# SPECTRAL CODING BY FAST VECTOR QUANTIZATION

*Erik Agrell*

Department of Information Theory
Chalmers University of Technology
S–412 96 Göteborg, Sweden

agrell@it.chalmers.se

***Abstract*—The "neighbor descent" method for fast vector quantization is well suited for speech coding. In a typical example, the time needed for encoding LPC coefficients was reduced to two percent of a full search. The price to pay is considerable precomputations and memory requirements.**

## I. INTRODUCTION

For the quantization of a number of analog values, a *vector quantizer* (VQ) allows in general a lower distortion than scalar quantization of each component [1]. The improvement increases with the number of scalars being quantized together, that is the dimension of the VQ. However, the encoding time also increases with the dimension, setting a practical limit on the size that can be handled in real-time applications such as speech coding.

This limit can be stretched by enforcing some structure upon the VQ. One example is *multi-stage vector quantization*, where the available number of bits is divided between two or more cascaded VQs [2]. Another approach is to divide the components between two or more separate VQs, so-called *split vector quantization* [3]. A scalar quantizer for every component is an extreme case of a split VQ. An alternative to structured vector quantization, without compromising output quality, is to apply an efficient search algorithm at the encoder. That is the theme of the present paper.

The input to a VQ is a $k$-dimensional vector $\mathbf{x}$ and the output is another vector $\mathbf{x}_j$, selected from a finite set $\{\mathbf{x}_1,\cdots,\mathbf{x}_n\}$. The index, or *codeword*, $j$ is represented as an $m$-bit integer. The number of available output vectors $n$ is thus equal to $2^m$. As a measure of the distortion induced by this quantization, the squared Euclidean distance

$$d(\mathbf{x},\mathbf{x}_j)=\left\|\mathbf{x}-\mathbf{x}_j\right\|^2 \tag{1}$$

is used. For a given input $\mathbf{x}$, an encoder should select the codeword $j$ that minimizes this distortion (the so-called *nearest-neighbor* rule).

The set of input vectors that are encoded as the same codeword $j$ is the *Voronoi region* $\mathcal{V}_j$:

$$\mathcal{V}_j=\left\{\mathbf{x}:\quad d(\mathbf{x},\mathbf{x}_j)\le d(\mathbf{x},\mathbf{x}_i);\ i=1,\ldots,n\right\} \tag{2}$$

An alternative specification of a Voronoi region is

$$\mathcal{V}_j=\left\{\mathbf{x}:\quad d(\mathbf{x},\mathbf{x}_j)\le d(\mathbf{x},\mathbf{x}_i);\ i\in\mathcal{A}_j\right\} \tag{3}$$

where all the redundant inequalities in (2) have been removed. $\mathcal{A}_j$, the *adjacency* of codeword $j$, is defined as

$$\mathcal{A}_j=\left\{i:\quad \mathcal{V}_i\cap\mathcal{V}_j\ne\varnothing\right\} \tag{4}$$

that is, the set of codewords whose Voronoi regions have a facet in common with $\mathcal{V}_j$. A pair of codewords being in each other's adjacency will be called *adjacent*.

To find a codeword whose Voronoi region comprises a given input vector $\mathbf{x}$, the natural approach is a *full search* (FS); every codeword is tested as a possible representation of $\mathbf{x}$ and the distortion is computed. The method is simple but slow. Many methods have been proposed to increase the speed. Surveys are found in [1, 4, 5].

This paper considers how to quantize linear predictive coding (LPC) coefficients of speech efficiently, avoiding a FS. It presents a family of encoding algorithms that use a precomputed table of adjacencies. Three members of the family are introduced. They are not restricted to VQs with a certain structure or a certain type of input, but they benefit from a source that has correlated vectors. This makes them suitable for speech coding.

## II. THE CONCEPT OF NEIGHBOR DESCENT

A FS for the optimal codeword includes much unnecessary work. When a few distortion measures have been computed, they should normally give a good indication of where to look for better codewords. From this notion arises the *neighbor descent* (ND) encoding method, which in its basic form (called "SND" below) was independently proposed by Okabe *et al.* [6] and Agrell [7]. The idea is to move from an initial hypothesis, over adjacent codewords with decreasing distortion, up to optimum. Note the similarity to descent methods for minimization of continuous functions.

