and granular gains given in [6] quantify these gains. The boundary gain of an $m$-dimensional uniform vector quantizer with a codebook bounded by a region $\mathcal{R}$ is defined as the inverse ratio of the normalized (to two dimensions) volume of the region $\mathcal{R}$ to the normalized volume of an $m$-cube that has the same overload probability[1]. The granular gain of an $m$-dimensional uniform vector quantizer (assuming squared-error distortion measure) is the inverse ratio of the second moment of the Voronoi regions of its codevectors to the second moment of an $m$-cube of the same volume.

These definitions are however too restrictive as they apply only to uniform vector quantizers. In practice, most vector quantizers are 'non-uniform' and the sizes and shapes of the Voronoi regions of their codevectors can be very different. For these quantizers the distinction between overload and granular noise becomes weaker and hence it can be difficult to differentiate between boundary and non-uniform density gains. Another problem with the above definition of the boundary gain, which is motivated by the coding gain in transmission, is that it effectively assumes the distortion measure to be the overload probability. This is rarely the case in quantization where the most commonly used distortion metric is the squared-error distortion. In spite of their drawbacks, these definitions are useful for uniform quantizers and it is shown in [6] that at high quantization rates, the boundary gain of these quantizers is decoupled from the granular gain and the two can be realized independently. The granular gain in this case is upper-bounded by 1.423 (1.53 dB) for the squared-error distortion and is independent of the source probability density. The boundary gain is usually larger for most non-uniform sources and is dependent on the source density. Next, we classify the various vector quantization schemes [6]-[15] in terms of the gains they realize.

Locally optimal $m$-dimensional vector quantizers introduced by Linde, Buzo and Gray in [9], perform arbitrarily close to the rate-distortion bound as $m$ becomes large. Their implementation complexity (both computational and storage) is exponential in $mr$ (where $r$ is the per sample rate), making them impractical even at modest rates and dimensions ($mr > 12$). Suboptimal tree-searched vector quantizers [10] solve the computational complexity problem but only at the cost of added storage complexity. Other vector quantizers such as multi-stage vector quantizers [11], are implementable for a large $mr$ but can result in a significant performance degradation over optimal vector quantizers.

To solve these problems, a variety of vector quantizers motivated by the AEP have

[1] An overload event in vector quantization occurs when the source vector lies outside the codebook boundary and results in overload noise. When the source vector is inside the codebook boundary, the resulting distortion is called granular noise.

been proposed for memoryless sources. Among these are the pyramid vector quantizer for Laplacian sources and the spherical vector quantizer for Gaussian sources [12], [13]. These quantizers place their codevectors at the intersection of a cubic lattice with the region $\mathcal{R}_m$ ($m$-pyramid shell for Laplacian and $m$-sphere shell for Gaussian sources) and as dictated by the AEP, for a large $m$, achieve the optimal boundary gain. Although asymptotically in $m$ these quantizers can realize all the boundary gain, for any fixed value of $m$, it is always preferable to place the codevectors not only on the $m$-shell but also inside it. This is because the probability density is higher inside the shell than on it. No granular gain or non-uniform density gain is realized by these quantizers as they are based on a cubic lattice.

The pyramid and the spherical vector quantizers described above are essentially special cases of a structured fixed-rate vector quantizer, called the scalar-vector quantizer, introduced by Laroia and Farvardin in [14]. This scalar-vector quantizer (denoted by SVQ) is a vector quantizer derived from a variable-length scalar quantizer and, for a class of useful sources, can (asymptotically in $m$) achieve optimal boundary gains by placing the codevectors on and inside the region $\mathcal{R}_m$. In addition, the SVQ allows non-uniformly spaced codevectors and hence the possibility of achieving some non-uniform density gain. No granular gain is realized by the SVQ as the underlying grid is rectangular.

The trellis coded quantizer (TCQ) introduced by Marcellin and Fischer in [15] is also derived from a scalar quantizer and uses Ungerboeck's idea of coding by set partitioning [16] to realize a significant portion of the maximum granular gain of 1.53 dB. This quantizer also realizes some non-uniform density gain by allowing the underlying scalar quantizer to have non-uniformly spaced levels, but makes no explicit attempt to exploit the boundary gain. The entropy-constrained TCQ (ECTCQ) of Fischer and Wang [28] uses variable-length coding to realize a gain equal to the optimal boundary gain (in addition to the granular gain of the TCQ) and for most sources with smooth densities performs within 0.5 dB of the rate-distortion bound. The problems associated with variable-length coding however limit the usefulness of the ECTCQ in most practical situations.

The lattice-bounded lattice and trellis-bounded trellis quantizers introduced by Eyuboglu and Forney [6] for Gaussian sources derive their motivation from data transmission and are dual of the multidimensional lattice-bounded lattice and the trellis-bounded trellis constellations, respectively. Like the TCQ, these quantizers use a lattice and a trellis the TB-SVQ are also described and their complexities considered. The design of the TB-SVQ is discussed in Section V along with the results on the performance of this quantizer for the generalized Gaussian source with three different parameters (including Gaussian and Laplacian sources). These results are also compared against those of some of the other quantization schemes described above.

## II. The Scalar-Vector Quantizer

The scalar-vector quantizer (SVQ) is a specific kind of vector quantizer in which the codebook structure is derived from a variable-length scalar quantizer [14]. Consider an $n$-level scalar quantizer $\mathcal{S}$ described in terms of the set of quantization levels $\mathcal{Q} = \{q_1, q_2, \ldots, q_n\}$ and the corresponding set of positive integer lengths $\mathcal{L} = \{\ell_1, \ell_2, \ldots, \ell_n\}$, where $\ell_i$ is the length associated with the level $q_i$, $i \in J_n \equiv \{1, 2, ..., n\}$. These lengths are not required to be the codeword lengths of a uniquely decodable code such as the Huffman code. If each component of an input $m$-vector (block of $m$ input samples) is separately quantized using $\mathcal{S}$, the quantized vector is in $\mathcal{Q}^m$ which is an $m$-dimensional grid of $n^m$ points. The codebook $\mathcal{Z}$ of the SVQ $\mathcal{V}$ derived from $\mathcal{S} \equiv (\mathcal{Q}, \mathcal{L})$ is a subset of $\mathcal{Q}^m$. Specifically, if $\mathcal{V}$ is a rate $r$ bits/sample SVQ, a vector in $\mathcal{Q}^m$ is a codevector of $\mathcal{V}$ only if its total length (defined as the sum of the lengths of its components) is no greater than a threshold $L$, where $L$ is chosen such that the codebook contains no more than $2^{mr}$ codevectors. A formal definition of the SVQ now follows.

