Journal of Applied Statistics

ISSN: 0266-4763 (Print) 1360-0532 (Online) Journal homepage: https://www.tandfonline.com/loi/cjas20

Vector quantization: clustering and classification trees

Pamela C. Cosman , Robert M. Gray & Richard A. Olshen

To cite this article: Pamela C. Cosman , Robert M. Gray & Richard A. Olshen (1994) Vector quantization: clustering and classification trees, Journal of Applied Statistics, 21:1-2, 93-108, DOI: 10.1080/757582970

To link to this article: https://doi.org/10.1080/757582970

CHAPTER 6

Vector quantization: clustering and classification trees

PAMELA C. COSMAN[1], ROBERT M. GRAY[1] & RICHARD A. OLSHEN[2],
[1]*Department of Electrical Engineering, Stanford University, and* [2]*Department of Health Research and Policy, Stanford University School of Medicine*

SUMMARY *An image that is mapped into a bit stream suitable for communication over or storage in a digital medium is said to have been compressed. Using tree-structured vector quantizers (TSVQs) is an approach to image compression in which clustering algorithms are combined with ideas from tree-structured classification to provide code books that can be searched quickly and simply. The overall goal is to optimize the quality of the compressed image subject to a constraint on the communication or storage capacity, i.e. on the allowed bit rate. General goals of image compression and vector quantization are summarized in this paper. There is discussion of methods for code book design, particularly the generalized Lloyd algorithm for clustering, and methods for splitting and pruning that have been extended from the design of classification trees to TSVQs. The resulting codes, called pruned TSVQs, are of variable rate, and yield lower distortion than fixed-rate, full-search vector quantizers for a given average bit rate. They have simple encoders and a natural successive approximation (progressive) property. Applications of pruned TSVQs are discussed, particularly compressing computerized tomography images. In this work, the key issue is not merely the subjective attractiveness of the compressed image but rather whether the diagnostic accuracy is adversely affected by compression. In recent work, TSVQs have been combined with other types of image processing, including segmentation and enhancement. The relationship between vector quantizer performance and the size of the training sequence used to design the code and other asymptotic properties of the codes are discussed.*

1 Introduction

Image compression maps an original image into a bit stream suitable for communication over or storage in a digital medium. The number of bits required to represent

the coded image should be smaller than that required for the original image, so that one can use less storage space or communication time. There are two basic types of compression. 'Lossless' compression—also called noiseless coding, data compaction, entropy coding or invertible coding—refers to algorithms which allow the original image to be perfectly recovered from the digital representation. Lossless compression is only applicable for already-digital images. For example, if the original image is an analog photograph, then it is impossible to recreate it exactly from a digital representation, regardless of how many bits are used. Lossless compression requires variable-rate coding techniques: a varying number of bits are needed for different pixels or pixel patterns. The basic idea of such codes is to use long code words for unlikely inputs and short code words for likely inputs. The codes are explicitly designed so that the average number of bits per input pixel is as small as possible. The most popular lossless coding techniques are Huffman, adaptive Huffman, run-length, Ziv-Lempel and arithmetic codes. Typical compression ratios for lossless codes on still-frame eight-bit grey scale images run from expansion (poor) to 4:1 compression (unusually good).

'Lossy' compression algorithms do not allow the original pixel intensities to be perfectly recovered from the compressed representation. Simple quantization or analog/digital (A/D) conversion (the mapping of a number from a continuous range of possible values into a finite set of approximating values) are examples of lossy compression. A possible goal is to minimize an average distortion, such as the average squared error for a given bit rate. Typical compression ratios range from 4:1 to 32:1 for still-frame eight-bit grey scale images using common techniques, and much better compression is reported for some more recent techniques. Lossy compression systems can be clustered into several basic overlapping types, including scalar quantization (PCM) with entropy coding, predictive coding with scalar quantizers (predictive DPCM), transform and subband coding (including the popular discrete cosine transform), 'second generation' codes based on image segmentation and contour extraction, multi-resolution codes, such as those based on Bert-Adelson pyramids and wavelet decomposition, and vector quantization.

Since lossy compression algorithms do not reproduce an image exactly, the issue of the quality of a compressed image becomes paramount. As one might expect, using enough bits in a digital representation should result in a coded image that is perceptually indistinguishable from the original. 'Enough bits', however, can be too many to fit into the available storage or to communicate over an available link in reasonable time. Furthermore, the term 'perceptually indistinguishable' may depend on the viewer. An untrained eye might detect no difference where an expert's eye perceives flaws. Furthermore, one image may be aesthetically less attractive than another, though at least as useful for the task for which the image was taken. Thus, the definition of quality is strongly dependent on the application.

The development of compression systems optimized for a particular application has proceeded along a variety of parallel paths. Early systems were based on engineering intuition, suggesting that predicting or transforming a signal would 'remove redundancy' and lead to more efficient simple quantization on the resulting linearly transformed signal components. High rate or asymptotic quantization theory provided an approximate theory for analyzing such systems when the bit rate was high. Shannon showed how digital information, such as quantized images (or inherently digital images), could be compressed in an invertible fashion, provided enough bits were available. Initially, the 'lossy' compression techniques typified by quantization were combined with the 'lossless' techniques introduced by Shannon

by simply cascading them to provide an overall compression system. Such a system typically consisted of linear operations, such as transforms or prediction, followed by scalar quantization of the transformed or predicted data using several quantizers with differing bit rates, followed by an invertible code. This basic approach has worked well and still dominates in practice.

Along with invertible coding, Shannon also introduced what he called 'source coding with a fidelity criterion'. This referred to the mapping of blocks or vectors produced by a signal into binary code words, so as to form a coded representation that was optimum in the sense of minimizing the average of some mathematically well-defined distortion measure between the original signal and the digital reproduction. Such coding schemes for vectors have become known as block quantizers or vector quantizers. The Shannon theory and geometric arguments indicate that one can achieve better performance by operating directly on vectors rather than using scalar operations on linearly transformed vectors; however, the classical theory does not provide design techniques for codes.

One approach to code design is to use clustering algorithms to form code books combined with tree-structured classification algorithms to provide low complexity code book searches with useful structure. The overall goal is to optimize the quality subject to a constraint on the communication or storage capacity (the allowed bit rate). Some of the basic ideas for such vector quantizer (VQ) design algorithms are sketched here.

