

EE3414

Multimedia Communication Systems - I

Source Coding Basics and Speech Coding

Yao Wang

Polytechnic University, Brooklyn, NY11201

<http://eeweb.poly.edu/~yao>

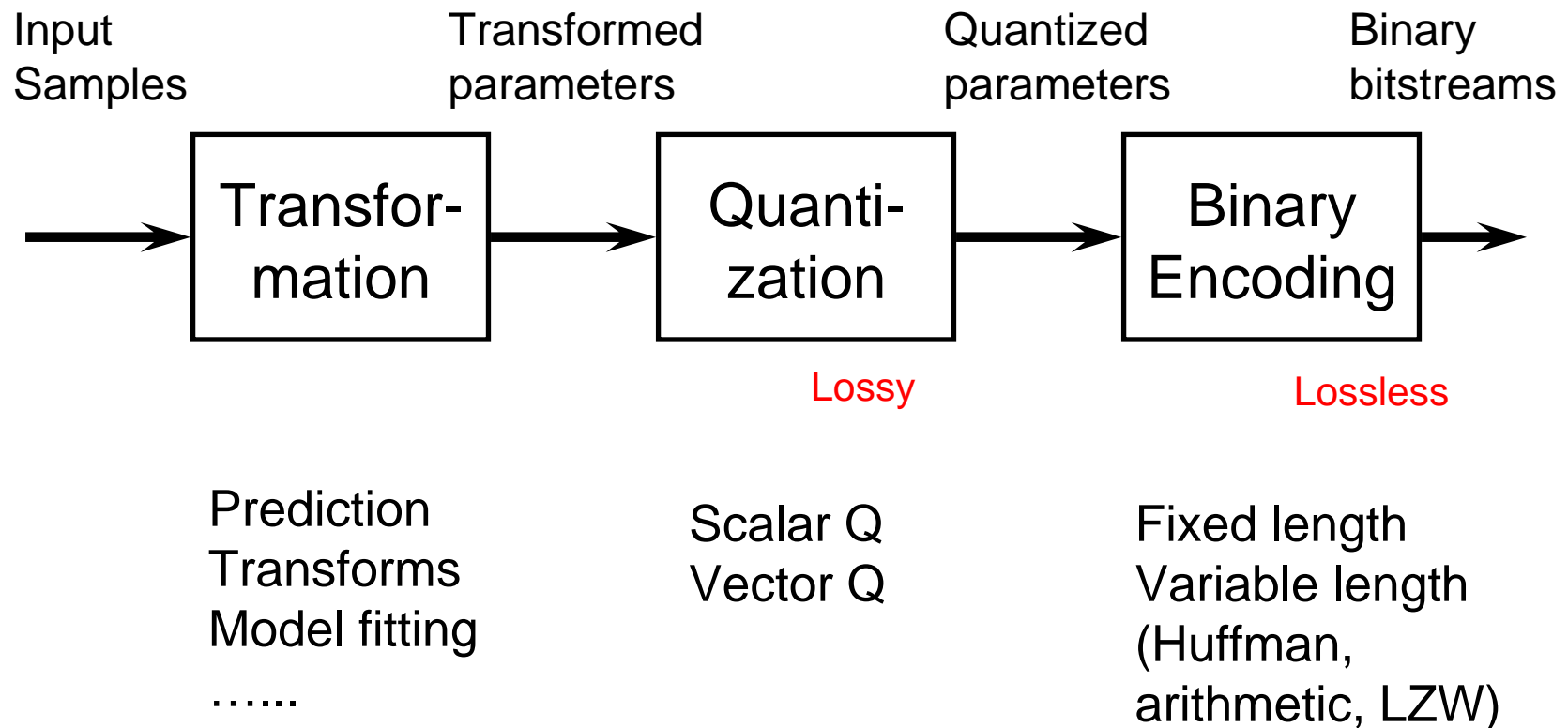
Outline

- Why do we need to compress speech signals
- Basic components in a source coding system
- Variable length binary encoding: Huffman coding
- Speech coding overview
- Predictive coding: DPCM, ADPCM
- Vocoder
- Hybrid coding
- Speech coding standards

Why do we need to compress?

- Raw PCM speech (sampled at 8 kbps, represented with 8 bit/sample) has data rate of 64 kbps
- Speech coding refers to a process that reduces the bit rate of a speech file
- Speech coding enables a telephone company to carry more voice calls in a single fiber or cable
- Speech coding is necessary for cellular phones, which has limited data rate for each user (≤ 16 kbps is desired!).
- For the same mobile user, the lower the bit rate for a voice call, the more other services (data/image/video) can be accommodated.
- Speech coding is also necessary for voice-over-IP, audio-visual teleconferencing, etc, to reduce the bandwidth consumption over the Internet

Basic Components in a Source Coding System



- Motivation for transformation ---
To yield a more efficient representation of the original samples.

Example Coding Methods

- “ZIP”: no transformation nor quantization, apply VLC (LZW) to the stream of letters (symbols) in a file directly, lossless coding
- PCM for speech: no transformation, quantize the speech samples directly using mu-law quantizer, apply fixed length binary coding
- ADPCM for speech: apply prediction to original samples, the predictor is adapted from one speech frame to the next, quantize the prediction error, error symbols coded using fixed length binary coding
- JPEG for image: apply discrete cosine transform to blocks of image pixels, quantize the transformed coefficients, code the quantized coefficients using variable length coding (runlength + Huffman coding)

Binary Encoding

- Binary encoding
 - To represent a finite set of symbols using binary codewords.
- Fixed length coding
 - N levels represented by $(\text{int}) \log_2(N)$ bits.
- Variable length coding (VLC)
 - more frequently appearing symbols represented by shorter codewords (Huffman, arithmetic, LZW=zip).
- The minimum number of bits required to represent a source is bounded by its entropy.

Entropy Bound on Bit Rate

- Shannon's Theorem:
 - A source with finite number of symbols $\{s_1, s_2, \dots, s_N\}$
 - Symbol s_n has a probability $\text{Prob}(s_n) = p_n$
 - If s_n is given a codeword with l_n bits, average bit rate (bits/symbol) is

$$\bar{l} = \sum p_n l_n;$$

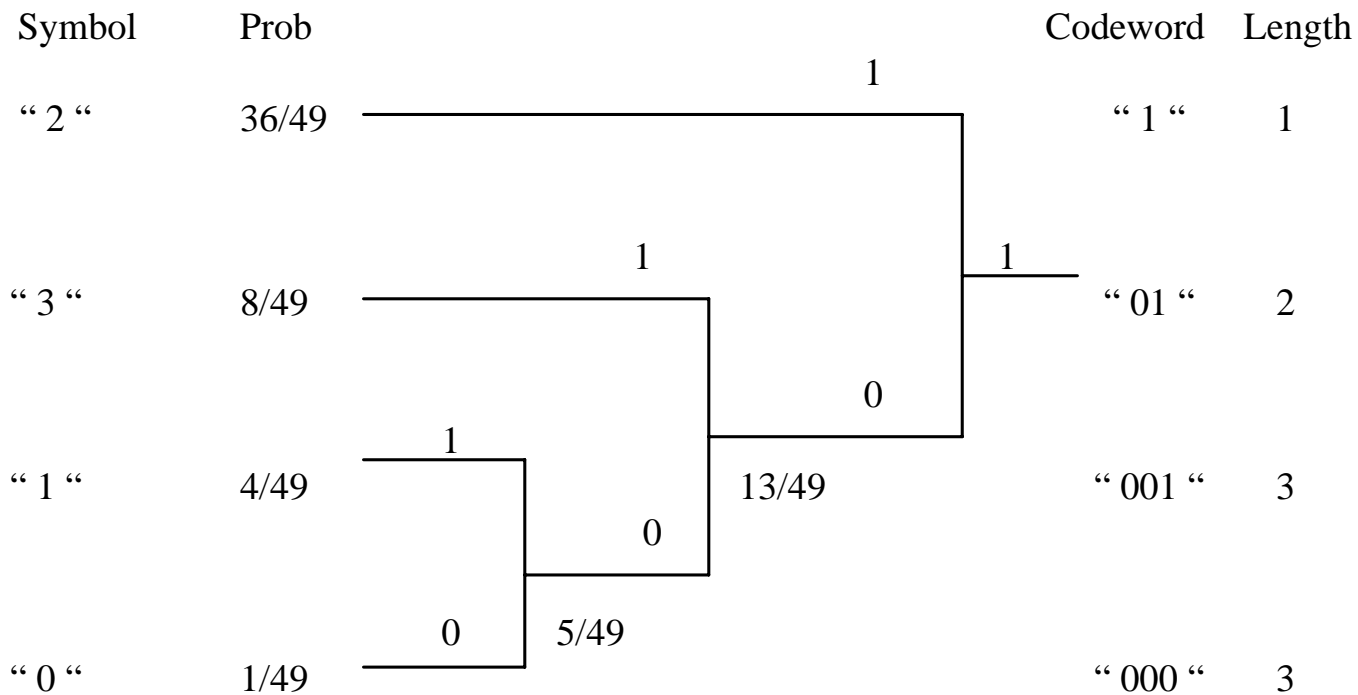
- Average bit rate is bounded by entropy of the source (H)

$$H \leq \bar{l} \leq H + 1;$$

$$H = -\sum p_n \log_2 p_n;$$

- For this reason, variable length coding is also known as entropy coding. The goal in VLC is to reach the entropy bound as closely as possible.