*Definition 1:* An $m$-dimensional SVQ $\mathcal{V}$ derived from a the variable-length scalar quantizer $\mathcal{S} = (\mathcal{Q}, \mathcal{L})$ is a vector quantizer with a codebook $\mathcal{Z}$ given as,

$$\mathcal{Z} = \{\mathbf{z} \equiv (z_1, z_2, \ldots, z_m) \in \mathcal{Q}^m \ : \ \ell_{f(z_1)} + \ell_{f(z_2)} + \cdots + \ell_{f(z_m)} \leq L\}, \tag{1}$$

where the index function $f(\cdot) : \mathcal{Q} \rightarrow J_n$ is defined as,

$$f(q_i) = i, \ i \in J_n \ . \tag{2}$$

For a rate $r$ bits/sample SVQ, the threshold $L$ is selected as the largest integer such that $\mathrm{card}(\mathcal{Z}) \leq 2^{mr}$.

An $m$-dimensional SVQ $\mathcal{V}$ is completely defined in terms of the triple $(\mathcal{Q}, \mathcal{L}, L)$, i.e., the quantization levels of the scalar quantizer $\mathcal{S}$, the set of lengths associated with these levels and a threshold. It is shown in [14] that this simple structure of the SVQ codebook results in fast and efficient algorithms for codebook search and codevector encoding/decoding. Several algorithms for the design of the SVQ are also given in [14]. Next we show that by choosing an appropriate set of lengths the SVQ can realize the boundary gain.

The AEP dictates that for a Gaussian source an $m$-sphere bounded codebook asymptotically realizes the optimal boundary gain as $m$ increases. Also, for a given dimension $m$ and rate $r$, an $m$-sphere codebook boundary minimizes the overload probability and maximizes the boundary gain [6]. An $m$-sphere bounded cubic lattice based codebook can be expressed as an SVQ codebook by choosing the underlying scalar quantizer $\mathcal{S}$ as a uniform scalar quantizer and defining the length $\ell_i$ associated with the level $q_i$ as $\ell_i = |q_i|^2$. Similarly, the asymptotically optimal codebook boundary for a Laplacian source is an $m$-pyramid. The SVQ can accomplish this if the level $q_i$ is assigned a length $\ell_i = |q_i|$.

While for the generalized Gaussian family of sources [2] the optimal codebook boundary $\mathcal{R}_m$ can be easily determined and visualized, for other sources with arbitrary distributions it is generally not easy to determine $\mathcal{R}_m$ and then place the codevectors on or inside it. Most AEP-motivated asymptotically optimal structured quantization schemes will therefore be difficult to implement for such sources. However, for these sources the SVQ can be derived from the optimal ECSQ designed for the source distribution. As shown in [14], this SVQ asymptotically (in block-length) achieves the ECSQ performance which is inferior to the rate-distortion limit only by an amount equal to the maximum granular gain (at high rates) [5], [27].

## III. Trellis Coded Quantization (TCQ)

In the previous section we have shown that the SVQ can asymptotically (in block-length) achieve the optimal boundary gain by placing its codevectors only in the high probability density region of the source space. However, the underlying lattice (grid) on which the codevectors are located is cubic (rectangular) and hence no granular gain is realized by the SVQ. To realize the granular gain the codevectors must be placed on a more densely packed multidimensional lattice than the cubic lattice. A dense lattice has a more spherical Voronoi region which therefore has a smaller second moment than a cube of

[2] The *pdf* of the generalized Gaussian distribution is parameterized by $\alpha$ and is of the form $p(x) = c_1 e^{-c_2|x|^\alpha}$, where $c_1$ is the normalization constant and $c_2$ determines the variance of the distribution. For $\alpha = 2$ the generalized Gaussian distribution reduces to the Gaussian distribution and for $\alpha = 1$ it reduces to the Laplacian distribution.

the same volume. Quantizers based on dense lattices hence result in a smaller (up to 1.53 dB) average squared-error distortion than those based on a cubic lattice. The granular gain of several lattices is given in [6]. Among these is the 24-dimensional Leech lattice, which is the densest known 24-dimensional lattice. The Leech lattice gives a granular gain of 1.03 dB but has a fairly high implementation complexity.

Trellis codes are generalizations of finite dimensional lattices to infinite dimensional space (sequence space) just as convolutional codes are generalizations of block codes. Trellis codes are constructed by using a trellis diagram to select a sequence of subsets (cosets of a sublattice) of a redundant lattice and were first proposed by Ungerboeck in his celebrated paper [16] which combines coding with modulation. Since then they have extensively been studied in the context of data transmission [18]-[22].

Marcellin and Fischer were the first to apply Ungerboeck's trellis coding ideas to quantization [15]. They showed that for a given complexity, trellis codes can result in higher granular gains than dense lattices. As pointed out in [6] and worth repeating here, the gains obtained in using trellis codes in transmission and quantization are of a different nature. Consider a trellis code and a cubic lattice with the same normalized point density. The coding gain in transmission results from a larger minimum distance between trellis sequences as compared to the cubic lattice. On the other hand, the granular gain in quantization results from a smaller second moment of the Voronoi region of the trellis sequence compared to the Voronoi region of the cubic lattice.

In [15], Marcellin and Fischer have studied the granular gains of several Ungerboeck's one-dimensional trellis codes based on partitioning $\mathbf{Z}$ into the four cosets of $4\mathbf{Z}$. These codes have a redundancy of 1 bit/dimension. The granular gains obtained in [15] for 4, 8, 16, 32, 64, 128 and 256-state trellis codes are given in Table 1. These values were obtained by high-rate trellis coded quantization of uniform sources. Because of the 1 bit/dimension redundancy of the trellis codes, the quantizer alphabet of the trellis coded quantizer (TCQ) has twice as many quantization levels as a scalar quantizer at the same rate. Since no boundary gain can be realized for uniform sources, the gain over uniform scalar quantization in this case is the granular gain. The gains in Table 1 are for large quantization delays and will be somewhat less for smaller delays. Due to the similarity between granular gain and the shaping gain in transmission, the dependence of granular gain on delay is the same as that of the shaping gain on delay as studied by Forney for trellis shaping [23].

