Enhancing the Performance of Vector Quantizers for Data Compression and Pattern Matching using Sorted Codebooks

John Leis

leis@usq.edu.au

http://www.usq.edu.au/users/leis/

March 2, 2004

Faculty of Engineering & Surveying Technical Reports
ISSN 1446-1846

Report TR-2004-01
ISBN 1 877078 07 7

Faculty of Engineering & Surveying
University of Southern Queensland
Toowoomba Qld 4350 Australia
http://www.usq.edu.au/

Abstract

Vector quantization (VQ) has found application in very low-rate speech and image encoding, together with content-based multimedia retrieval. In order to reduce the average distortion, large codebooks are required to store sufficient representative source vectors. The fundamental drawback with large codebooks is the search time required at the encoder. In order to reduce this search time, several methods have been proposed in the literature. This note describes an improvement on the so-called triangle-inequality based search. Experimental results are presented to demonstrate the usefulness of the method, which is able to reduce search times to as little as 10-20% of the full-search equivalent method under certain circumstances.

1 Background

Vector quantization (VQ) is a technique for matching a vector (or matrix) of data to the nearest match in a precomputed codebook of representative data [1]. It finds application in low-rate encoding for speech and images, and may be used in pattern recognition applications for searching a template database for the nearest match to a given feature vector. One of the problems with VQ is its very high computational complexity. Since the training phase is usually done separately, the complexity problem is not of particular concern when designing the codebook. However, when encoding vectors, the encoding time can be a significant problem, especially for real-time encoders.

Several computationally efficient VQ search algorithms have been proposed in the past. The computational reduction of these is somewhat dependent on the nature of the source distribution, and usually involves a tradeoff between search time and memory requirements.

Due to the nature of the search method, several simplifications may be made to the search process. More advanced methods, which are able to give a more significant reduction in the search time, utilize precomputed parameter tables.

2 Vector Quantization

In a vector quantizer, a codebook $\mathbb{C}$ of size $N \times k$ maps the k-dimensional space $\mathcal{R}^k$ onto the reproduction vectors (also called *codevectors* or *codewords*):

$$Q\colon \mathcal{R}^k \rightarrow \mathbb{C} \quad : \quad \mathbb{C} \triangleq (\mathbf{c}_1, \mathbf{c}_2, \cdots, \mathbf{c}_N,)^T \quad , \quad \mathbf{c}_i \in \mathcal{R}^k \tag{1}$$

The codebook consists of the codevectors $\mathbf{c}_i : \ i = 0, \ldots, N-1$. The codebook vectors are selected through a clustering or training process, involving representative source training data.

Encoding or pattern matching for a block of k samples in vector $\mathbf{x}$ consists of searching the codebook for an entry $\mathbf{c}_i$ that is the closest approximation for the current input vector $\mathbf{x}$. The encoder minimizes the distortion $d(\cdot)$ to give the optimal estimated vector $\hat{\mathbf{x}}$:

$$\hat{\mathbf{x}} \quad = \quad \arg\min_{\mathbf{c}_i \in \mathbb{C}} d(\mathbf{x}, \mathbf{c}_i) \tag{2}$$

The index i thus derived constitutes the best (minimum error) VQ representation $\mathbf{y} = \mathbf{c}_i$ of the input vector $\mathbf{x}$. The usual choice for the distance minimization is the Euclidean distance metric:

$$d_2(\mathbf{x}, \mathbf{y}) \quad = \quad \|\mathbf{x}_i - \mathbf{y}\|^2 = \sum_{j=0}^{K-1} (x_j - y_j)^2 \tag{3}$$

where $\mathbf{x}$ is the source vector, $\mathbf{y} = \mathbf{c}_i$ is each codevector in turn, and $\mathbb{C} = \{\mathbf{c}_i\}_{i=0}^{N-1}$ represents the set of codebook vectors.