Huffman Coding Example



$$l = \frac{36}{49} \cdot 1 + \frac{8}{49} \cdot 2 + \left(\frac{4}{49} + \frac{1}{49}\right) \cdot 3 = \frac{67}{49} = 1.4; \quad H = -\sum p_k \log p_k = 1.16.$$

Huffman Coding Example Continued

- Code the sequence of symbols {3,2,2,0,1,1,2,3,2,2} using the Huffman code designed previously

- Code Table

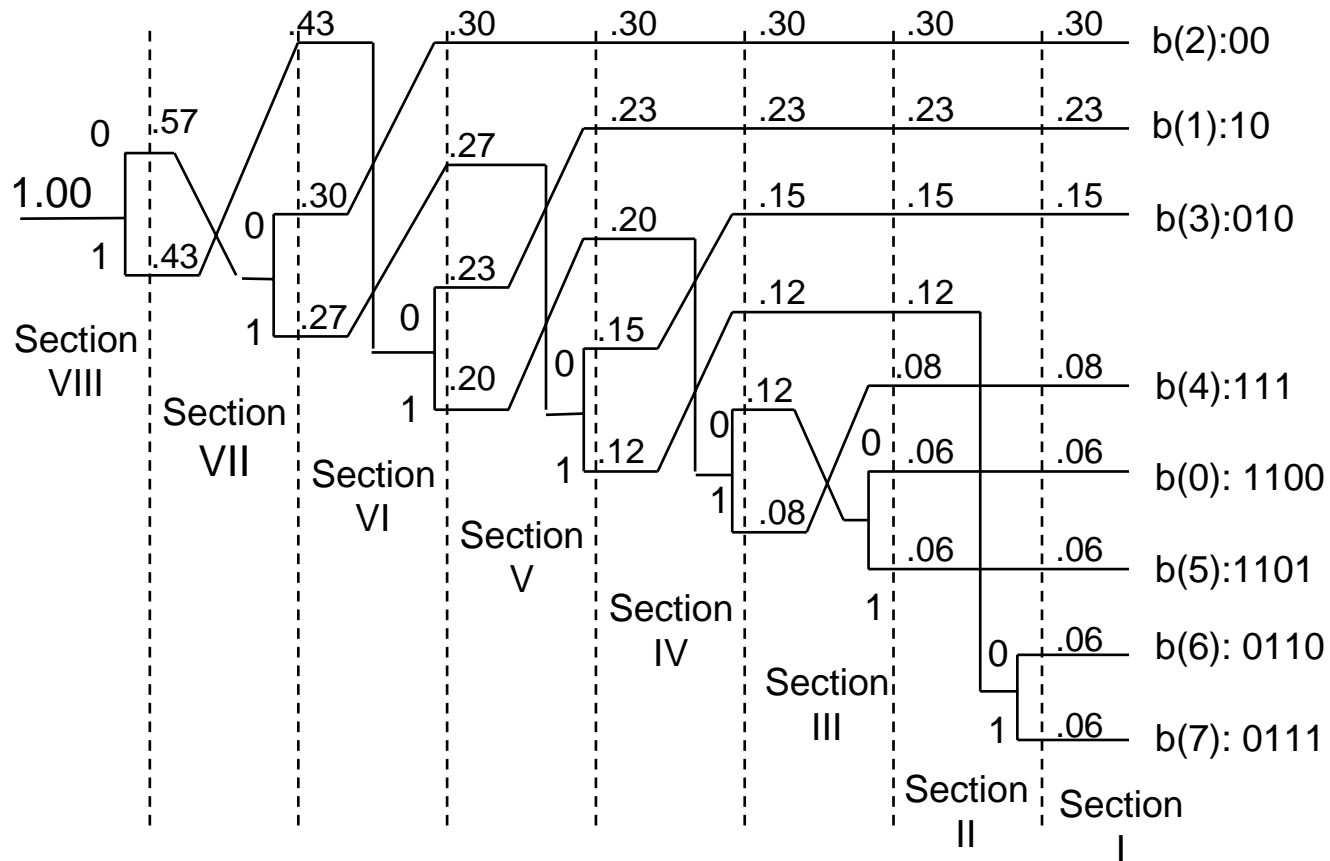
symbol	codeword
0	000
1	001
2	1
3	01

- Coded sequence: {01,1,1,000,001,001,1,01,1,1}
 - Average bit rate: $18 \text{ bits}/10 = 1.8 \text{ bits/symbol}$
 - Fixed length coding rate: 2 bits/symbol
 - Saving is more obvious for a longer sequence of symbols
- Decoding: table look-up

Steps in Huffman Code Design

- Step 1: arrange the symbol probabilities in a decreasing order and consider them as leaf nodes of a tree
- Step 2: while there are more than one node:
 - Find the two nodes with the smallest probability and assign the one with the lowest probability a “0”, and the other one a “1” (or the other way, but be consistent)
 - Merge the two nodes to form a new node whose probability is the sum of the two merged nodes.
 - Go back to Step 1
- Step 3: For each symbol, determine its codeword by tracing the assigned bits from the corresponding leaf node to the top of the tree. The bit at the leaf node is the last bit of the codeword

Another Example of Huffman Coding



$$\text{Average Bit Rate} = \sum_n p_n l_n = 2.71 \text{ bits}; \quad \text{Entropy} = \sum_n p_n \log_2 p_n = 2.68 \text{ bits}.$$

