International Conference on
Information, Communications,
and Signal Processing
Singapore
9 September 1997

Fundamentals of Data Compression

Robert M. Gray
Information Systems Laboratory
Department of Electrical Engineering
Stanford University
rmgray@stanford.edu
http://www-isl.stanford.edu/~gray/

Part I Introduction and Overview

Part II Lossless Compression

Part III Lossy Compression

Part IV "Optimal" Compression

Closing Observations

I. Introduction and Overview

Signal compression: motivation

World is increasingly digital:

- Digitally acquired signals
- Analog signals converted to digital

Why??

Digital signals (images, speech, etc.) can

- be stored (saved, archived) in a digital medium (computer disk)
- be transmitted locally or remotely over a digital communications link (telecommunications, T1, optical fiber)
- be archived in public databases (ftp, Web)

- be processed by computer:
  - computer assisted diagnosis/decision
  - automatic search for specified patterns
  - context searches
  - highlight or mark interesting suspicious regions
  - focus on specific regions of image intensity
  - statistical signal processing
    * enhancement/restoration
    * denoising
    * classification/detection
    * regression/estimation/prediction
    * feature extraction/pattern recognition
    * filtering

Problem: Digital image files are large
Massive quantities of data can overwhelm resources.

- low-resolution, TV quality, color video image: 512×512 pixels/color, 8 bits/pixel, and 3 colors $\Rightarrow \approx 6 \times 10^6$ bits
- 24×36 mm (35-mm) negative photograph scanned at 12μm: 3000×2000 pixels/color, 8 bits/pixel, and 3 colors $\Rightarrow \approx 144 \times 10^6$ bits
- 14×17 inch radiograph scanned at 70μm: 5000×6000 pixels, 12 bits/pixel $\Rightarrow \approx 360 \times 10^6$ bits. Medium size hospital generates terrabytes each year.
- Scanning a modest sized Durer print at 12bpp, 2400dpi, produces a file of over 40Mbytes.

- LANDSAT Thematic Mapper scene: 6000×6000 pixels/spectral band, 8 bits/pixel, and 6 nonthernmal spectral bands $\Rightarrow \approx 1.7 \times 10^9$ bits

Solution?

DATA COMPRESSION

Compression required for efficient **transmission**

- send more data in available bandwidth
- send the same data in less bandwidth
- more users on same same bandwidth

and **storage**

- can store more data
- can compress for local storage, put details on cheaper media

Also useful for progressive reconstruction, scalable delivery, browsing

and as a front end to other signal processing.

Future: Combine compression and subsequent user-specific processing.

General types of compression

Lossless compression

noiseless coding, lossless coding, invertible coding, entropy coding, data compaction.

- Can perfectly recover original data (if no storage or transmission bit errors).
- Variable length binary codewords (or no compression)
- Only works for *digital sources*.

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

Morse code Consider dots and dashes as binary.

Chose codeword length inversely proportional to letter relative frequencies

Huffman code 1952, in unix *compact* utility and many standards

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, DOS, GIF

lots of patent suits

arithmetic codes Fano, Elias, Rissanen, Pasco

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
Can do better on specific data types

Lossy compression

Not invertible, information lost.

Disadvantage: Lose information and quality, but if have enough (?) bits, loss is invisible.

Advantage: Greater compression possible. (6:1 to 16:1 typical, 80:1 to 100:1 promising in some applications)

Design Goal: Get "best" possible quality for available bits.

Big issue: is "best" good enough?

Examples: DPCM, ADPCM, transform coding, LPC, H.26*, JPEG, MPEG, EZW, SPHIT, CREW, StackRun

In between: "perceptually lossless"
(Perhaps) need not retain accuracy past

- what the eye can see (which may depend on context),
- the noise level of the acquisition device,
- what can be squeezed through a transmission or storage channel, i.e., an imperfect picture may be better than no picture at all.

This can be controversial (fear of lurking lawyers if misdiagnose using "lossy" reproduction, even if "perceptually lossless")

Is lossy compression acceptable?

NO in some applications: Computer programs, bank statements, espionage

Thought necessary for Science and Medical Images.

Is it??

Loss may be unavoidable, may have to choose between imperfect image or no data at all (or long delays).

Loss not a problem with

- Followup studies
- Archives
- Research and Education
- Progressive reconstruction.
- Entertainment

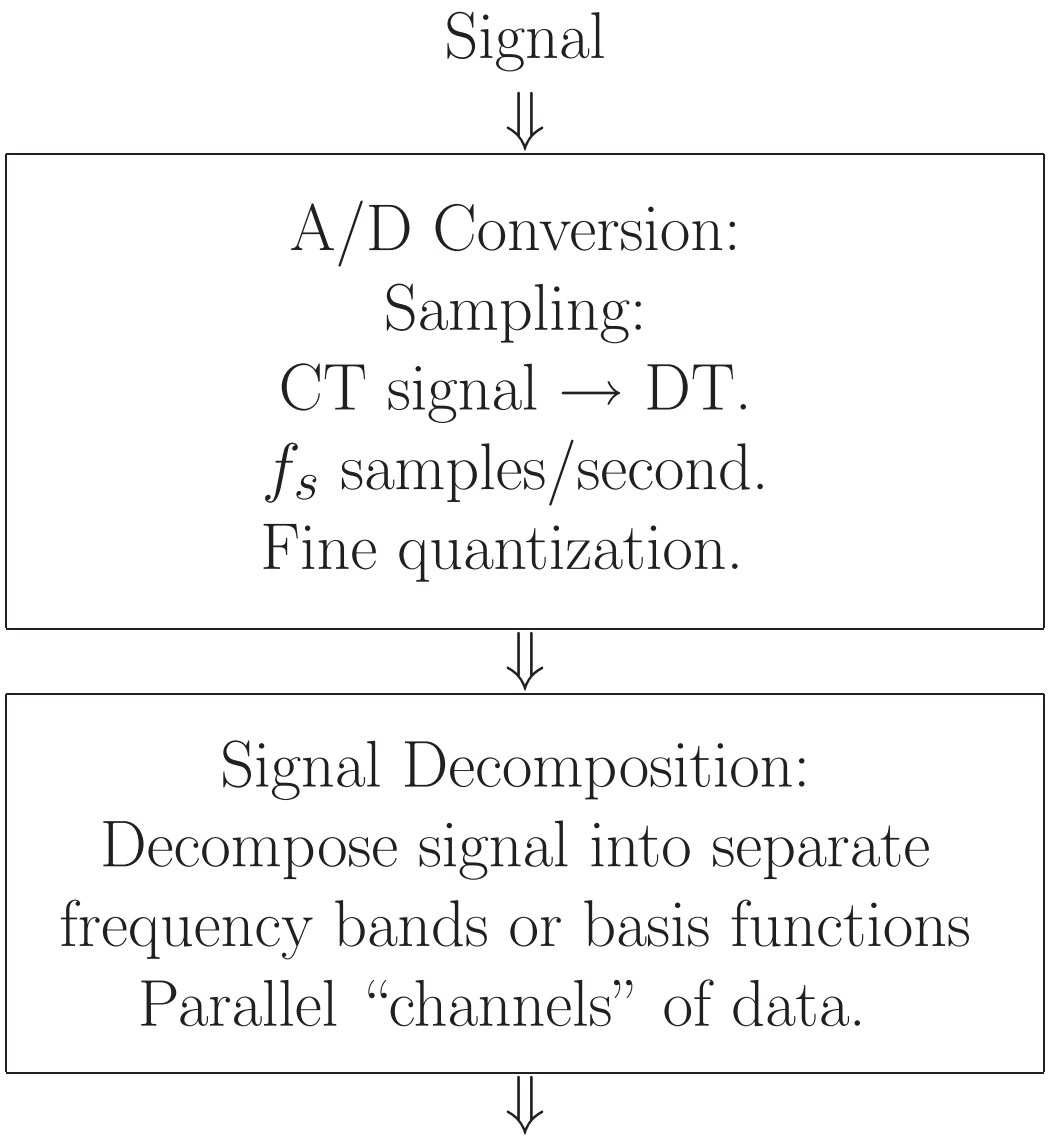
Growing evidence suggests lossy compression does not damage diagnosis.

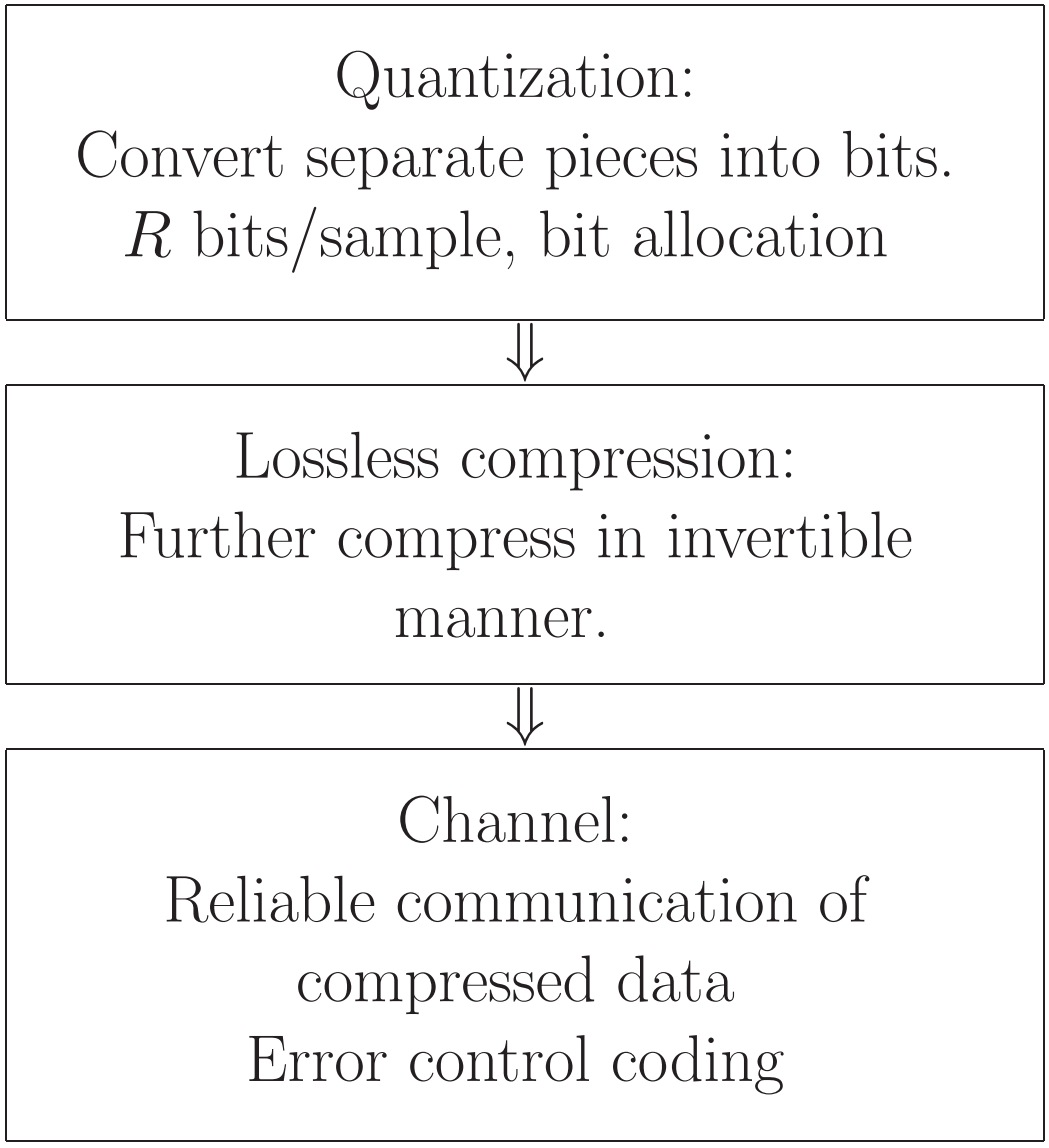
In general lossy compression may also include a lossless compression component, to squeeze a few more bits out.

Compression Context

- Compression is usually part of general system for data acquisition, transmission, storage, and display.
- Analog-to-digital conversion and digital acquisition.
- Quality and utility of "modified" digital images.

Transmitter (Encoding, including compression)





Decoding/Decompression reverses process, **except** for quantization, which cannot be reversed.

Signal The information to be communicated to a user

- speech/audio
- images/graphics
- video/animation
- telemetry
- computer files (binary data)

Encoder What do to signal before channel. Can include preprocessing, sampling, A/D conversion, signal decompositions, modulation, compression. Goal is to prepare signal for channel in a way decoder can recover good reproduction.

Decoder What do to channel output in order to reconstruct or render a version of the signal for the user. Can include inverses or approximate inverses of encoder operations, or other stuff to enhance reproduction.

Channel Portion of communication system out of designer's control: random or deterministic alteration of signal. E.g., addition of random noise, linear or nonlinear filtering.

*transparent channel	*wireless (aka radio)
deep space	*satellite
telephone lines (POTS, ISDN, fiber)	*ethernet
fiber	*CD
magnetic tape	*magnetic disk
computer memory	*modem

Often modeled as linear filtering + conditional probability distribution

Here ignore the channel coding/error control portion, save for anecdotal remarks.

Consider individual components

Sometimes combined. Sometimes omitted.

