3. LOSSLESS/LOSSY COMPRESSION

Overview of Lossless Compression:

It is also known as:

- Noiseless coding
- Lossless coding
- Invertible coding
- Entropy coding
- Data compaction codes.

They can perfectly recover original data (if no storage or transmission bit errors, i.e., noiseless channel).

1. They normally have variable length binary codewords to produce variable numbers of bits per symbol.
2. Only works for ***digital sources***.

Properties of Variable rate (length) systems:

- They can more efficiently trade off quality/distortion (noise) and rate.
- They are generally more complex and costly to build.
- They require buffering to match fixed-rate transmission system requirements.
- They tend to suffer catastrophic transmission error propagation; in other words, once in error then the error grows quickly.

3. But they can provide produce superior rate/distortion (noise) trade-off. For instance, in image compression we can use more bits for edges, fewer for flat areas. Similarly, more bits can be assigned for plosives, fewer for vowels in speech compression.

General idea: Code highly probable symbols into short binary sequences, low probability symbols into long binary sequences, so that average is minimized. Most famous examples:

- **Morse code:** Consider dots and dashes as binary levels "0" and "1".

  Assign codeword length inversely proportional to letter relative frequencies. In other words, most frequent letter (message) is assigned the smallest number of bits, whereas the least likely message has the longest codeword, such as "e" and ""z" in English alphabet.

  Total number of bits per second would much less than if we have used fixed-rate ASCII representation for alphanumerical data.

- **Huffman code** 1952, employed in UNIX *compact* utility and many standards, known to be optimal under specific constraints. (We will study later.)
- **Run-length codes** popularized by Golomb in early 1960s, used in JPEG standard.
- **Lempel-Ziv(-Welch) codes** 1977,78 in Unix *compress* utility, diskdouble, stuffit, stacker, PKzip, winzip, DOS, GIF.
- **Arithmetic codes** by Fano, Berlecamp, Rissanen, Pasco, Langdon. Example: IBM Q-coder.

Warning: To get average rate down, need to let maximum instantaneous rate grow. This means that can get data expansion instead of compression in the short run.

Typical lossless compression ratios: 2:1 to 4:1

Overview of Lossy Compression:

These schemes are non-invertible and information is always lost. However, they permit more compression.
Examples:

1. Memoryless, non-predictive compression techniques:

- PCM (Shannon (1938), Oliver, Pierce (1948).
- Sampling + scalar quantization; analog to digital conversion.
- By introducing loss in a controlled fashion, we could prevent further loss in the channel. Birth of digital communication.

2. Predictive coding:

Developed by Derjavitch, Deloraine, and Van Mierlo (1947), Elias (1950, 1955), Cutler (1952), DeJager (1952).

Idea: Predict next sample based on previous reconstructions (decisions) and code the differences (residual, prediction error). Specific systems:

- Predictive scalar quantization: Differential pulse code modulation (DPCM), delta modulation (DM), adaptive DPCM (ADPCM), sigma-delta modulation ($\Sigma - \Lambda$ *Modulation*).
- Almost all speech coders, in particular, cellular systems, use some form of predictive coding.
- CD players use $\Sigma - \Lambda$ *Modulation* and delta modulation, both forms of predictive codes.
- Most video codecs use frame-to-frame predictive coding: Motion compensation (MC).
- **Optimal quantization** Lloyd (1956) Optimizing PCM. connection of quantization and statistics (clustering)

3. Transform coding:

Developed Mathews and Kramer (1956), Huang (1962, 1963, Habibi, Chen (Compression Labs Inc). Dominant image coding (lossy compression) method in ISO and other standards:

- p*64, H.26*, JPEG, JPEG2000
- MPEG I, II, IV, and now VII.
- C-Cubed, CLI, Picture-Tel, and many others
- JPEG is ubiquitous in WWW and use transform coding (DCT) + custom uniform quantizers + runlength coding + Huffman or arithmetic coding..

4. Linear predictive coding (LPC)

First very low bit rate speech coding based on sophisticated signal processing and cellular technology, implementable in DSP chips.

- Itakura and Saito (1968), Atal et al. (1971), Markel and Gray (1972-76), NTT, Bell, Signal Technology, TI (Speak and Spell).

5. Vector Quantization (VQ)

Initial development late 1970s early 1980s.

- Very low rate LPC speech coding, Chaffee and Omura (1974-75), Linde, Buzo, Gray, et al. (1978-1980), Adoul et al. (1978), Abut, Gray, et al. (1982-84).
- Low rate image/video coding 1.0 bit/pixel or lower, Hilbert (1977), Yamada, et al. (1980-83), Gersho (1982), Baker (1982-83), Caprio et al. (1978), Menez, et al. (1979).
- Code excited LPC (CELP) basis for the cell phone technology: Steward (1981-82), Schroeder and Atal (1984-85)
- Software-based video: table look-ups instead of computation. Tekalp and others (1986). Quicktime, Sorenson, Indeo, Cell, Supermac Cinepak, Motive, VXtreme.