Of the three members of the ND family, *steepest neighbor descent* (SND) takes the shortest route to optimum. The distortion is computed for all codewords in the adjacency of the current hypothesis, and the one with the smallest distortion is appointed the new hypothesis. When finally a hypothesis has emerged whose adjacency provides no lower distortion, the search stops. It can be proved that such a codeword is optimal; no suboptimal local minima exist.

*Random neighbor descent* (RND) does not wait until the whole adjacency of the current hypothesis is tested. As soon as an adjacent codeword is found with a lower distortion, the search

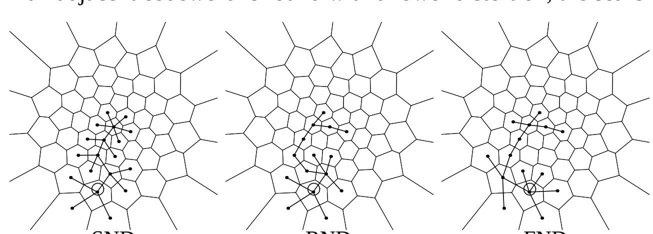


**Fig. 1.** A two-dimensional vector (circle) is encoded. Thin lines mark the Voronoi regions. The fat lines indicate how the ND algorithms search and which codewords they test, if they start at the center.

continues from there. The performance of this algorithm depends to some extent on the order in which the codewords are tested. Specifically, it should be ensured that, when the adjacency of a hypothesis is tested, the first two codewords lie on geometrically opposite sides. If the first of them has a higher distortion than the hypothesis, it is likely that the second one will provide an improvement.

We now observe that the change from one hypothesis to a better one indicates the location of the optimum. Most likely, it is favorable to continue in the same direction—that is "forwards". *Forward neighbor descent* (FND) uses a second table, complementing the adjacency table, whose entries answer the question "If a line is extended from $\mathbf{x}_j$ through $\mathbf{x}_k$, where $k \in \mathcal{A}_j$, into which Voronoi region does it pass when it leaves $\mathcal{V}_k$?". This information is precomputed for all codewords $j$ and all codewords $k$ in the adjacency of $j$. When a new hypothesis has been found, the "forward neighbor" is tested first. If its distortion is lower, it becomes the next hypothesis, otherwise the search continues as in RND.

Figure 1 gives an example on how the three ND algorithms behave. Even in this small example, ND avoids many of the distortion computations. For further efficiency, a table of the codewords that have already been tested is maintained during the encoding procedure. Otherwise some duplication of work would occur.

Essential for the method is that the adjacency of every codeword is known in advance. To compute them is a heavy task, but it is done only once for a given VQ. It can be accomplished using a linear programming technique [7] or by solving an equivalent convex hull problem [8].

## III. Quantization of LPC Coefficients

VQs were designed for line spectrum pair (LSP) frequencies of continuous speech, recorded from several speakers [9]. A split VQ technique was used, similar to the one described in [3]. The LSP frequencies obtained from a tenth order linear predictor were divided into two groups. The quantizer focused on in this section was designed for the first five frequencies. Twelve bits, 4096 codewords, were used.

ND is able to exploit the correlation between consecutive frames of speech. Because parameters tend not to change too rapidly, a good choice of initial hypothesis for the algorithms is the previous output. With this initialization, ND will operate as a method to continuously track the speech in a discrete parameter space.

32,184 input vectors were encoded, using FS and the three ND algorithms. They all arrived at the same results, but in different search time. Table 1 shows the average and maximum number of distortion computations. A remarkable improvement over FS is seen; RND, for instance, requires on an average 2.0 % of all distortion measures to find the optimal codeword.