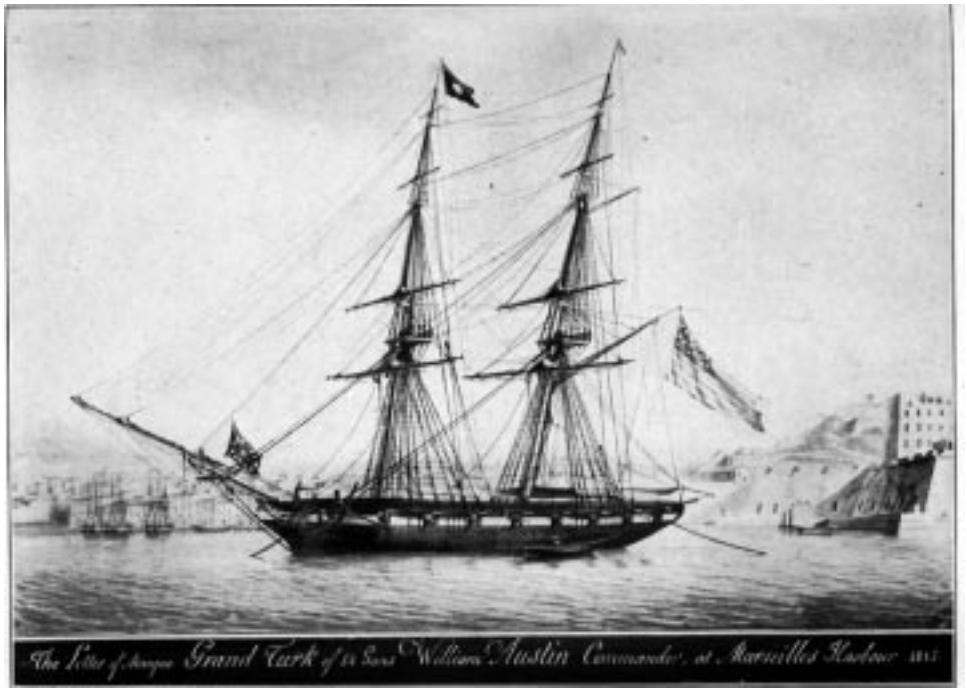
Sampling/High Resolution Quantization
A/D conversion, digitize space and amplitude.

$N \times M$ pixels, $G = 2^r$ gray levels.
Number of bits $= N \times M \times r$

Note: This step not necessary for digitally acquired image.

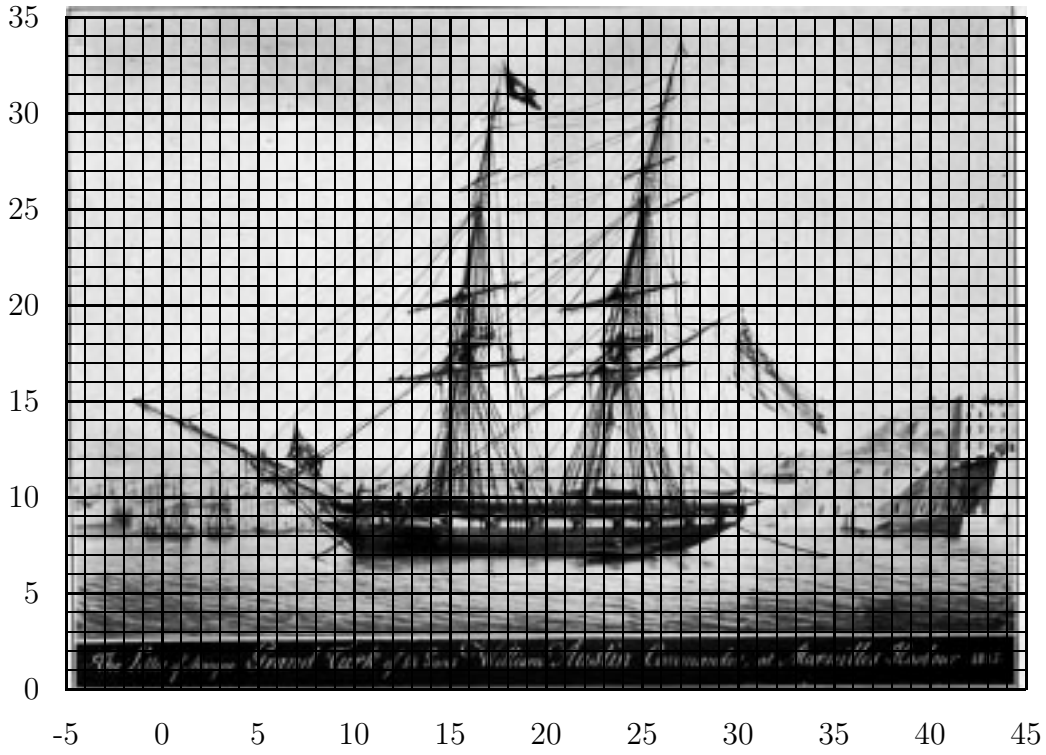
Basic idea: Sample and quantize

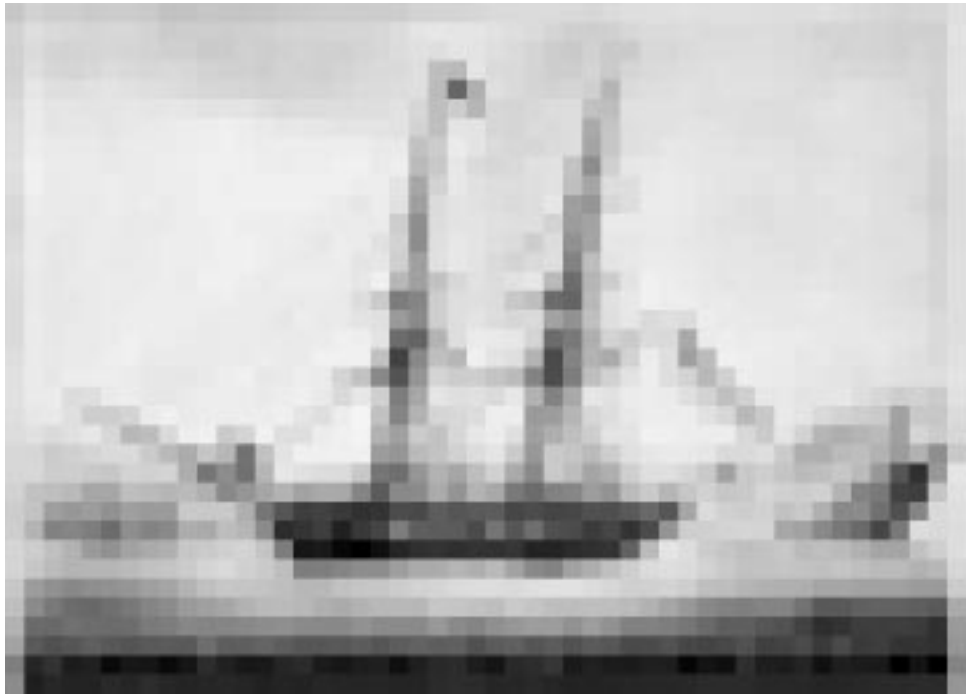
$$\{f(x,y);\ x,y \in \mathcal{R}\}$$
$$\Downarrow$$
$$\{f_{n,m} = f(n\Delta x, m\Delta y);\ n,m \in \mathcal{Z}\}$$
$$\Downarrow$$
$$\{q(f_{n,m});\ n,m \in \mathcal{Z}\}$$



The Brig Grand Turk

Most compression work assumes a digital input as a starting point since DSP is used to accomplish the compression.


35
30
25
20
15
10
5
0
-5
0
5
10
15
20
25
30
35
40
45



Spatiatially sampled and quantized

(4 bpp)

Signal decomposition (transformation, mapping): Decompose image into collection of separate images (bands, coefficients)

E.g., Fourier, DCT, wavelet, subband.

Also: Karhunen-Loeve, Hotelling, Principal Value Decomposition, Hadamard, Walsh, Sine, Hartley, Fractal

(Typically done digitally.)

Why transform?

Several reasons:

- Good transforms tend to compact energy into a few coefficients, allowing many to be quantized to 0 bits without affecting quality.
- Good transforms tend to decorrelate (reduce linear dependence) among coefficients, causing scalar quantizers to be more efficient. (folk theorem)
- Good transforms are effectively expanding the signals in good basis functions. (Mathematical intuition.)
- The eye and ear tend be sensitive to behavior in the frequency domain, so coding in the frequency domain allows the use of perceptually based distortion measures, e.g., incorporating masking.

Quantization: Convert high rate digital pixels into a relatively small number of bits.
Lossy (not invertible), nonlinear

scalar Operate on individual pixels

- uniform
- nonuniform (companded)

vector operate on groups of pixels (VQ)

Vector sources:

- Blocked signals (audio, speech, images, video)
- (R,G,B) pixels
- Multispectral imagery, microphone array
- Stereo images, audio
- LPC frames

Codeword assignment ("Coding" (Lossless compression, Entropy Coding): binary words are chosen to represent what comes out of the quantization step.

Desirable attributes of compression/coding:

- Successive approximation/progressive reconstruction
  Each additional bit improves accuracy of signal, don't need to wait for all bits to start.
- Scalable: Same bit stream produces different quality/cost options (rate, complexity, power, . . .)

Decoding/Decompression reverses process, **except** for quantization, which cannot be reversed.

Quantization involved in

- A/D conversion (high resolution quantization)
- digital arithmetic, errors in transforms
- quantization for lossy compression

For computer applications can view original signal as already digital, yielding a general compression encoder block diagram:

Typical Compression System

Important parameters of compression systems

- Compression efficiency
  - Bitrate and compression ratio.
- Fidelity, average distortion
  - MSE, SNR, subjective
- Complexity
  - algorithmic, hardware, cost
- Memory
- Coding Delay
- Robustness

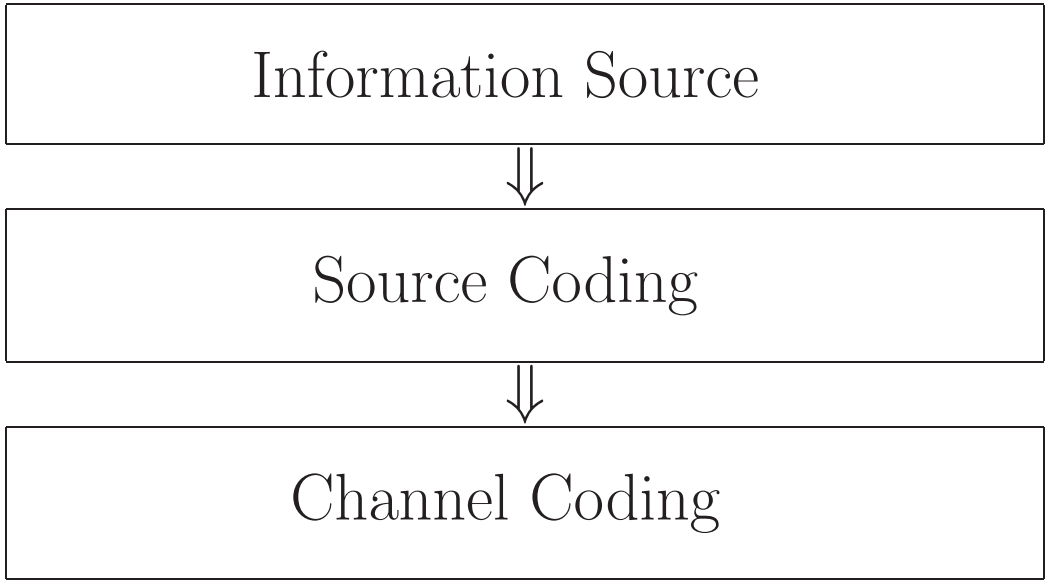
Issues in Research and Practice

- 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
- Overall quality in application: Subjective quality and diagnostic accuracy
- systems and standards

Other issues:

- Variable-rate vs. fixed-rate
- Bitrate control
- Progressive, scalable (power, bitrate, cost, complexity)

Information theory changes blocks slightly and usually ignores signal decompositions:



Source coding reduces the number of bits in order to save on transmission time or storage space.
Channel coding typically increases the number of bits or chooses different bits in order to protect against channel errors.

"Image coding" and "speech coding" mean source coding.

"Error correction coding" means channel coding.

Information theory (Shannon) $\Rightarrow$ for single source, single user, can do nearly optimally if separately source and channel code. NOT true in networks! Better to consider jointly.

Care needed, channel errors can have catastrophic effects on reconstructing compressed data.
Some work done on joint source-channel coding. Some consideration given to the effect of bit errors.
Catastrophic errors vs. minor errors of fixed rate schemes.

If *enough* bits (samples and quantizer values), cannot tell the difference between original and digital version.

But how much is enough and how do you prove it?

How much can you compress?
Do you lose important information?
Do you get sued?

Survey the field of compression: methods, theory, and standards.

Some critiques and personal biases.

For a list of many compression-related Web sites with demonstrations, examples, explanations, and software, visit

`http://www-isl.stanford.edu/~gray/iii.html`

Part II. Lossless Coding

noiseless coding, entropy coding, invertible coding, data compaction.
In much of CS world simply called "data compression"

- Can perfectly recover original data (if no storage or transmission bit errors) *transparent*
- Variable length binary codewords.
- Only works for *digital sources*.
- Can also *expand* data!

Simple binary invertible code:
Example: fixed length (rate) code

Input Letter	Codeword
a_0	00
a_1	01
a_2	10
a_3	11

Can uniquely decode individual codewords and sequences if know where blocks start. E.g.,
0000011011 decodes as $a_0a_0a_1a_2a_3$.

