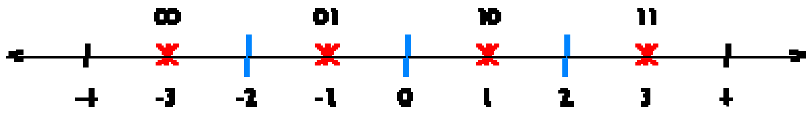
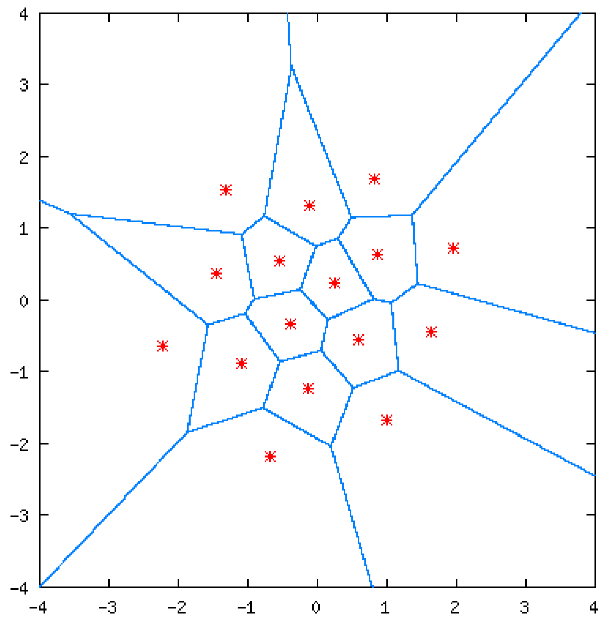
6. VECTOR QUANTIZATION METHODS

General idea: Code highly probable symbols into short binary sequences without regard to their statistical, temporal, or spatial behavior.

Introduction: Vector quantization (VQ) is a lossy data compression method based on the principle of block coding, i.e., coding vectors of information into codewords composed of string of bits. It is a fixed-to-fixed length algorithm. In 1980s, the design of a vector quantizer (VQ) is considered to be a challenging problem due to the need for multi-dimensional integration. Linde, Buzo, and Gray (LBG) proposed a VQ design algorithm following the ideas of Lloyd (1957) based on a training sequence[1]. The use of a training sequence bypasses the need for probability density assumption. The idea is similar to that of ``rounding-off" (say to the nearest integer). An example of a 1-dimensional VQ, as it was suggested by Llyod in his famous unpublished memo, is shown below:



Here, every number less than -2 are approximated by -3. Every number between -2 and 0 are approximated by -1. Every number between 0 and 2 are approximated by +1. Every number greater than 2 is approximated by +3. It is worth noting 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. An example of a 2-dimensional VQ is shown below:

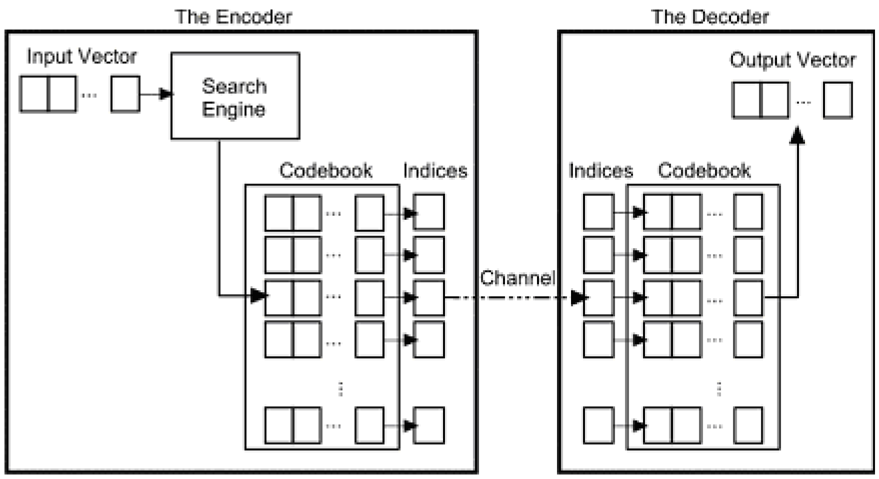


Here, every pair of numbers falling in a particular region is approximated by a red star associated with that region. Note that there are 16 regions and 16 red stars -- each of which can be uniquely represented by 4 bits. Thus, this is a 2-dimensional, 4-bit VQ. Its rate is also 2 bits/dimension.

[1] Some people call it LBG Algorithm even though the people involved in the early stages of the VQ evolution including Linde, Buzo, Gray, themselves, Gersho, the author, and many other scholars prefer it to be called as Lloyd Algorithm.

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.

Communication systems based on VQ can be illustrated with the following encoder/decoder pair:



Llyod Algorithm Design: Assume that there is a *training sequence* T consisting of M source vectors: $T = \{x_1, x_2, \cdots, x_M\}$, a distortion measure (distance measure, such as MSE or MAE), and the number of codevectors, the task is to find a codebook (the set of all red stars) and a partition, which result in the smallest average distortion.