The granular gain of the simple 4-state Ungerboeck trellis code is about 1 dB which is close to the 1.03 dB of the 24-dimensional Leech lattice. The 4-state code is however considerably simpler to implement. The trellis diagram of this code along with the four partitions of the quantizer alphabet are shown in Fig. 1.

Trellis coded quantization has also been applied to some non-uniform sources such as Gaussian and Laplacian sources [15]. For these sources, the quantization levels are not restricted to (scaled) integers but can have non-uniform spacing, permitting the TCQ to realize some non-uniform density gain. The underlying 'trellis code' hence is no longer a geometrically uniform code [24] and we shall call it a non-uniform trellis code. In spite of the fact that no explicit attempt is made to realize the boundary gain, the TCQ performs remarkably well for both Gaussian and Laplacian sources at rates of 3 bits/sample and less [15].

## IV. Trellis-Based Structured Vector Quantizer (TB-SVQ)

Having studied the boundary gain property of the SVQ and the granular gain property of the TCQ, the most logical next step is to try and combine these two into a single quantizer that realizes both these gains. We call this quantizer the trellis-based scalar-vector quantizer (TB-SVQ). The TB-SVQ uses the SVQ idea to shape the 'codebook' boundary, but unlike the SVQ the 'codevectors' of the TB-SVQ — referred to as code-sequences, do not lie on a rectangular grid but are sequences from a (non-uniform) trellis code[3]. Note that the SVQ as defined in Section II has a finite block-length $m$ while the TB-SVQ code-sequences are vectors in an infinite dimensional sequence space. We will use the term TB-SVQ block-length (or dimension) to refer to the block-length of the underlying SVQ. The TB-SVQ is now defined as follows.

An $m$-dimensional TB-SVQ is specified by an alphabet (set of quantization levels) $\mathcal{Q} \equiv \{q_1, q_2, \ldots, q_{2n}\}$, a corresponding set of lengths $\mathcal{L} \equiv \{\ell_1, \ell_2, \ldots, \ell_{2n}\}$ (where $\ell_i$ is the length of $q_i$), a threshold $L$ and a trellis code $\mathcal{T}(\mathcal{Q})$ whose branches are labeled with elements in $\mathcal{Q}$. The TB-SVQ 'codebook' is a collection of code-sequences such that every code-sequence $\{c_i\}$ is a sequence from the trellis code $\mathcal{T}(\mathcal{Q})$ with the additional constraint that when $\{c_i\}$ is partioned into $m$-vectors (each in $\mathcal{Q}^m$), the total length of each vector is

[3] Unless stated otherwise, for the rest of this paper we will assume that the trellis code is one dimensional and has a redundancy of 1 bit/dimension.

no greater than the threshold $L$. As before, the total length of a vector is the sum of the lengths of its components.

Unlike a TCQ, the number of quantization levels ($2n$) of a TB-SVQ is not restricted to $2^{r+1}$, where $r$ is the quantizer rate in bits/sample. In general, the TB-SVQ has more than $2^{r+1}$ levels ($n > 2^r$). This additional alphabet expansion is similar to the shaping constellation expansion in transmission [8]. If there is no additional alphabet expansion ($n = 2^r$), the TB-SVQ effectively reduces to a TCQ which is analogous to an unshaped trellis-coded constellation.

Given a sequence $\{x_i\}$ of source samples, the quantization operation of the TB-SVQ consists of searching for the code-sequence that is closest to $\{x_i\}$ in some given distortion metric — usually squared-error. Later in this section we will describe an algorithm to do the code-sequence search but first we address the problem of encoding and decoding these sequences. It is shown next that by imposing some minor constraints on the TB-SVQ the corresponding algorithms of the SVQ can be used for this purpose. The encoding and decoding algorithms described here are respectively similar to the decoding and encoding algorithms for the SVQ-shaped trellis-coded constellations given in [25].

## *A. Encoding and decoding algorithms for the TB-SVQ*

A convolutional code based implementation of a trellis code provides a natural mapping from a bit sequence to a trellis sequence and vice versa. This leads to simple encoding and decoding algorithms for the TCQ. Since in general, all trellis sequences are not code-sequences of the TB-SVQ, the task of encoding and decoding for the TB-SVQ is not as straightforward. However, by placing the following two constraints on the TB-SVQ we will show that the corresponding SVQ algorithms can be used for this purpose.

(i) The trellis code $\mathcal{T}(\mathcal{Q})$ must partition the alphabet $\mathcal{Q}$ (containing $2n$ quantization levels) into two subsets $\mathcal{Q}_A$ and $\mathcal{Q}_B$ such that both $\mathcal{Q}_A$ and $\mathcal{Q}_B$ have $n$ levels each and all outgoing transitions from every trellis state are either associated with levels in $\mathcal{Q}_A$ or $\mathcal{Q}_B$ but not both. In other words, the set of allowed outputs in any trellis state is either $\mathcal{Q}_A$ or $\mathcal{Q}_B$. This constraint is satisfied by all of Ungerboeck's trellis codes.

(ii) It should be possible to pair every level $a_i$ in $\mathcal{Q}_A$ with a distinct level $b_i$ in $\mathcal{Q}_B$, such that $a_i$ and $b_i$ have the same length. Let the set of pairs be denoted by $\mathcal{P} \equiv \{\mathbf{P}_1, \mathbf{P}_2, \ldots, \mathbf{P}_n\}$, where $\mathbf{P}_i \equiv (a_i, b_i)$. To every pair $\mathbf{P}_i$, $i \in J_n$, assign a length $l_i$ equal to the length of the level $a_i$ (or $b_i$). Note that according to the first constraint, codebook search algorithm of the SVQ [14] and the Viterbi trellis search algorithm of the TCQ [15]. Before describing the algorithm we introduce some notation that will be helpful for this purpose.