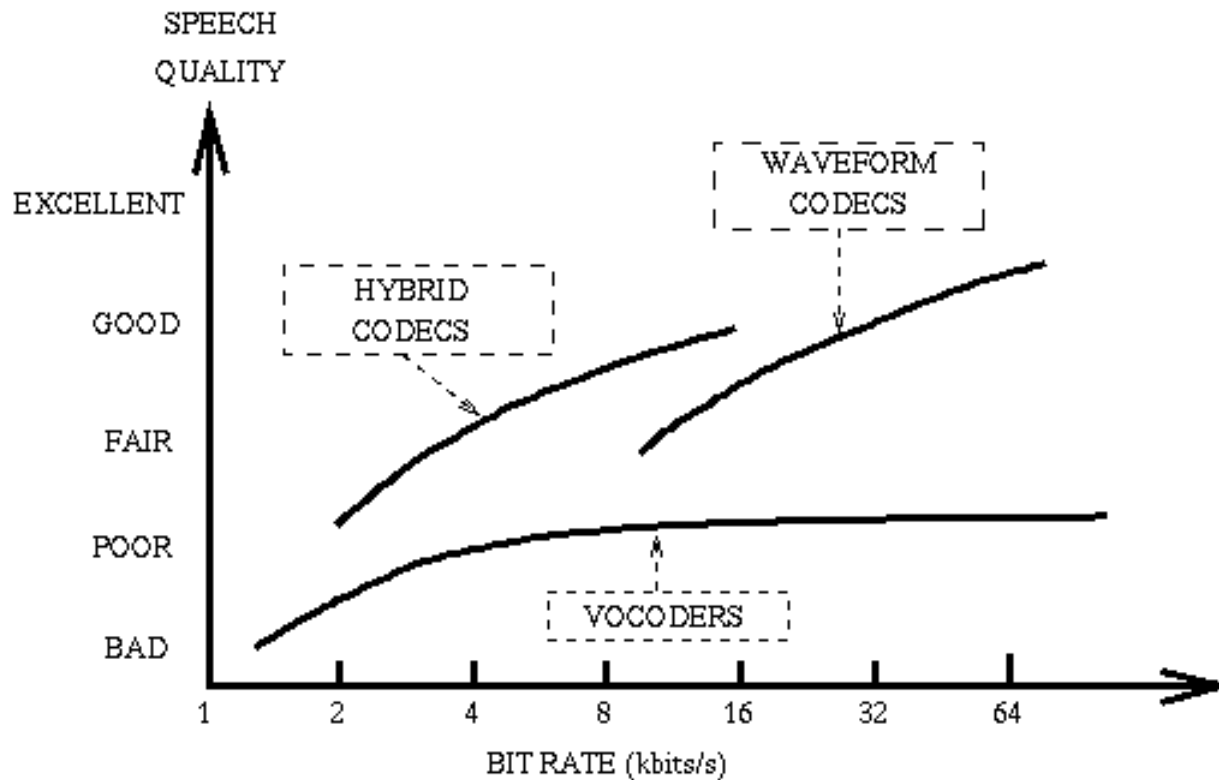
More on Huffman Coding

- Huffman coding achieves the upper entropy bound
- One can code one symbol at a time (scalar coding) or a group of symbols at a time (vector coding)
- If the probability distribution is known and accurate, Huffman coding is very good (off from the entropy by 1 bit at most).
- Adaptive Huffman coding: estimate the probability from the sequence on line
- Other lossless coding method:
 - Arithmetic coding: reaching the entropy lower bound more closely, but also more complex than Huffman coding, can adapt to change probability more easily
 - Lempel Ziv Welch (LZW): does not employ a probability table, universally applicable, but less efficient

Speech Coding

- Speech coders are lossy coders, i.e. the decoded signal is different from the original
- The goal in speech coding is to minimize the distortion at a given bit rate, or minimize the bit rate to reach a given distortion
- Figure of Merit
 - Objective measure of distortion is SNR (Signal to noise ratio)
 - SNR does not correlate well with perceived speech quality
- Speech quality is often measured by *MOS (mean opinion score)*
 - 5: excellent
 - 4: good
 - 3: fair
 - 2: poor
 - 1: bad
- PCM at 64 kbps with mu-law or A-law has MOS = 4.5 to 5.0

Speech Quality Versus Bit Rate For Common Classes of Codecs



From: J. Wooward, "Speech coding overview",
http://www-mobile.ecs.soton.ac.uk/speech_codecs

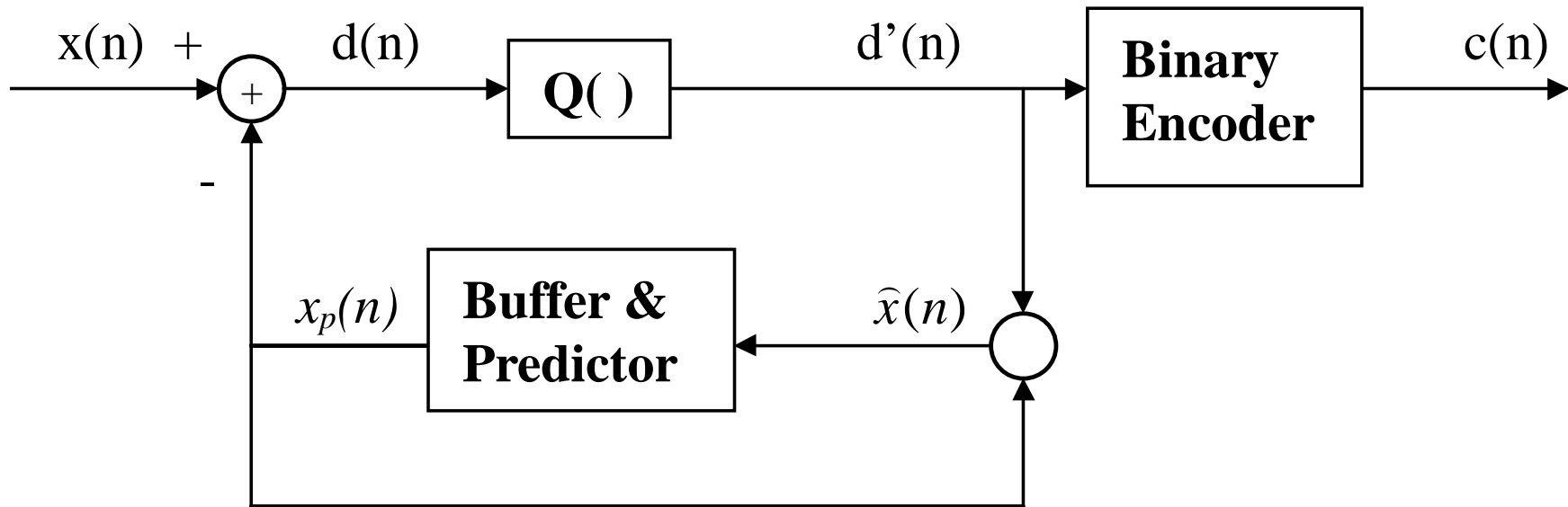
Predictive Coding (LPC or DPCM)

- Observation:
 - Adjacent samples are often similar
- Predictive coding:
 - Predict the current sample from previous samples, quantize and code the prediction error, instead of the original sample.
 - If the prediction is accurate most of the time, the prediction error is concentrated near zeros and can be coded with fewer bits than the original signal
 - Usually a linear predictor is used (linear predictive coding)

$$x_p(n) = \sum_{k=1}^P a_k x(n-k)$$

- Class of **waveform-based coders**
- Other names:
 - Non-predictive coding (uniform or non-uniform) -> PCM
 - Predictive coding -> Differential PCM or DPCM

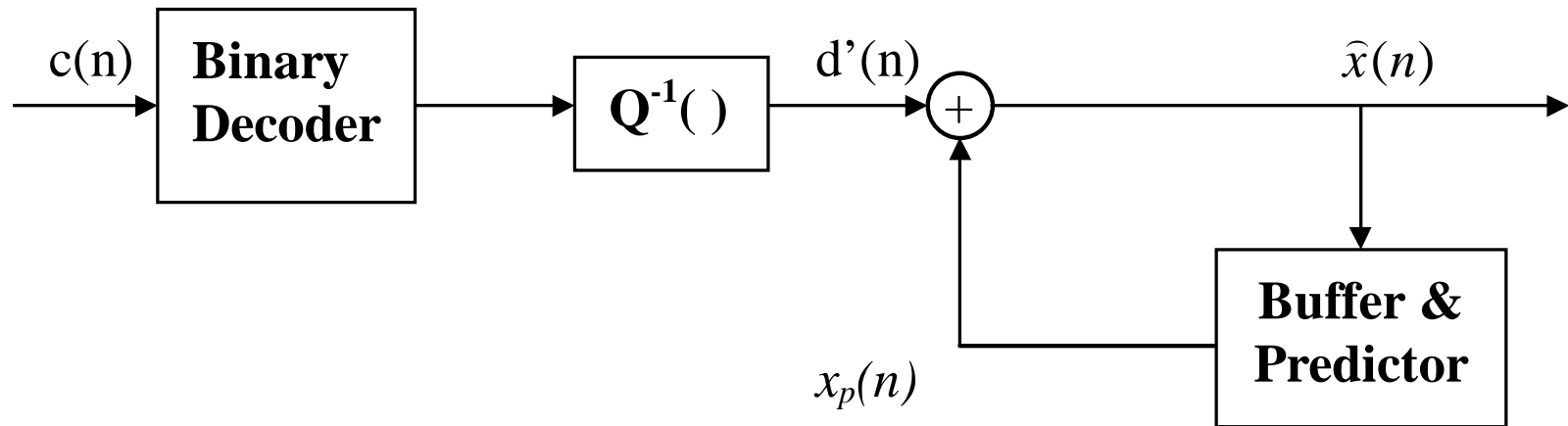
Encoder Block Diagram



$$x_p(n) = \sum a_k \hat{x}(n-k) \quad d(n) = x(n) - x_p(n)$$
$$\hat{x}(n) = x_p(n) + \hat{d}(n)$$

Note: the predictor is closed-loop: based on reconstructed (not original) past values

Decoder Block Diagram



Note: the decoder predictor MUST track the encoder predictor, hence BOTH based on reconstructed (not original) past values