Details of the fundamentals can be found in Gersho and Gray (1992). This provides the basic theory of vector quantization, along with a wide variety of code design algorithms and applications. Extensive citations to the literature are provided, especially for speech compression and image compression. Also, see Breiman *et al.* (1984) for details relating to some aspects of the algorithms. This includes algorithms, applications and mathematics related to CART (classification and regression trees) algorithms for binary tree-structured recursive partitioning approaches to classification, probability class estimation and regression; algorithms for pruning CART trees are closely related to those of the lossy data compression algorithms termed pruned, tree-structured vector quantizers (PTSVQs).

Riskin *et al.* (1990) described the first application of pruned tree-structured VQs to the lossy compression of medical images. Cosman *et al.* (1991a) in turn described an early example of how compression can be combined with enhancement by using dual code books for reconstruction. Here code books were subjected to off-line histogram equalization, a technique for improving the dynamic range of images. The user can then see the unprocessed, reconstructed image, or an equalized reconstruction, by simply selecting the appropriate code book, without any additional computation.

Cosman *et al.* (1991b) showed how input-weighted quadratic distortion measures are used to assign larger distortion to visually important inputs, thereby improving the perceived quality of minimum average distortion code books. Here, the emphasis was on weightings that depended on intensity and texture. Similarly, Oehler *et al.* (1991) considered input-weighted distortion measures; however, here, the weighting was determined by the classification error to force the quantizer to classify implicitly and then highlight man-made objects.

Gray *et al.* (1992) gave an expanded survey of the topics treated briefly here. Also, Gray *et al.* (1993) surveys methods for incorporating visual factors into the design of VQs, with an emphasis on the use of input-weighted distortion measures such as those considered by Cosman *et al.* (1991b) and Oehler *et al.* (1991).

Cosman *et al.* (1991c) reported on the performance of VQs as a function of learning sequence size for both tree-structured and full-search VQs. The results suggest that distortion decreases algebraically to an asymptote (as a function of the number of training images that are used in generating the code book).

2 Vector quantization

Our VQs for image data parse the image into blocks of typically 2×2 or 4×4 sub-blocks of pixels. The encoder views an input vector $\mathbf{X}_n$ at time n and produces a channel code word i, which is a binary R-tuple if the code has R bits per vector and the system is of fixed rate. We will consider later the situation where the indices i can be of variable dimension. The decoder is a table look-up: on receiving a channel code word i, the decoder puts out a stored code word or template $\hat{\mathbf{Y}}_i$, i.e. a word in memory indexed by the channel code word. The decoder is completely described by a code book containing all 2^R of the possible code words. The basic Shannon source code model provides a means of operation that is optimal for a given code book if the goal is to minimize an average distortion. If we assume $d(\mathbf{X}, \hat{\mathbf{X}}) \geqslant 0$ measures the distortion or cost of reproducing an input vector $\mathbf{X}$ as a reproduction $\hat{\mathbf{X}}$, and if we further assume that the overall distortion (or lack of fidelity) of the system is measured by an average distortion, then the optimal encoder for a given code book selects the vector $\mathbf{Y}_i$ if

$$d(\mathbf{X}_n, \mathbf{Y}_i) \leqslant d(\mathbf{X}_n, \mathbf{Y}_j), \qquad \forall j$$

In other words, the encoder operates in a nearest neighbor or minimum distortion fashion. For the moment, the code is not assumed to have any structure. A VQ for image compression is depicted in Fig. 1, with the vectors being pictured as 2×2 square pixel blocks.

If the code book of reproductions is fixed for all input vectors, then the vector quantization operates in a memoryless fashion on each input vector; i.e. each vector is encoded independently of previous inputs and outputs of the encoder. In general, an encoder can have memory by varying the code book according to past actions. Predictive and finite-state VQs are examples of vector quantization with memory.

The choice of distortion measure permits us to quantify the performance of a VQ in a manner that can be computed, used in analysis and used in design optimization. The theory focuses on average distortion in the sense of a probabilistic average or expectation. Practice emphasizes the time average or sample average distortion

$$D = \frac{1}{L} \sum_{n=1}^{L} d(\mathbf{X}_n, \hat{\mathbf{X}}_n)$$

for large L. This is frequently normalized and a logarithm taken to form a signal-to-noise ratio (SNR) or peak signal-to-noise ratio (PSNR). With well-defined stationary ergodic random process models instead of real signals, the sample average and expectation are effectively equal.

Here, we restrict our interest to the class of input-weighted squared-error distortion measures of the form

$$d(\mathbf{X}, \hat{\mathbf{X}}) = (\mathbf{X} - \hat{\mathbf{X}})^* \mathbf{B}_X (\mathbf{X} - \hat{\mathbf{X}}) \tag{1}$$

where $\mathbf{B}_X$ is a positive-definite symmetric matrix. Weightings alternative to the identity matrix often yield perceptually superior reproductions, since the weighting allows one to count the distortion weighted according to the behaviour of the input

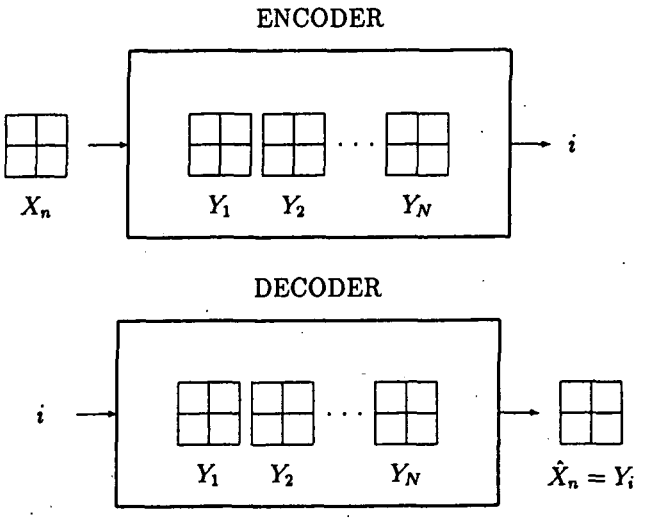

FIG. 1. Vector quantizer.

vector. For example, $\mathbf{B}_X$ can be used to adjust the distortion with the input $\mathbf{X}$ being classified in some fashion (e.g. active, textured, bright, dim, important, unimportant (Cosman *et al.*, 1991; Oehler *et al.*, 1991; Gray *et al.*, forthcoming). All the basic vector quantization design methods work for such input-weighted quadratic distortion measures.

Although a tractable distortion measure is needed for optimizing the design, other explicit or implicit measures of distortion may be required to validate the worth of a compression system. For example, in medical imaging, it may be far more important to verify that diagnostic accuracy remains as good or better with compressed images than it is to compare SNRs (Cosman *et al.*, 1993b, 1994).