Real world example:
ASCII (American Standard Code for Information Interchange): assigns binary numbers to all letters, numbers, punctuation for use in computers and digital communication and storage.

All letters have same number of binary symbols (7).

Example:

$$\begin{aligned} \text{a} &\rightarrow 110001 \\ \text{b} &\rightarrow 110010 \\ \text{c} &\rightarrow 110011 \\ \text{d} &\rightarrow 110100 \\ \text{e} &\rightarrow 110101 \\ &\vdots \end{aligned}$$

May get compression.

Need at least R bits, where $2^R > M$.

Generally a fixed rate code gives no lossless compression unless original representation was inefficient.
To get lossless compression need a variable length code.

Goal of noiseless coding is to reduce the average number of symbols sent while suffering no loss of fidelity.

variable rate
ambiguous

Input Letter	Codeword
a_0	0
a_1	10
a_2	101
a_3	0101

Not work
For example, $0101101010\cdots$ could be produced by
$a_0a_2a_2a_0a_1\cdots$ or by $a_3a_2a_0a_1\cdots$.
Ambiguity can never be resolved regardless of future received bits.

uniquely decodable: if decoder receives a valid encoded sequence of finite length, there is only one possible input sequence.

i.e., lossless coding of any *sequence*.

variable rate
uniquely decodable

Input Letter	Codeword
a_0	0
a_1	10
a_2	110
a_3	111

Goal of noiseless coding is to reduce the average number of symbols sent while suffering no loss of fidelity. Basic approaches:

• Statistical: Code likely symbols or sequences with short binary words, unlikely with long. (Morse code, runlength coding, Huffman coding, Fano/Elias codes, arithmetic codes)

• Dictionary: Build a table of variable length sequences as they occur and then index into table. (Lempel-Ziv codes)

A *noiseless code* or *lossless code* consists of

- an encoder $\alpha : A \to \mathcal{W} \subset \{0, 1, \}^*$ and
- a decoder $\beta : \mathcal{W} \to A$ such that $\beta(\alpha(x)) = x$.

One-to-one, invertible

Length: $l(\alpha(x))$ = length (# bits) of encoded symbol x

Kraft Inequality If code is uniquely decodable, then

$$\sum_x 2^{l(\alpha(x))} \leq 1$$

average length $\bar{l}(\alpha) = El(\alpha(X)) = \Sigma_{a \in A} p(a) l(a)$.

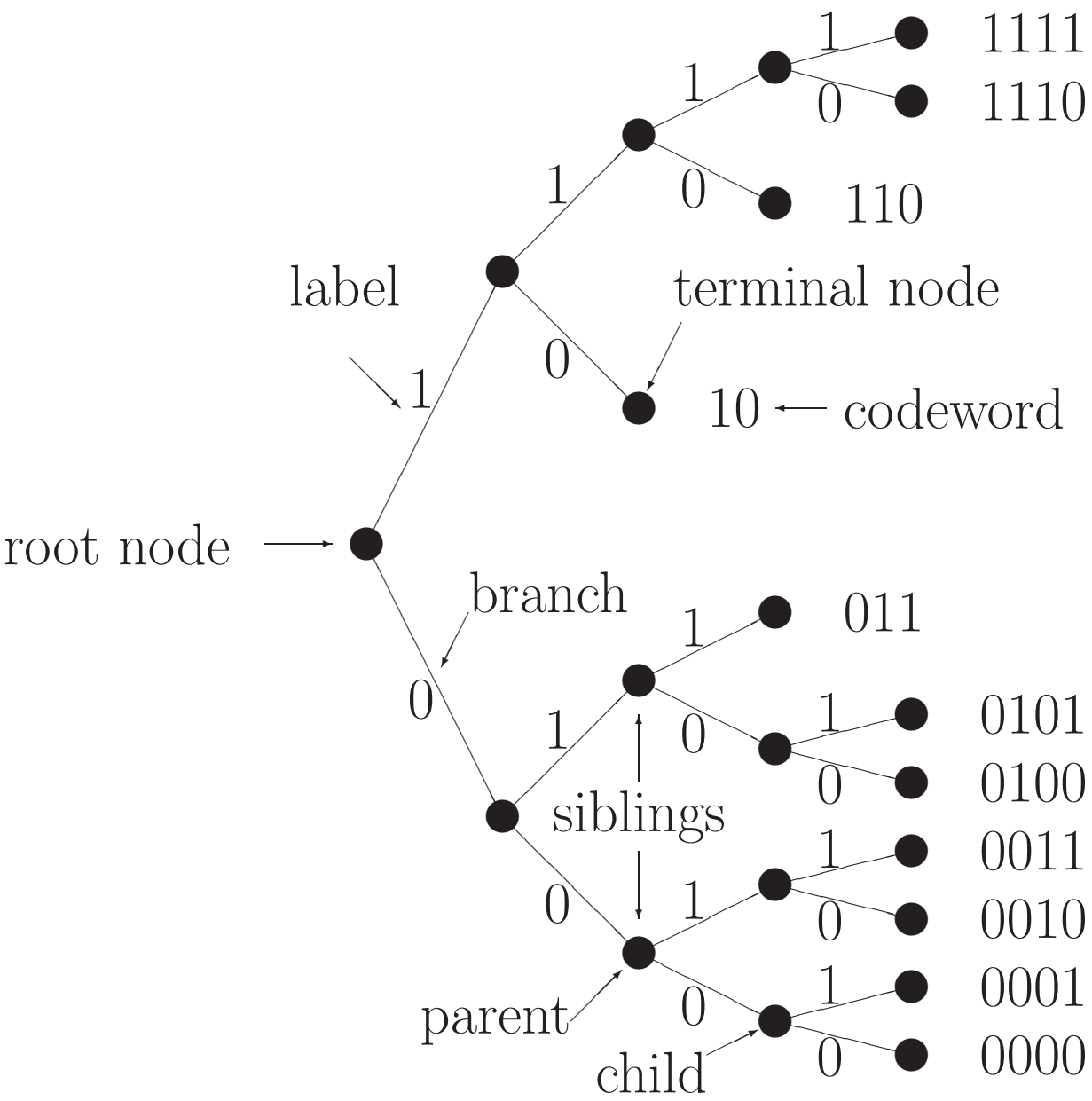
- How small can $\bar{l}(\alpha)$ be?
- How achieve performance close to optimal?

Prefix and Tree-structured Codes

Prefix condition: No codeword is a prefix of another codeword.

Ensures uniquely decodable.

Essentially no loss of generality.
Binary prefix codes can be depicted as a binary *tree*:


1
1111
1
0
1110
1
0
110
label
terminal node
0
1
10 ← codeword
root node
branch
1
011
0
1
0
1
0101
0
0100
siblings
1
0011
0
1
0
1
0010
parent
0
1
0001
child
0
0000

Shannon's lossless coding theorem: Define entropy $H(X)$ or $H(p)$ by

$$H(p) \equiv \sum_{a \in A} p(a) \log \frac{1}{p(a)}$$

Then for any uniquely decodable lossless code

$$\bar{l}(\alpha) \geq H(p);$$

and there exists a prefix code for which

$$\bar{l}(\alpha) < H(p) + 1.$$

Proof of lower bound to $\bar{l}$:
Simple application of Kraft inequality and Divergence inequality, which follows from definition of divergence (relative entropy) and Jensen's inequality.
See any elementary text on information theory or compression.

Proof of upper bound to $\bar{l}$:
Simple suboptimal coding construction

Huffman: simple algorithm yielding the *optimum* code in the sense of providing the minimum average length over all uniquely decodable codes.

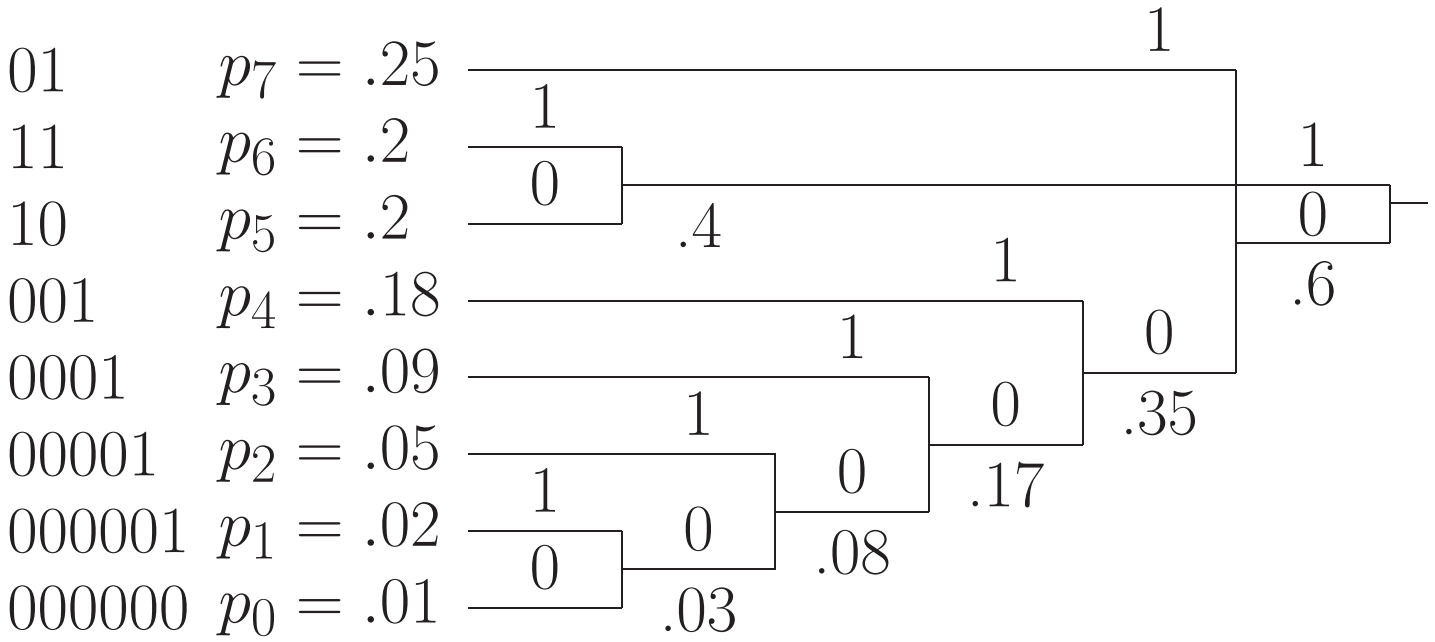
Notes

- "optimum" for specified code structure, i.e., memoryless operation on each successive symbol
- assumes that underlying pmf p is known

Huffman Codes:
Based on following facts (proved in any compression or information theory text)
A prefix code is optimal if it has the following properties:

- If symbol a is more probable than symbol b ($p(a) > p(b)$) then $l(a) \leq l(b)$
- The two least probable input symbols have codewords which are equal in length and differ only in the final symbol

Huffman gave a constructive algorithm for designing such a code. The input alphabet symbols can be considered to correspond to the terminal nodes of a code tree . "Grow" trea from leaves to root, each time satisfying optimality conditions



A Huffman Code

So far considered only one source symbol at a time.

First order (marginal entropy)

All ideas extend to larger alphabets; e.g., to vectors.

Code $X^N = (X_0, \cdots, X_{N-1}) \sim p_{X^N}$
Shannon:
For any integer N average length satisfies

$$\bar{l} \geq \frac{1}{N} H(X^N).$$

There exists a prefix code for which

$$\bar{l} < \frac{1}{N} H(X^N) + \frac{1}{N}.$$

For any fixed N, Huffman optimal.

If process stationary, *entropy rate*

$$\overline{H} = \min_N \frac{H(X^N)}{N} = \lim_{N \to \infty} \frac{H(X^N)}{N}.$$

is impossible to beat, and asymptotically achievable by coding very large blocks.

Question If "optimal" and can get arbitrarily close to theoretical limit, why not always use?

Because *too complicated*: Big N needed, which can make Huffman tables enormous.

Also: Do not always know distributions, and they can change.

Leads to many methods that are "suboptimal," but which outperform any (implementable) Huffman code.

Good ones are also asymptotically optimal, e.g., arithmetic and Lempel-Ziv.

Note: codes have global block structure on a sequence, but locally they operate very differently. Nonblock, memory

Note: Frequently someone announces a lossless code that does better than the Shannon optimum.

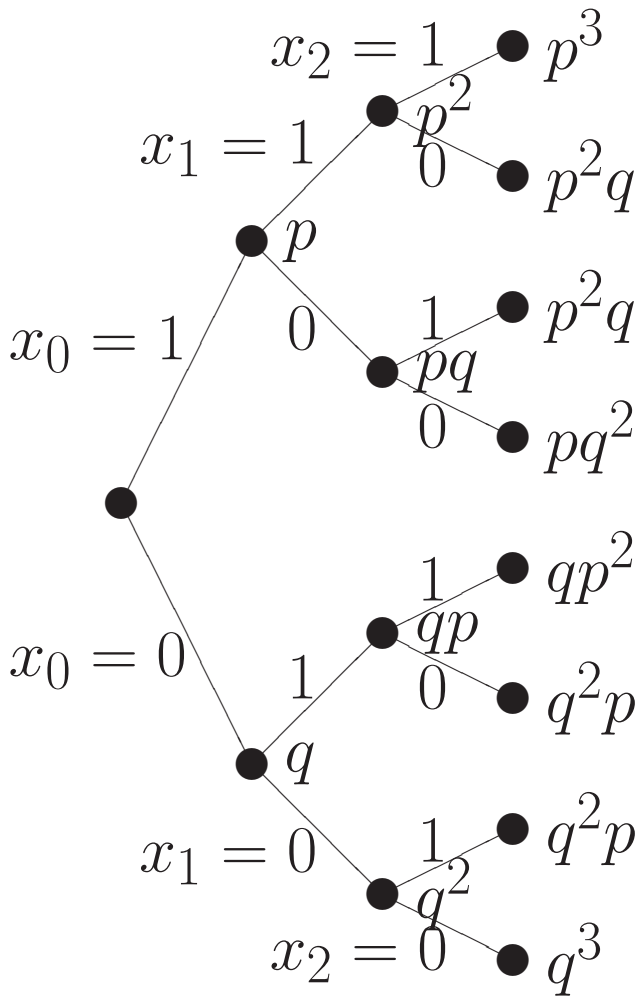
Usually a confusion of first order entropy with entropy rate, often a confusion of what "optimal" means or what the constraints are.