Example

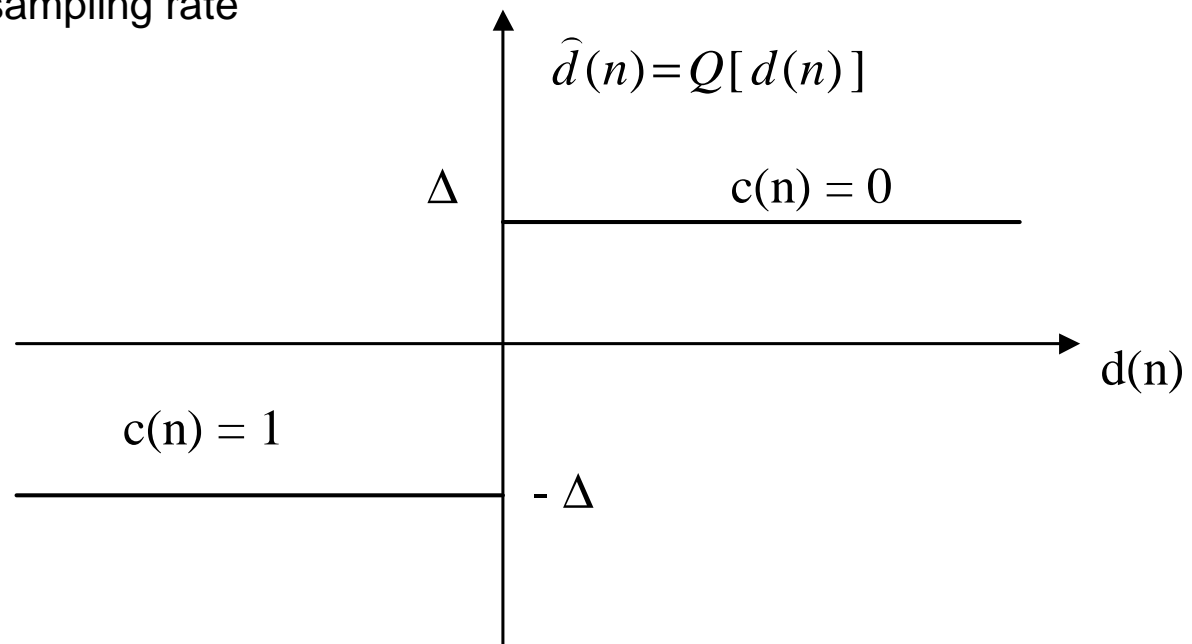
- Code the following sequence of samples using a DPCM coder:
 - Sequence: {1,3,4,4,7,8,6,5,3,1,...}
 - Using a simple predictor: predict the current value by the previous one

$$x_p(n) = a_1 x(n-1)$$

- Using a 3-level quantizer:
$$Q(d) = \begin{cases} 2 & d \geq 1 \\ 0 & |d| < 1 \\ -2 & d \leq -1 \end{cases}$$
- Show the reconstructed sequence
- For simplicity, assume the first sample is known.
- Show the coded binary stream if the following code is used to code the difference signal
 - Error “0” -> “1”, Error “2” -> “01”, Error “-2” -> “00”

Delta-Modulation (DM)

- Delta-Modulation
 - Quantize the prediction error to 2 levels, Δ and $-\Delta$.
 - Each sample takes 1 bit
 - Bit rate=sampling rate



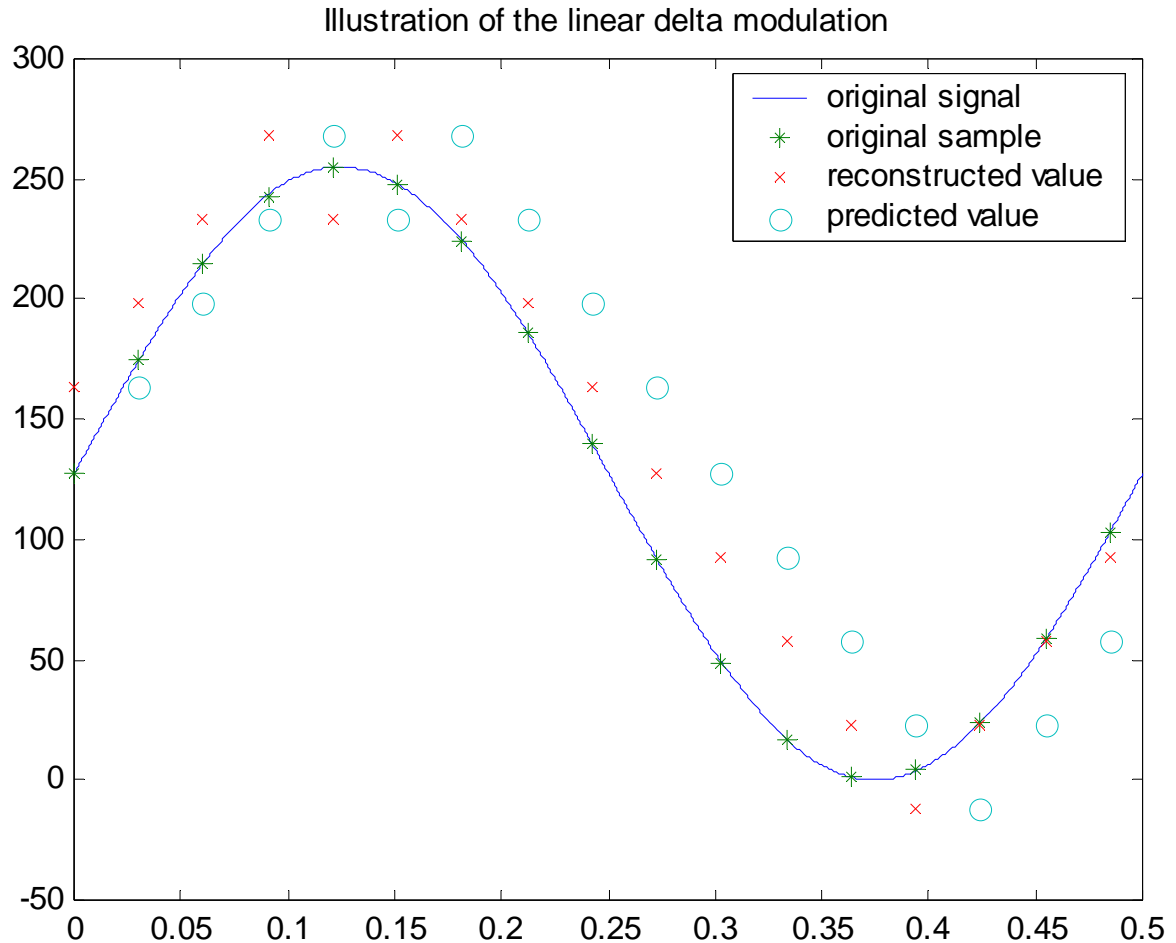
Example

- Code the following sequence of samples using a DM:
 - $\{1,3,4,4,7,8,6,5,3,1,\dots\}$
 - Using $\Delta = 2$, assuming the first sample is known
 - Show the reconstructed sequence
 - Show the coded binary sequence