3 Code design

There are many approaches to VQ design, including linear transformation of the input vector followed by scalar quantization; random code book population; lattice codes and lattice-based codes; clustering techniques, such as Lloyd, Forgey, Isodata, k-means and pairwise nearest neighbor algorithms; simulated annealing; deterministic annealing; and stochastic relaxation techniques.

Here, we emphasize the generalized Lloyd algorith (GLA), which is a simple clustering technique that is the vectorial extension of Lloyd's (1957) algorithm for finding optimal scalar quantizers based on a distribution. The idea is essentially the same as the later k-means and basic Isodata algorithms. One iteratively improves a code book $\mathscr{C} = \{\hat{\mathbf{X}}_i; i = 1, \ldots, N\}$ of vectors by alternately optimizing the encoder for the decoder and vice versa. Given a decoder or code book $\mathscr{C}$, the optimum encoder $\mathscr{E}$ is a minimum distortion or nearest neighbor mapping: $\mathscr{E}(\mathbf{X}) = i$ if

$$d(\mathbf{X}, \hat{\mathbf{X}}_i) \leqslant d(\mathbf{X}, \hat{\mathbf{X}}_j), \qquad \forall j \neq i$$

Given an encoder mapping $\mathscr{E}$ mapping vector space into an index set $\{0, 1, \ldots, N\}$, one can optimize the code book $\mathscr{C}$ (decoder) for the encoder by choosing the code words to be the generalized centroids of all inputs mapping into a given index:

$$\hat{\mathbf{X}}_i = \min_y{}^{-1} E(d(\mathbf{X}, y) | \mathscr{E}(\mathbf{X}) = i)$$

For the input-weighted squared-error distortion measure, we have

$$\hat{\mathbf{X}}_i = E[\mathbf{B}_X | \mathscr{E}(\mathbf{X}) = i]^{-1} E[\mathbf{B}_X \mathbf{X} | \mathscr{E}(\mathbf{X}) = i]$$

where the expectation is with respect to a sample distribution, because code design is usually based on a training set of typical data, not on mathematical models of the data. For example, in applications to medical imaging, we have trained on several images of the modality and organ scanned for a particular application. This avoids the necessity of modeling the data and bases the design on an empirical distribution. The Lloyd iteration begins with an initial code book and iterates until convergence or some error threshold is achieved; a variety of techniques are available for designing the initial code book.

4 Tree-structured vector quantization

The Shannon theory states that VQs can perform arbitrarily close to the theoretical optimal performance for a given rate if the vectors have a sufficiently large dimension; unfortunately, however, the complexity of the codes grows exponentially with the vector dimension. The practical solution to this 'curse of dimensionality' is to constrain the structure of codes. A wide variety of constrained code structures have been proposed and used, as was surveyed by Gersho and Gray (1992). We focus here on one particular code structure that has a natural progressive or successive approximation form and provides a good balance between performance and complexity: tree-structured VQ (TSVQ).

The key idea in TSVQs is to perform a tree search on a code book designed for such a search, rather than perform a full search of an unstructured code book. The code word is selected by a sequence of binary decisions. The code book can be thought of as a tree where each node is labeled by a reproduction. The search begins at the root node, where the encoder compares the input vector with two possible candidate reproductions and picks the one with the minimum distortion. This is a full search of a simple binary code book. The encoder advances to the selected node and produces a binary symbol to represent its decision. For example, '0' indicates the left-hand child is selected and '1' indicates the right-hand child. If the node is not a terminal node or leaf of the tree, the encoder continues and chooses the best available node of the new pair presented, putting out another binary symbol, and so on. When the encoder reaches a leaf, the sequence of binary decisions (the binary code word) is then a path map through the tree to the terminal node, which is labeled by the final reproduction vector or code word. The tree can be balanced or unbalanced. In the balanced case, every binary sequence has the same length, resulting in a fixed-rate tree. The unbalanced case permits the use of more bits for describing complicated vectors and fewer for simple vectors.

Potentially a great advantage of the general code structure is that, if properly designed, each successive decision should further refine the reproduction. Such a code will have a built-in successive approximation or progressive character. As more bits describing a given vector arrive, the quality of the reproduction improves. This progressive structure also means that the rate can be adjusted according to the available communication capacity. If the available rate is cut, one codes less deeply into the tree, and vice versa. The same tree is used, regardless of the allowed rate, and the quality is as good as possible for the allowed rate. How is a tree-structured

code designed so as to have these properties? This is accomplished by combining clustering with ideas from classification and regression tree design.

4.1 Growing trees

The first problem in designing a tree is to use the training set to determine the binary splits of the data set into smaller and smaller pieces. The fundamental idea is to select each split of a subset so that the data in each of the descendant subsets are 'purer' than the data in the parent subset (Breiman *et al.*, 1984). If $p(j|t)$ is the proportion of the cases in node t that belong to class j, then one defines an 'impurity' measure $i(t)$ that is a non-negative function of $p(j|t)$. The impurity measure should have the property that it is largest when all classes are equally mixed together in the node, and is smallest when the node contains only one class. For any node t, let us suppose that there is a candidate split s of the node which divides it into t_L and t_R, such that a proportion p_L of the cases in t go to t_L and a proportion p_R go to t_R. Then the 'goodness' of the split is defined to be the decrease in impurity:

$$\Delta i(s,t) = i(t) - p_L i(t_L) - p_R i(t_R) \tag{2}$$

How is this paradigm useful for vector quantization? If $\mathcal{T}_t$ is the set of all training vectors $\mathbf{x}_k$ mapping into node t, then (by analogy with CART regression (Breiman *et al.*, 1984)), we can take our node impurity function to be the distortion measured at node t:

$$d(t) = \frac{1}{\| \mathcal{T}_t \|} \sum_{k:\mathbf{x}_k \in \mathcal{T}_t} d(\mathbf{X}_k, \hat{\mathbf{X}}_t) \tag{3}$$

where $\| \mathcal{T}_t \|$ is the number of vectors in $\mathcal{T}_t$. The goodness of a node split is defined as the decrease in node impurity, i.e.

$$\Delta d(s,t) = d(t) - p_L d(t_L) - p_R d(t_R) \tag{4}$$

where, in the vector quantization context, the binary test s is a nearest neighbor selection (hyper-plane test) designed by the GLA.