It is imperative for HDTV systems. (Already part of the Grand-Alliance Proposal for the standard.) It is currently in use in software based video: table lookups instead of computation Apple's QuickTime, Intel's Indeo, Sun's Cell, Supermac Technology's Cinepak, Media Vision's Motive, VXtreme

Various structures:

- Product codes (scalar quantization, gain/shape)
- Successive approximation/ tree-structured
- Predictive and finite-state
- Fractal coding
- Video streaming.

6. Subband/pyramid/wavelet Coding: (late1980s).

- MUSICAM digital audio (European standard)
- EZW (winzip) (Knowles, Shapiro)
- SPHIT (Said and Pearlman)
- CREW (RICOH Calif.),
- JPEG 2000 (Marcellin 2002)
- MPEG 4, MPEG7

7. Properties: (Possibly) in between lossless and lossy: "perceptually lossless" compression:

- what the eye can see (which may depend on context).
- Noise level of the acquisition device.
- What can be squeezed through transmission or storage, i.e., an imperfect picture may be better than no picture at all.
- Loss may be unavoidable, may have to choose between imperfect image or long delays or no data at all.
- However, some loss is not a problem with follow-up studies, archives, R&D, education, entertainment.

8. Fundamental Question: Is lossy compression acceptable?

- Growing evidence suggests that lossy compression even in medicine does not damage diagnosis and may in fact improve it if performed intelligently.
- However, it is NOT acceptable for computer programs, banking, espionage and a number of other security emphasized fields.

Compression application areas:

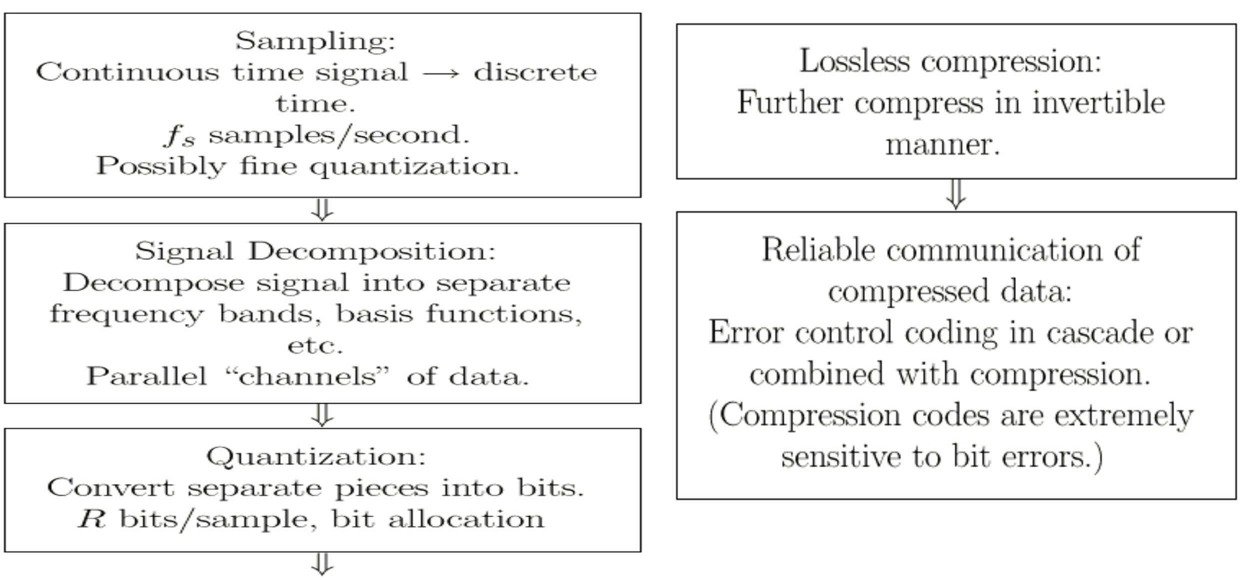
- Teleconferencing, FAX
- Medical records (archival, educational, remote diagnosis)
- Remote sensing, Remote control
- Space communications
- Multimedia, Web
- Binary, gray-scale, color, video, audio.
- Instructional TV on the net.
- Surveillance.
- Classification and feature extraction in person identification, gene clustering, bioinformatics.
- Image processing (texture modeling, synthesis)
- Data mining

Compression Context:

- It is generally part of a general system for data acquisition, transmission, storage, and display within the Shannon point-to-point communication system model.
- It involves A/D and D/A conversions and digital acquisition.
- Quality and utility of “modified” digital images/sounds/video varies depending on the specifics of the application and the bandwidth available.

Components of Typical Encoded Communication System

Transmitter (Encoding, Compression)



Sampling:

It is also known as A/D conversion, high resolution quantization achieved by digitizing amplitude if it is one-dimensional signal or digitizing space and amplitude for 2-D signals, i.e., image and video.

$N \times M$ pixels, $G = 2^r$ gray levels. Number of bits = $N \times M \times r$
Note: This step not necessary for digitally acquired image.