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 or pictures from various sources. 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.,

$$x_m = (x_{m,1}, x_{m,2}, \cdots, x_{m,k}) \qquad \text{for } m = 1,2,\cdots,M$$

Let N be the number of codevectors and let

$$C = \{c_1, c_2, \cdots, c_N\}$$

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

$$c_n = (c_{n,1}, c_{n,2}, \cdots, c_{n,k}) \qquad \text{for } n = 1,2,\cdots,N$$

Let S_n be the encoding region associated with codevector C_n and let

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

denote the partition of the space. If the source vector x_m is in the encoding region S_n, then its approximation is given by:

$$Q(x_m) = c_n \quad if \quad x_m \in S_n \tag{6.1}$$

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

$$D_{ave} = \frac{1}{Mk}\sum_{m=1}^{M}\|x_m - Q(x_m)\|^2 = \frac{1}{Mk}\sum_{m=1}^{M}\|e\|^2 \tag{6.2}$$

where the error-square is the sum of error terms on each dimension:

$$\|e\|^2 = e_1^2 + e_2^2 + \cdots + e_k^2 \quad . \tag{6.3}$$

Design problem can be stated as follows: **Given T and N, find C and P and such that D_{ave} is minimized.**

Optimality Criteria: If the codebook $\boldsymbol{C}$ and the partition $\boldsymbol{P}$ are a solution to the above minimization problem, then it must satisfied the following two criteria together with the iteration stopping criterion.

1. **Nearest Neighbor Condition:**

$$S_n = \{x : \|x - c_n\|^2 \le \|x - c_j\|^2 \quad for\ all\ n, j = 1,2,\cdots,N\} \tag{6.4}$$

This condition says that the encoding region S_n should consist of all vectors that are closer to c_n than any of the other codevectors. For those vectors lying on the boundary, any tie-breaking procedure will do.

2. **Centroid Condition:**

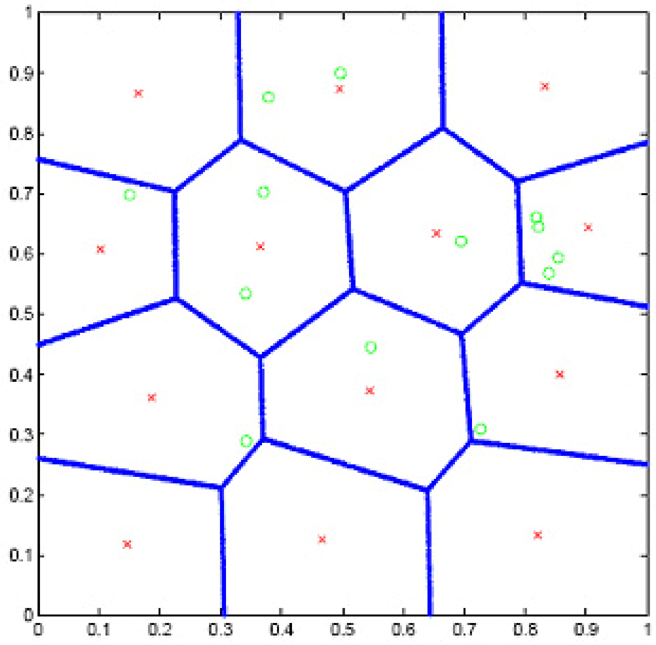
$$c_n = \frac{\sum_{x_m \in S_n} x_m}{\sum_{x_m \in S_n} 1} \quad for \quad n = 1,2,\cdots,N \tag{6.5}$$

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).

3. Stopping (Decision) Test: Suppose that the average distortion from NN mapping in the previous and current iterations were $D_{ave}^{(i)}, D_{ave}^{(i-1)}$ the iteration process would terminate if

$$\left| \frac{D_{ave}^{(i)} - D_{ave}^{(i-1)}}{D_{ave}^{(i-1)}} \right| < \varepsilon \tag{6.6}$$

where ε is a small number.



× denotes centroid, ○ denotes initial points.

Detailed Steps of VQ (Llyod) Design Algorithm

A. Initial Codebook Selection Step:

In order to start the Generalized Lloyd VQ Design Algorithm (GLA), we need to have an initial guess codebook in addition to an appropriate distortion measure. There are a number of initial codebooks used in literature with good success. Some of these are:

- **Random Codebooks:** Simplest way to assign an initial codebook to use random numbers generated according to an uniform distribution.
- **K-Means Initial Codbook:** Well-known method in experimental statistics is to use the first 2^R vectors in the training algorithm as the initial codebook.
- **Product Codebooks:** Use a scalar quantizer code, such as a uniform quantizer k times in succession and then prune the resulting codebook down to the correct size.
- **Splitting:** Grow bigger codes with a fixed dimension from smaller ones.
  1. Find an optimum rate=0 code (centroid of the entire database.)
  2. Split this code into 2 by perturbing the original by a small amount.
  3. Obtain the optimum rate=1 code via GLA.
  4. Split and optimize again until codebook with correct size is found.