In growing a VQ tree, one begins with the root node, which can be considered to be labeled by the optimum rate '0' code word which is the centroid of the learning set or distribution. One splits the node into two new 'child' nodes. The split can involve simply perturbing the root node label slightly and using the root node label and its perturbation as the two new child labels; alternatively, it can be more clever, such as forming two new labels along the axis produced by a principal components analysis. One then runs the GLA (or some other clustering algorithm) on the pair to produce a good one-bit code book as level '1' of the tree.

There are now two quite different options. A standard approach is to grow a 'balanced' tree by splitting all the terminal nodes and clustering. This results in a fixed-rate code. For each current terminal node, all the learning set (or the conditional probability) which survives to that node will be used in the clustering algorithm to design a good one-bit code which forms the given node's children. When the clustering algorithm converges for all the split nodes, one will have a new tree with twice as many nodes. One can continue in this fashion until one obtains a very large tree, although the nodes can become quite sparse in training vectors.

A different option is to split nodes one at a time rather than an entire level at a time. This method is a natural extension of the fundamental design technique discussed above for classification and regression tree design, as exemplified in

CART algorithms (Breiman *et al.*, 1984). After the level 1 code book has converged, we choose one of the two nodes to split, and run a clustering algorithm on that node alone to obtain a new, unbalanced tree. We then repeat this, splitting one node at a time until the tree reaches the desired average rate. How do we choose which node to split? We choose the node that provides the best goodness of split, as defined in equation (4).

Given a tree $\mathbf{T}$, let $\tilde{\mathbf{T}}$ denote its leaves (or terminal nodes). Let us assume that we split $t \in \tilde{\mathbf{T}}$ into two new leaves t_L and t_R. Let D and R stand for the distortion and rate, respectively, measured by $\mathbf{T}$, and let D' and R' denote the distortion and rate of the tree after t is split. Let $\Delta D = D' - D$ and $\Delta R = R' - R$ be the change in the distortion and rate, respectively, resulting from splitting t, and let $l(t)$ be the depth of node t. Then, we have

$$D = \sum_{\substack{i \in \tilde{\mathbf{T}} \\ i \neq t}} p(i)d(i) + p(t)d(t) \tag{5}$$

$$R = \sum_{\substack{i \in \tilde{\mathbf{T}} \\ i \neq t}} p(i)l(i) + p(t)l(t) \tag{6}$$

$$D' = \sum_{\substack{i \in \tilde{\mathbf{T}} \\ i \neq t}} p(i)d(i) + p(t_L)d(t_L) + p(t_R)d(t_R) \tag{7}$$

$$R' = \sum_{\substack{i \in \tilde{\mathbf{T}} \\ i \neq t}} p(i)l(i) + p(t_L)l(t_L) + p(t_R)l(t_R) \tag{8}$$

and the ratio of the change in distortion to the change in rate resulting from splitting leaf t is

$$\lambda = -\frac{\Delta D}{\Delta R} = d(t) - p_L d(t_L) - p_R d(t_R) = \Delta d(s, t) \tag{9}$$

which is simply the goodness of split for leaf t. Therefore, we choose to split the node that maximizes the magnitude slope $|\Delta D/\Delta R|$, so obtaining the largest possible decrease in average distortion per increase in average bit rate. This is optimal in an incremental or greedy fashion, and it is simple and effective. As we can see, for vector quantization, the object is to trade off the average distortion and average rate as measured by the length of the binary paths through the tree.

4.2 Pruning trees

Whether balanced or unbalanced, the growing algorithm greedily optimizes for the current split only, not looking forward to the impact which the current split can have on future splits. Furthermore, even the unbalanced tree can result in sparsely populated or improbable nodes that cannot be fully trusted to typify the long-term behavior. A solution to both these problems is to take a grown tree and prune it back using a similar strategy. Pruning achieves some of the advantage of looking ahead without the enormous additional computation. Pruning by removing a node and all its descendants will reduce the average bit rate, but it will increase the average distortion. The idea is to minimize the increase in average distortion per decrease in bit rate, i.e. to minimize the magnitude slope $|\Delta D/\Delta R|$. Now, however, we can consider the effect of removing entire branches rather than individual nodes. This permits one to find the optimal subtrees of an initial tree, in the sense of providing

the best rate–distortion trade-off. The key property that makes such pruning work is that the optimal subtrees of decreasing rate are nested (Breiman *et al.*, 1984), i.e. the optimal TSVQs formed by pruning an initial tree form embedded codes. This means, in particular, that these codes indeed have the successive approximation character: the distortion decreases on average as the bit rate increases.

A TSVQ designed by growing and pruning is called a pruned TSVQ or PTSVQ. An unbalanced tree is well matched to variable-rate environments, such as storage or packet communications. PTSVQs tend to yield lower distortion than fixed-rate full-search VQs for a given average rate. Furthermore, as we have seen, it has a simple encoder (a sequence of binary decisions) and a natural successive approximation (progressive) property.

5 Diagnostic accuracy of compressed medical images

Variations on the basic vector quantization techniques have been applied to a variety of image compression examples, including still-frame photographs, medical images and video. Typical results show that high quality reproductions of eight-bit monochrome still-frame photographs can be achieved at 0.5 bpp and less for moderate complexity. However, the definition of 'high quality' is often dependent on the application, and we have been particularly interested in questions of the diagnostic accuracy of compressed medical images.

Recently, we conducted an experiment in which the diagnostic accuracy of predictive PTSVQs was assessed for CT chest scans with 11-bit originals. The radiologic tasks involved two important medical problems: detection of mediastinal adenopathy and of lung nodules (see Cosman *et al.* (1993a, b, 1994) for more details). In this experiment, each pixel intensity was predicted from three nearby values and the residuals were coded in 2×2 blocks. Each of three Stanford radiologists studied a total of 360 CT images, 180 of each problem type. Some 30 were originals that were not compressed; the other 150 were compressed at levels that ranged from 0.5 bpp to 2.75 bpp. The three judging sessions were weeks apart. Viewing was from hard-copy on a images lightbox in an environment designed to match that of clinical practice. The presentation of the images was randomized but the randomization was subject to many constraints in an effort to reduce learning effects. An example of an original lung image is shown in Fig. 2. It contains three lung nodules, according to the gold standard, and these are marked by black arrows. Figure 3 shows a compressed version of this image at 1.21 bpp, in which there is a slight degradation in image quality; however, all three judges were still able to locate all three tumors.