Consider a $\nu$-state trellis code $\mathcal{T}(\mathcal{Q})$ defined on an alphabet $\mathcal{Q}$ with $2n$ quantization levels. The trellis diagram for this code can be thought of as a $\nu$-state finite-state machine (FSM). In the $k^{th}$ sample interval, the input $\theta_k \in \Theta \equiv \{0, 1, 2, \ldots, n-1\}$ to the FSM chooses its output $o_k$ from one of $n$ quantization levels allowed in the current state $s_k \in \Sigma \equiv \{0, 1, 2, \ldots, \nu - 1\}$ and also determines the state $s_{k+1}$ for the next sample interval. The FSM can be completely described by two functions, the next-state function $\eta$ and the output function $\mathcal{O}$. The next-state function $\eta(\cdot,\cdot) : \Sigma \times \Theta \rightarrow \Sigma$, with $s_{k+1} = \eta(s_k, \theta_k)$ and the output function $\mathcal{O}(\cdot,\cdot) : \Sigma \times \Theta \rightarrow \mathcal{Q}$, with $o_k = \mathcal{O}(s_k, \theta_k)$. For all trellis codes of interest, the next-state function $\eta$ is invertible for a given input $\theta_i$ and hence we can define a previous-state function $\mu(\cdot,\cdot) : \Sigma \times \Theta \rightarrow \Sigma$, with $s_k = \mu(s_{k+1}, \theta_k)$. The above notation will simplify the presentation of the search algorithm as the details of the trellis code are now buried in the two functions $\mathcal{O}$ and $\eta$ (or $\mathcal{O}$ and $\mu$).

Let $d(x, y)$ denote the distortion between a source sample $x$ and its reproduction $y$. We wish to quantize the given source output sequence to the code-sequence that minimizes the average distortion. Partition the source sequence into blocks of $m$ samples ($m$-vectors) and let $\{\mathbf{x}_k\}$ denote the sequence of source $m$-vectors, where $\mathbf{x}_k \equiv (x_{1k}, x_{2k}, \ldots, x_{mk})$.

Denote by $\Delta_k^s$ the minimum distortion that results when the first $k$ source vectors are quantized to a code-sequence such that the final trellis state is $s$, where $s \in \Sigma$. Now consider the quantization of the $k^{th}$ source vector. Denote by $D_{i,k}^{j,s}$ the minimum cumulative distortion (sum of the distortions of all source samples up to the $i^{th}$ sample of the $k^{th}$ source vector) that results when the first $i$ components of the $k^{th}$ quantized vector have a total length $j$ and the final trellis state is $s$. To make the notation simpler, we shall suppress the dependence on $k$ and write this distortion as $D_i^{j,s}$.

The distortion $D_{i+1}^{j,s}$ can recursively be obtained as,

$$D_{i+1}^{j,s} = \min_{\theta \in \Theta} [D_i^{j-\ell(\theta),s'} + d(x_{(i+1)k} - q(\theta))], \ i = 0, 1, 2, \ldots, m-1, \tag{3}$$

where $s' = \mu(s, \theta)$, $q(\theta) = \mathcal{O}(s', \theta)$ and $\ell(\theta)$ is the length of the quantization level $q(\theta)$, i.e., $\ell(\theta) = \ell_{f_{q(\theta)}}$, $f$ being the index function of (2). Also, define $D_i^{j,s} = \infty$ for $j < 0$, $i \leq 0$ and $D_0^{0,s} = \Delta_{k-1}^s$.

By using a dynamic programming algorithm similar to the one in [14], the equations in (3) can be solved for $D_m^{j,s}$, $\forall j \in J_L = \{1, 2, \ldots, L\}$ and $\forall s \in \Sigma$. From the $D_m^{j,s}$ we can obtain $\Delta_k^s$ as,

$$\Delta_k^s = \min_{j \in J_L} D_m^{j,s}, \ s \in \Sigma. \tag{4}$$

If in the dynamic programming algorithm for solving (3), we not only keep track of the distortion of the survivor path but also the path itself, we can easily obtain the code-sequence that resulted in the distortion $\Delta_k^s$, $s \in \Sigma$.

It is impractical in any system to wait for the entire source sequence (possibly very long) before deciding the code-sequence to which it should be quantized. Such a system would have very large quantization delays. In practice little is lost by making these decisions after a fixed delay of $d$ source $m$-vectors. Hence, decision about the $k^{th}$ quantized vector $\mathbf{v}_k$ of the desired code-sequence can be made after determining $\Delta_{k+d}^s$, $\forall s \in \Sigma$. The performance of the TB-SVQ will depend on the quantization delay $d$ and this dependence is examined in the next section.

The complexity of the search algorithm presented above is higher than the complexity of the SVQ codebook search algorithm. This is because the $D_i^{j,s}$ must be determined not only for all $i \in J_m$ and $j \in J_L$ but also for $s \in \Sigma$. Hence the complexity of the TB-SVQ search increases approximately linearly with the number of trellis states. For Ungerboeck's 4-state one-dimensional trellis code, the complexity is approximately four times that of the equivalent SVQ search complexity. The SVQ however realizes no granular gain while the TB-SVQ can easily realize up to 1 dB granular gain. Because of the granular gain advantage, the TB-SVQ complexity can be reduced significantly while maintaining a performance better than the SVQ. This can be done by assigning smaller lengths to quantization levels (coarser length quantization as described in [14]) and hence reducing the threshold $L$. For some applications however, the complexity can still be quite high if the quantization rate $r$ is high, or the trellis code has a large number of states $\nu$, or the block-length $m$ is large. To overcome this problem, in the next section we describe a suboptimal code-sequence search algorithm that performs very well but has a very low complexity.

## V. Design and Performance

We consider two different approaches for designing the TB-SVQ. In the first approach

(Method 1), the boundary gain of the TB-SVQ is obtained by deriving it from an entropy-constrained scalar quantizer and the granular gain is realized by the trellis code. The second approach (Method 2) uses the AEP to realize the boundary gain and works only for those sources for which the region $\mathcal{R}_m$ has a simple geometrical shape such as an $m$-sphere or an $m$-pyramid. While the latter approach requires the explicit knowledge of the source probability density and works only for a small (but important) class of sources, the former has the advantage that it can be used to design quantizers from a training sequence of source samples. Assuming squared-error as the distortion measure, we now describe these two design methods and present their performance results based on simulations. To reduce the search complexity, a suboptimal code-sequence search algorithm is presented for Method 2. The complexity of this algorithm is effectively that of a trellis search as in a TCQ.