An advantage of the last method, which is the norm nowadays, is that all smaller codebooks are also optimum.

Let us assume that $\mathbf{C}^{(0)}$ is an initial codebook either assumed as a random vector or obtained by the ***splitting method***, in this latter case, an initial codevector is set as the average of the entire training sequence. Let $\varepsilon > 0$ and $\delta > 0$ be small decision threshold value (used for stopping/continuing) and a splitting increment, respectively. Start with the size of the codebook as *N=1*. 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 split into four and the process is repeated until the desired number of codevectors is obtained.

B. Llyod Iteration Steps:

1. Calculate the **centroids** and the resulting **distortion**:

$$c_n^* = \frac{1}{M} \sum_{x_m \in S_n} x_m \qquad \text{and} \qquad D_{ave}^* = \frac{1}{Mk} \sum_{m=1}^{M} \left\| x - c_1^* \right\|^2 \tag{6.7}$$

2. **Splitting:** For $i = 1, 2, \cdots, N$ set

$$c_i^{(0)} = (1+\delta).c_i^*; \quad c_{N+i}^{(0)} = (1-\delta).c_i^* \tag{6.8}$$

   Now set *N=2N*.

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

   a. For $m = 1, 2, \cdots, M$, , find the minimum value of $\left\| x_m - c_n^{(i)} \right\|$ over all $n = 1, 2, \cdots, N$. Let n^* be the index which achieves the minimum. Set $Q(x_m) = c_n^{(i)}$.

   b. For $n = 1, 2, \cdots, N.$, update the codevector

$$c_n^{(i+1)} = \frac{\sum_{Q(x_m) \in c_n^{(i)}} x_m}{\sum_{Q(x_m) \in c_n^{(i)}} 1} \tag{6.9a}$$

c. Set $i=i+1$

d. Calculate

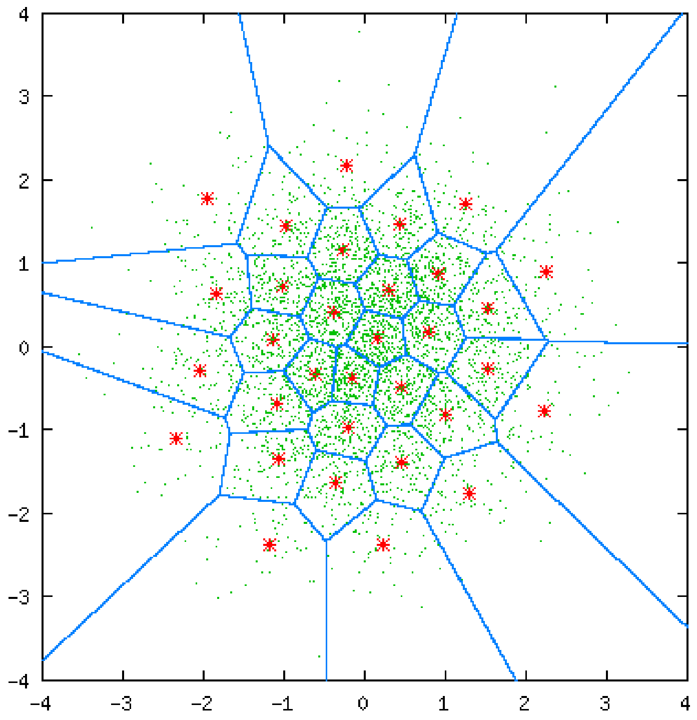
$$D_{ave}^{(i)} = \frac{1}{Mk}\sum_{m=1}^{M}\left\| x_m - Q(x_m)\right\|^2 \tag{6.9b}$$

e. If $\dfrac{D_{ave}^{(i-1)} - D_{ave}^{(i)}}{D_{ave}^{(i-1)}} > \varepsilon$, go back to Step (i) to increase the iteration number by 1.

f. If not, set $D_{ave}^{(*)} = D_{ave}^{(i)}$ and for $n = 1,2,\cdots,N.$, set $c_n^* = c_n^{(i)}$ as the final codebook.

4. Repeat Steps **2** and **3** until the desired number of codevectors is obtained.

Two-Dimensional Animation: Click on the figure below to begin the animation.

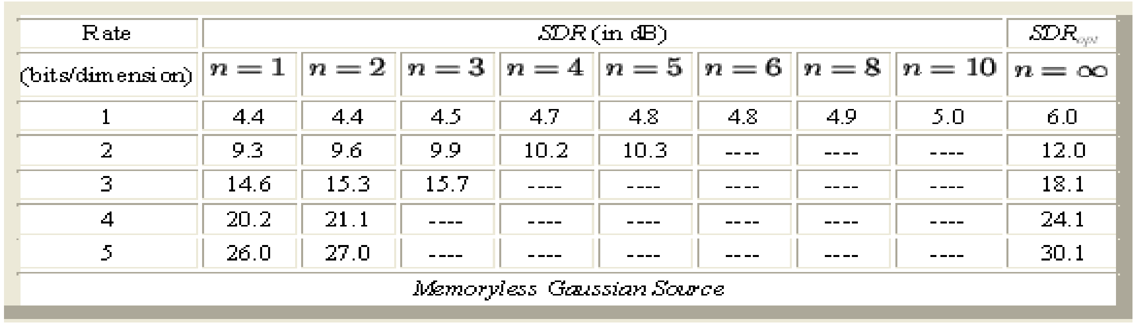


- The source for the above is a memoryless Gaussian source with zero-mean and unit variance.
- The size of the training sequence should be sufficiently large. It is recommended that $M \geq 1000N$. In this animation, tiny green dots represent 4096 training vectors.
- Lloyd design algorithm is run with $\varepsilon = 0.001$.
- The algorithm guarantees a locally optimal solution.