Basic idea: Sample and quantize:

For instance, for an image the process can be formulated by:

$$\{f(x,y);\ x,y \in \mathcal{R}\}$$
$$\Downarrow$$
$$\{f_{n,m} = f(n\Delta x, m\Delta y);\ n,m \in \mathcal{Z}\}$$
$$\Downarrow$$
$$\{q(f_{n,m});\ n,m \in \mathcal{Z}\}$$

Terminology: $f(x,y)$: image intensity (amplitude, brightness, luminance) at location (x,y)

$f_{n,m}$: sampled image value with integer indices (n,m), which is obtained at:

$f(n\Delta x, m\Delta y)$ of the original image with resolution Δx horizontally and Δy vertically.

$q(f_{n,m})$: digitized/quantized value of the image $f(x,y)$

Sampling (Nyquist) Theorem:

If a continuous-time (analog) signal *x(t)* has no frequency components (harmonics) at values greater than a frequency value f_{max} then this signal can be UNIQUELY represented by its equally spaced samples if the sampling frequency F_s is greater than or equal to $2f_{max}$. This is known as the analog-to-digital (A/D) conversion at F_s samples/s. Furthermore, the original continuous signal x(t) can be TOTALLY recovered from its samples x(n) after passing them through an ideal integrator (ideal low-pass filter) with an appropriate bandwidth.

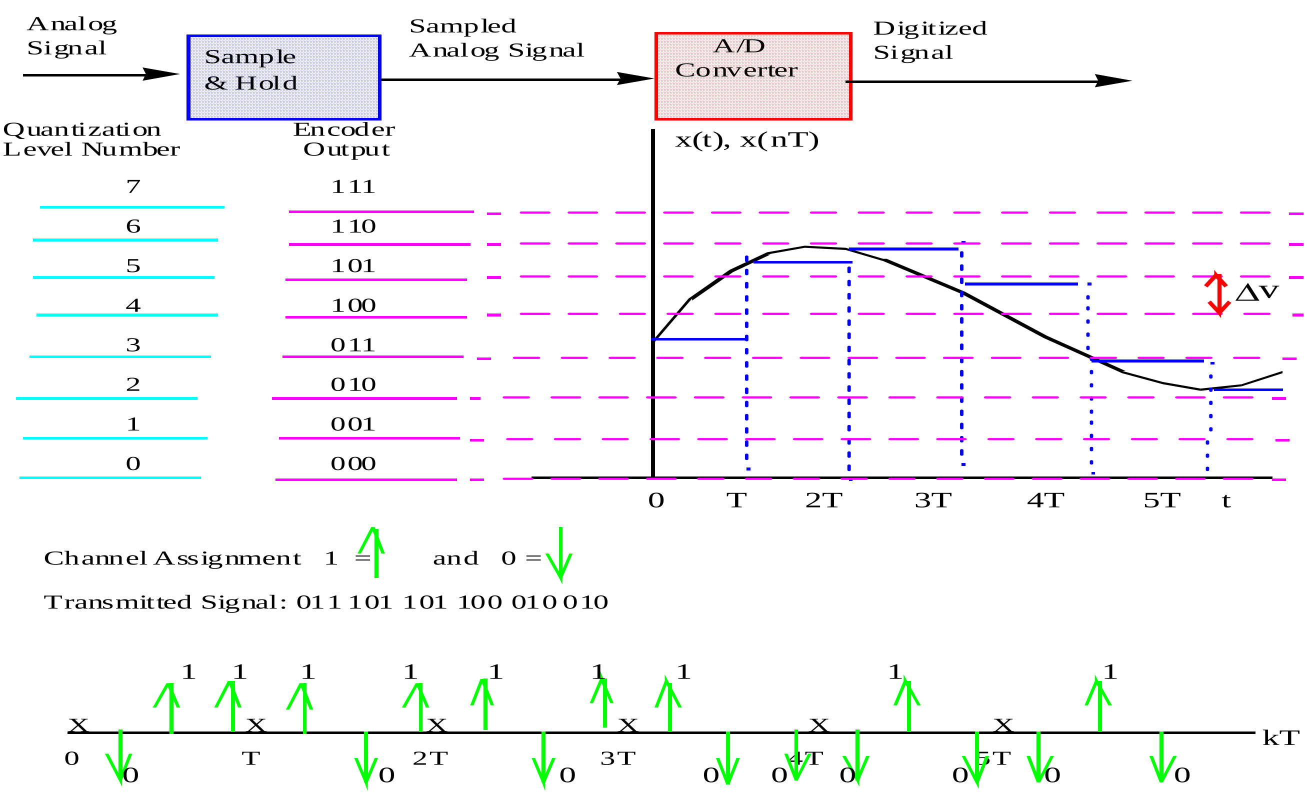


Figure 3.1 Quantization and coding of analog signals into binary information.

***Signal decomposition* (transformation, mapping):** Decompose image or speech into collection of separate images/speech (bands, coefficients). Examples:

- Fourier, Hadamard, Walsh, Sine, Discrete Cosine (DCT) and Karhunen-Loeve (KL),
- Wavelet and Sub-band (multi-resolution),
- Also: Hotelling, Principal Value Decomposition, Hartley, Fractal (Typically done digitally.)