### *A. Method 1*

In this method the TB-SVQ is first derived from an ECSQ designed for the source probability distribution; its quantization levels are then optimized using an algorithm similar to the Step A of the SVQ design algorithm [14]. In practice, a uniform-threshold[4] scalar quantizer (UTSQ) is used in place of the optimal ECSQ. The UTSQ is considerably simpler to design and its performance is very close to the performance of the optimal ECSQ for most sources [4], [27]. The spacing $\beta$ between the quantization thresholds of the (symmetric) UTSQ is determined such that its output entropy is equal to the desired rate $r$ (bits/sample) of the TB-SVQ. Since all trellis codes considered here have a redundancy of 1 bit/sample, we actually start with a symmetric UTSQ that has a spacing of $\beta/2$ between its thresholds and has an even number of levels. The quantization levels are placed between the thresholds at the center of the probability mass. As in the SVQ design, the length $\ell_i$ of the quantization level $q_i$ is determined as, $\ell_i = [b \log 1/p_i]$, where $p_i$ is the probability of the quantization region of $q_i$, square brackets indicate rounding off to the nearest integer and $b$ is a scalar factor that determines the effective round-off error [14] (for all results obtained here, we have taken $b = 1$). For this length assignment, the second constraint in Section IV.A will automatically be satisfied for sources with symmetric[5] *pdf*'s provided the trellis

[4] Not to be confused with the threshold $L$ of the TB-SVQ.

[5] For sources that are not symmetric, the lengths determined above might have to be altered to satisfy the constraint.

showing the variation with quantization delay $d$ (in number of source $m$-vectors; $m = 32$) of the TB-SVQ performance at 1 bit/sample for a Gaussian source. A similar plot for a Laplacian source at 3 bit/sample is given in Fig. 3. As expected, these plots indicate that the delay required to saturate the performance is smaller when the TB-SVQ has fewer states. A 4-state TB-SVQ requires a delay of only 1 source vector (32 samples) to perform within 0.03 dB of the results in Table 2.

*B. Method 2*

This method of designing TB-SVQs is based on the AEP and uniformly places the codevectors on (and inside) the region $\mathcal{R}_m$. This is similar to the approach presented in [6] for Gaussian and Laplacian sources, except that for these sources the TB-SVQ can implement the optimal codebook boundary exactly instead of approximating it by the (generalized) Voronoi region of some lattice[6]. For a Gaussian source, the region $\mathcal{R}_m$ is an $m$-sphere and therefore the desired trellis-based 'codebook' has an $m$-sphere boundary. This can be implemented by the TB-SVQ if the quantization levels are uniformly spaced — $\mathcal{Q} \equiv \{\ldots, -3\beta, -\beta, \beta, 3\beta, \ldots\}$ and the lengths are are assigned as $\ell_i = |q_i|^2/\beta^2$. The threshold $L$ is determined such that the primary codebook has $2^{mr}$ codevectors. The resulting codebook boundary is hence an $m$-sphere of squared-radius $\beta^2 L$. The trellis code $\mathcal{T}(\mathcal{Q})$ here is geometrically uniform and no non-uniform density gain is realized. This is however not a big drawback since the suboptimal code-sequence search algorithm described below enables quantization at large block-lengths, reducing the significance of the non-uniform density gain. For a Laplacian source the region $\mathcal{R}_m$ is an $m$-pyramid. The $m$-pyramid codebook boundary in this case can be implemented as above provided the lengths are assigned as $\ell_i = |q_i|/\beta$. The only parameter that needs to be determined to design quantizers for Gaussian and Laplacian sources is the spacing $2\beta$ between the quantization levels. This can be done by plotting the quantizer distortion as a function of $\beta$ and choosing that value of $\beta$ for which the distortion is minimized.

To reduce the implementation complexity of the TB-SVQs designed using Method 2, we have used the following suboptimal code-sequence search algorithm. A simple trellis search is first performed on the given sequence of source $m$-vectors. In other words, the source sequence is quantized just as in a TCQ. If a source $m$-vector is quantized to a

[6] Even for most other generalized Gaussian sources, the TB-SVQ implements a much better approximation of the optimal codebook boundary than the approach given in [6].

vector of total length less than or equal to $L$, the quantized output is accepted and the next $m$-vector in the source sequence is quantized. If, on the other hand, the source $m$-vector is quantized to a vector of total length greater than $L$ — corresponding to an overload event — the source vector is (gradually) moved towards the closest point on the codebook boundary and the trellis search is attempted again starting from the previous quantized vector that did not result in an overload. This process continues until the resulting quantized vector satisfies the total length constraint. For the Gaussian source the 'overload' source $m$-vector can be moved towards the closest point on the $m$-sphere codebook boundary by simply scaling by a factor less than unity. For Laplacian sources this is done by subtracting a fixed amount from the magnitude of each of the $m$ components of the source vector.

The suboptimality of the above search algorithm arises from the fact that in case of an overload, the search does not necessarily result in quantization to the closest vector inside the codebook boundary. In general, we found that the performance of this algorithm is not significantly affected even if the overload source vector is moved towards the codebook boundary in large steps.

The complexity of this search algorithm is essentially equal to the complexity of a TCQ based on the same trellis code, and is significantly less than the complexity of the optimal search algorithm of Section IV. This reduced complexity makes it possible to increase the block-length of the TB-SVQ, possibly compensating for some of the loss due to the suboptimality. Table 3 gives the performance of the suboptimally searched TB-SVQs of Method 2 for Gaussian and Laplacian sources at a block-length $m = 64$. A comparison with Table 2 reveals that for a Gaussian source the performance of the suboptimally searched quantizer for $m = 64$ is similar (within 0.2 dB) to the performance of the optimally searched TB-SVQ of Method 1 for $m = 32$. For the Laplacian case, the performances are comparable at rates 2 and 3 bits/sample but the suboptimal search performs considerably worse (up to about 0.6 dB) at 1 bit/sample.

We have also tried to use a two codebook approach similar to the one described in [7]. A second codebook — called the overload codebook — is used to quantize the overload source vectors directly without having to move them inside the boundary. It was found that at rates less than or equal to 2 bits/sample, this did not result in any performance improvement. For higher rates this leads to some gain. The results for Gaussian and

Laplacian sources at a rate of 3 bits/sample are given in Table 4. These results were obtained for equal number of vectors in the two codebooks and the spacing between the quantization levels for each codebook was determined by experimentation. The results in Table 4 are even better than the corresponding results for the optimally searched quantizers (Table 2) and are the best results reported for the fixed-rate quantization of Gaussian and Laplacian sources at 3 bits/sample.