Performance: The performance of VQ are typically given in terms of the signal-to-distortion ratio (*SDR*):

$$SDR = 10.\log_{10}\frac{\sigma^2}{D_{ave}} \quad in \quad dB \tag{6.10}$$

where σ^2 is the variance of the source and D_{ave} is the average squared-error distortion. The higher the *SDR* the better the performance. The following tables show the performance of VQ for the memoryless Gaussian source and comparisons can be made with the optimal performance theoretically attainable, SDR_{opt}, which is obtained by evaluating the rate-distortion function.



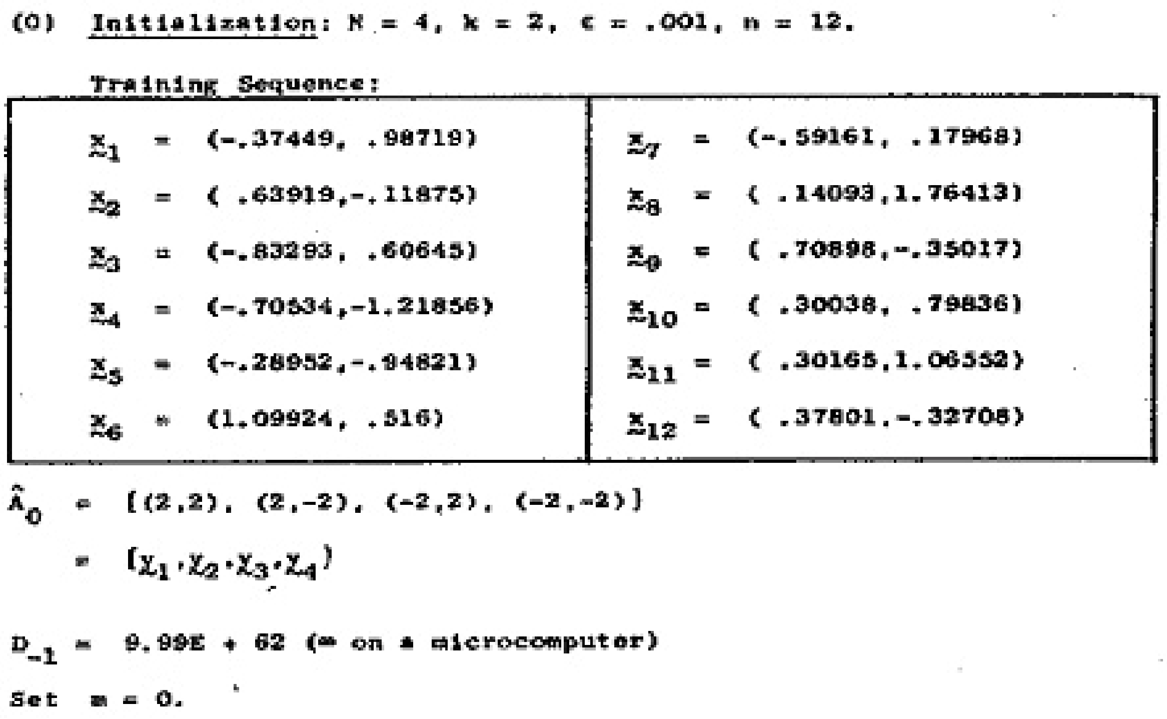
Rate	SDR (in dB)								SDR_{opt}
(bits/dimension)	$n=1$	$n=2$	$n=3$	$n=4$	$n=5$	$n=6$	$n=8$	$n=10$	$n=\infty$
1	4.4	4.4	4.5	4.7	4.8	4.8	4.9	5.0	6.0
2	9.3	9.6	9.9	10.2	10.3	----	----	----	12.0
3	14.6	15.3	15.7	----	----	----	----	----	18.1
4	20.2	21.1	----	----	----	----	----	----	24.1
5	26.0	27.0	----	----	----	----	----	----	30.1

Memoryless Gaussian Source

References

1. P. Namdo: http://www.data-compression.com/vq.shtml
2. A. Gersho and R. M. Gray, Vector Quantization and Signal Compression, Kluver Academic Press, 1991
3. H. Abut, Vector Quantization, IEEE Press, 1990.
4. R. M. Gray, ``Vector Quantization," *IEEE ASSP Magazine*, pp. 4--29, April 1984.
5. Y. Linde, A. Buzo, and R. M. Gray, ``An Algorithm for Vector Quantizer Design," *IEEE Transactions on Communications*, pp. 702--710, January 1980.