3 Search Methods

A direct calculation of the distance metric for each codevector in the codebook for a given source vector is generally termed an *exhaustive search*. A number of simplifications may be made to the VQ search algorithm to reduce the complexity. These fall into the categories of:

Partial-distance search techniques. This approach aims to eliminate certain codevectors from the search before the full distortion is calculated. The computational reduction attainable depends upon how early the codevector may be eliminated as a candidate in the current search.

Algebraic simplifications. This approach uses a mathematical simplification of the distortion metric to both precalculate certain constants and apply preliminary tests to remove certain codevectors from candidature.

Region of Interest (ROI) partitioning. By considering the search process in $\mathcal{R}^k$, a region of interest may be used to prune the search to a much smaller number of admissable candidate codevectors. This method is the most powerful of all, but requires a very large memory space.

The exact reduction attainable is somewhat dependent upon the source characteristics. The partial-search and algebraic techniques require at least one scalar comparison to be performed on each vector of the codebook, and hence the upper bound on the reduction in complexity is limited by the codebook size. The ROI methods eliminate most codevectors in the codebook without the need for a vector comparison, and thus form an extremely powerful approach, with the potential for substantial performance improvements.

3.1 Partial Distance Search

For K components, the Euclidean metric requires the accumulation of K partial distortion values, after each successive term is added into the total distortion. The idea of the partial distance search is to terminate the search if the accumulated partial distortion exceeds the minimum distortion so far in the codebook search. This approach requires an additional test within the loop, and may or may not result in any net savings. Processors which heavily utilize caching and branch prediction are less likely to be able to effectively utilize a partial distance search, since the overhead of branch testing and pipeline misses must be traded off against fewer floating-point computations.

Representative work in this area includes [2], [3], [4], and [5]. Note that the title of [3] implies a similar technique to that presented here, whereas it in fact presents a method for ordering the components *within* each codevector, rather than ordering the codebook itself.

3.2 Algebraic Simplification

Several algebraic simplification methods have been reported in the literature, such as [6], [7], [8], [9], and [10]. The method of [7] has been used in the experimental results presented here for the purpose of comparison.

Let $\mathbf{x}$ be the input vector of dimension K, x_k be component k of $\mathbf{x}$, and $\overline{x}$ the mean of vector $\mathbf{x}$. Defining

$$d'(\mathbf{x}, \mathbf{y}_i) = K(\overline{x} - \overline{y}_i)^2 \tag{4}$$

it may be shown that

$$d(\mathbf{x}, \mathbf{y}_i) \geq d'(\mathbf{x}, \mathbf{y}_i) \tag{5}$$

This depends only on precomputed values, and is easily calculated for each codevector. The full distortion calculation is only performed for codevectors which satisfy

$$d'(\mathbf{x}, \mathbf{y}_i) \quad < \quad d_{min}(\mathbf{x}, \mathbf{y}_i) \tag{6}$$

where $d_{min}(\mathbf{x}, \mathbf{y}_i)$ is the minimum distortion found up until that point in the search.

3.3 Region-of-Interest Search

These methods utilize precomputed parameter tables, and hence increase the memory requirements. However, substantial reductions in computational requirements may be achieved. Methods which include partitioning the vector space include those in [10], [11], [12], [13] and [14], although our work is based on the distance and triangle-inequality methods of [15].

Given an input vector $\mathbf{x}$ and a starting point $\mathbf{c}_i$, Figure 1 shows that the best match in the codebook must satisfy

$$r_x - h_i \;\leq\; r_k \;\leq\; r_x + h_i \tag{7}$$

where h_i is the distance from $\mathbf{x}$ to $\mathbf{c}_i$. This may be visualized in two dimensions as a region bounding $\|\mathbf{x}\| \pm h_i$.

The values of r_i are stored alongside each codevector. From any current codevector $\mathbf{c}_i$, the value h_i is calculated. Only codevectors which fall inside the bounding area given by $r_x \pm h_i$ need be tested. This requires two scalar comparisons to determine if the codevector is a candidate. If the codevector is a possible candidate, a full distance search is performed.