The analyses began by determining a gold standard, which was the consensus of the three judges on the original images. For 20% of the images, the judges could not arrive at a consensus, so those images were discarded from the study. After a gold standard was determined, we could calculate the sensitivity (the chance something is correctly detected, given that it is there) and predictive value positive (the chance something is correctly detected, given that it was detected). Determining the specificity (the chance something is not detected, given that it is not there) was regarded as inapplicable for our study, because it is not possible to say how many abnormalities are absent for these non-binary diagnostic tasks. Without a sensible way of assigning specificity values, it was not possible to use receiver-operating characteristic curves. Therefore, our principal analyses focused on spline fits to the sensitivity and predictive value positive data. An example is shown in Fig. 4, which

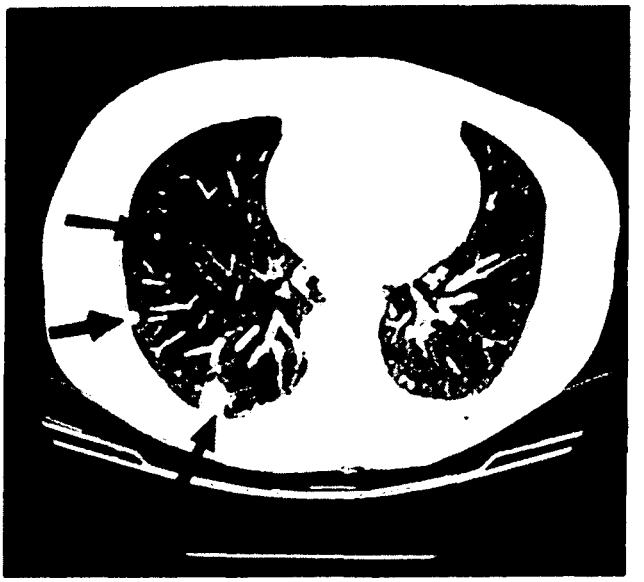

FIG. 2. Original lung image at 12 bpp. The black arrows mark the three tumors, as determined by the gold standard.

shows the sensitivity for the lung nodule detection task. The curve is a least-squares quadratic spline fit to the data with a single knot at 1.5 bpp (Powell, 1981), together with the two-sided 95% confidence regions. In view of the highly non-Gaussian nature of the data, Scheffé simultaneous confidence regions were obtained by a bootstrapping procedure, using an algorithm adapted from Miller (1981) and Efron (1982).

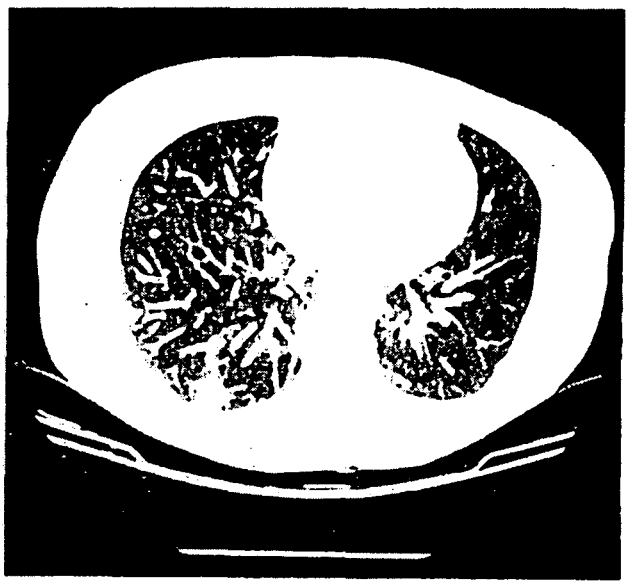

FIG. 3. Compressed lung image at 1.21 bpp.

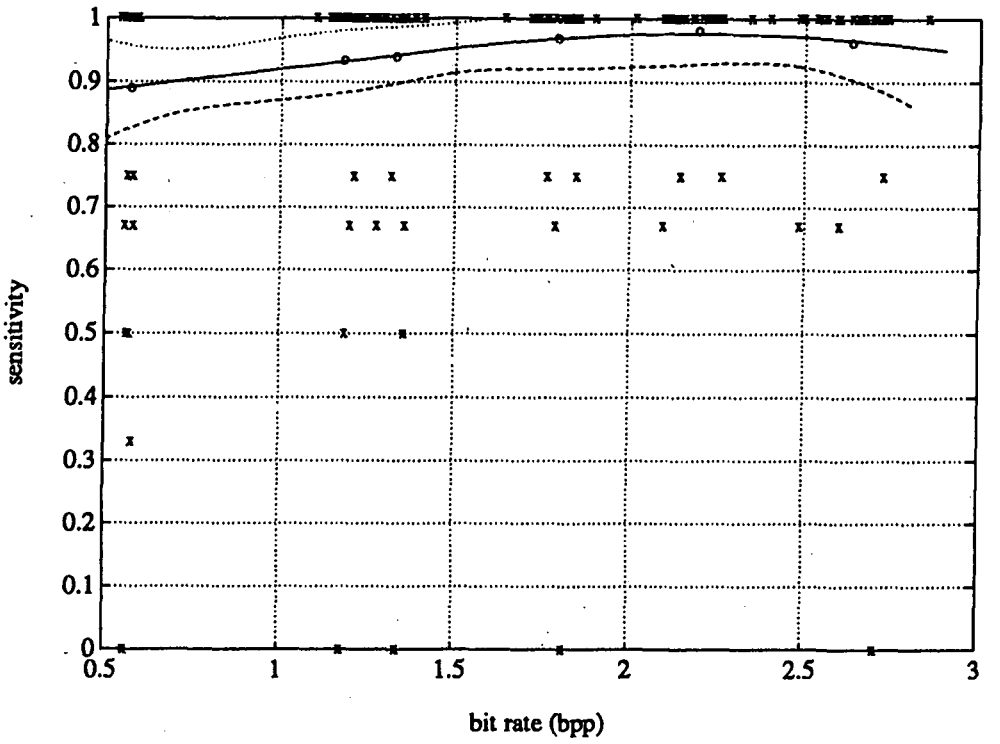

FIG. 4. Lung sensitivity: r.m.s. = 0.177.

From the graphs it appears that the sensitivity for the lung seems to be nearly as good at the low rates of compression (0.57 bpp) as at higher rates. At bit rates in excess of about 1.7, there is no perceptible difference in sensitivity for the lung tasks. The sensitivity for the mediastinum is roughly constant for all but the lowest bit rate, and that is driven by the results for one judge. The predictive value positive (PVP) for the lung is roughly constant across the bit rates, and the same is true for the mediastinum. The judges were of somewhat different opinions for the mediastinal images but not for the lung images. The images themselves were quite different in terms of difficulty for the lung, as was the case for two of the three judges for the mediastinum images.