## VI. Conclusions

In this paper we have combined the scalar-vector quantizer of [14] with the trellis coded quantizer of [15] and proposed a new quantizer called the trellis-based scalar-vector quantizer. The SVQ structure enables the TB-SVQ to realize a significant boundary gain while the trellis code allows it to realize the granular gain. In addition, by having non-uniformly spaced quantization levels in the TB-SVQ alphabet, some non-uniform density gain can also be attained. The TB-SVQ is hence capable of realizing all the three gains over the baseline uniform scalar quantizer. It can, in principle, approach the rate-distortion bound at large block-lengths and with powerful (possibly complex) trellis codes while maintaining a fixed rate. Even with the relatively simple Ungerboeck's one-dimensional trellis code, it can realize a significant portion of the total granular gain and performs better than all other reasonable complexity fixed-rate quantizers.

## References

1. N. S. Jayant and P. Noll, *Digital Coding of Waveforms: Principles and Application to Speech and Video*, Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1984.
2. J. Max, "Quantizing for Minimum Distortion," *IRE Trans. Inform. Theory*, Vol. IT-6, pp. 7-12, March 1960.
3. S. P. Lloyd, "Least Squares Quantization in PCM," *IEEE Trans. Inform. Theory*, Vol. IT-28, pp. 129-137, March 1982.
4. N. Farvardin and J. W. Modestino, "Optimum Quantizer Performance for a Class of Non-Gaussian Memoryless Sources," *IEEE Trans. Inform. Theory*, Vol. IT-30, pp. 485-497, May 1984.
5. H. Gish and J. N. Pierce, "Asymptotically Efficient Quantizing," *IEEE Trans. Inform. Theory*, Vol. IT-14, pp. 676-683, Sept. 1968.
6. M.V. Eyuboglu and G.D. Forney, Jr., "Lattice and Trellis Quantization with Lattice-

and Trellis-Bounded Codebooks – Part I: High-Rate Theory for Memoryless Sources," submitted to *IEEE Trans. Inform. Theory*, Dec. 1990.

7. M.V. Eyuboglu and A.S. Balamesh, "Lattice and Trellis Quantization with Lattice- and Trellis-Bounded Codebooks – Construction and Implementation for Memoryless Sources," submitted to *IEEE Trans. Inform, Theory*, Oct. 1991.
8. G.D. Forney, Jr. and L.F. Wei, "Multidimensional Constellations – Part I: Introduction, Figures of Merit, and Generalized Cross Constellations," *IEEE J. Select. Area Commun.*, Vol. 7, No. 6, pp. 877-892, Aug. 1989.
9. Y. Linde, A. Buzo and R. M. Gray, "An Algorithm for Vector Quantizer Design," *IEEE Trans. Commun.*, Vol. COM-28, pp. 84-95, Jan. 1980.
10. R. M. Gray and Y. Linde, "Vector Quantizers and Predictive Quantizers for Gauss-Markov Sources," *IEEE Trans. Commun.*, Vol. COM-30, pp. 381-389, Feb. 1982.
11. B.-H. Juang and A. H. Gray, "Multiple stage vector quantization for speech coding," *Proc. ICASSP*, pp. 597-600, April 1982.
12. T. R. Fischer, "A Pyramid Vector Quantizer," *IEEE Trans. Inform. Theory*, Vol. IT-32, pp. 568-583, July 1986.
13. T. R. Fischer, "Geometric Source Coding and Vector Quantization," *IEEE Trans. Inform. Theory*, Vol. IT-35, pp. 137-145, Jan. 1989.
14. R. Laroia and N. Farvardin, "A Structured Fixed-Rate Vector Quantizer Derived from a Variable-Length Scalar Quantizer," submitted to *IEEE Trans. Inform. Theory*, August 1991.
15. M.W. Marcellin and T.R. Fischer, "Trellis Coded Quantization of Memoryless and Gauss-Markov sources," *IEEE Trans. Commun.*, Vol. 38, pp. 82-93, Jan. 1990.
16. G. Ungerboeck, "Channel Coding with Multilevel/Phase Signals," *IEEE Trans. Inform. Theory*, Vol. IT-28, pp. 55-67, Jan. 1982.
17. R. Laroia, *Design and Analysis of a Fixed-Rate Structured Vector Quantizer Derived From Variable-Length Scalar Quantizers*, Ph.D. Thesis, Dept. of Electrical Engineering, Univ. of Maryland, College Park, May 1992.
18. G. D. Forney, Jr., "Coset Codes — Part I: Introduction and Geometrical Classification," *IEEE Trans. Inform. Theory*, Vol. IT-34, pp. 1123-1151, Sept. 1988.
19. G. D. Forney, Jr., "Coset Codes — Part II: Binary Lattices and Related Codes," *IEEE Trans. Inform. Theory*, Vol. IT-34, pp. 1152-1187, Sept. 1988.

20. A. R. Calderbank and N. J. Sloane, "New Trellis Codes Based on Lattices and Cosets," *IEEE Trans. Inform. Theory*, Vol. IT-33, pp. 177-195, March 1987.

21. L. F. Wei, "Trellis Coded Modulation with Multidimensional Constellations," *IEEE Trans. Inform. Theory*, Vol. IT-33, pp. 483-501, July 1987.

22. G. D. Forney, Jr. and A. R. Calderbank, "Coset Codes for Partial Response Channels; or, Coset Codes with Spectral Nulls," *IEEE Trans. Inform. Theory*, Vol. 35, pp. 925-943, Sept. 1989.

23. G. D. Forney, Jr., "Trellis Shaping," *IEEE Trans. Inform. Theory*, Vol. 38, pp. 281-300, March 1992.

24. G. D. Forney, Jr., "Geometrically Uniform Codes," *IEEE Trans. Inform. Theory*, Vol. 37, pp. 1241-1260, Sept. 1991.

25. R. Laroia, "On Optimal Shaping of Multidimensional Constellations — An Alternative Approach to Lattice-Bounded (Voronoi) Constellations," submitted to *IEEE Trans. Inform. Theory*, Nov. 1991.

26. R. Laroia, N. Farvardin and S. Tretter, "On SVQ Shaping of Multidimensional Constellations — High-Rate Large-Dimensional Constellations," submitted to *IEEE Trans. Inform. Theory*, Jan. 1992.