Arithmetic codes

Finite-precision version of Elias (& Fano)code developed by Pasco and Rissanen.
Heart of IBM Q-coder.

Here consider only special case of lossless coding a binary iid source $\{X_n\}$ with $\Pr(X_n = 1) = p$ and $\Pr(X_n = 0) = q$.
(It is convenient to give a separate name to $q = 1 - p$.)

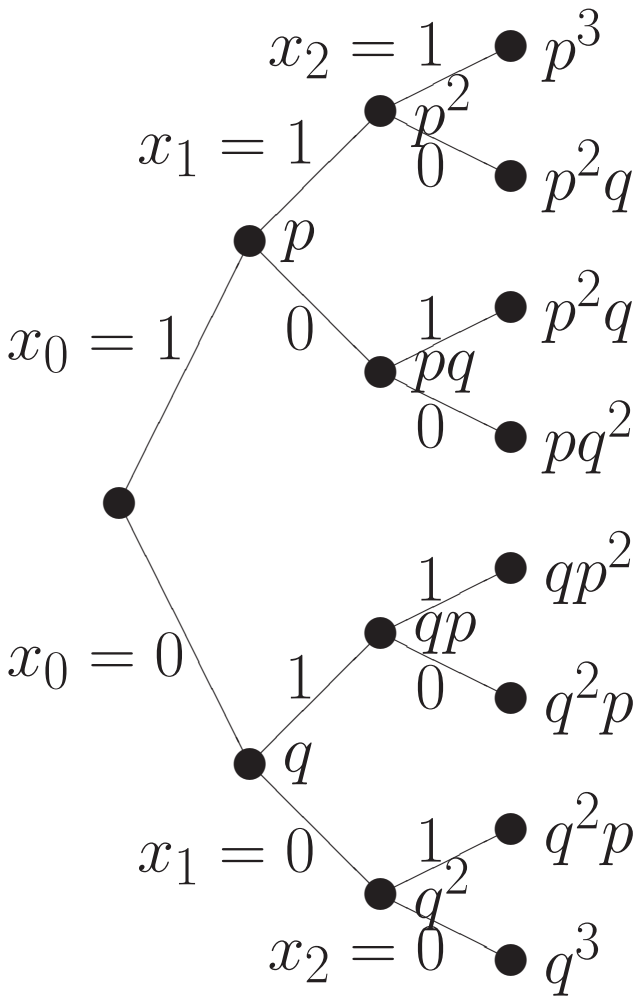
Key idea: Produce a code that results in codelengths near the Shannon optimal, i.e., code a sequence $x^n = (x_0, x_1, \ldots, x_{n-1})$ into a codeword u^L where $L = l(x^n) \approx -\log p_{X^n}(x^n)$.

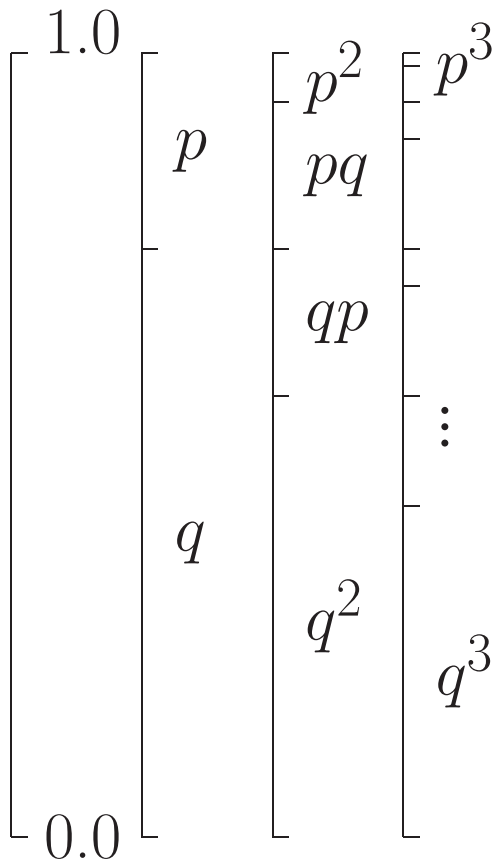


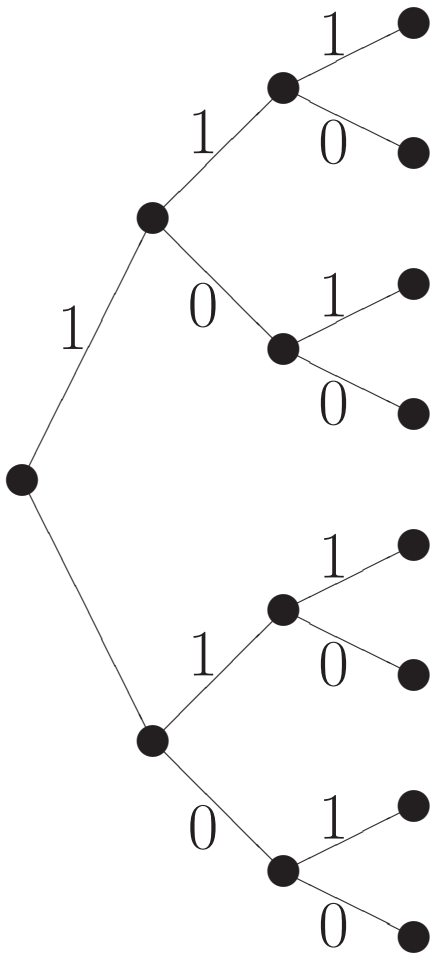
SOURCE TREE

x^n corresponds to
interval $I_n(x^n) = I_n = [a_n, b_n)$
such that $|I_n| = b_n - a_n = p_{X^n}(x^n)$

$$[a_n, b_n) = \begin{cases} [a_{n-1}, a_{n-1} + q(b_n - a_n)) & \text{if } x_n = 0 \\ \\ [a_{n-1} + q(b_n - a_n), b_{n-1}) & \text{if } x_n = 1 \end{cases}$$


$x_2 = 1$
p^3
p^2
$x_1 = 1$
0
p^2q
p
$x_0 = 1$
0
1
p^2q
pq
0
pq^2
1
qp^2
qp
$x_0 = 0$
1
0
q^2p
q
$x_1 = 0$
1
q^2p
q^2
$x_2 = 0$
q^3

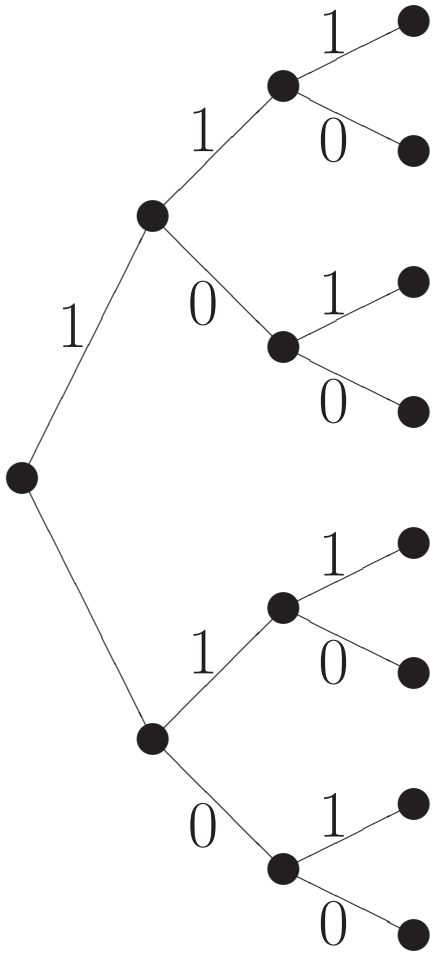

1.0
p
q
0.0
p^2
pq
qp
q^2
p^3
q^3

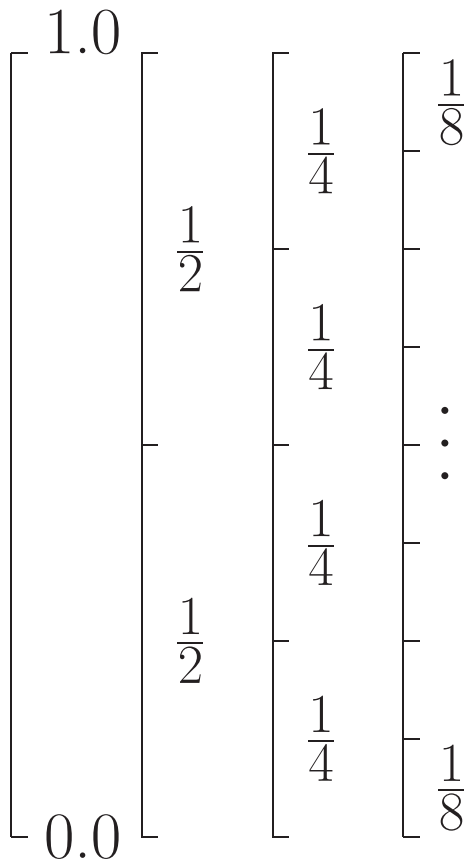


CODE TREE

u^L corresponds to
interval J_L such that $|J_L| = 2^{-L}$

$$J_L = [\Sigma_{l=0}^{L-1} u_l 2^{-l-1}, \Sigma_{l=0}^{L-1} u_l 2^{-l-1} + 2^{-L})$$


1
1
0
1
0
1
0
1
1
0
1
0
1
1
0
0


1.0
0.0
1/2
1/2
1/4
1/4
1/4
1/4
1/8
1/8

Suppose encoder and decoder both know n

Encoder: $x^n \to I_n \to J_L \subseteq I_n \to u^L$

Lots of ways to choose L and J_L. Choose smallest possible L and hence biggest possible interval J_L.

$$|I_n| \geq |J_L| = 2^{-L} \geq |\frac{I_n}{2}| \qquad \star$$

Reason: Given I_n, divide unit interval into 2^L subintervals of size 2^{-L} where L chosen as smallest integer for which $|I_n| \geq 2 \times 2^{-L}$. Then one of the subintervals must be contained in I_n and hence $|J_L| \geq |\frac{I_n}{2}|$

Decode: $u^L \to J_L \to I_n \supseteq J_L \to x^n$

Key fact: Taking logarithms implies that

$$-\log|I_n| \leq L \leq -\log|\frac{I_n}{2}|$$

$$-\log p_{X^n}(x^n) \leq L \leq -\log\frac{p_{X^n}(x^n)}{2} = -\log p_{X^n}(x^n) + 1$$

Thus if $x^n \to J_L$, then

$$-\log p_{X^n}(x^n) \leq l(\alpha(x^n)) =\leq -\log p_{X^n}(x^n) + 1$$

or

so $l(x^n) \approx -\log p_{X^n}(x^n)$ as desired.

Taking expectations and normalizing:

$$\frac{1}{n}H(X^n) \leq El(X^n) \leq \frac{1}{n}H(X^n) + \frac{1}{n}$$

which converges to the optimum $\overline{H}$ as n gets large!

Collection of leaves $u^L(x^n)$ is a prefix-free code since a subtree.

Why "better" than Huffman?

Incrementality of algorithm permits much larger n then does Huffman. E.g., suppose $n = 256 \times 256$ for a modest size image.

Huffman out of the question, would need a tree of depth 2^{65536}.

Elias encoder can compute $[a_k, b_k)$ incrementally as $k = 1, 2, \ldots, n$.

As I_k shrinks, can send "release" bits incrementally: Send up to l bits where

$$J_l \supseteq I_k,$$

J_l is the smallest interval containing I_k.
Decoder receives J_l, can decode up to x^m, where

$$I_m \supseteq J_l \supseteq I_k, \text{ where } m < k.$$

Finally, encoder sends final bits, or can stop if no ambiguity at decoder (knowing n).

- practical implementations need finite precision, result is arithmetic code.
- trees can be m-ary

Extend to sources with memory by using "context modeling"
Compute

$$[a_n.b_n) = \begin{cases} [a_{n-1}, a_{n-1} + (b_{n-1} - a_{n-1}) \Pr(X_n = 0|x^{n-1})) & \text{if } x_n = 0 \\ [a_{n-1} + (b_{n-1} - a_{n-1}) \Pr(X_n = 0|x^{n-1}), b_{n-1}) & \text{if } x_n = 1 \end{cases}$$

$\Pr(X_n = 0|x^{n-1}))$ can be learned on the fly, e.g., assume
Markov order and estimate from histograms.

Adaptive arithmetic codes: learn conditional pmf's on the fly.

Performance depends on how well (accuracy and complexity)
estimate underlying conditional distributions.

Witten, Bell, & Cleary

Adaptive and Universal Entropy Coding:

Both Huffman codes and arithmetic codes assume *a priori* knowledge of the input probabilities. Often not known in practice.