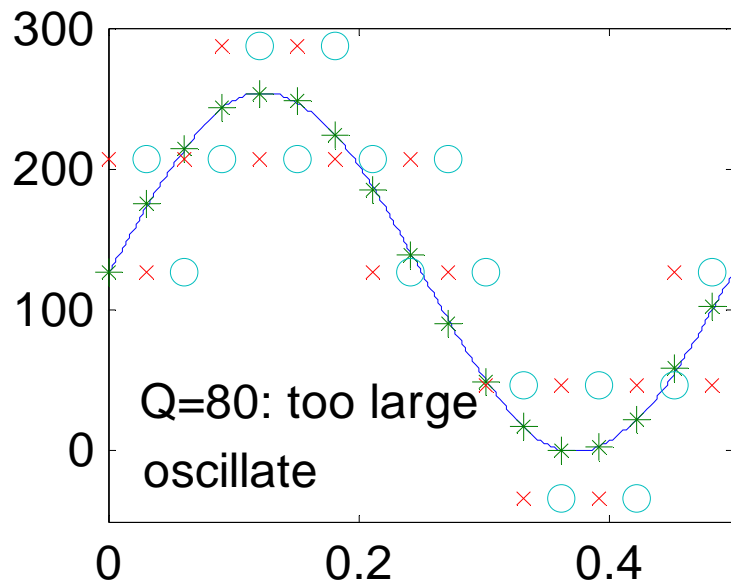
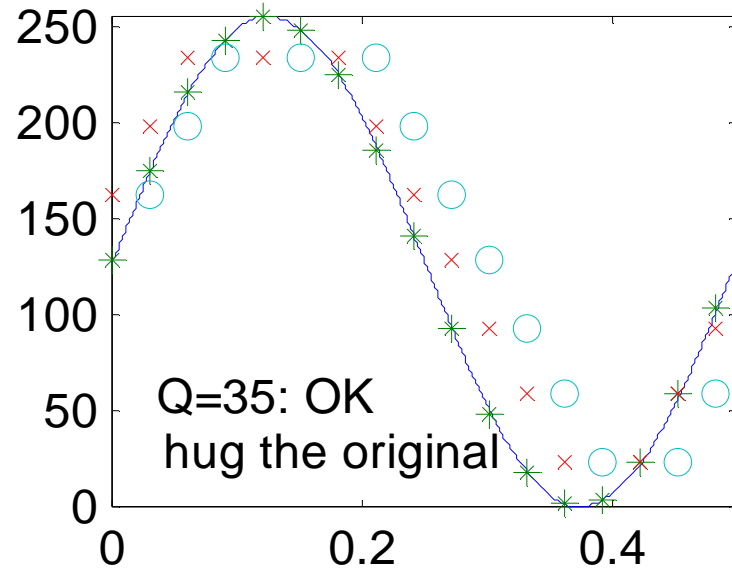
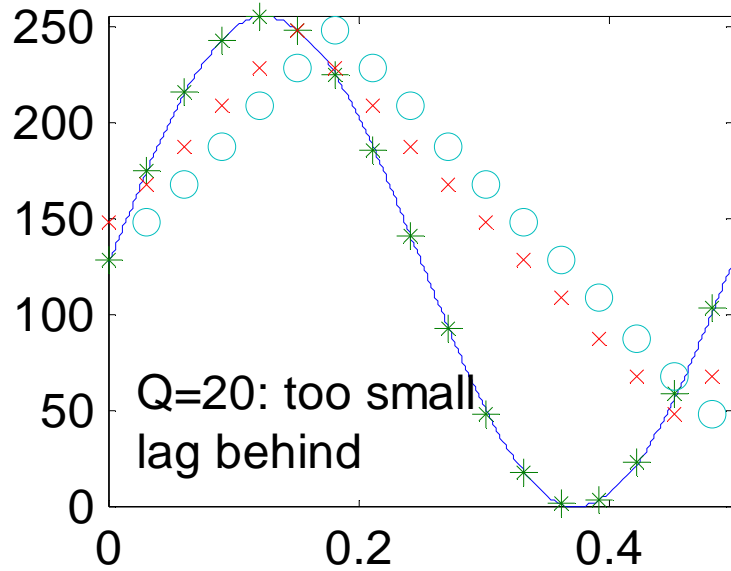
DM With a Simple Predictor

Predictor: predict the current pixel by the previous one

Stepsize: $\Delta=35$



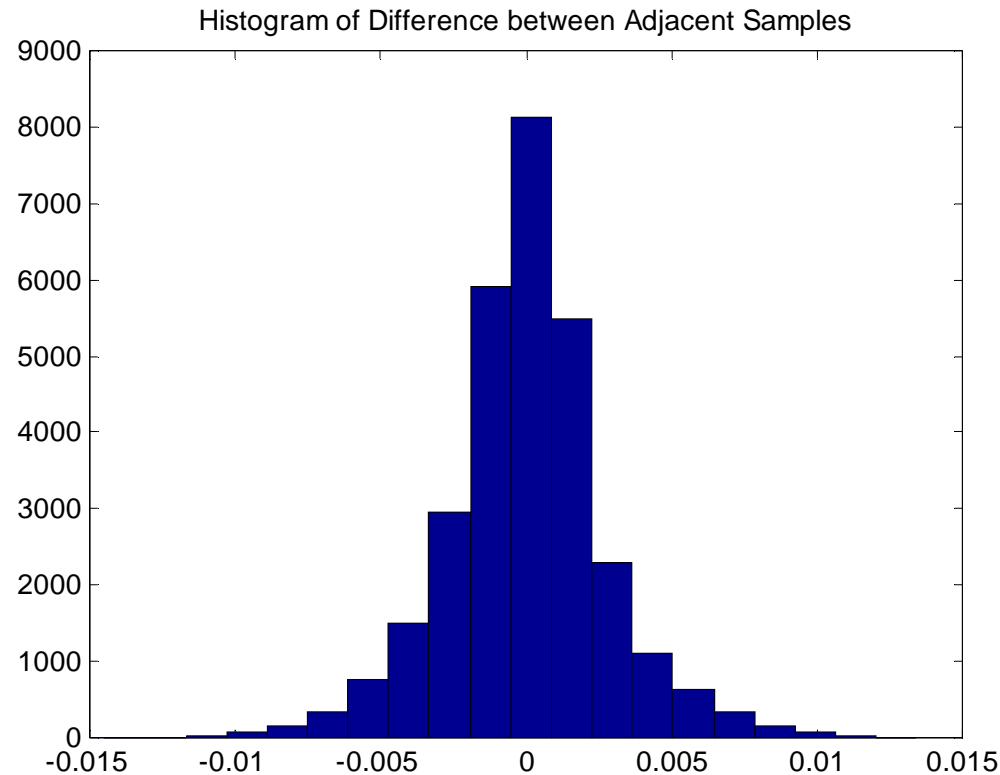
How to determine stepsize?



- * original value
- o predicted value
- x reconstructed value

demo_sindm.m

Selecting DM Parameters Based on Signal Statistics



```
L=length(x);  
d=x(2:L)-x(1:L-1);  
hist(d,20);
```

Δ should be chosen to be the difference value that has a relatively large percentage
In the above example, $\Delta = 0.002 \sim 0.003$

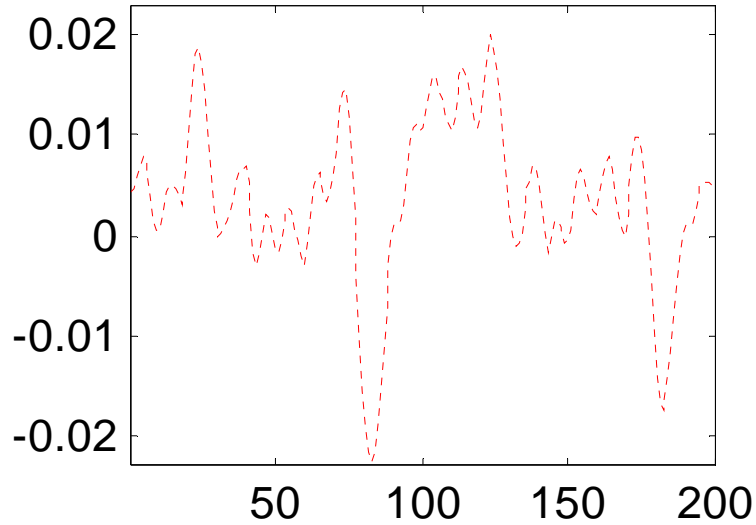
In adaptive DM, Δ is varied automatically so that the reconstructed signal hugs the original signal