The initial codevector $\mathbf{c}_i$ may be chosen at random, or as the codevector whose r_i is closest to r_x. At each stage, if a new codevector with lower distance is found, the current codevector $\mathbf{c}_i$ is replaced with that codevector.

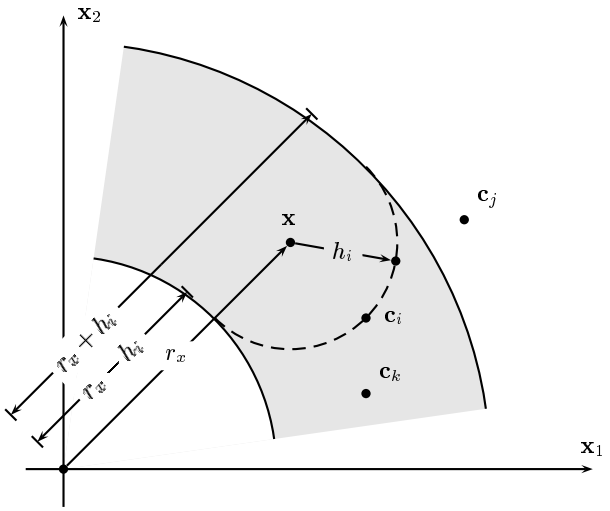

Figure 1: Search method based the distance from the origin. Given a target vector $\mathbf{x}$ and current best-match $\mathbf{c}_i$, only vectors within the radius $r_x \pm h_i$ (shaded) need be examined.

Another region-based method is based on the triangle inequality, which for a test vector $\mathbf{x}$ and codebook vectors $\mathbf{c}_i$ and $\mathbf{c}_j$ states that

$$d(\mathbf{c}_i, \mathbf{c}_j) \quad \leq \quad d(\mathbf{x}, \mathbf{c}_i) + d(\mathbf{x}, \mathbf{c}_j) \tag{8}$$

This is illustrated in Figure 2. The search proceeds by calculating h_i for the current candidate codevector $\mathbf{c}_i$. Only codevectors which have $d(\mathbf{c}_i, \mathbf{c}_k) \leq 2\,h_i$ are tested further, since only these could possibly yield less distortion. In the illustration, codevector $\mathbf{c}_k$ would be tested, but codevector $\mathbf{c}_j$ would not be tested. The key to the method is a precomputed table of all the distances $d(\mathbf{c}_i, \mathbf{c}_j)$ in the codebook $\mathbb{C}$, requiring $N(N-1)/2$ entries. A good choice of the initial starting vector $\mathbf{c}_i$ will minimize the size of the search hypersphere.

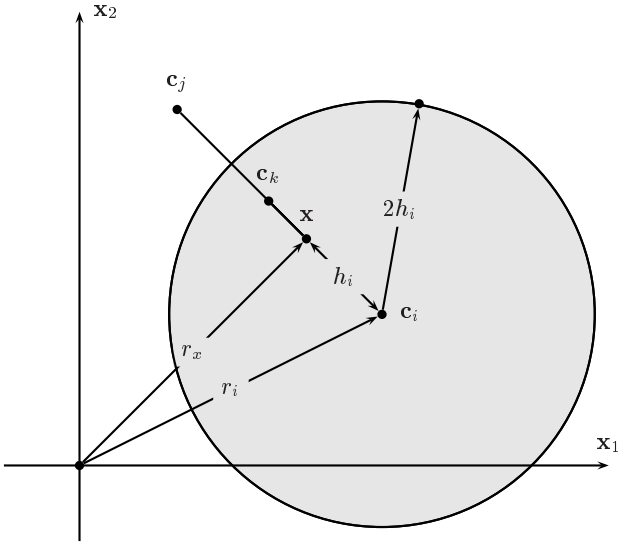

Figure 2: Search method based on the triangle-inequality. Given a target vector $\mathbf{x}$ and current best-matching vector $\mathbf{c}_i$, only codevectors within a radius $2h_i$ of $\mathbf{c}_i$ (shaded disk) need be examined.