Example: 6.1 Llyod design of a 2-bit (N=4), 2-Dimensional (k=2) with $\varepsilon = 0.001$ based on MSE and 12 training vectors (2-dimensional). (Curtesy of Bob Gray, ref. 3, 5.)



(0) Initialization: $N = 4$, $k = 2$, $\epsilon = .001$, $n = 12$.

Training Sequence:

$\underset{\sim}{x}_1 = (-.37449, .98719)$	$\underset{\sim}{x}_7 = (-.59161, .17968)$
$\underset{\sim}{x}_2 = (.63919, -.11875)$	$\underset{\sim}{x}_8 = (.14093, 1.76413)$
$\underset{\sim}{x}_3 = (-.83293, .60645)$	$\underset{\sim}{x}_9 = (.70898, -.35017)$
$\underset{\sim}{x}_4 = (-.70534, -1.21856)$	$\underset{\sim}{x}_{10} = (.30038, .79836)$
$\underset{\sim}{x}_5 = (-.28952, -.94821)$	$\underset{\sim}{x}_{11} = (.30165, 1.06352)$
$\underset{\sim}{x}_6 = (1.09924, .516)$	$\underset{\sim}{x}_{12} = (.37801, -.32708)$

$\hat{A}_0 = [(2,2), (2,-2), (-2,2), (-2,-2)]$

$= \{y_1, y_2, y_3, y_4\}$

$D_{-1} = 9.99E + 62$ (∞ on a microcomputer)

Set $m = 0$.

$\underline{\underline{m=0}}$ (1) Find $P(\hat{A}_0) = \{S_1, S_2, S_3, S_4\}$:

$\underset{\sim}{x}_j \in S_i$ if $d(\underset{\sim}{x}_j, \underset{\sim}{y}_i) \le d(\underset{\sim}{x}_j, \underset{\sim}{y}_m)$, all m.

$S_1 = \{\underset{\sim}{x}_6, \underset{\sim}{x}_8, \underset{\sim}{x}_{10}, \underset{\sim}{x}_{11}\}$

$S_2 = \{\underset{\sim}{x}_2, \underset{\sim}{x}_9\}$

$S_3 = \{\underset{\sim}{x}_1, \underset{\sim}{x}_3, \underset{\sim}{x}_7\}$

$S_4 = \{\underset{\sim}{x}_4, \underset{\sim}{x}_5, \underset{\sim}{x}_{12}\}$

Compute D_0:

$$D_0 = \frac{1}{12} \sum_{j=1}^{12} \min_{\underset{\sim}{y} \in \hat{A}_0} d(\underset{\sim}{x}_j, \underset{\sim}{y}) = 2.0172 .$$

(2) $(D_{-1} - D_0)/D_0 > .001$, continue.

(3) Find the optimal reproduction alphabet $\hat{A}_1 \triangleq \hat{\underset{\sim}{x}}(P(\hat{A}_0)) = \{\hat{\underset{\sim}{x}}(S_i), i=1,\ldots,4\}$:

$\hat{\underset{\sim}{x}}(S_1) = (\underset{\sim}{x}_6 + \underset{\sim}{x}_8 + \underset{\sim}{x}_{10} + \underset{\sim}{x}_{11})/4 = (.46055, 1.036)$

$\hat{\underset{\sim}{x}}(S_2) = (\underset{\sim}{x}_2 + \underset{\sim}{x}_9)/2 = (.674085, -.23446)$

$\hat{\underset{\sim}{x}}(S_3) = (\underset{\sim}{x}_1 + \underset{\sim}{x}_3 + \underset{\sim}{x}_7)/3 = (-.599676, .591106)$

$\hat{\underset{\sim}{x}}(S_4) = (\underset{\sim}{x}_4 + \underset{\sim}{x}_5 + \underset{\sim}{x}_{12})/3 = (-.457623, -.831283)$

Set $m = 1$. Go to (1).

$\underline{\underline{m=1}}$ (1) Find $P(\hat{A}_1)$:

Evaluating distortions shows $P(\hat{A}_1) = P(\hat{A}_0)$ (no change in partition)

Compute D_1:

$$D_1 = \frac{1}{12} \sum_{j=1}^{12} \min_{\underset{\sim}{y} \in \hat{A}_1} d(\underset{\sim}{x}_j, \underset{\sim}{y}) = .0997308 .$$

(2) $(D_0 - D_1)/D_1 \cong 19 > .001$

(3) $\hat{A}_2 \triangleq \hat{\underset{\sim}{x}}(P(\hat{A}_1)) = \hat{A}_1$, since $P(\hat{A}_1) = P(\hat{A}_0)$ and hence

$\hat{\underset{\sim}{x}}(P(\hat{A}_1)) = \hat{\underset{\sim}{x}}(P(\hat{A}_0)) = \hat{A}_1$. Thus $\hat{A}_1$ is a fixed point. Set $m=2$. Go to (1).