• Adaptive Codes: Adjust code parameters in real time. Often by fitting model to source and then coding for model
Examples: adaptive Huffman (Gallager) and adaptive arithmetic codes. (JBIG, lossless JPEG, CALIC, LOCO-I, Witten, Bell, & Cleary)

Alternative approach:

• Universal Codes: Collection of codes, pick best

or

build code dictionary from data with no explicit modeling

Examples of Universal Codes:

Rice machine multiple codebooks, pick one that does best over some time period.
(4 Huffman codes in Voyager, 1978–1990; 32 Huffman codes in MPEG-Audio Layer 3)

Lynch-Davisson codes View binary N-vector. First send integer n giving the number of ones (this is an estimate n/N of underlying p), then send binary vector index saying which of the $\binom{N}{n}$ weight n N-dimensional vectors was seen.

Rissanen MDL, send prefix describing estimated model, followed by optimal encoding of data assuming model.

Lempel-Ziv codes

Dictionary code, not statistical in construction.
Inherently universal
Variable numbers of input symbols are required to produce each code symbol.
Unlike both Huffman and arithmetic codes, the code does not use any knowledge of the probability distribution of the input.
Achieves the entropy lower bound when applied to a suitably well-behaved source.

Basic idea (LZ78, LZW): recursively parse input sequence into nonoverlapping blocks of variable size while constructing a dictionary of blocks seen thus far.

Each sequence found in the dictionary is coded into its index in the dictionary.

The dictionary is initialized with the available single symbols, here 0 and 1.

Each successive block in the parsing is chosen to be the largest (longest) word ω that has appeared in the dictionary. Hence the word ωa formed by concatenating ω and the following symbol is not in the dictionary.

Before continuing the parsing ωa is added to the dictionary and a becomes the first symbol in the next block.

Example of parsing and dictionary construction:

$$01100110010110000100110.$$

parsed blocks enclosed in parentheses and the new dictionary word (the parsed block followed by the first symbol of the next block) written as a subscript:

$$(0)_{01}(1)_{11}(1)_{10}(0)_{00}(01)_{011}(10)_{100}(01)_{010}(011)_{0110}$$

$$(00)_{000}(00)_{001}(100)_{1001}(11)_{110}(0).$$

The parsed data implies a code by indexing the dictionary words in the order in which they were added and sending the indices to the decoder.

Decoder can recursively reconstruct dictionary and input sequence as bits arrive.

LZ77 (PKZip)

Assume encoder and decoder have fixed length window of memory.

In window store the N most frequently coded symbols.

Encoder and decoder both know this information since code lossless.

Initialize window with standard set of values.

To encode data:

- search through the window and look for the longest matching string to this window.
- describe length of the match
- describe the location of match in the earlier window
- shift the window to incorporate the latest information and repeat

Offset & length information also described by lossless code.

Final thoughts on lossless coding:

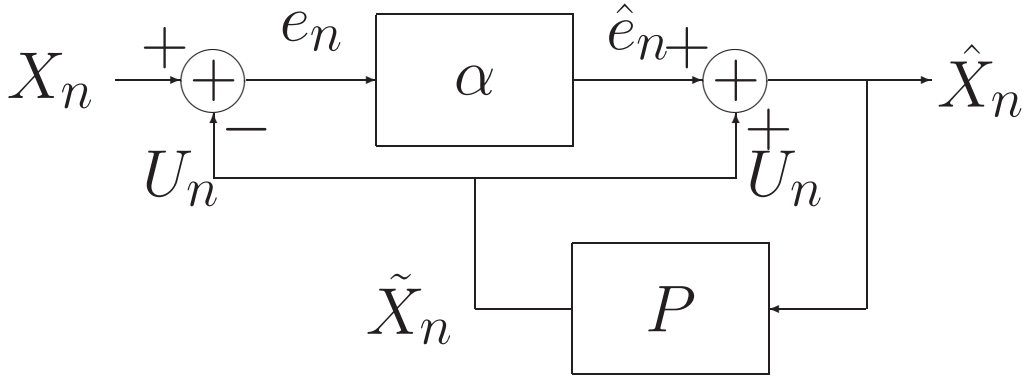
- Run Length Encoding
- Predictive Coding

Run length encoding: Simple but still useful.

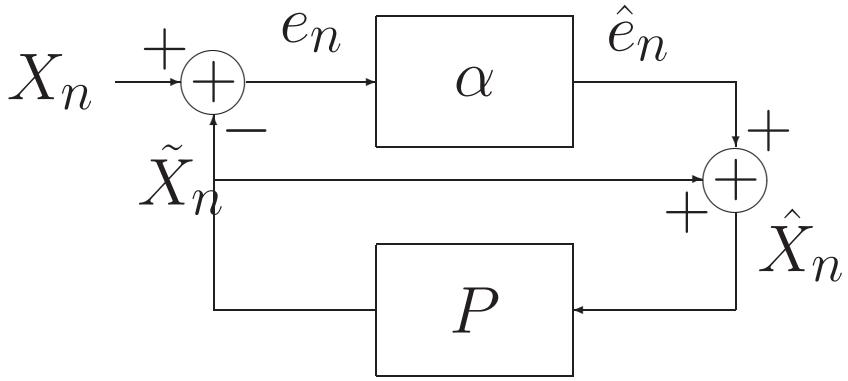
Often used for data with many repeated symbols, e.g., FAX and quantized transform coefficients (DCT, wavelet)

Can combine with other methods, e.g., JPEG does lossless coding (Huffman or arithmetic) of combined quantizer levels/runlengths.

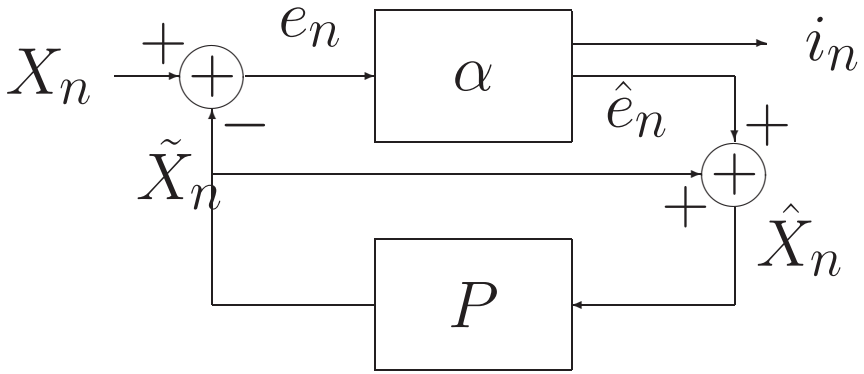
Predictive: Instead of coding each symbol in a memoryless fashion, predict the symbol based on information that the decoder also possesses, form a prediction residual, and code it. Decoder then adds decoded residual to its version of prediction. One view: Encoder has copy of decoder and uses current info to predict next symbol viewed in way known to decoder. Then forms difference.



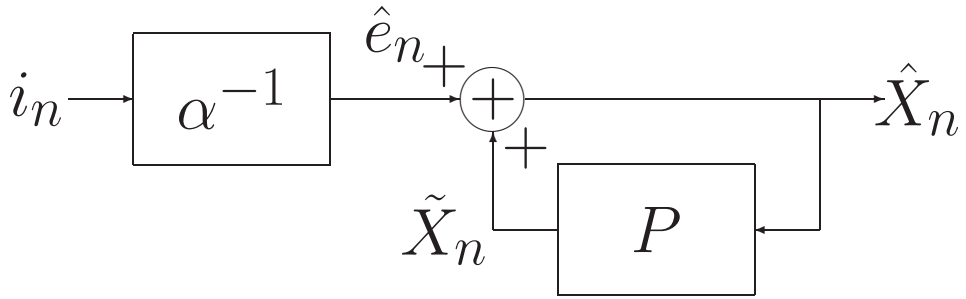
$$\hat{X}_n - X_n = (\hat{e}_n + U_n) - (e_n + U_n) = \hat{e_n} - e_n$$



Predictive Code Encoder (discrete DPCM)



Encoder



Decoder

Lossless DPCM Encoder/Decoder

Part III: Lossy Compression

Uniform Scalar Quatization

Roundoff numbers to nearest number in discrete set.
$>$ 100 years old.

Original A/D converter was pulse coded modulation (PCM)
(Oliver, Pierce, Shannon)

Uniform scalar quantization coded by binary indexes.

Mathematical "compression" since mapped analog space into digital space.

Not practical "compression" since expanded required bandwidth given modulation techniques of the time.

Example

Speech $\approx$ 4 kHz BW.

Sample @ 8 kHz, Quantize to 8bpp $\Rightarrow$ 64 kbps
With traditional ASK, PSK, or FSK, get about 1 bit per Hz
Thus new BW $\approx$ 64 kHz, 16 times greater than original analog waveform!!

Optimal Scalar Quatization

Lloyd (1956)

Optimize PCM by formulating as a constrained optimization problem

X is scalar real-valued random variable, with pdf f_X

Quantizer:

Encoder $\alpha : \Re \rightarrow \mathcal{I}$

$\mathcal{I}$ is $\{0, 1, 2, \ldots, N-1\}$, or an equivalent set of binary vectors.

Decoder $\beta : \mathcal{I} \rightarrow \Re$

Overall Quantizer $Q(x) = \beta(\alpha(x))$

Reproduction codebook $\mathcal{C} = \{\beta(i); i \in \mathcal{I}\}$

Problem: Given N, find the encoder α and decoder β that minimize the average distortion (MSE)

$$\begin{aligned} E[(X - Q(X))^2] &= \int (x - Q(x))^2 f_X(x)\, dx \\ &= \sum_{i=1}^{N} \Pr(\alpha(X) = i)) E[(X - \beta(i))^2 | \alpha(X) = i] \end{aligned}$$

An example of statistical clustering.
Lloyd algorithm for optimal PCM was one of the original clustering algorithms. (k-means, isodata, principal points)

Strong parallel between optimal quantization and statistical algorithms for clustering/classification/pattern recognition.

Idea:

$\star$ Suppose the decoder β is fixed, then the optimal encoder is

$$\alpha(x) = i \text{ which minimizes } |x - \beta(i)|^2$$

(minimum distortion encoder, nearest neighbor encoder)

$\star$ Suppose the encoder α is fixed, then the optimal decoder is

$$\beta(i) = E[X|\alpha(X) = \beta(i)]$$

Conditional expectation is optimal nonlinear estimate.

This gives a descent algorithm for improving a code:

1. Optimize the encoder for the decoder.
2. Optimize the decoder for the encoder.

Continue until convergence.

General formulation: Lloyd (1957)
Special case (calculus derivation) Max (1960)
"Lloyd-Max Quantizer"

In general only a local optimum, sometimes global (Gaussian)

Most common techniques employ *scalar* quantization, but improve performance via prediction, transforms, or other linear operations.

Uniform simple vs. Lloyd-optimal better performance

Compromise: Companding (μ-law, A-law, etc.)

Caveat Optimal for *fixed rate*, i.e., number of quantizer levels N is fixed. *Not* optimized for variable rate (variable length) codes, e.g., if fix entropy instead of N. (Entropy constrained quantization.)

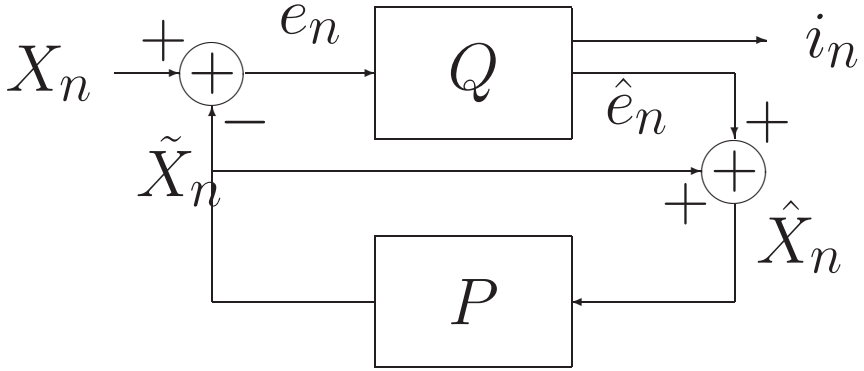
There are generalized Lloyd algorithms for designing ECQ, *but* there are theoretical results (will sketch later) that show:

> If the rate is large, then the quantizer solving the ECQ problem, i.e., minimizing MSE for a given entropy (hence assuming a subsequent entropy coder), is *approximately uniform*!

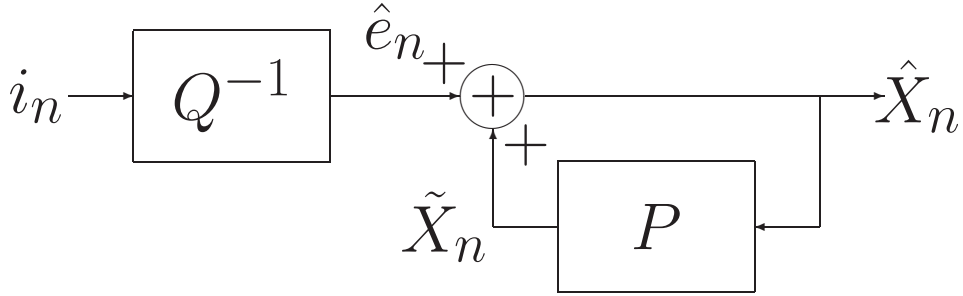
Thus if variable-rate coding is allowed and the rate is large, uniform quantization is nearly optimal.