4 Improvements to the Triangle-Inequality Method

The triangle inequality, when applied to fast search methods in many dimensions, effectively produces a shrinking hypersphere, bounded by the distance between the candidate codevector and the most recently found minimum-distance codevector. The aim is to shrink the hypersphere as rapidly as possible, and thereby exclude the largest possible number of codevectors in the shortest possible time.

Reducing the search space as rapidly as possible may be achieved by reordering the distance table as follows. For each codevector i, we sort each entry in the corresponding row of the distance table $D(i,k)$, $k \in N$, in increasing order of distance from the codevector with index i. The search space for each candidate codevector is then guaranteed to reduce at the fastest possible rate for a given codebook, since the closest codevectors are examined first. This re-ordering requires a permutation table to keep track of the original codebook indexes corresponding to each row of the sorted distance table. However, the search time is substantially reduced, as will be shown.

Several factors impact on the exact amount of computational reduction possible. These are primarily:

1. The vector dimension k.
2. The codebook size N.
3. The statistical distribution of the codevectors, and any intra-vector and inter-vector correlation.

In order to evaluate the effectiveness of the proposed method, two sets of experiments were conducted. In the first experiment, the codebook was populated with randomly generated test vectors; the second experiment utilized real data from test images, applied to a product-code VQ.

The results from the first experiment, utilizing codebooks assembled from independent, random variates, are shown in Table 1. Firstly, it is clear that the triangle-inequality based methods outperform algebraic and magnitude searches, by a considerable margin. For a 1024-element codebook, the triangle method was able to reduce the average number of floating-point computations to about 20% of the original. Application of the magnitude-sorting permutation to the distance table was further able to reduce the number of computations, down to about 10% of the exhaustive search.

Codebook	Search Method			
Size	r-search	$\triangle$-search	sorted-$\triangle$	algebraic
1024	0.90	0.20	0.10	0.99
2048	0.86	0.15	0.09	1.00

Codebook	Search Method			
Size	r-search	$\triangle$-search	sorted-$\triangle$	algebraic
1024	1.06	0.79	0.66	1.05
2048	1.06	0.68	0.57	1.03

Table 1: Results for 8-dimensional (upper) and 16-dimensional (lower) codevectors. The search methods are radius-search (r-search), triangle inequality ($\triangle$-search), sorted triangle inequality (sorted-$\triangle$) and algebraic simplification based on Poggi's derivation [7]. The results are normalized, so that 1.0 corresponds to the complexity of a full (exhaustive) search for a codebook of the same dimension.

Next, an image encoding problem utilizing VQ was examined. A product-code image VQ was developed, similar to that reported elsewhere [1]. The mean/shape/gain image encoding VQ transmits the mean and gain of an image block as scalars, with the shape of the image block encoded using a vector index. Five standard images were used to generate the codebook – "bridge", "camera", "goldhill" and "lena". Testing was carried out using the "bird" image. The images were blocked into 4×4 sub-blocks, the mean removed, and the gain factored out.

Table 2 illustrates the advantage of the proposed method. For a large codebook (1024 codevectors), the number of operations required for a triangular-search — approximately 58% that of a full-search — was reduced to 48% when using the sorted-permutation distance matrix as proposed here. For a smaller codebook (256 codevectors), the number of operations required for a triangular-search — approximately 74% that of a full-search — was reduced to 65% when using the sorted-permutation distance matrix method.

	Search Method			
Codebook	r-search	$\triangle$-search	sorted-$\triangle$	algebraic
1024×16	1.08	0.58	0.48	1.06
256×16	1.08	0.74	0.65	1.06

Table 2: Results for mean/shape/gain image VQ. The image shape vectors are 4×4, codebook sizes 256 and 1024. The search methods are radius-search (r-search), triangle inequality ($\triangle$-search), sorted triangle inequality (sorted-$\triangle$), and algebraic simplification.The results are normalized, so that 1.0 corresponds to 50176 flops for the larger codebook, and 12544 for the smaller codebook.