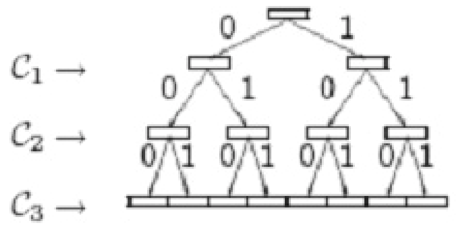
$\underline{\underline{m=2}}$ (1) $P(\hat{A}_1) = P(\hat{A}_0)$ and hence $D_2 = D_1$ and hence $(D_1 - D_2)/D_2 = 0 < .001$.

Halt with final quantizer described by $\{\hat{A}_1, P(\hat{A}_1)\}$.

As we can see from the above hand calculations the algorithm converged after two iterations.

Full Search vs Tree Search: If we search and compute each and every codevector in the design or the encoding modes of the VQ system then it is called a full search. All the discussion until this point has been on full search as shown in the figure below all the computations are done at level

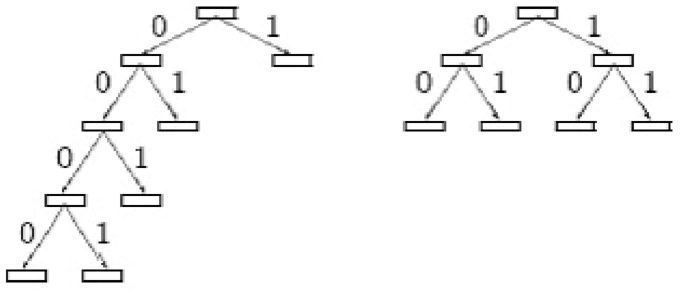
C_3. On the other hand, if we search the codebook in a tree fashion then the search is called a tree search.



Example 6.2: Computational complexity of search:

- Full search: For this figure we need 8 distortion computations at level C_3.
- Tree search: We need two distortion computations in each level C_1, C_2, C_3. 6 computations all together.
- Similarly, for a level 10 (10-bit per sample) quantizer, the computations would $2^{10} = 1024$ versus 2x10=20, resulting at a computational savings of 50:1!
- Cost: The performance of tree search VQs are inferior to full search 0.5-3.0 dB depending upon the source and its statistics.

Complete Tree vs. Pruned-Tree Search: If the occupancy statistics of all the terminal nodes (children) in a tree is computed they are all significant then every branch (child, terminal node) is critical and the full tree needs to be searched. If it turns out that some offspring branches (children) are rarely used then they can be combined and represented by their parent branches. Trees of this sort are called pruned tree.



In the full tree every terminal branch is assigned a code of equal length. However, shorter codes are assigned to top branches and longer codes are needed for grand-grand children just like Huffman coding schemes. Therefore, variable-length codes are used in pruned tree encoders.

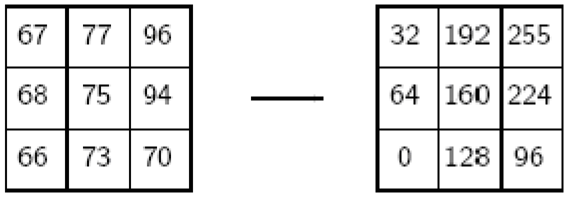
- Increased storage, possible performance loss.
- Code is successive approximation (progressive), each bit provides improved quality.
- Table lookup decoder (simple, cheap, software).
- Unbalanced (incomplete) tree provides a variable-rate code.
- Throw more bits at more active blocks.

Two TSVQ design approaches:

1. Begin with a reproduction codebook and design a tree-search into the designed codebook by a pruning technique.
2. Design from scratch.

VQ Compression Off-springs:

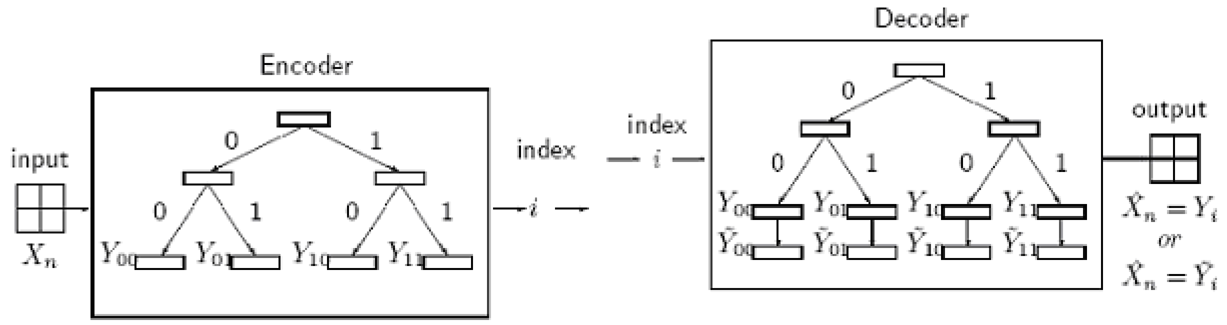
1. Histogram Equalized VQ: Used in contrast enhancement by mapping each pixel to an intensity proportional to its rank in histogram (probability) ordering among its pixels. The resulting output image has a more uniform histogram.