Uniform vs. DM to Audio

Mozart_short.wav

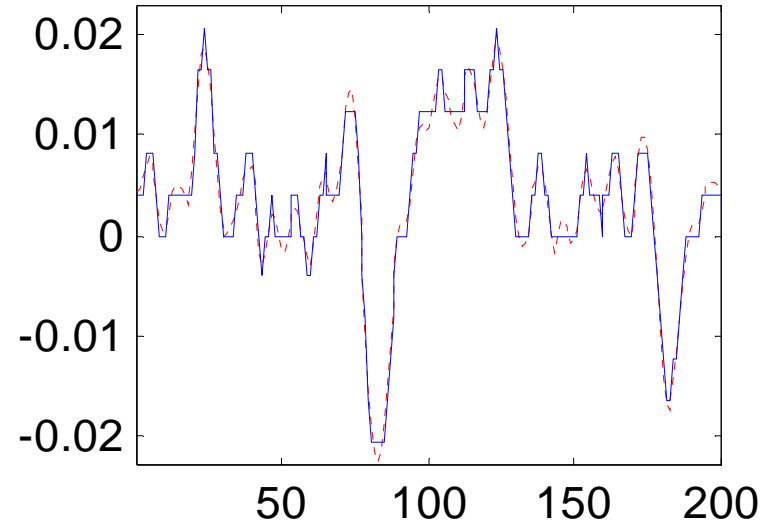


original, 705 Kbps

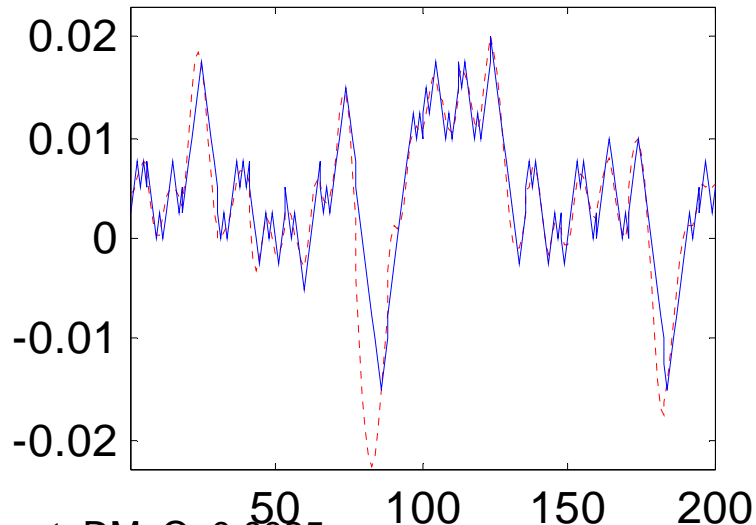


Mozart_q32_short.wav

Uniform Q=32, 220 Kbps

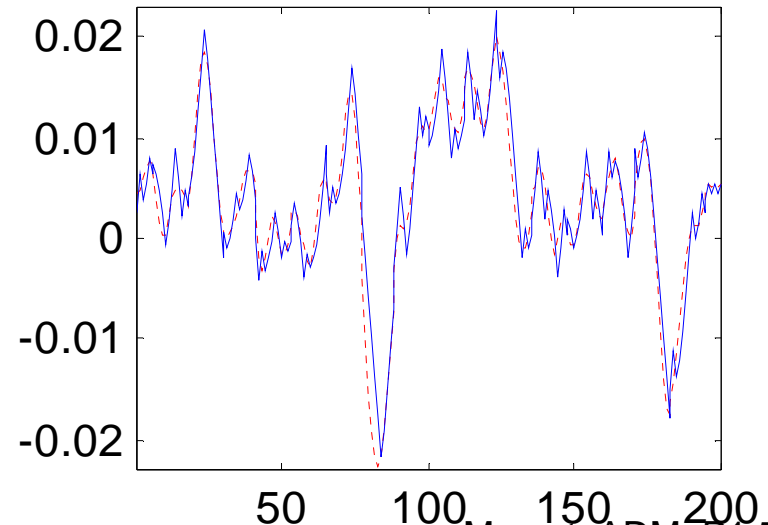


DM Q=0.0025, 44 Kbps



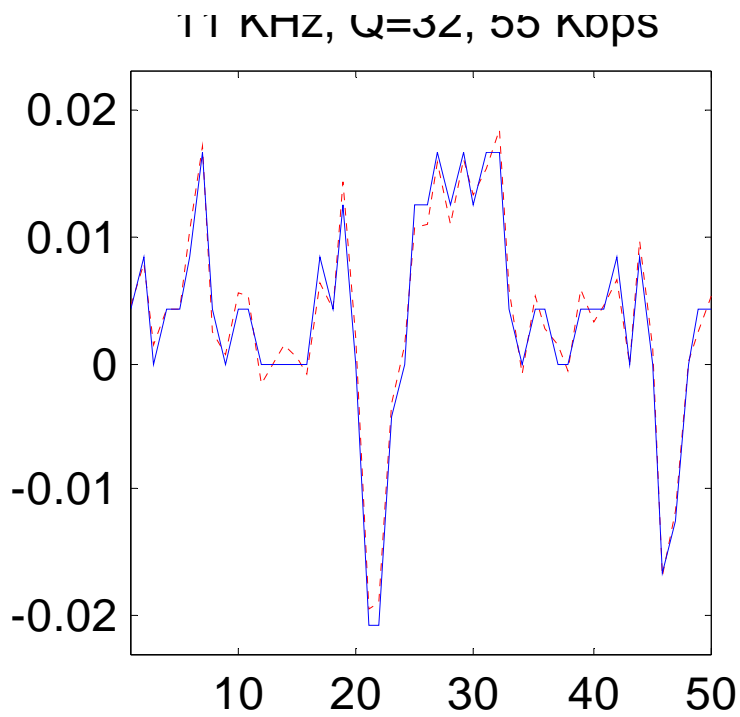
Mozart_DM_Q_0.0025.wav

ADM P=1.5, 44 Kbps

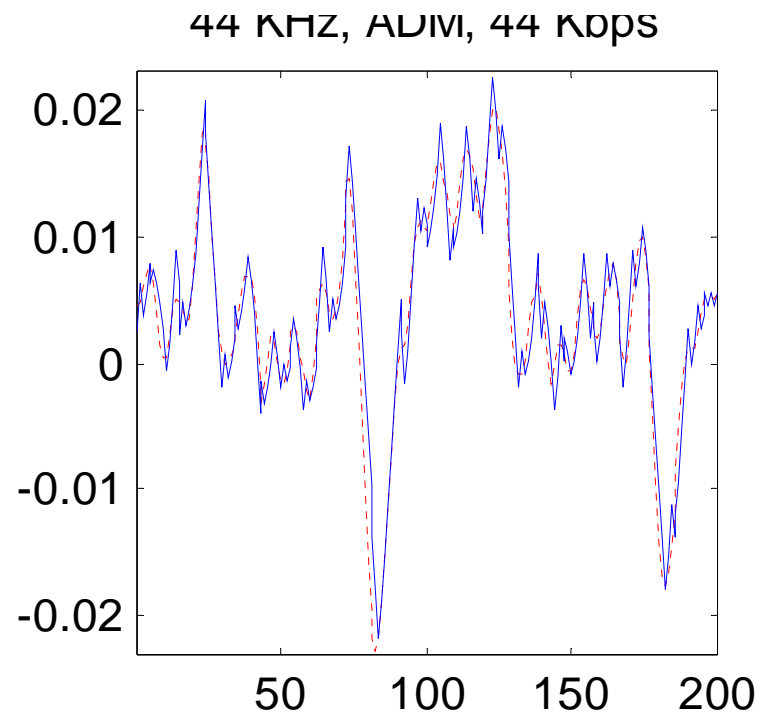


Mozart_ADM_P1.5.wav

Uniform vs. DM at Different Sampling Rates



Mozart_f_11_q32.wav



Mozart_ADM_P1.5.wav



Adaptive Predictive Coding (ADPCM)

- The optimal linear predictor for a signal depends on the correlation between adjacent coefficients.
- The short term correlation in a speech signal is time varying, depending on the sound being produced.
- To reduce the prediction error, the predictor should be adapted from frame to frame, where a frame is a group of speech samples, typically 20ms long (160 samples at 8 KHz sampling rate).
- The quantizer for the prediction error is also adapted.