Why?
Several reasons:

1. Good transforms tend to compact energy into a few coefficients, which allow many to be quantized to zero bits without affecting quality.
2. Good transforms tend to decorrelate (reduce linear dependence) among coefficients, causing scalar quantizers to be more efficient. (folk theorem)
3. Good transforms are effectively expanding the signals in good basis functions. (Mathematical intuition.)
4. The eye and ear tend be sensitive to behavior in the frequency domain, so coding in the frequency domain allows the use o perceptually based distortion measures, e.g., incorporating masking.

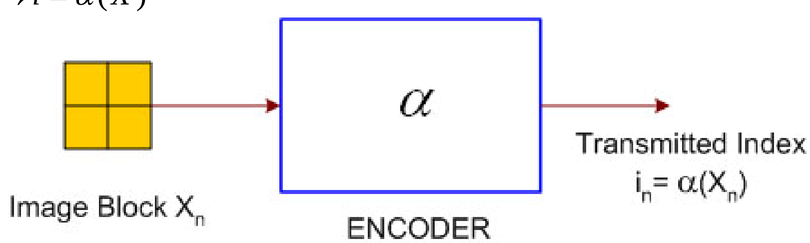
Issues

- Encoder and decoder structures/ Encoder and decoder algorithms,
- Encoder and decoder design and optimization/ Encoder and decoder performance (*R; D*),
- Encoder and decoder complexity/cost,
- Theoretical bounds and limits, and
- Systems

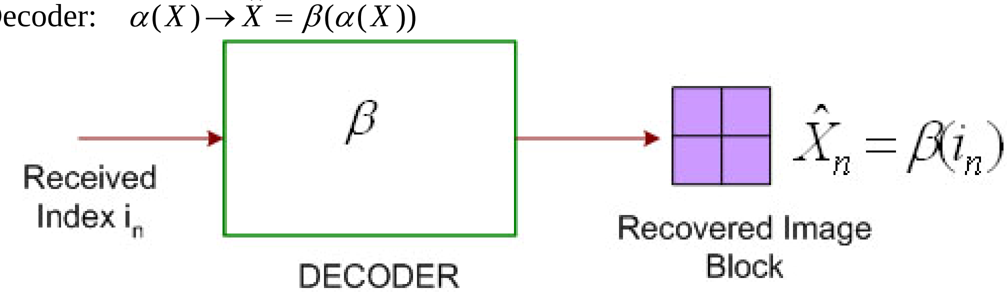
Encoder - Decoder Pair Memoryless Block Coder

Memoryless: We use information from the current and only current sample to code the signal.
Encoding and Decoding Operations:

Encoder: $X \xrightarrow{\alpha} i = \alpha(X)$ (3.1)



Decoder: $\alpha(X) \xrightarrow{\beta} \hat{X} = \beta(\alpha(X))$ (3.2)



Code Length, Rate:

Consider a binary vector i composed of $\{0,1\}$ we define the length of a binary vector as:

$$l(i) = \textit{length of the binary string } i$$

Example: $l(0) = 1;$ $l(101) = 3;$ $l(1011000) = 7$

Instantaneous rate of a binary vector i is defined by:

$$r(i) = \frac{i}{k} \quad \text{bits/symbol} \tag{3.3}$$

where k is the number of possible binary vectors, symbols.

The average rate (average codeword length) of the encoder applied to the source is defined by:

$$R = E[r(\alpha(X))] \quad \text{bits/symbol} \tag{3.4}$$

where E stands for average (expected value).

An encoder is "fixed-rate" or "fixed-length" if all the codewords sent to the channel has the same length:

$$l(i) = R.k \quad \textit{for all } i \tag{3.5}$$

Otherwise, it a "variable-rate" or "variable-length" coder.

Remarks: Selection of fixed or variable rate codes can have important implications in practice.

- Variable rate codes can cost more as they may require data buffering if the encoded data is to be transmitted over a fixed rate channel.
- They are harder to synchronize. Single errors in decoding, lost, or added bits on the channel can have catastrophic effects.
- Buffers can overflow (causing data loss) or underflow (causing wasted time or bandwidth).
- However, variable rate codes can provide superior compression performance. For example, in image compression we can use more bits for edges, fewer for background. In voice compression, more bits for plosives, fewer for vowels.

A source code is **invertible** or **noiseless** or **lossless** if

$$\beta(\alpha(x)) = x \tag{3.6}$$

which is the same as saying, the code is invertible if

$$\beta = \alpha^{-1} \tag{3.7}$$

A code is **lossy** if it is not lossless. In this case, a notion of **distortion *d*** between input vector and reconstructed replica has to be developed and it must be measured to quantify the seriousness of the loss.

Distortion Measure: It measures $d(X, \hat{X})$ loss resulting if an original input X is reproduced as $\hat{X}$ at the decoder. Mathematically, a distortion measure satisfies: $d(X, \hat{X}) \geq 0$. To be useful d should be:

- Easy to compute
- Easy to tract

- Meaningful for perception or application.

Types of Distortion: No single distortion measure accomplishes all of these goals, although the well-known **mean squared-error (MSE)** distortion (distance) is defined by:

$$d(X,\hat{X}) = \left\| X - \hat{X} \right\|^2 = \sum_{l=0}^{k-1} |x_l - \hat{x}_l \,| \qquad (3.8)$$

where $X = (x_0, x_1, \cdots x_{k-1})$ and $\hat{X} = (\hat{x}_0, \hat{x}_1, \cdots \hat{x}_{k-1})$ individual vectors. This measure satisfies the first two properties and occasionally accomplishes the third and most difficult.