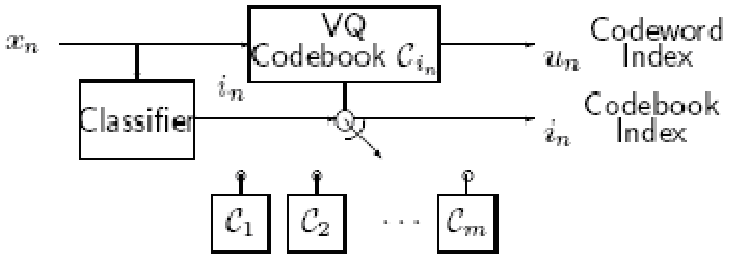
67	77	96
68	75	94
66	73	70

32	192	255
64	160	224
0	128	96

2. TSVQ and Histogram Equalization: A tree-structured vector quantizer can be built with both equalized and un-equalized codebooks.

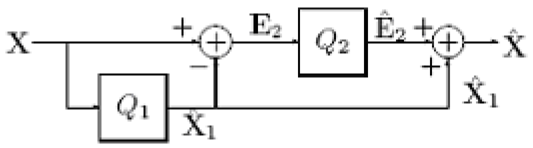


3. Classified VQ (CVQ, Switched VQ): Separate codebook for each input class (e.g., active, inactive, textured, background, edge with orientation, etc.) and switch among classes according to some rule.

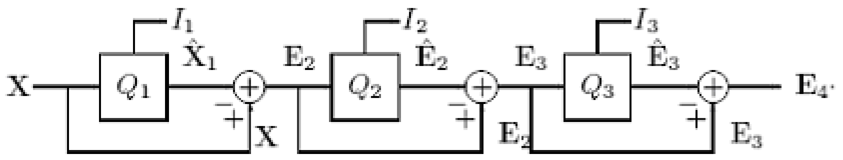


- Requires on-line classification and side information.
- Rate allocation issue: How divide bit rate among sub-codes? Often an optimization rule is used.
- Structure often useful for proving theorems, e.g., classic and modern high rate systems uses “composite" codes which are just CVQ.
- If search all subcodes to find the minimum distortion word, then pick best codebook and best word: also known as the **universal quantizer.**
- Better performance, but usually larger complexity.

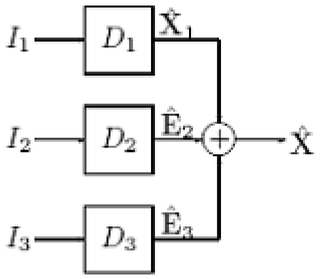
4. Multistage VQ (Multistep or Cascade or Residual VQ)



2-Stage Encoder

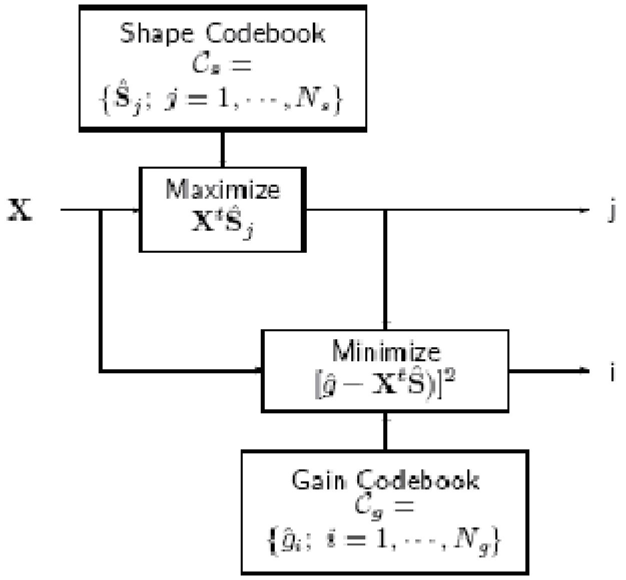


Multistage Encoder

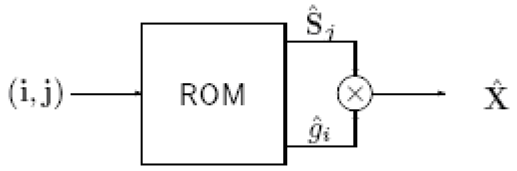


3-stage Decoder

5. Shape-Gain VQ: Separate the gain information and code it and then the shape information is encoded by a normalized second quantizer.