Predictive coding

Elias (1955). Code differences (residual, prediction error)
Predictive scalar quantization (DPCM, delta modulation, Sigma Delta modulation. Cutler (1960), Inose & Yasuda (1963), Candy (1974))



Encoder



Decoder

DPCM Encoder/Decoder

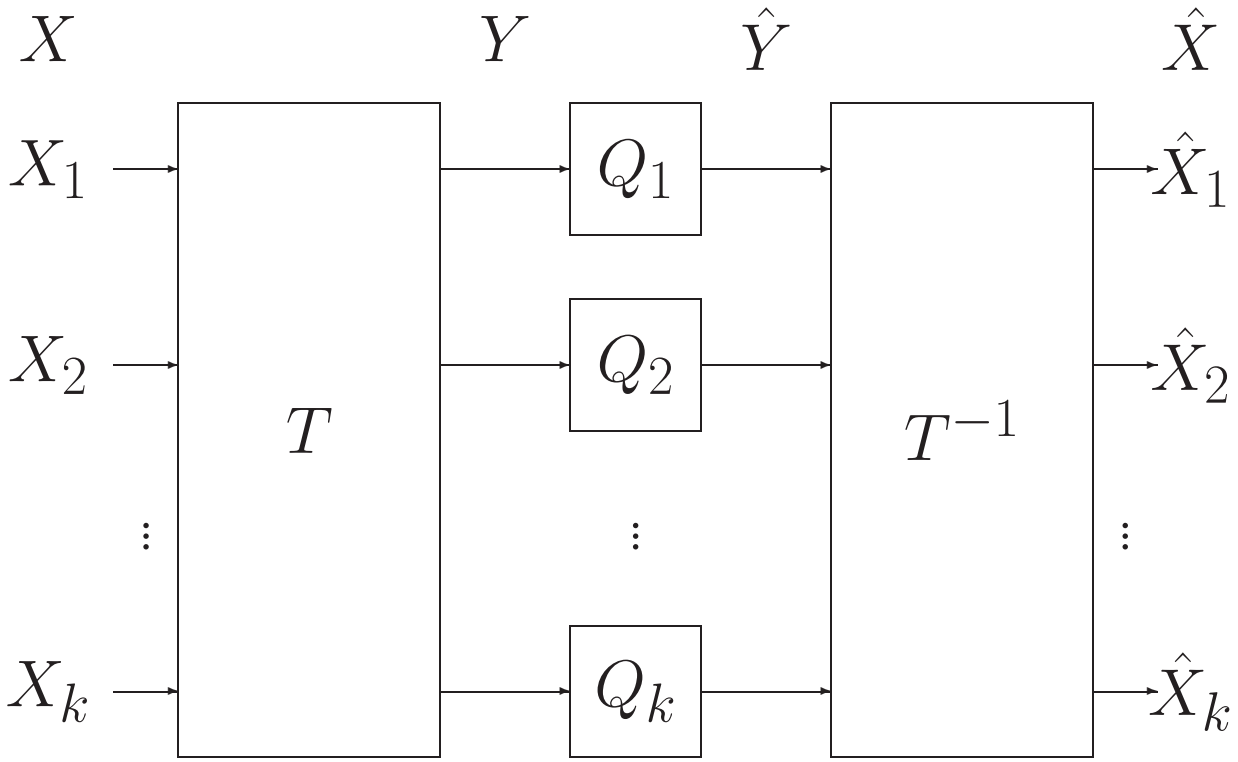
As in lossless case, $\hat{X}_n - X_n = (\hat{e}_n + U_n) - (e_n + U_n) = \hat{e_n} - e_n$

Overall quality = quality of reconstructed prediction error.

Transform Coding

Mathews and Kramer (1956)
Huang, Habibi, Chen (1960s)

Dominant image coding (lossy compression) method: ITU and other standards p*64, H.261, H.263, JPEG, MPEG
C-Cubed, PictureTel, CLI



Transform Coding

$$X = (X_1, X_2, \ldots, X_k)^t$$

$$Y = T(X), \hat{Y} = Q(Y), \hat{X} = T^{-1}(\hat{Y})$$

MSE:

$$D_{tc} = \sum_{i=1}^{k} E[|X_i - \hat{X}_i|^2] = E[||X - \hat{X}||^2]$$

Easy to relate MSE of Xs to MSE of Ys if use *orthogonal* transform:
If $T^* = T^{-1}$ so that rows are orthogonal, then

$$E[||X - \hat{X}||^2] = E[||Y - \hat{Y}||^2]$$

(linear algebra).
Ensures that inverse transform will not amplify quantization error.

Which **Transforms**??

Karhunen-Loeve Transform

Often referred to as "optimal transform"

Idea: Decorrelate components of vector X. If also Gaussian, makes independent.

Consider 1D case, assume real:

$R_X = E[XX^t]$
$X \rightarrow Y = TX$ with $R_Y = E[YY^t]$ diagonal.

u_i denote the eigenvectors of R_X (normalized to unit norm)
λ_i the corresponding eigenvalues (ordered)

The Karhunen-Loeve transform matrix is then defined as $T = U^t$ where

$$U = [u_i u_2 \dots u_k],$$

(the columns of U are the eigenvectors of R_X)

Then

$$R_Y = E[YY^t] = E[U^t X X^t U] = U^t R_X U = diag(\lambda_i).$$

Note that the variances of the transform coefficients are the eigenvalues of the autocorrelation matrix R_X.

Many other transforms, but dominant ones are discrete cosine transform (DCT, the 800 lb gorilla) and discrete wavelet transform (DWT).

Theorems to show that if image $\approx$ Markovian, then DCT approximates KL

Wavelet fans argue that wavelet transforms $\approx$ decorrelate wider class of images.

Often claimed KL "optimal" for Xform codes

Only true (approximately) if high rate, optimal bit allocation among coefficients, and Gaussian.

Discrete Cosine Transform

The 1-D discrete cosine transform is defined by

$$C(u) = \alpha(u) \sum_{x=0}^{N-1} f(x) \cos\left[\frac{(2x+1)u\pi}{2N}\right]$$

for $u = 0, 1, 2, \ldots, N-1$. The inverse DCT is define by

$$f(x) = \sum_{u=0}^{N-1} \alpha(u) C(u) \cos\left[\frac{(2x+1)u\pi}{2N}\right]$$

for $x = 0, 1, 2, \ldots, N-1$. In these equations, α is

$$\alpha(u) = \begin{cases} \sqrt{\frac{1}{N}} & \text{for } u = 0 \\ \sqrt{\frac{2}{N}} & \text{for } u = 1, 2, \ldots N-1. \end{cases}$$

The corresponding 2-D DCT pair is:

$$C(u,v) = \alpha(u)\alpha(v) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) \times \cos\left[\frac{(2x+1)u\pi}{2N}\right] \cos\left[\frac{(2x+1)v\pi}{2N}\right]$$

for $u, v = 0, 1, 2, \ldots, N-1$, and

$$f(x,y) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} \alpha(u)\alpha(v)C(u,v) \times \cos\left[\frac{(2x+1)u\pi}{2N}\right] \cos\left[\frac{(2x+1)v\pi}{2N}\right]$$

for $x, y = 0, 1, 2, \ldots, N-1$.

Linear and reversible.

Why DCT so important?

- real
- can be computed by an FFT
- excellent energy compaction for highly correlated data
- good approximation to Karhunen-Loeve transform for Markov behavior and large images

Won in the open competition for an international standard.

- 1985: Joint Photographic Experts Group (JPEG) formed as an ISO/CCITT joint working group
- 6/87 12 proposals reduced to 3 in blind subjective assessment of picture quality
- 1/88 DCT-based algorithm wins
- 92-93 JPEG becomes intl standard

Once have transform, how actually quantize & code?

Several options:

- Scalar quantize coefficients
  - $\star$ Uniform simple and pretty good if high rate and use subsequent entropy coding. Only need to pick bin size for each quantizer.
  - Lloyd-optimal good for fixed rate applications and very low bit rate ECQ.
- Quantize groups (subbands) as vectors (complicated)
- Entropy code
  - Many zeros after quantizer, runlength code
  - Huffman or arithmetic
  - Combination

JPEG is basically

- DCT transform coding (DCT) on 8×8 pixel blocks
- custom uniform quantizers (user-specified Quantization tables)
- runlength coding
- Huffman or arithmetic coding on (runs, levels)

DC coefficient coded separately, DPCM

Sample quantization table

$$S = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

Uniform quantization: Suppose DCT transformed $8\times$ block is $\{\theta_{ij}\}$. Then quantizer output index is

$$l_{ij} = \lfloor \frac{\theta_{ij}}{S_{ij}} + .5 \rfloor \tag{1}$$

$\lfloor x \rfloor$ = largest integer smaller than x.

JPEG ubiquitous in WWW

Vector quantization Quantize vectors instead of scalars.
(theory in Shannon (1959), design algorithms in late 70s)

Motivation: Shannon theory $\Rightarrow$ better distortion/rate tradeoffs of code vectors.

Added benefit: Simple decompression in general, just table lookup.

Problem: High encoder complexity in usual implementations.

Applications:
Software based video: table lookups instead of computation
Part of speech coding standards based on CELP and variations.

Basic idea: Quantize vectors instead of scalars.

As with a scalar quantizer, VQ consists of an *encoder* and *decoder*.

Encoder $\alpha : A \to \mathcal{W} \subset \{0,1\}^*$ (or *source encoder*) α is a mapping of the input vectors into a collection $\mathcal{W}$ of finite length binary sequences.

$\mathcal{W} =$ *channel codebook*, members are *channel codewords*. set of binary sequences that will be stored or transmitted

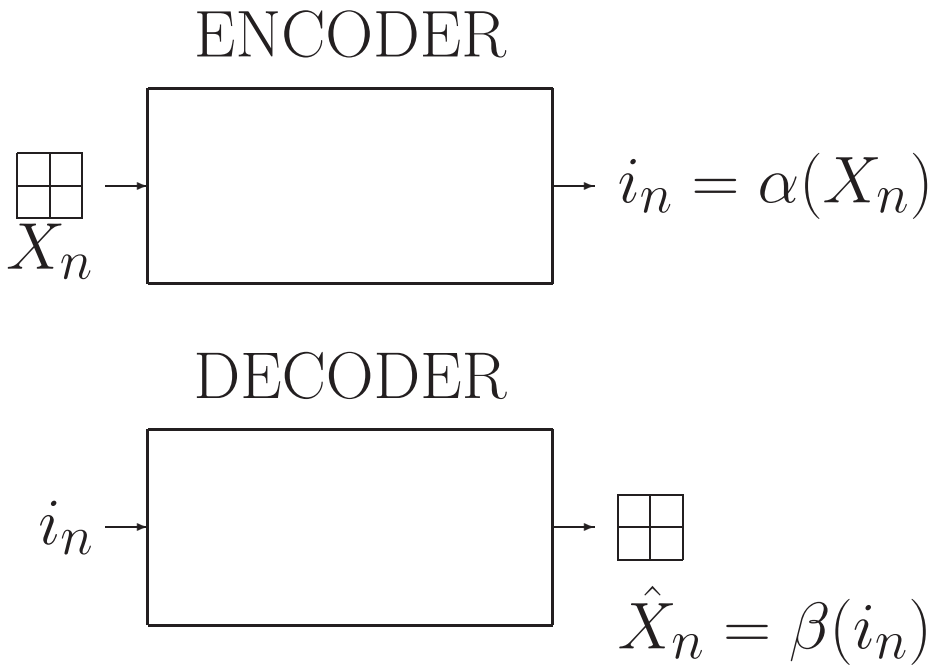
$\mathcal{W}$ must satisfy the prefix condition

Decoder $\beta : \mathcal{W} \to \hat{A}$

(table lookup)

reproduction codebook $\mathcal{C} \equiv \{\beta(i);\ i \in \mathcal{W}\}$, $\beta(i)$ called *reproduction codewords* or *templates*.

A *source code* or *compression code* for the source $\{X_n\}$ consists of a triple $(\alpha, \mathcal{W}, \beta)$ of encoder, channel codebook, and decoder.



General block memoryless source code

A VQ is essentially the Shannon model of a general compression system operating on successive blocks of data.
Theory of compression mostly based on this model.

Model is sufficiently general to include most real-world methods.

Performance: Rate and Distortion

Rate:

Given an $i \in \{0,1\}^*$, define

$$l(i) = \text{ length of binary vector } i$$

instantaneous rate $r(i) = \frac{l(i)}{k}$ bits/input symbol.

Average rate $R(\alpha, \mathcal{W}) = E[r(\alpha(X))]$.

An encoder is said to be *fixed length* or *fixed rate* if all channel codewords have the same length, i.e., if $l(i) = Rk$ for all $i \in \mathcal{W}$.

- Variable rate codes may require data buffering, expensive and can overflow and underflow
- Harder to synchronize variable-rate codes. Channel bit errors can have catastrophic effects.

But variable rate codes can provide superior rate/distortion tradeoffs.
E.g., in image compression can use more bits for edges, fewer for flat areas. In voice compression, more bits for plosives, fewer for vowels.

Distortion

Distortion measure $d(x, \hat{x})$ measures the distortion or loss resulting if an original input x is reproduced as $\hat{x}$.

Mathematically: A distortion measure satisfies

$$d(x, \hat{x}) \geq 0$$

To be useful, d should be

- easy to compute
- tractable
- meaningful for perception or application.

No single distortion measure accomplishes all of these goals. Most common is MSE:

$$d(x, y) = ||x - y||^2 = \sum_{l=0}^{k-1} |x_l - y_l|^2$$

Weighted or transform/weighted versions are used for perceptual coding.