27. T. Berger, "Optimum Quantizers and Permutation Codes," *IEEE Trans. Inform. Theory*, Vol. IT-18, pp. 759-765, Nov. 1972.

28. T. R. Fischer and M. Wang, "Entropy-Constrained Trellis-Coded Quantization." *IEEE Trans. Inform. Theory*, Vol. 38, pp. 415-425, March 1992.

29. T. D. Lookabaugh and R. M. Gray, "High-Resolution Quantization Theory and the Vector Quantizer Advantage," *IEEE Trans. Inform. Theory*, Vol. 35, pp. 1020-1033, Sept. 1989.

| States | 4 | 8 | 16 | 32 | 64 | 128 | 256 |
|---|---|---|---|---|---|---|---|
| Gain (dB) | 0.99 | 1.10 | 1.15 | 1.23 | 1.28 | 1.33 | 1.36 |

**Table 1:** Granular gains of Ungerboeck's one-dimensional 4 to 256-state trellis codes obtained by Marcellin and Fischer [15].

| | | TB-SVQ (States) | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Rate | SVQ | 4 | 8 | 16 | 32 | TCQ | ECSQ | R(D) |
| 1 | 4.63 | 5.14 | 5.27 | 5.29 | 5.32 | 5.56 | 4.64 | 6.02 |
| 2 | 10.40 | 11.11 | 11.19 | 11.25 | 11.31 | 11.04 | 10.55 | 12.04 |
| 3 | 15.97 | 16.77 (2.98) | 16.86 (2.98) | 16.92 (2.98) | 16.95 (2.98) | 16.64 | 16.56 | 18.06 |

**Table 2a:** Performance (SNR in dB) of the TB-SVQ (Method 1) for four different trellis codes (4, 8, 16 and 32-state) and three different rates (1, 2 and 3 bits/sample). The source is Gaussian, the block-length of the SVQ and the TB-SVQ is 32 and the TCQ has 256 states [15]. The numbers in () indicate the actual rate if different from the desired rate.

| | | TB-SVQ (States) | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Rate | SVQ | 4 | 8 | 16 | 32 | TCQ | ECSQ | R(D) |
| 1 | 5.59 | 5.85 | 5.91 | 5.95 | 5.96 | 5.54 | 5.76 | 6.62 |
| 2 | 10.87 | 11.52 (1.99) | 11.62 (1.99) | 11.67 (1.99) | 11.72 (1.99) | 11.22 | 11.31 | 12.66 |
| 3 | 16.14 | 17.08 | 17.15 | 17.23 | 17.30 | 16.96 | 17.20 | 18.68 |

**Table 2b:** Performance (SNR in dB) of the TB-SVQ (Method 1) for four different trellis codes (4, 8, 16 and 32-state) and three different rates (1, 2 and 3 bits/sample). The source is Laplacian, the block-length of the SVQ and the TB-SVQ is 32 and the TCQ has 256 states [15]. The numbers in () indicate the actual rate if different from the desired rate.

| | | TB-SVQ (States) | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Rate | SVQ | 4 | 8 | 16 | 32 | TCQ | ECSQ | R(D) |
| 1 | 7.80 | 8.18 (0.99) | 8.23 (0.99) | 8.25 (0.99) | 8.28 (0.99) | — | 8.53 | — |
| 2 | 13.58 | 13.90 | 14.02 | 14.08 | 14.16 | — | 14.53 | — |
| 3 | 18.53 | 18.81 | 18.95 | 19.10 | 19.17 | — | 20.41 | — |

**Table 2c:** Performance (SNR in dB) of the TB-SVQ (Method 1) for four different trellis codes (4, 8, 16 and 32-state) and three different rates (1, 2 and 3 bits/sample). The source is generalized Gaussian with $\alpha = 0.5$, the block-length of the SVQ and the TB-SVQ is 32 and the SVQ has 256 states [15]. The numbers in () indicate the actual rate if different from the desired rate.

|  | TB-SVQ (States) |  |  |  |
|---|---|---|---|---|
| Rate | 4 | 8 | 16 | 32 |
| 1 | 5.39 (1.02) | 5.43 (1.02) | 5.46 (1.02) | 5.49 (1.02) |
| 2 | 11.18 | 11.22 | 11.25 | 11.28 |
| 3 | 16.92 | 16.98 | 17.01 | 17.05 |

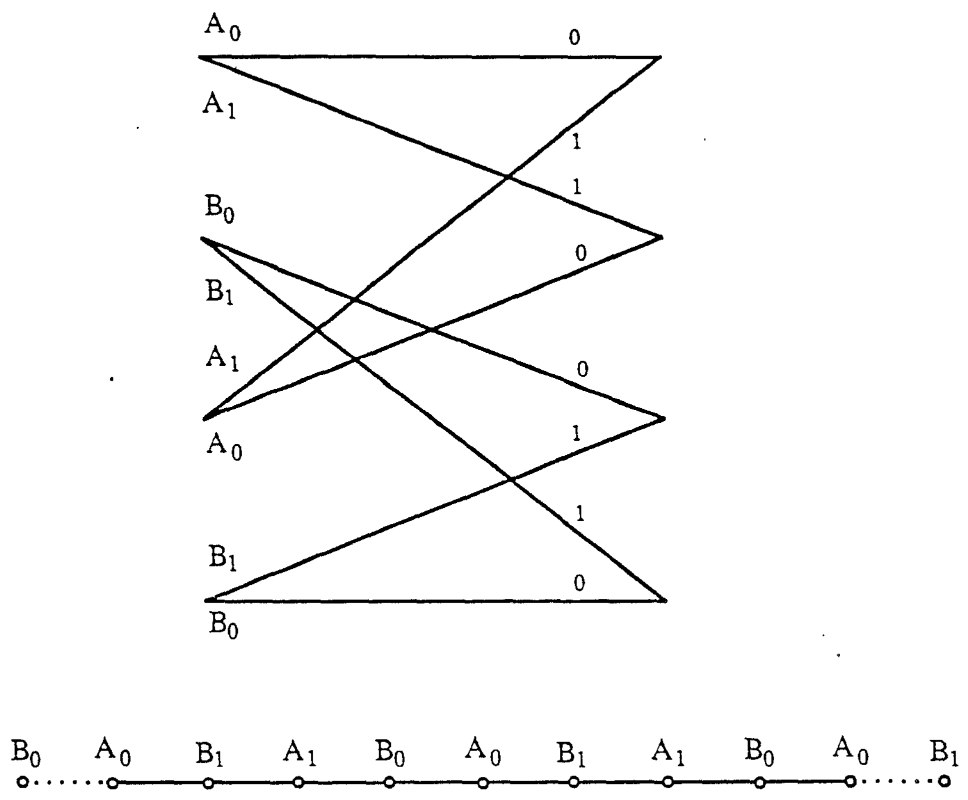
**Table 3a:** Performance (SNR in dB) of the 64-dimensional TB-SVQ designed using Method 2 for a Gaussian source. The code-sequence search algorithm is suboptimal. The results for four different trellis codes (4, 8, 16 and 32-state) and three different rates (1, 2 and 3 bits/sample) are given. The numbers in () indicate the actual rate if different from the desired rate.