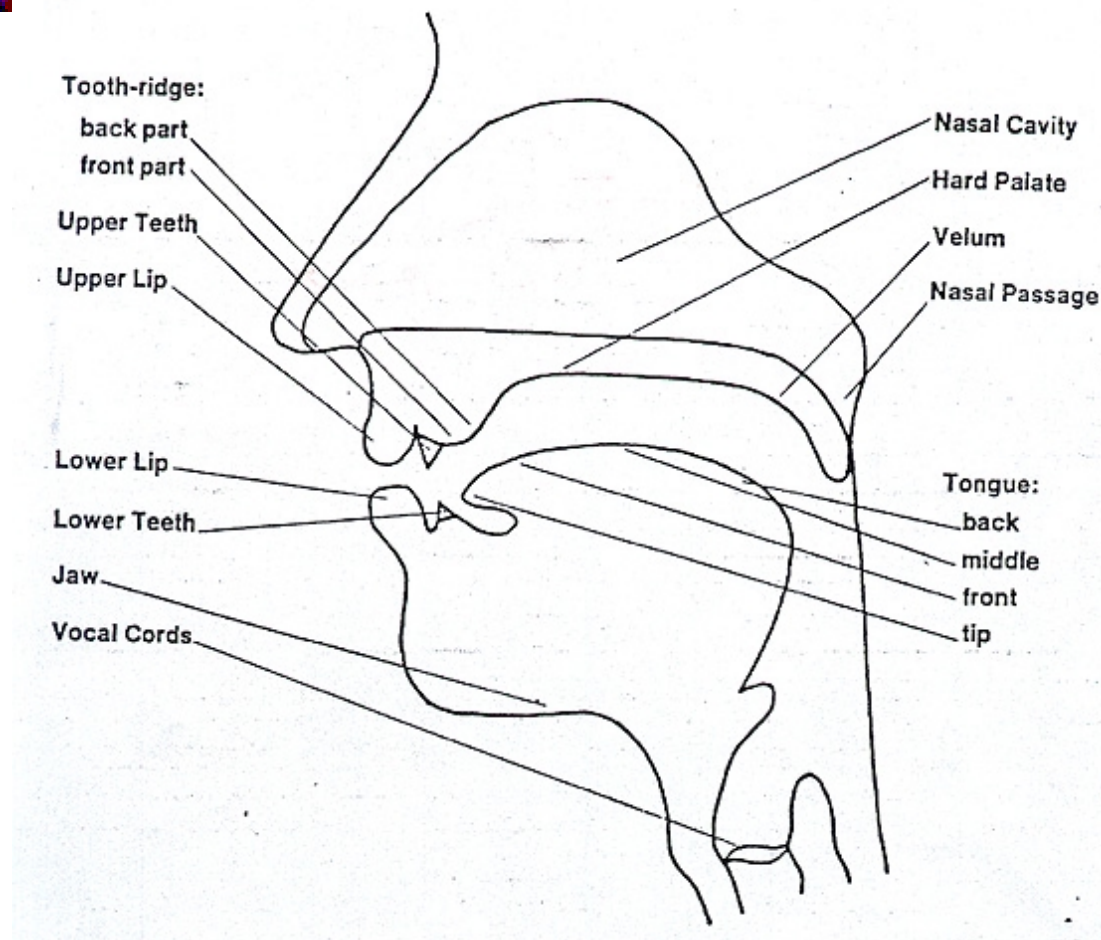
Forward vs. Backward Adaptation

- Forward adaptation
 - Before coding a frame of data, find the optimal predictor and quantizer to use based on this frame, the resulting predictor and quantizer information needs to be transmitted along with the prediction error information
- Backward adaptation
 - The predictor and quantizer are adapted based on the past data frame, the decoder does the same adaptation as the encoder.
 - The predictor and quantizer information does not need to be transmitted: “side information” data rate is lower!
 - But the prediction error is typically larger than forward adaptation!

How to Further Reduce the Bit Rate

- ADPCM cannot produce satisfactory quality when bit rate is lower than 16 Kbps
- To further reduce the bit rate, the speech production model should be exploited -> *model based coding* or *vocoder*
- Non-model-based methods are called *waveform-based coding* or *waveform coder*

The Organs of Speech



From: <http://www.phon.ox.ac.uk/~jcoleman/phonation.htm>

Speech Production Model

- Speech is produced when air is forced from the lungs through the vocal cords and along the vocal tract.
- Speech production in human can be crudely modeled as an excitation signal driving a linear system (modeled by a IIR filter)
- The filter coefficients depend on shape of the vocal tract, which changes when producing different sounds, by changing the positions of the tongue and jaw.
- Vowels are produced when the excitation signal is a periodic pulse with different periods
- Consonants are generated when the excitation signal is noise like

The Speech Model

- Vocal tract model
 - Short-term predictor

$$x_p(n) = \sum_{k=1}^P a_k x(n-k)$$

- Excitation model
 - Vocoder
 - Pitch pulse train / noise sequence
 - Analysis-by-synthesis coder
 - Long-term predictor

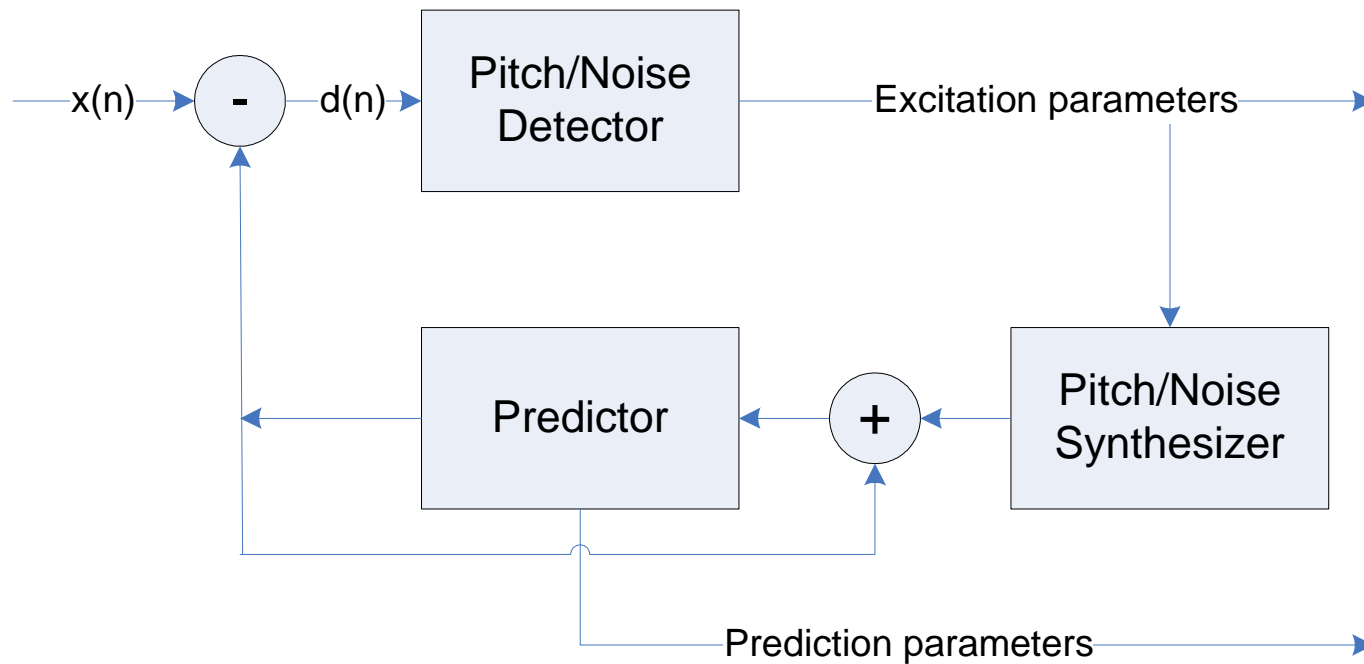
$$e_p(n) = b_p x(n-m)$$

- Optimal excitation sequence

Vocoder Principle

- For every short frame
 - Code the filter coefficient
 - Code the excitation signal
 - Send a indicator for “voiced” “unvoiced”
 - For “voiced” send the period
- Produce “unnatural” but “intelligible” sounds at very low bit rate (≤ 2.4 kbps)