5 Conclusions

This note has presented an improvement to the triangular-inequality fast search method for searching vector codebooks. Some additional memory over that required for the distance table is needed, in order to store the sorted permutations for each codevector distance. This simple enhancement to the previously-published algorithm was demonstrated to be able to reduce the search/encoding times somewhat, depending on the codebook size and the distribution of the source vectors.

References

[1] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*, Kluwer Academic Publishers, 1992.

[2] C-D. Bei and R. M. Gray, "An Improvement of the Minimum Distortion Encoding Algorithm for Vector Quantization", *IEEE Transactions on Communications*, vol. COM-33, no. 10, pp. 1132–1133, Oct. 1985.

[3] K. K. Paliwal and V. Ramasubramanian, "Effect of Ordering the Codebook on the Efficiency of the Partial Distance Search Algorithm for Vector Quantization", *IEEE Transactions on Communications*, vol. 37, no. 5, pp. 538–540, May 1989.

[4] James McNames, "Rotated Partial Distance Search for Faster Vector Quantization Encoding", *IEEE Signal Processing Letters*, vol. 7, no. 9, pp. 244–246, Sept. 2000, also available `http://ece.pdx.edu/~mcnames/Publications/RPDS.pdf`

[5] S. C. Tai, C. C. Lai, and Y. C. Lin, "Two Fast Nearest Neighbour Searching Algorithms for Image Vector Quantization", *IEEE Transactions on Communications*, vol. 44, no. 12, pp. 1623–1628, Dec. 1996.

[6] C.-H. Lee and L.-H. Chen, "Fast Closest Codeword Search Algorithm for Vector Quantization", *IEE Proceedings*, vol. 141, no. 3, pp. 143–148, June 1994.

[7] G. Poggi, "Fast Algorithm for Full-Search VQ Encoding", *Electronics Letters*, vol. 29, no. 12, pp. 1141–1142, June 1993, also available `http://diesun.die.unina.it/GruppoTLC/poggi/R03.pdf`

[8] K.-L. Chung, W.-M., and Yan J.-G. Wu, "A Simple Improved Full Search for Vector Quantization Based on Winograd's Identity", *IEEE Signal Processing Letters*, vol. 7, no. 12, pp. 342–344, Dec. 2000, also available `http://ece.pdx.edu/~mcnames/Publications/RPDS.pdf`

[9] L. Torres and J. Huguet, "An Improvement on Codebook Search for Vector Quantization", *IEEE Transactions on Communications*, vol. 42, no. 2/3/4, pp. 656–659, Feb. 1994.

[10] S. Baek, B. Jeon, and K-M. Sung, "A Fast Encoding Algorithm for Vector Quantization", *IEEE Signal Processing Letters*, vol. 4, no. 12, pp. 325–327, Dec. 1997.

[11] V. Ramasubramanian and K. K. Paliwal, "Voronoi Projection-Based Fast Nearest-Neighbour Search Algorithms: Box-Search and Mapping Table-Based Search Techniques", *Digital Signal Processing*, vol. 7, no. 4, pp. 260–277, Oct. 1997.

[12] J. Mielikainen, "A Novel Full-Search Vector Quantization Algorithm Based on the Law of Cosines", *IEEE Signal Processing Letters*, vol. 9, no. 6, pp. 175–176, June 2002.

[13] M. R. Soleymani and S. D. Morgera, "A Fast MMSE Encoding Technique for Vector Quantization", *IEEE Transactions on Communications*, vol. 37, no. 6, pp. 656–659, June 1989.

[14] M. R. Soleymani and S. D. Morgera, "An Efficient Nearest Neighbour Search Method", *IEEE Transactions on Communications*, vol. COM-35, no. 6, pp. 677–679, June 1987.

[15] C-M. Huang, Q. Bi, G. S. Stiles, and R. W. Harris, "Fast Full Search Equivalent Encoding Algorithms for Image Compression Using Vector Quantization", *IEEE Transactions on Image Processing*, vol. 1, no. 3, pp. 413–416, July 1992.