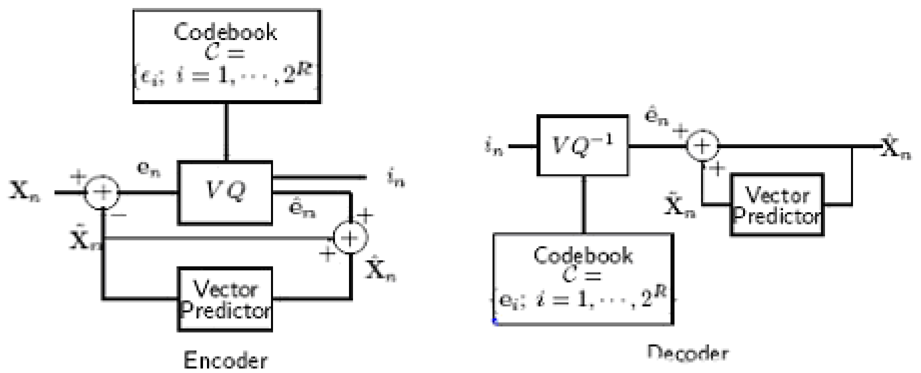


Shape Codebook
$C_s = \{\hat{S}_j;\ j = 1, \cdots, N_s\}$
X
Maximize
$X^t\hat{S}_j$
j
Minimize
$[\hat{g} - X^t\hat{S})]^2$
i
Gain Codebook
$C_g = \{\hat{g}_i;\ i = 1, \cdots, N_g\}$

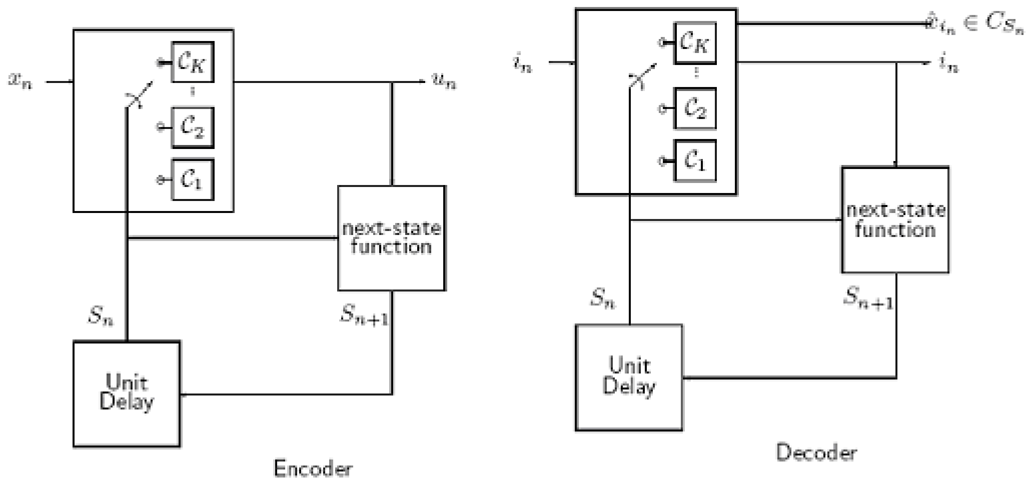


- Do not need to normalize input vector to encode shape, no online division.
- Two step encoding of shape then gain is optimal for the constrained codebook, chooses best combination in overall MSE sense.

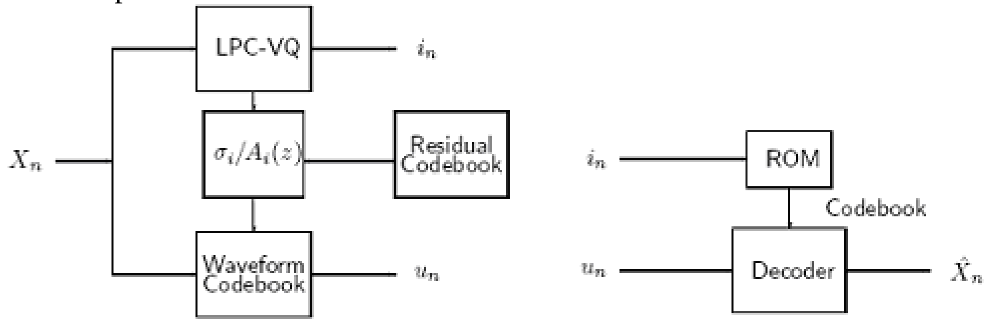
6. Predictive VQ (Vector DPCM): Used in some speech compression tasks and motion-compensated video coding.



7. Recursive VQ: Finite State VQ (FSVQ): Switched Vector Quantizer: Different codebook for each state and need a Next State Rule, also known as a classified VQ with backward adaptation.



8. Model-based VQ: Linear Predictive Coding (LPC): Founding building block in all cellular phones, which is .mathematically equivalent to a minimum distortion selection of a "vocal tract" model and the gain of the speech segment. Done over 20-30 ms long segments of speech, where the statistical behavior of the speaker's vocal tract does not change. The famed Itakura-Saito distortion (spectral distance of the vocal tracts, i.e, distance between the Fourier transform of the vocal tract mathematical model. Encoded parameters are: gain of a speech segment, a set of 10-12 LPC coefficients to represent the vocal tract, the pitch period (quasi-periodic harmonics of the oscillations in the vocal cords) and the residuals between the actual speech and the locally synthesized model speech.



Example 6.4: Explore the performance of image codecs based on VQ using VcDemo.