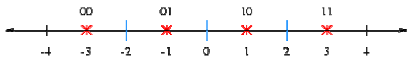
Vector Quantization

http://www.data-compression.com/vq.shtml

Vector quantization is used in many applications such as image and voice compression, voice recognition (in general statistical pattern recognition).

A VQ is nothing more than an approximator. The idea is similar to that of ``rounding-off'' (say to the nearest integer).

An example of a 1-dimensional VQ is shown below:

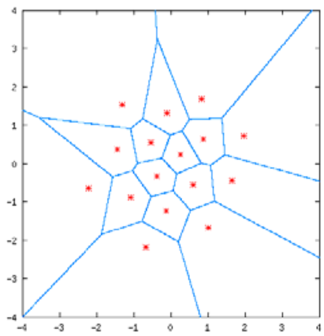


Here, every number less than -2 are approximated by -3. Every number between -2 and 0 are approximated by -1...

Note that the approximate values are uniquely represented by 2 bits. This is a 1-dimensional, 2-bit VQ. It has a rate of 2 bits/dimension.

Vector Quantization

An example of a 2-dimensional VQ is shown below:



Here, every pair of numbers falling in a particular region are approximated by a red star associated with that region. Note that there are 16 regions and 16 red stars -- each of which can be uniquely represented by 4 bits. Thus, this is a 2 dimensional, 4-bit VQ. Its rate is also 2 bits/dimension.

Design Problem

In the above two examples, the red stars are called *codevectors* and the regions defined by the blue borders are called *encoding regions*. The set of all codevectors is called the *codebook* and the set of all encoding regions is called the *partition* of the space.

The VQ design problem can be stated as follows. Given a vector source with its statistical properties known, given a distortion measure, and given the number of codevectors, find a codebook (the set of all red stars) and a partition (the set of blue lines) which result in the smallest average distortion.

LBG Algorithm

We assume that there is a *training sequence* consisting of source vectors:

$$\mathrm{T} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_M\}$$

This training sequence can be obtained from some large database. For example, if the source is a speech signal, then the training sequence can be obtained by recording several long telephone conversations. M is assumed to be sufficiently large so that all the statistical properties of the source are captured by the training sequence.

We assume that the source vectors are k-dimensional, e.g.,

$$\mathbf{x}_m = (x_{m,1}, x_{m,2}, \ldots x_{m,k})$$

Let N be the number of codevectors and let

$$C = (\mathbf{c}_1, \mathbf{c}_2, \ldots \mathbf{c}_N)$$

represents the codebook. Each codevector is k-dimensional, e.g.,

$$c_n = (\mathbf{c}_{n,1}, \mathbf{c}_{n,2}, \ldots \mathbf{c}_{n,k})$$

Let S_n be the encoding region associated with codevector $\mathbf{c}_n$ and let

$$P = \{S_1, S_2, \ldots, S_N\}$$

denote the partition of the space. If the source vector x_m is in the encoding region S_n , then its approximation (denoted by $Q(\mathbf{x}_m)$) is c_n :

$$Q(\mathbf{x}_m) = \mathbf{c}_n$$

If $\mathbf{x}_m \in S_n$.

Design Problem

Assuming a squared-error distortion measure, the average distortion is given by:

$$D_{ave} = \frac{1}{Mk} \sum_{m=1}^{M} \| \mathbf{x}_m - Q(\mathbf{x}_m) \|^2$$

Where $||\mathbf{e}||^2 = e_1^2 + e_2^2 + \ldots$..

The design problem can be succinctly stated as follows:

Given T and N, find C and P such that D_{ave} is minimized.

Optimality Criteria

If C and P are a solution to the above minimization problem, then it must satisfied the following two criteria:

- Nearest Neighbor Condition: $S_n = \left\{ \mathbf{x} : \left\| \mathbf{x} - \mathbf{c}_n \right\|^2 \le \left\| \mathbf{x} - \mathbf{c}_{n'} \right\|^2 \forall n' = 1,2,\ldots, N \right\}$

This condition says that the encoding region S_n should consists of all vectors that are closer to $\mathbf{c}_n$ than any of the other codevectors. For those vectors lying on the boundary (blue lines), any tie-breaking procedure will do.

- Centroid Condition $$\mathbf{c}_n = \frac{\sum_{\mathbf{x}_m \in S_n} \mathbf{x}_m}{\sum_{\mathbf{x}_m \in S_n} 1} \qquad n=1,2,\ldots N$$

This condition says that the codevector c_n should be average of all those training vectors that are in encoding region S_n. In implementation, one should ensure that at least one training vector belongs to each encoding region (so that the denominator in the above equation is never 0).

LBG Design Algorithm

The LBG VQ design algorithm is an iterative algorithm which alternatively solves the above two optimality criteria.

The algorithm requires an initial codebook $C^{(0)}$.

This initial codebook is obtained by the *splitting* method. In this method, an initial codevector is set as the average of the entire training sequence. This codevector is then split into two. The iterative algorithm is run with these two vectors as the initial codebook.

The final two codevectors are splitted into four and the process is repeated until the desired number of codevectors is obtained. The algorithm is summarized below.

LBG

1. Given T. Fixed ε>0 to be a small number.
2. Let N=1

$$\mathbf{c}_1^* = \frac{1}{M}\sum_{m=1}^{M}\mathbf{x}_m$$

$$D_{ave}^* = \frac{1}{Mk}\sum_{m=1}^{M}\left\|x_m - c_1^*\right\|^2$$

3. For i=1,2,…N set

$$\mathbf{c}_i^{(0)} = (1+\varepsilon)\mathbf{c}_i^*$$
$$\mathbf{c}_{N+i}^{(0)} = (1-\varepsilon)\mathbf{c}_i^*$$

Set N=2N

LBG

4. Iteration: Let $D_{ave}^{(i)} = D_{ave}^*$. Set the index iteration i = 0

   i. For m=1,2,…,M, find the minimum value of

$$\left\|x_m - c_n^{(i)}\right\|^2$$

   over all n=1,2,….,N. over all . Let n^* be the index which achieves the minimum. Set

$$Q(x_m) = c_{n^*}^{(i)}$$

   ii. For n=1,2,…,N update the codevector

$$c_n^{(i+1)} = \frac{\sum_{Q(x_m)=c_n^{(i)}} x_m}{\sum_{Q(x_m)=c_n^{(i)}} 1}$$

   iii. Set i=i+1.

LBG

iv. Calculate

$$D_{ave}^{(i)} = \frac{1}{Mk}\sum_{m=1}^{M}\left\|x_m - Q(x_m)\right\|^2$$

v. if $(D_{ave}^{(i-1)}-D_{ave}^{(i)})/D_{ave}^{(i-1)}>\varepsilon$, go back to step (i)

vi. Set $D_{ave}^{*}=D_{ave}^{(i)}$. For n=1,2,…N, set

$$c_n^{\ *} = c_n^{\ (i)}$$

as the final codevector.

5. Repeat steps 3 and 4 until the desired number of codevectors is obtained

Suggested Exercizes

Channel capacity: 2,3,4,9,11
Gaussian channel: 2,3,4
Rate distortion: 1,2,3,5,7,(9)