In particular: Input-weighted squared error:

$$d(X, \hat{X}) = (X - \hat{X})^* B_X (X - \hat{X}),$$

B_X positive definite.

most common $B_X = I$, $d(X, \hat{X}) = ||X - \hat{X}||^2$ (MSE)

Other measures:

B_x takes DCT and uses perceptual masking
Or weights distortion according to importance of x
(Safranek and Johnson, Watson et al.)

Performance measured by average distortion

$$D(\alpha, \mathcal{W}, \beta) = D(\alpha, \beta) = E[d(X, \beta(\alpha(X))]$$

Goal of Compression Keep both rate and distortion small.

Optimal tradeoff??

Given all else equal, one code is better than another if it has lower rate or lower distortion.

Various structures

- Lattice codes
- Product codes (scalar Quantization, gain/shape)
- Successive approximation/ tree-structured
- Trellis encoded, trellis coded
- Predictive and finite-state
- Fractal coding

Can be applied to transform coefficients

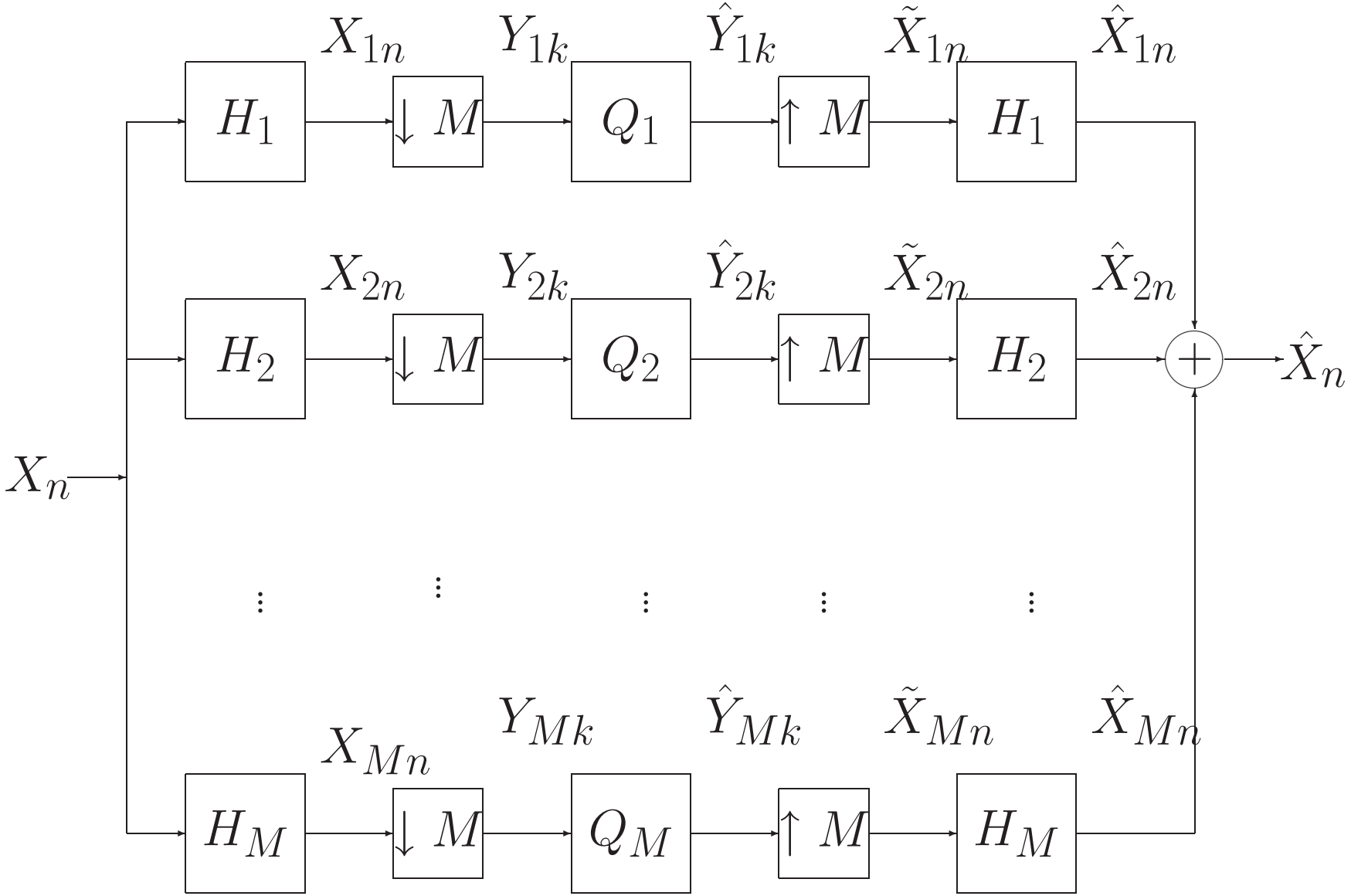
Lloyd algorithm extends to vectors, clustering approach to design. (Many other clustering algorithms.)
Computationally costly to design.

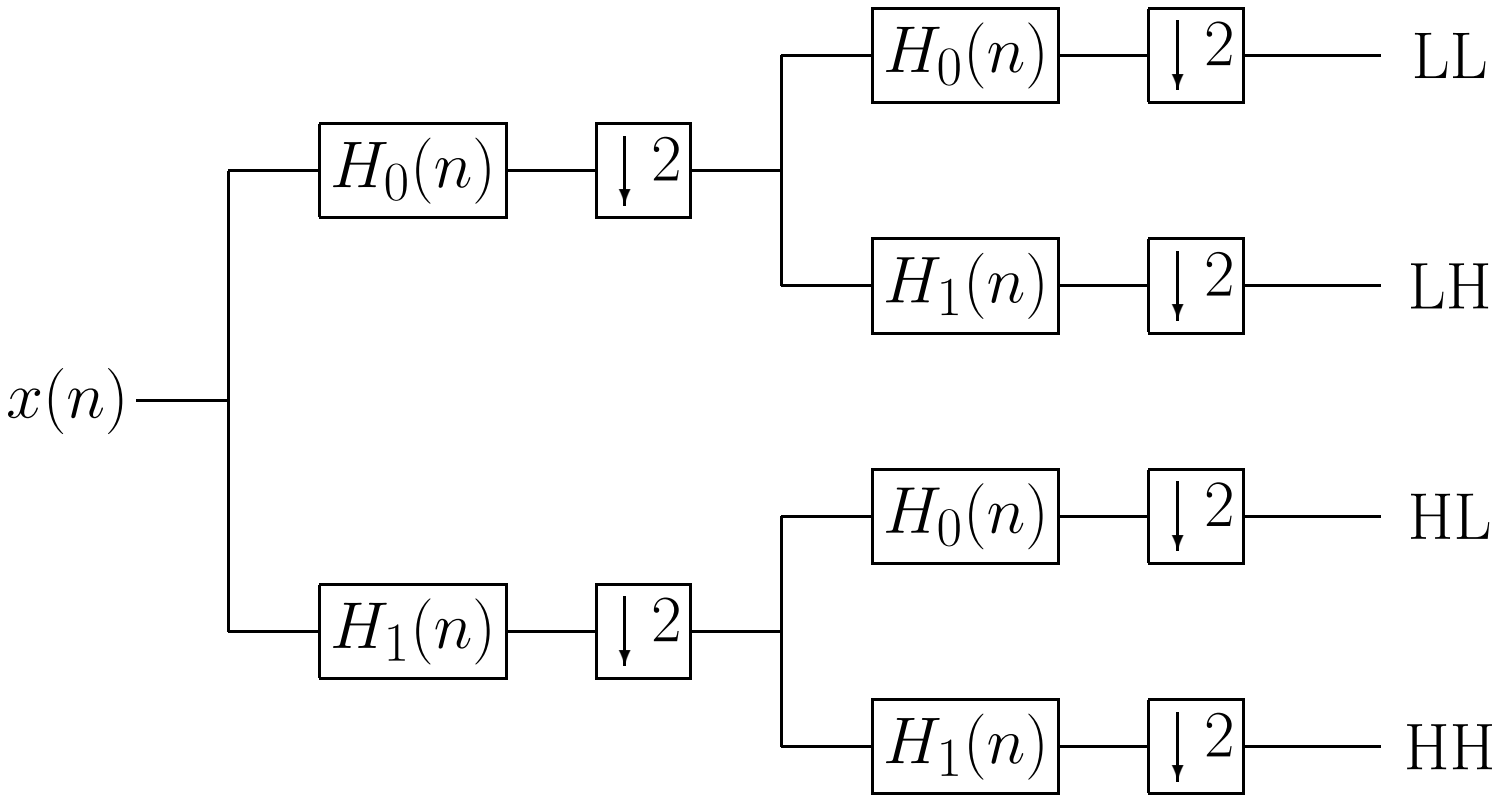
Subband/pyramid/wavelet coding (late 1980s)
(Lots of people, e.g., Barnwell, Smith, Adelson, Burt, Mallat, Vetterli)
Filter and downsample.
Traditional uniform subbands: MUSICAM digital audio (European standard)

Use filter bank of roughly orthogonal equal bandwidth filters.
(Classical subband decomposition.)
Generalize to nonorthogonal with *perfect reconstruction property*

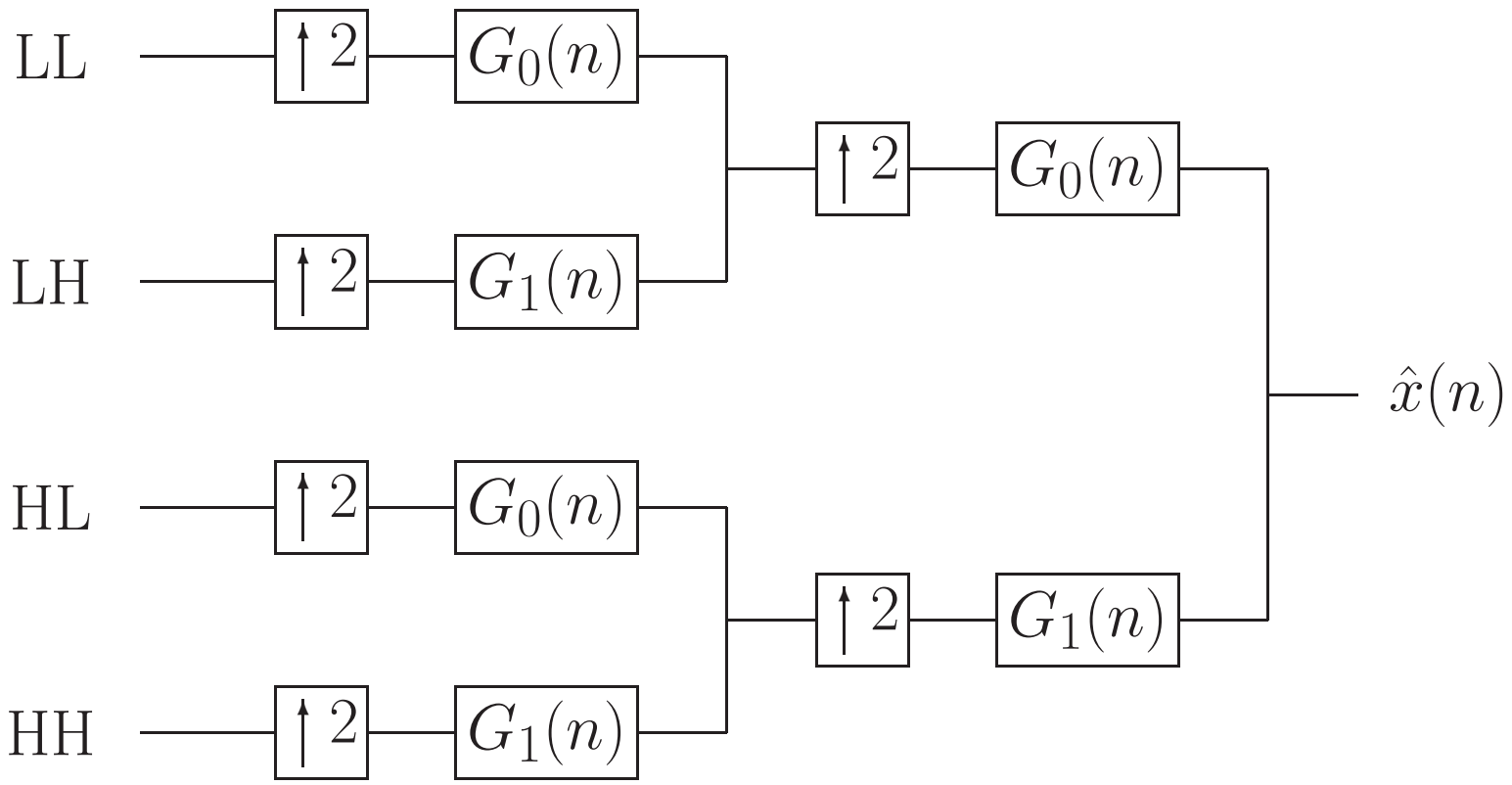
Two dimensional subband decomposition: Separately decompose rows and columns.

Wavelet, pyramid, or octave subbands: Only split low frequency band each time.