Weighted or transform versions are used perceptual coding:

$$d(X,\hat{X}) = (X - \hat{X})^* B(X - \hat{X}) \qquad (3.9)$$

where "$*$" stands for complex conjugate of a complex vector and B_X is a matrix with some special characteristic, hopefully, correlating with perception. Note that if B_X is an identity matrix then the distortion measure reduces to the mean-square error (MSE) case of (3.8).

Mean absolute difference (MAE) distortion (distance) is defined by:

$$d(X,\hat{X}) = \left\| X - \hat{X} \right\| = \sum_{l=0}^{k-1} |x_l - \hat{x}_l \,| \qquad (3.8)$$

where $X = (x_0, x_1, \cdots x_{k-1})$ and $\hat{X} = (\hat{x}_0, \hat{x}_1, \cdots \hat{x}_{k-1})$ individual vectors. This measure more frequently used in image compression due to its computational simplicity and it also satisfies the first two properties.

Hamming Distortion or Distance:

$$d_H(X,\hat{X}) = \begin{cases} 0 & if \;\; X = \hat{X} \\ 1 & f \;\; X \neq \hat{X} \end{cases} \qquad (3.10)$$

Average Hamming Distance for vectors:

$$d_H(X,\hat{X}) = \sum_{l=0}^{k-1} d_H(x_l, \hat{x}_l) \qquad (3.11)$$

Common assumption: *d* is an ***additive*** distortion measure. We often normalize distortion with respect to magnitude, power or other factors.

Remarks: If $d(X,\hat{X}) = 0 \; if \;\; and \;\; only \;\; if \;\; X = \hat{X}$

- This clearly implies that zero distortion is equivalent to *lossless* coding. We normally make this assumption if the source is discrete.

- Performance of a data compression system is measured by the expected values of the distortion $d(X,\hat{X})$ and the associated rate *R*.

Units of rate will be:

- Bits/symbol for binary communication systems.
- Bits/sample for one-dimensional signals such as sampled speech.

- Bits/pixel for still imagery, and
- Bits/second for image sequence coding.

Example 3.1: 1101 and 0101 have a Hamming distance 1
1101 and 0001 have a Hamming distance 2.

The significance of the Hamming distance: When two binary vectors have Hamming distance *d*, then it would take *d* single-bit errors (i.e. inversions of bit values) to convert one into the other.

Example 3.2: Above three vectors corresponds to digits: 13, 5, and 1. The MSE difference for the first pair would be: $(13-5)^2=64$ and the second one is: $(13-1)^2=144$. Similarly, MAE values would be: 8 and 12, which would have very little value.

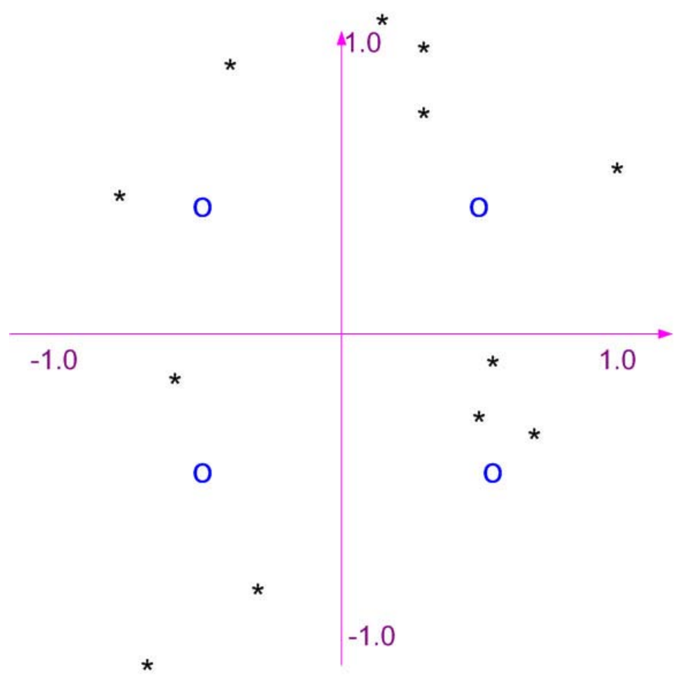
Example 3.3: Let us consider the following 12 vectors (stars) clustered in a 2-D space and suppose that we will represent them with four codewords (circles) located at (0.5,0.5), (-0.5,0.5), (-0.5,-0.5) and (0.5,-0.5). In doing so, we will select the circles in the same quarter-circle where the point lies. Let us compute the average MSE and MAE for this dataset.