|  | TB-SVQ (States) |  |  |  |
|---|---|---|---|---|
| Rate | 4 | 8 | 16 | 32 |
| 1 | 5.22 (0.99) | 5.29 (0.99) | 5.34 (0.99) | 5.39 (0.99) |
| 2 | 11.44 | 11.50 | 11.54 | 11.57 |
| 3 | 17.15 | 17.20 | 17.24 | 17.27 |

**Table 3b:** Performance (SNR in dB) of the 64-dimensional TB-SVQ designed using Method 2 for a Laplacian source. The code-sequence search algorithm is suboptimal. The results for four different trellis codes (4, 8, 16 and 32-state) and three different rates (1, 2 and 3 bits/sample) are given. The numbers in () indicate the actual rate if different from the desired rate.

| Source | TB-SVQ (States) | | | |
|---|---|---|---|---|
| | 4 | 8 | 16 | 32 |
| Gaussian | 17.09 | 17.13 | 17.16 | 17.19 |
| Laplacian | 17.44 | 17.48 | 17.51 | 17.54 |

**Table 4:** Performance (SNR in dB) of the 64-dimensional TB-SVQ with an overload codebook. The results for four different trellis codes (4, 8, 16 and 32-state) at rate 3 bits/sample are given.



$A_0$
0
$A_1$
1
1
$B_0$
0
$B_1$
$A_1$
0
$A_0$
1
1
$B_1$
0
$B_0$
$B_0$ $A_0$ $B_1$ $A_1$ $B_0$ $A_0$ $B_1$ $A_1$ $B_0$ $A_0$ $B_1$

**Fig. 1:** Ungerboeck's 4-state one-dimensional trellis code.

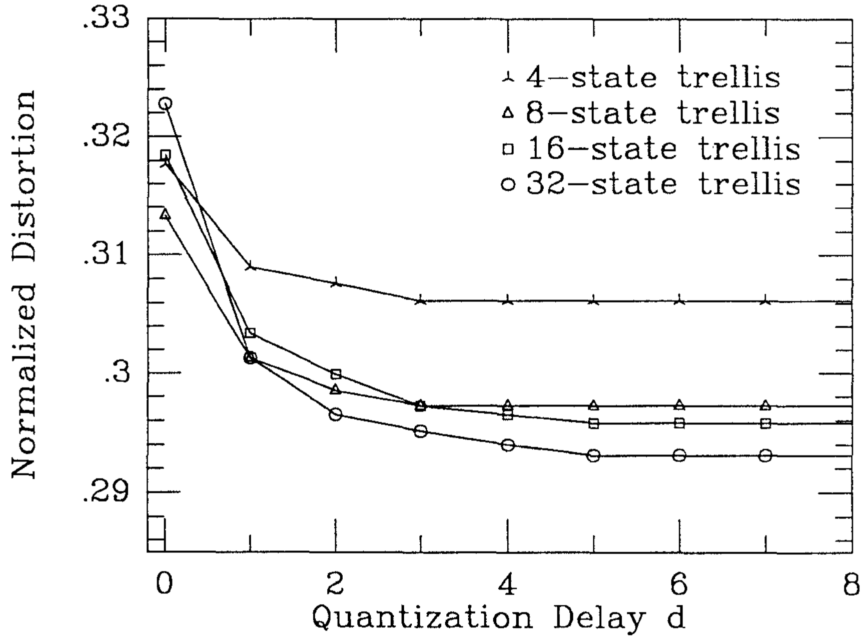


**Fig. 2:** TB-SVQ performance as a function of the quantization delay $d$ in number of $m$-vectors ($m = 32$) for a Gaussian source at 1 bit/sample.

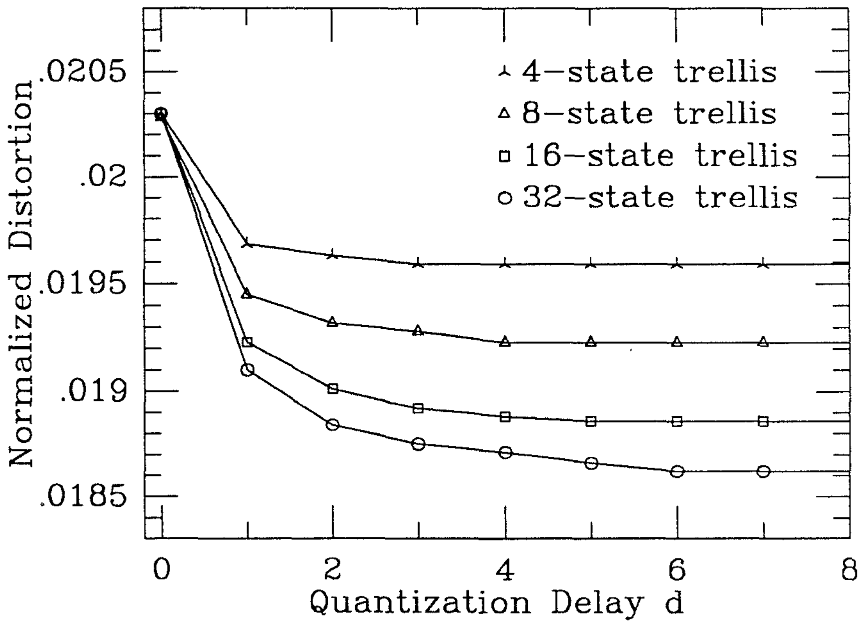


**Fig. 3:** TB-SVQ performance as a function of the quantization delay $d$ in number of $m$-vectors ($m = 32$) for a Laplacian source at 3 bits/sample.