We note that these conclusions are conservative, because the study is biased in favor of the original images. Since the consensus that determined the gold standard was clearly more likely to be attained for those original images where the judges were in perfect agreement initially, and thus where the original images would have perfect sensitivity and PVP relative to that gold standard, the original images have an advantage when compared with the other images. It is not far-fetched to suppose that, were the gold standard to be based on consensus on the most compressed images, a different subset of the 30 images would have been retained, and the most compressed images would perform well relative to the gold standard based on them.

In addition to the spline fits, the compression levels were also compared in a pairwise fashion, using the permutation distribution of McNemar's (1947) statistic. In this test, each image seen by a given judge at a given rate was paired with the same image seen by the same judge at another rate. For purposes of comparison, each image in the pair was assigned a value of 'perfect' (sensitivity = 1, PVP = 1) or

not perfect. Examining the data according to this formulation with all judges pooled revealed that the most compressed level (0.56 bpp) is unacceptable, because it differs from uncompressed images and other compression levels with highly significant p values. The next most compressed levels (1.18 and 1.33 bpp) are marginal, and a determination of the accuracy of these levels would require more testing with a larger number of images and a reformulation of the protocol so as to remove the bias favoring the originals. However, we have no difficulty concluding that the three least compressed levels (1.80, 2.20 and 2.64 bpp) are certainly acceptable, because no differences between these and the originals were found, despite this bias. We believe that, if our work is validated by subsequent studies, including some that we will undertake ourselves, then it can have serious implications for how digital radiologic data are archived and transmitted.

6 Vector quantization and other image processing

When considering segmentation it should be noted that many classes of images possess a strong degree of spatial stationarity, such that certain features of the image reliably appear in certain regions of the image. This spatial information can be used to improve compression. Perlmutter *et al.* (1992) used the CART regression tree algorithm to segment the images of the training sequence, using the x, y pixel location as a predictor for the intensity. For magnetic resonance brain scans, this segmentation tended to separate out the background areas from the medically significant areas such as the cortex and cerebellum. This segmentation could then be used to partition the training data by region, and generate separate code books for each region, or to allocate differing numbers of bits to the regions; both the SNR and the perceptual quality of the images were enhanced.

By changing decoder code books without any changes to the encoder, other signal processing can be performed off-line to improve the reproduced images. A survey of such algorithms was presented by Cosman *et al.* (1993a). For example, the decoder code book can have its terminal nodes undergo a grey scale transformation, such as histogram equalization (empirical probability integral transformation), using the histogram of the training sequences images. When the decoder uses these equalized leaves to decode a test image, the resulting reproduction has an improved dynamic range and contrast. This could also be accomplished by decoding using an ordinary TSVQ, and following that by equalization using the histogram of the test image itself. However, by equalizing the code book in advance, these two operations can be accomplished simultaneously by the decoder (Cosman *et al.*, 1991a; Gray *et al.*, 1992).

input-weighted distortion measures, the distortion can be counted more heavily for important classes of input vectors. Weighting according to intensity-, texture- and human-labeled important classes have been considered (Cosman *et al.*, 1991b; Oehler *et al.*, 1991; Gray *et al.*, 1993). This can take advantage of perceptual masking effects in which the human visual system is less sensitive to certain types of noise than others. Alternatively, it can take advantage of the fact that, in certain classes of images, some features are more important than others.

7 Training sequence size and VQ performance

An interesting question involves quantifying how large a training set is needed for a specific application, and an approach that borrows from non-parametric function estimation in statistics may be informative. (Cosman *et al.*, 1991c). Typically, one

has a set of predictors $\mathbf{X}$ and outcome $\mathbf{Y}$ being predicted. Let us write $f=f(\mathbf{X})$ for $E(\mathbf{Y}|\mathbf{X})$. We are given $\mathbf{X}$ but not $\mathbf{Y}$, so we might guess $\mathbf{Y}$ by f if we could learn what it is. To that end, we have a learning sample $\mathscr{L}=\{(\mathbf{X}_1,\mathbf{Y}_1),\ldots(\mathbf{X}_n,\mathbf{Y}_n)\}$, independent of $(\mathbf{X},\mathbf{Y})$, but with each pair $(\mathbf{X}_i,\mathbf{Y}_i)$ distributed as $(\mathbf{X},\mathbf{Y})$. The exact functional form of f is unknown, though it is presumed to lie in some function space $\mathscr{F}$. A variety of difficult results have been obtained. Many require that the learning sample be independent and are of the form

$$\inf_{f}\sup_{\mathscr{F}} E\|\hat{f}(\mathbf{X})-f(\mathbf{X})\|^2 \asymp Cn^{-r}, \qquad r>0 \tag{10}$$

where f is a (measurable) function of the learning sample and the norm indicates some notion of distance. All but the most recent results are inappropriate for our applications to imaging, since, for these results, the function space $\mathscr{F}$ was taken to be Sobolev-like, so consisting of inappropriately smooth functions. Furthermore, functions $\mathscr{F}$ that achieve the 'minimax' rates of convergence of equation (10) for some such $\mathscr{F}$ are not adaptive. This precludes algorithms such as CART from achieving minimax rates. More recent research—for example, the results of Donoho (1992) and Donoho and I. Johnstone (1992) which give examples of function spaces and statistical estimation problems for which adaptive estimators are minimax—suggest that, for some other $\mathscr{F}$ that includes certain discontinuous functions, non-linear adaptive procedures such as CART can be minimax; and rates of convergence will be of the form given by equation (10).

With PTSVQ, $\mathbf{X}=\mathbf{Y}$ is a vector of pixel intensities, and f has a bit rate constraint of the form $E(\text{depth } f)\leqslant D$; therefore, it will not be the case that $E\|\hat{f}(\mathbf{X})-f(\mathbf{X})\|^2\to 0$. Instead, that norm will tend to some asymptote A, which might be computed (in some cases) in theory from information-theoretic considerations. In addition, we do not know exactly the 'correct' $\mathscr{F}$ that would render PTSVQ minimax in any sense. Even so, we conjecture that, with suitable assumptions, there is a result for it of the form

$$E\|\hat{f}(\mathbf{X})-f(\mathbf{X})\|^2 = A+Bn^{-r}, \qquad \text{for some } r>0 \tag{11}$$

although the functional form may have a change-point that depends on n and D. An example of the empirical result is shown in Fig. 5, in which the average mean-squared error is plotted against training sequence size.