Horizontal	-0.37	.63	-.83	-.70	-.28	1.09	.59	.14	.70	.30	.30	.37
Vertical	.98	-.11	.60	-1.21	-.94	.51	.17	1.76	-.35	.79	1.06	-.32
Circle Used	-.5;.5	.5;-.5	-.5;.5	-.5;-.5	-.5;-.5	.5;.5	.5;.5	.5;.5	.5;-.5	.5;.5	.5;.5	.5;-.5
Abs. Diff.	.13;.48	.13;.30	.33;.10	.20;.71	.22;.44	.59;.01	.09;.33	.36;1.26	.2;.15	.2;.29	.2;.56	.13;.18
ΣDiff. Sq.	.2473	0.171	.1189	.5441	.242	.3482	.117	.4777	.0625	.1241	.3536	.0493

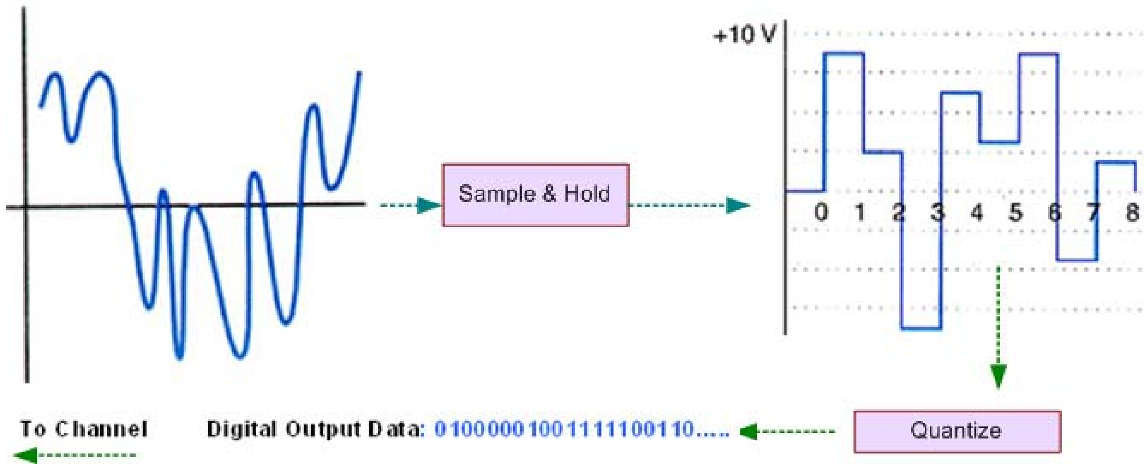
Average Absolute Difference= .13+.48+.33+.39+….+.18/12=0.72 units

MSE=.2473+.171+…+.0493/12=0.238 units

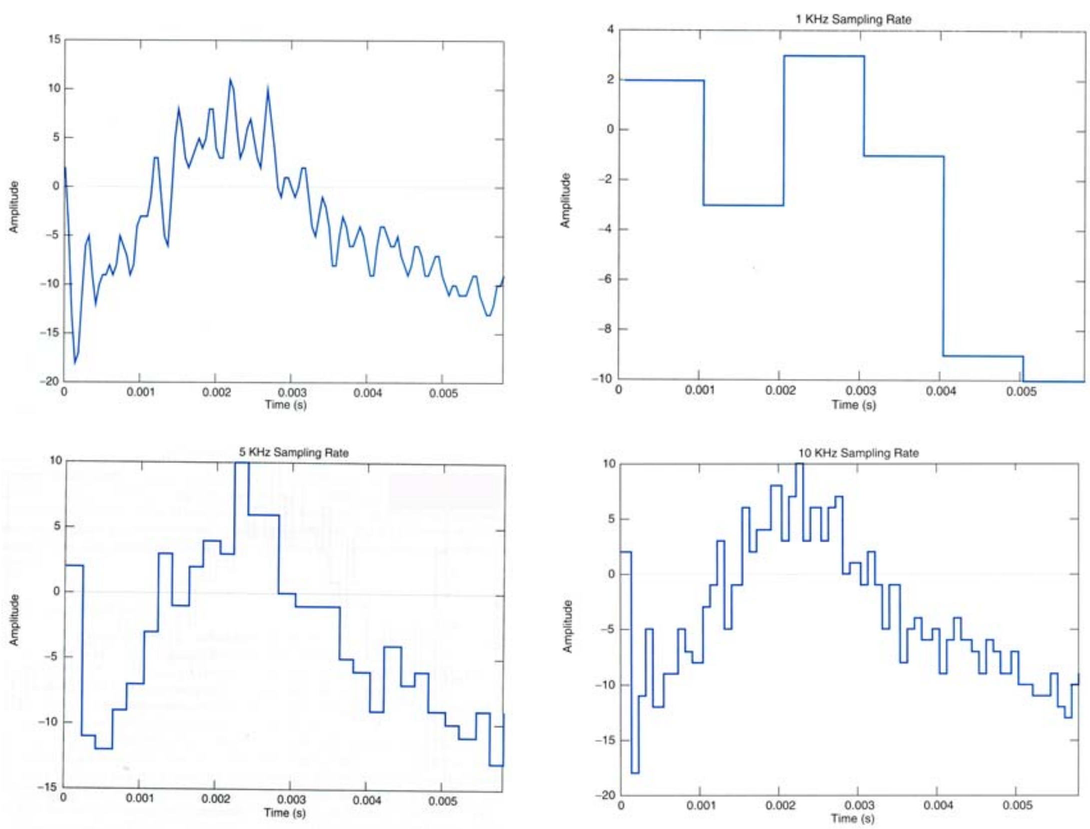
These would have significant if the samples were digitized values of imagery in a unit-size scale.