This beats even a two-stage VQ [2] applied to the same input; if the available twelve bits are divided between two six-bit VQs, both of which are scanned using FS, a total of 128 distortions is computed, considerably more, on an average, than when RND is

**Table 1.** The number of distortion computations required by four methods when 5-dimensional LSP vectors were quantized to 12 bits.

| Method | Average | Maximum |
|---|---|---|
| FS | 4096 | 4096 |
| SND | 152.7 | 654 |
| RND | 82.1 | 238 |
| FND | 81.8 | 239 |

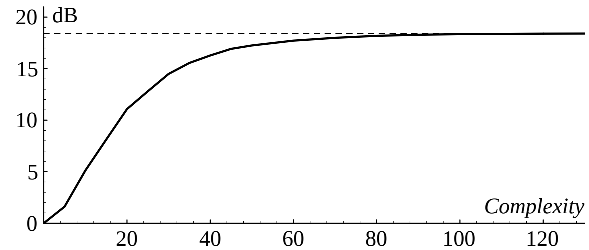

**Fig. 2.** The performance of RND, if it is interrupted after a fixed number of distortion computations. Parametric *SNR* versus this bound is shown, for an LSP VQ with $k = 5$ and $m = 12$.

applied to the one-stage VQ. Recall also that a multi-stage VQ is a suboptimal structure, which gives a higher distortion than the optimal structure that ND operates on.

Despite the shorter route, SND requires almost twice as many distortion measures as RND. FND displays practically no improvement at all over RND, so the extra memory demanded by FND cannot be justified in this case.

We should, however, not be too concerned with average values. In real-time speech coding, the time gained during one frame cannot freely be saved to another frame. The maxima are what matters, at least if we persist in that the true optimum must always be found. However, if we relax that requirement just a little, we can attain a considerable speed gain, at a moderate degradation of output quality. This is illustrated in figure 2, where a bound has been set on the number of distortion computations that are allowed for each input vector. For instance, if RND encoding is interrupted after 82 computations, which according to table 1 is the average number, the quantization error (in the parameter space) is only 0.22 dB above that of a FS. The main explanation is that so many computations are used just to discover that the last hypothesis is optimal. Actually, of the distortions computed by an uninterrupted RND for this VQ, this "wasteful" verification requires as much as 56 %.

The main drawback with the ND concept, beside the burdensome precomputations, is the storage of the adjacency table. For a five-dimensional application, such as this LSP VQ, the table requires a few times more memory than the output vectors themselves.

## References


[1] A. Gersho and R. M. Gray, *Vector quantization and signal compression*. Boston, MA: Kluwer, 1992.

[2] B.-H. Juang and A. H. Gray, Jr., "Multiple stage vector quantization for speech coding," in *Proc. IEEE ICASSP*, vol. 1, pp. 597–600, Paris, France, May 1982.

[3] K. K. Paliwal and B. S. Atal, "Efficient vector quantization of LPC parameters at 24 bits/frame," in *Proc. IEEE ICASSP*, vol. 1, pp. 661–664, Toronto, ON, Canada, May 1991.

[4] V. Ramasubramanian and K. K. Paliwal, "Fast *K*-dimensional tree algorithms for nearest neighbor search with application to vector quantization encoding," *IEEE Trans. Signal Processing*, vol. 40, no. 3, pp. 518–531, Mar. 1992.

[5] E. Vidal, H. Rulot, F. Casacuberta, and J. Benedí, "Searching for nearest neighbours in constant average time with applications to discrete utterance speech recognition," in *Proc. Int. Conf. Pattern Recognition*, pp. 808–810, Paris, France, Oct. 1986.

[6] A. Okabe, B. Boots, and K. Sugihara, *Spatial tessellations: Concepts and applications of Voronoi diagrams*. Chichester, England, U.K.: Wiley, 1992.

[7] E. Agrell, "A method for examining vector quantizer structures," in *Proc. Int. Symp. Information Theory*, p. 394, San Antonio, TX, Jan. 1993.

[8] F. P. Preparata and M. I. Shamos, *Computational geometry: An introduction*. New York, NY: Springer-Verlag, 1985.

[9] R. Hagen, "On spectral coding in CELP speech coders," Technical Report No. 141L, School of Electrical and Computer Eng., Chalmers Univ. of Technology, Göteborg, Sweden, Nov. 1992.