One interesting feature of this study involves its cross-validatory approach which we hope to see become standard practice for evaluating compression performance. Unlike most studies which use a large set of images for training and then report results on one or two test images, Cosman *et al.* (1991c) used the images repeatedly as training images in different combinations, and used them as test images whenever they were not part of the corresponding training sequence. Evidence for the conjectured algebraic form was shown by Cosman *et al.* (1991c) but much more work remains to be done before anything can be said with certainty.

8 Termination and continuity

While our knowledge of the rates of convergence of expected distortion rests on empirical evidence and conjecture, we do know that, for greedy, binary tree-structured algorithms like those we have described, their consistency in a strong sense has been established. The conclusions require that the learning sample of pixel vectors is stationary and ergodic. For a summary, see Nobel and Olshen (1993a, b).

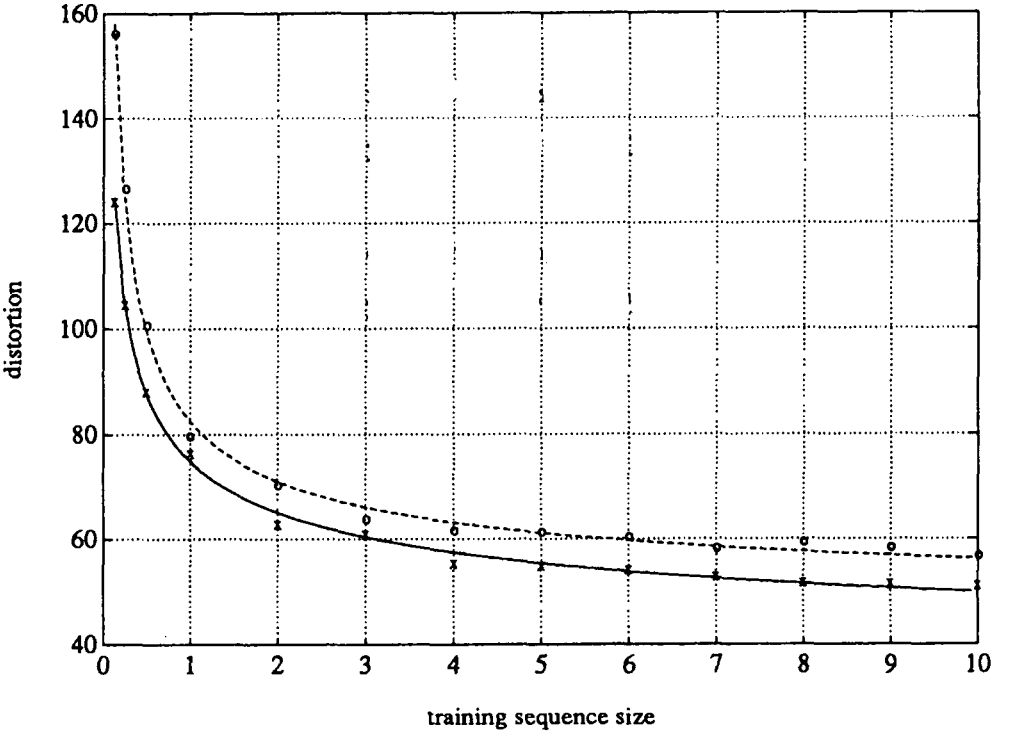

FIG. 5. Distortion versus training sequence size for 2 bpp TSVQ. The '×' symbols denote the case where the cross-validation scheme sampled entire images; the fitted curve has equation $y=31.1+43.7n^{-0.366}$. The 'O' symbols denote the case where the sampling units were the training vectors; the fitted curve is $y=45.3+37.2n^{-0.533}$.

To describe the conclusions, it is useful to denote the quantizer of a binary tree **T** by Q_T. Here $Q_T(\mathbf{X})$ is the centroid of the terminal node to which the 'test' pixel vector **X** belongs. F, the marginal distribution of **X** and the learning sample vectors, is assumed to be absolutely continuous, and to have bounded support. It is helpful to study the greedy growing algorithm first as it applies to F itself. Of course, there might be ties between and within nodes in the choice of the optimal split. We break them arbitrarily and recognize that there may be infinitely many possible trees **T** optimally grown on F. Let $\mathscr{A}(F, R)$ be the collection of binary trees produced by the greedy algorithm acting on F, with a bit rate of at most R. From the bit rate constraints, we know that each tree in $\mathscr{A}(F, R)$ has a bounded expected depth. Nobel and Olshen proved also that every such tree is itself finite with depth not more than $K=K(R,F)<\infty$; note that K does not depend on **T**. There are simple examples for which F lacks bounded support and $\mathscr{A}(F, R)$ has infinite trees.

Let $\hat{F}_n$ be the empirical distribution based on the learning sample $\mathscr{L}$ that has n-pixel vector members. Also, let $\mathscr{A}(\hat{F}_n, R)$ be the corresponding collection of possible trees produced by the greedy growing algorithm with bit rate constraint R. Fix $\varepsilon>0$. Then, for almost every learning sequence, if for every n $\mathbf{T}_n\in\mathscr{A}(\hat{F}_n, R)$, then there is a $\hat{\mathbf{T}}_n\in\mathscr{A}(F, R)$ for which $P\{\|Q_{T_n}(\mathbf{X})-Q_{\hat{T}_n}(\mathbf{X})\|>\varepsilon\}\to 0$. In other words, with probability tending to 1, the code word assigned to test pixel vector **X** by tree $\mathbf{T}_n\in\mathscr{A}(\hat{F}_n, R)$ will be within a pre-assigned distance of the code word assigned to **X** by some tree $\mathbf{T}_n\in\mathscr{A}(F, R)$.

9 Concluding remarks

We have provided a brief survey of the basic ideas behind vector quantization and their application to image compression. Compression can be considered as a form of classification, where the classes are not predetermined and the quality is measured by an average distortion, such as the squared error, rather than by the Bayes risk. From this viewpoint, it is natural that two approaches to statistical classification—clustering and classification trees—are well suited to the design of VQs. Together, clustering and classification trees provide a good balance between average distortion, bit rate and algorithmic complexity. The quality of any compression system must be judged by its usefulness in a particular application, something which SNRs and bit rates may not reveal. In the medical image application, we argued that the quality is measured by the degree to which diagnoses based on compressed images are consistent with those based on uncompressed images. Spline fits to sensitivity and PVP and statistical tests of difference were used to demonstrate such consistency at several levels of compression. The design of such validation experiments is itself an interesting statistical exercise, and it is necessary if lossy compressed images are to be accepted by the medical policy makers.