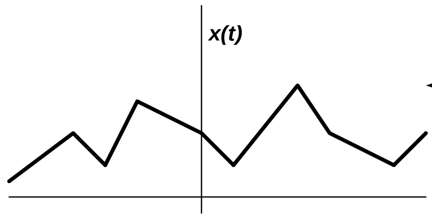
Example: 3.4 Sampling & PCM: From the information processing perspective key issue in the digital communication system is to have proper sampling and PCM rates. If we do not sample at the Nyquist rate or above distortion is unavoidable.



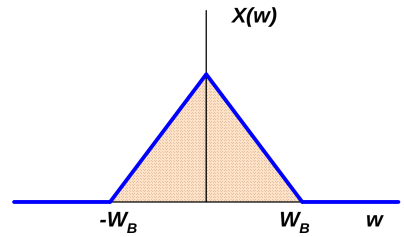
Encoding Stage: The effect of sampling rate on analog signals can be illustrated as shown below, where the first 5.0 ms segment of the sound "I" in the word "Information" spoken by a male speaker. Upper left is the original analog signal and the rest are the versions sampled at 1.0 KHz, 5.0 kHz, and 10.0 kHz, respectively. It is to see that the first two sampling rates have poor tracking capability of the original, whereas, the last figure has an envelope very close to that of the analog version.



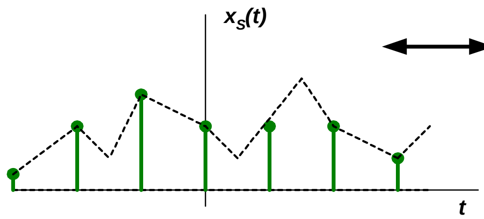
Decoding and Reconstruction Stage: It is commonly known as the D/A conversion and achieved by interpolating the discrete samples by use of an ideal Low-Pass filter with a bandwidth *B Hz*. This reconstruction process for an under-sampled (below Nyquist rate) can be illustrated as:



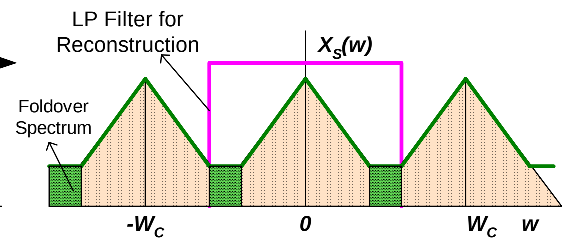
(a) Analog Signal



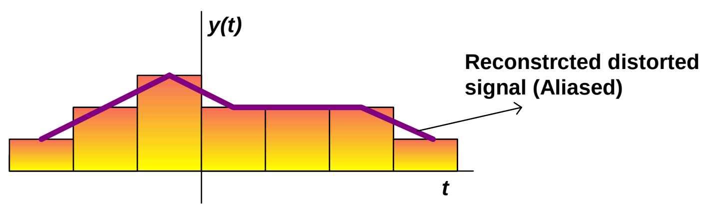
(b) Spectrum of analog signal in (a)



(c) Undersampled version of x(t) in (a).



(d) Spectrum of undersampled x(t).