Vocoder

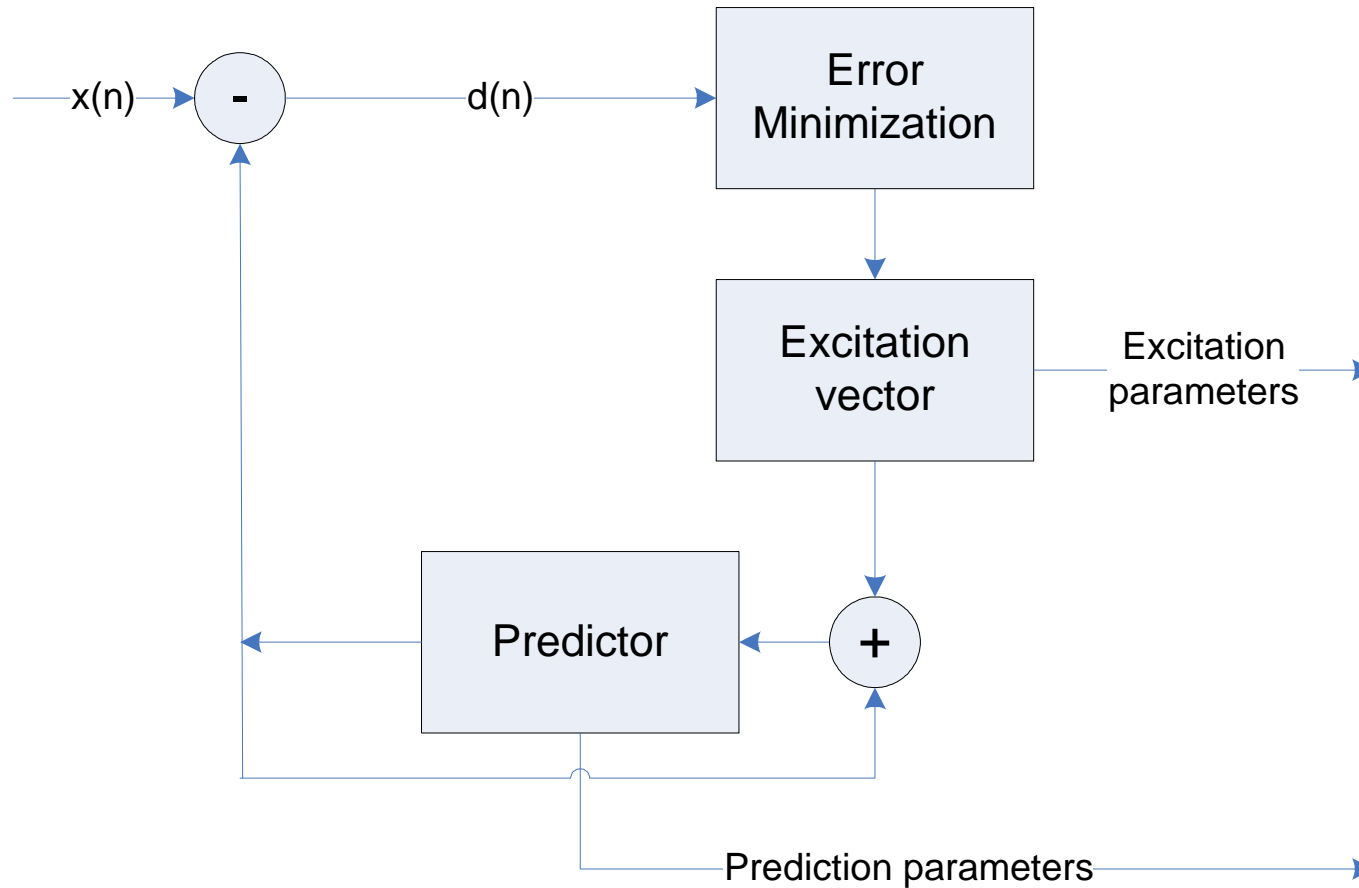


by S. Quackenbush

Hybrid Coder

- To produce more natural sounds, let the excitation signal be arbitrary, chosen so that the produced speech waveform matches with the actual waveform as closely as possible
- Hybrid coder: code the filter model plus the excitation signal as a waveform
- Code-excited linear prediction (CELP) coder: choose the excitation signal from codewords in a predesigned codebook
- This principle leads to acceptable speech quality in the rate range 4.8-16 kbps, used in various wireless cellular phone systems

Hybrid Coder

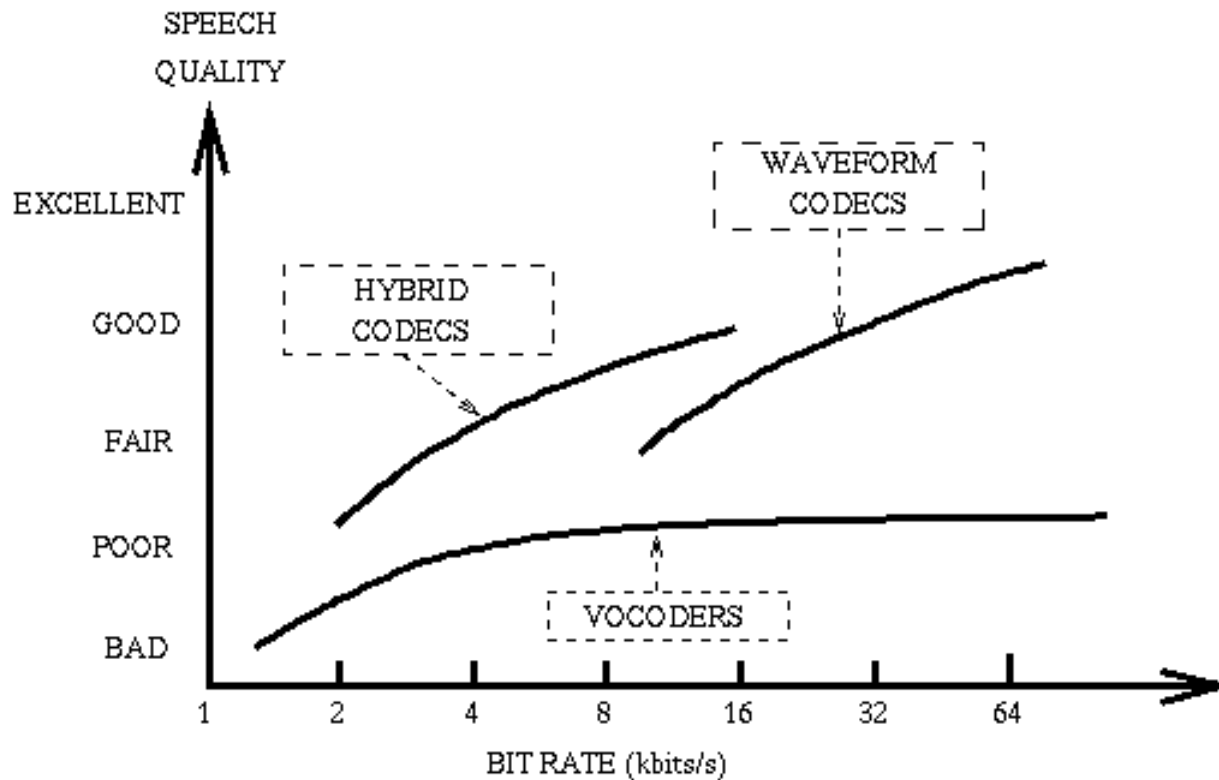


by S. Quackenbush

Speech Coding Standards





- Toll quality speech coder (digital wireline phone)
 - G.711 (A-LAW and μ -LAW at 64 kbits/sec)
 - G.721 (ADPCM at 32 kbits/ sec)
 - G.723 (ADPCM at 40, 24 kbps)
 - G.726 (ADPCM at 16,24,32,40 kbps)
- Low bit rate speech coder (cellular phone/IP phone)
 - G.728 low delay (16 Kbps, delay <2ms, same or better quality than G.721)
 - G. 723.1 (CELP Based, 5.3 and 6.4 kbits/sec)
 - G.729 (CELP based, 8 bps)
 - GSM 06.10 (13 and 6.5 kbits/sec, simple to implement, used in GSM phones)

Speech Quality Versus Bit Rate For Common Classes of Codecs



From: J. Wooward, "Speech coding overview",
http://www-mobile.ecs.soton.ac.uk/speech_codecs

Demonstration of speech quality with different standards

-  PCM, mu-law, 64 kbps
-  GSM 06.10 vocoder, 13 kbps
-  FED-STD-1016 CELP at 4.8 kbps
-  FED-STD-1015 LPC-10 at 2.4 kb/s

Speech files downloaded from

<http://people.qualcomm.com/karn/voicedemo/>

What Should You Know

- Huffman coding:
 - Given a probability table, can construct a Huffman code and apply it to a string of symbols for coding. Can also decode from the bit stream
 - Understand the general idea of variable length coding
- Predictive coding
 - Understand the predictive coding process (the encoder and decoder block diagram)
 - Understand how the simple predictive coder (including delta modulator) works and can apply it to a sequence
 - Understand the need for adaptation of both the predictor and the quantizer, and why ADPCM improves performance
- Vocoder and hybrid coding
 - Know the principle of model-based coding, understand the concept of vocoder and hybrid speech coder
- Standards for speech coding
 - Know why do we need different standards and difference between them

References

- Yao Wang, EL514 Lab Manual, Exp3: Speech and audio compression, Sec. 2.2, 2.3 (copies provided)
- Z. N. Li and M. Drew, Fundamentals of multimedia, Prentice Hall, 2004. Sec. 6.3 (Quantization and transmission of audio) and Chap.13 (Basic audio compression techniques)
- A. Spanias, “Speech coding: a tutorial review,” Proc. IEEE, vol. 82, pp 1541-1582, 1994. (and several other articles in this issue)
- “Speech Coding Overview” Jason Woodard, http://www-mobile.ecs.soton.ac.uk/speech_codecs/
- “Lossless compression algorithms”, <http://www.cs.sfu.ca/CourseCentral/365/D1/2003-1/material/notes/Chap4/Chap4.1/Chap4.1.html>
- The Lossless Compression (Squeeze) Page, by Dominik Szopa, <http://www.cs.sfu.ca/CourseCentral/365/li/squeeze/index.html>
 - Has good demo/applets for Huffman coding and LZ coding