We closed with recent results on the convergence of code performance with the size of the learning set. This issue is of extreme practical importance, because only rules of thumb existed previously as a guide for selecting a training set size (or, if the size is fixed, as a guide for knowing how much confidence to have in the resulting code). These preliminary results show promise for quantifying these convergence rates for realistic code design techniques.

Vector quantization is not so much a single compression technique as a collection of tools based on clustering and classification that can be applied to traditional compression algorithms as well as towards new algorithms. By explicitly optimizing the distortion–rate trade-off within the constraint of a low complexity tree structure, vector quantization provides a conceptually simple method that can be easily simulated and used as a benchmark against other compression algorithms. Its incorporation of statistical methods makes it amenable to combination with other statistical signal processing techniques such as enhancement. Its experimentally demonstrated convergence properties have raised a variety of interesting theoretical questions that are only now beginning to yield results.

Acknowledgements

This work was supported by the National Institutes for Health under Grants CA49697-02 and 5 RO1 CA55325, and by the National Science Foundation under Grant DMS-9101528.

Correspondence: Robert N. Gray, Information Systems Laboratory, Department of Electrical Engineering, Stanford University, Stanford, CA 94305-4055, USA.

REFERENCES

Breiman, L., Friedman, J. H., Olshen, R. A. & Stone, C. J. (1984) *Classification and Regression Trees* (Belmont, CA, Wadsworth).

Cosman, P. C., Riskin, E. A. & Gray, R. M. (1991a) Combining vector quantization and histogram equalization. *In:* J. A. Storer & J. H. Reif (Eds), *Proceedings of the 1991 IEEE Data Compression Conference* (Snowbird, UT, IEEE Computer Society Press), pp. 113–118.

COSMAN, P. C., OEHLER, K. L., HEATON, A. A. & GRAY, R. M. (1991b) Tree-structured vector quantization with input-weighted distortion measures, *Proceedings of SPIE: Visual Communications and Image Processing*, 1605, pp. 162–171.

COSMAN, P. C., PERLMUTTER, K. O., PERLMUTTER, S. M., OLSHEN, R. A. & GRAY, R. M. (1991c) Training sequence size and vector quantizer performance, *Proceedings of the 25th Annual Asilomar Conference on Signals, Systems, and Computers, Pacific Grove, CA*, pp. 434–438.

COSMAN, P. C., OCHLER, K. L., RISKIN, E. A. & GRAY, R. M. (1993a) Using vector quantization for image processing, *Proceedings of the IEEE*, 81, pp. 1326–1341.

COSMAN, P. C., TSENG, C., GRAY, R. M., OLSHEN, R. A., MOSES, L. E., DAVIDSON, H. C., BERGIN, C. J. & RISKIN, E. A. (1993b) Tree-structured vector quantization of CT chest scans: image quality and diagnostic accuracy, *IEEE Transactions on Medical Imaging*, 12, pp. 727–739.

COSMAN, P. C., DAVIDSON, H. C., BERGIN, C. J., TSENG, C., MOSES, L. E., RISKIN, E. A., OLSHEN, R. A. & GRAY, R. M. (1994) Thoracic CT images: effect of lossy image compression on diagnostic accuracy, *Radiology*, 190, pp. 517–524.

DONOHO, D. (1992) De-noising by soft-thresholding, *Technical Report 409*, Department of Statistics, Stanford University, Stanford, CA.

DONOHO, D. & JOHNSTONE, I. M. (1992) Minimax estimation via wavelet shrinkage, *Technical Report 402*, Department of Statistics, Stanford University, Stanford, CA.

EFRON, B. (1982) *The Jackknife, the Bootstrap, and Other Resampling Plans* (Philadeliphia, PA, Society for Industrial and Applied Mathematics).

GERSHO, A. & GRAY, R. M. (1992) *Vector Quantization and Signal Compression* (Norwell, MA, Kluwer).

GRAY, R. M., COSMAN, P. C. & RISKIN, E. A. (1992) Image compression and tree-structured vector quantization. *In:* J. STORER & J. REIF, (Eds), *Image and Text Compression* (Norwell, MA, Kluwer), pp. 3–34.

GRAY, R. M., COSMAN, P. C. & OEHLER, K. L. (1993) Incorporating visual factors into vector quantizers for image compression. *In:* B. WATSON (Ed.), *Digital Images and Human Vision* (Cambridge, MA, MIT Press), pp. 35–52.

LLOYD, S. P. (1957) Least squares quantization in PCM, Unpublished Bell Laboratories Technical note; published in 1982 in the *IEEE Transactions on Information Theory*, 28, pp. 127–135.

MCNEMAR, I. (1947) Note on the sampling errors of the differences between correlated proportions of percentages, *Psychometrika*, 12, pp. 153–157.

MILLER, R. G. Jr, (1981) *Simultaneous Statistical Inference*, 2nd edn (New York, Springer).

NOBEL, A. B. & OLSHEN, R. A. (1993a) Boundedness and consistency of greedy growing for tree-structured vector quantizers, *Proceedings of the 1993 IEEE International Symposium on Information Theory, San Antonio, TX, January*, p. 334.

NOBEL, A. & OLSHEN, R. A. (1993b) Termination and continuity of greedy growing for tree structured vector quantizers, *Technical Report 164*, Stanford University.

OEHLER, K. L., COSMAN, P. C., GRAY, R. M. & MAY, J. (1991) Classification using vector quantization, *Proceedings of the 25th Annual Asilomar Conference on Signals, Systems, and Computers, Pacific Grove, CA*, pp. 439–445.

PERLMUTTER, K. O., PERLMUTTER, S. M., COSMAN, P. C., RISKIN, E. A., OLSHEN, R. A. & GRAY, R. M. (1992) Tree-structured vector quantization with region-based classification, *Proceedings of the 26th Annual Asilomar Conference on Signals, Systems, and Computers, Pacific Grove, CA*.

POWELL, M. J. D. (1981) *Approximation Theory and Methods* (Cambridge, Cambridge University Press).

RISKIN, E. A., LOOKABOUGH, T., CHOU, P. A. & GRAY, R. M. (1990) Variable rate vector quantization for medical image compression, *IEEE Transactions on Medical Imaging*, 9, pp. 290–298.