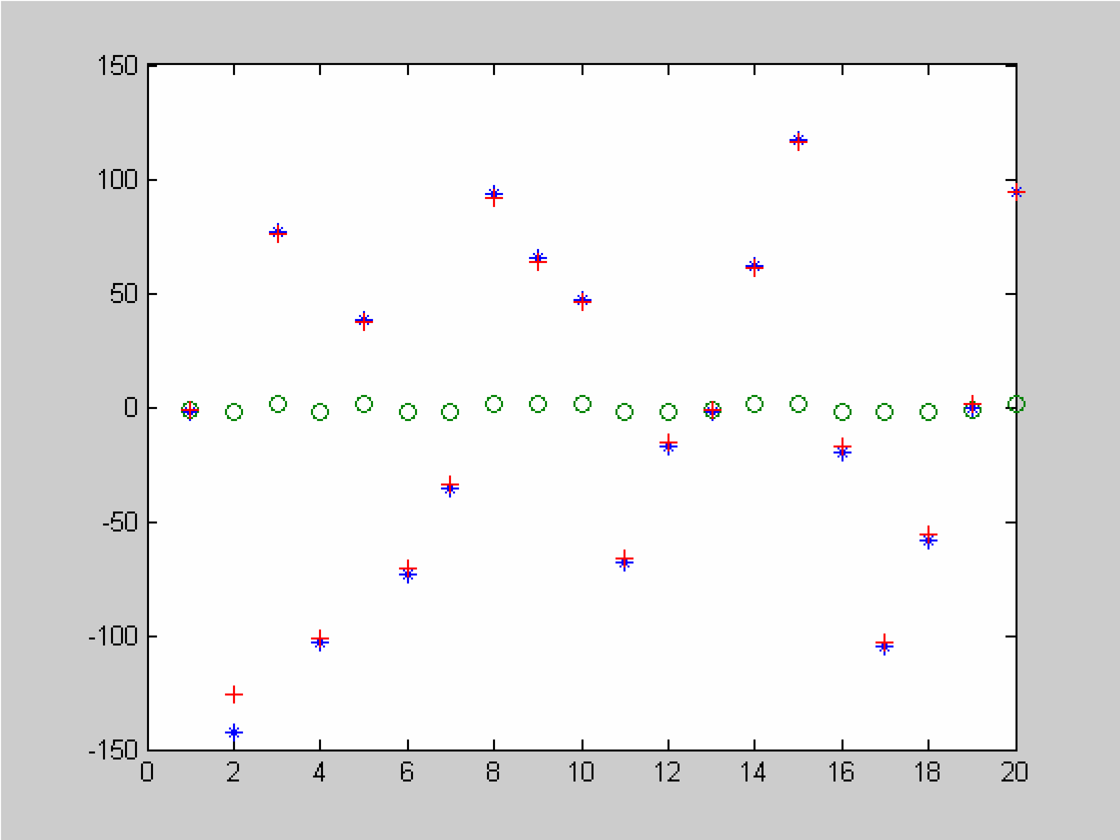


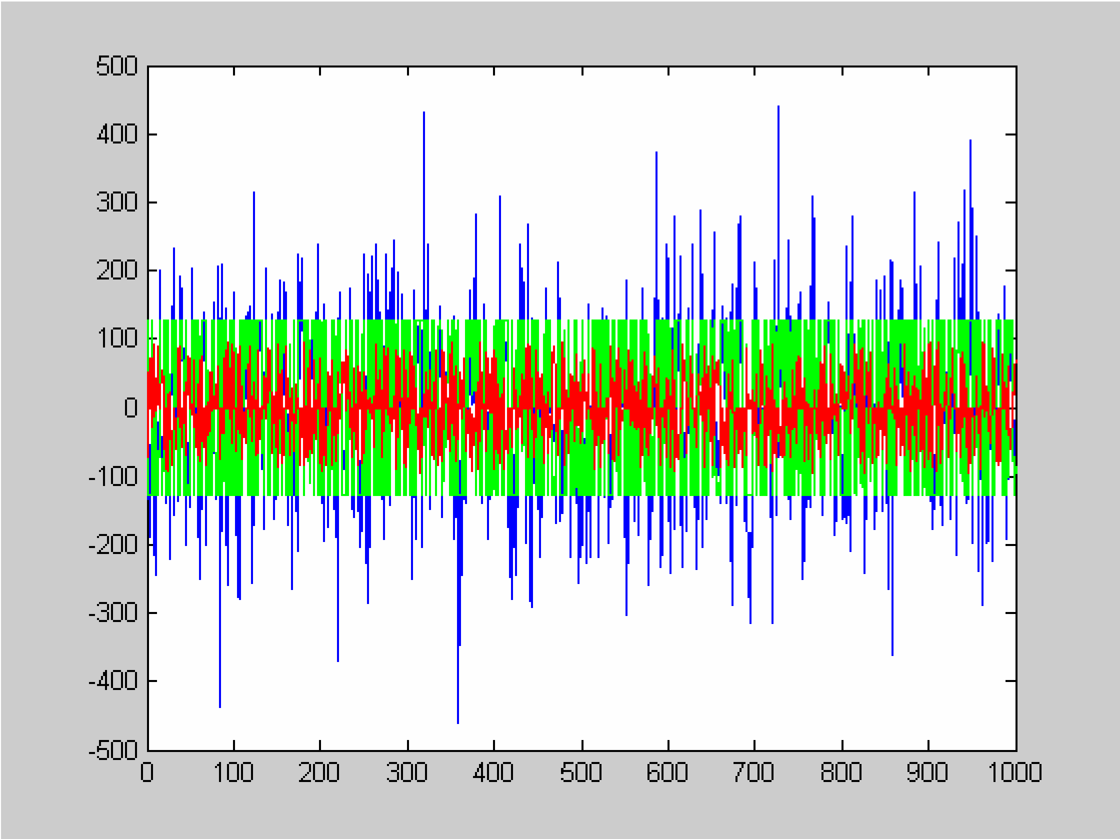
(e) Distorted output signal y(t).

Example: 3.4 Uniform PCM of random numbers

```
% PCM Codec for encoding random numbers or speech samples
%
a=randn(1,20); index=1:20;
a= a .* 64; disp (a)
a_quan=uencode(a,2,32,'signed'); disp (a_quan);
error=int8(a)-a_quan
plot(index,a,'*',index, a_quan,'o',index,error,'+')

% Repeat the example for 1000 points and 6-bits
a=randn(1,1000); index=1:1000;
a= a .* 128; disp (a)
a_quan=uencode(a,8,32,'signed'); disp (a_quan);
error=int8(a)-a_quan
figure;
plot(index,a,'b',index, a_quan,'g',index,error,'r')
```


150
100
50
0
-50
-100
-150
0
2
4
6
8
10
12
14
16
18
20


500
400
300
200
100
0
-100
-200
-300
-400
-500
0
100
200
300
400
500
600
700
800
900
1000