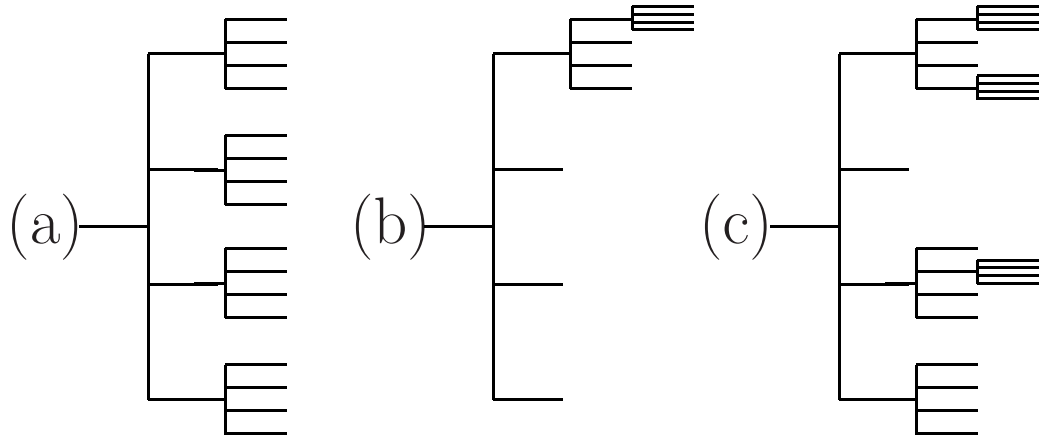

X_n
H_1
$\downarrow M$
Q_1
$\uparrow M$
H_1
X_{1n}
Y_{1k}
$\hat{Y}_{1k}$
$\tilde{X}_{1n}$
$\hat{X}_{1n}$
H_2
$\downarrow M$
Q_2
$\uparrow M$
H_2
X_{2n}
Y_{2k}
$\hat{Y}_{2k}$
$\tilde{X}_{2n}$
$\hat{X}_{2n}$
$+$
$\hat{X}_n$
H_M
$\downarrow M$
Q_M
$\uparrow M$
H_M
X_{Mn}
Y_{Mk}
$\hat{Y}_{Mk}$
$\tilde{X}_{Mn}$
$\hat{X}_{Mn}$



One stage of subband decomposition with 2-D separable filters

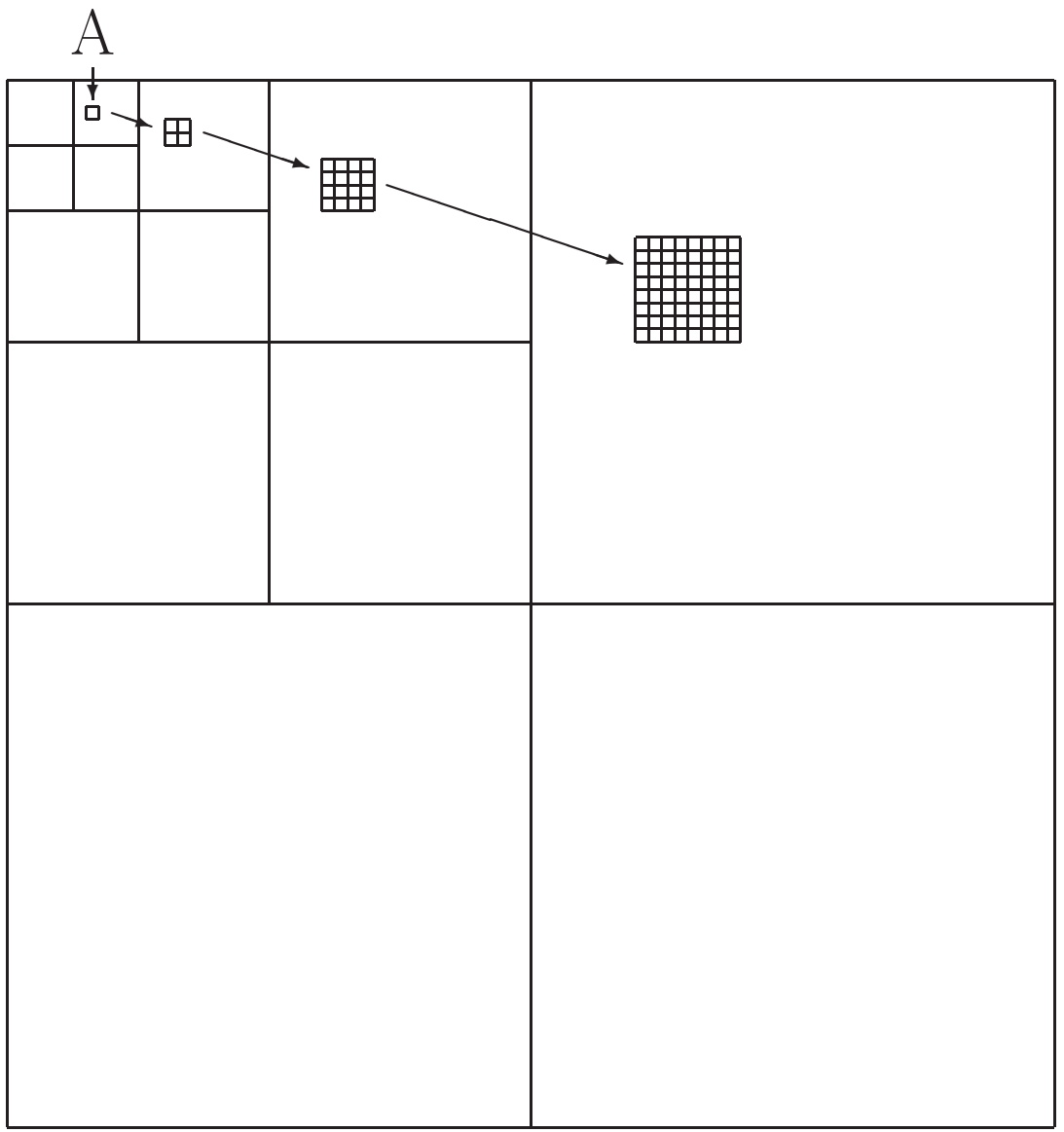


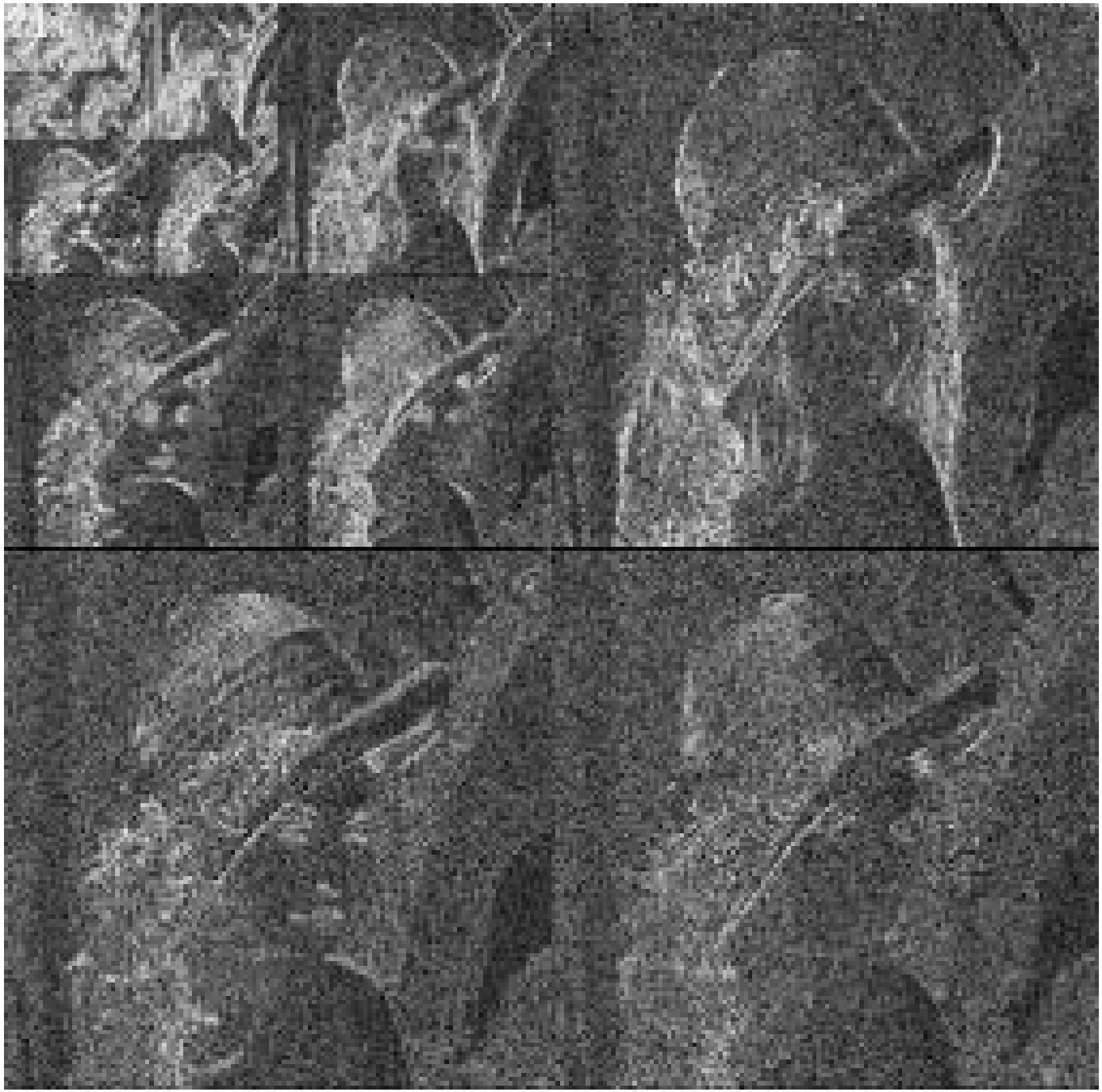
One stage of subband reconstruction with 2-D separable filters



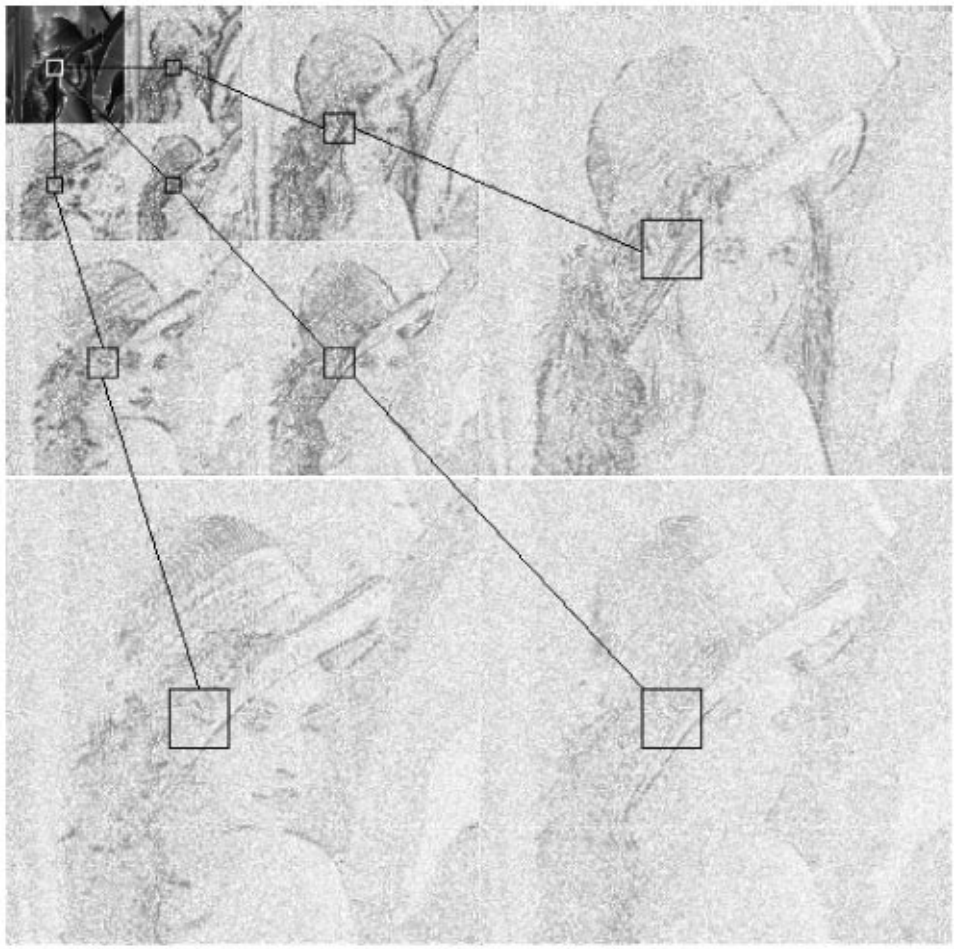
(a) Two stages of uniform decomposition leading to 16 uniform subbands, (b) Three stages of pyramidal decomposition, leading to 10 octave-band subbands, (c) a wavelet packet

For each subband in an image (except the very lowest one, and the three highest ones), each coefficient A is related to exactly 4 coefficients in the next higher band.


A



Lena
decomposed
Note:
multiresolution



Parent-descendants

Question is then: How code?

I.e., how quantize wavelet coefficients and then lossy code them?

Early attempts used traditional (JPEG-like & VQ) techniques.
Not impressive performance.

Breakthrough came with Knowles and Shapiro (1993):
Embedded zerotree coding.

Idea: Coefficients have a tree structure, high frequency wavelet pixels are descended from low frequency pixels.

If a low frequency pixel has low energy (and hence will be quantized into zero), then the same will likely be true for its descendents.
$\Rightarrow$ zerotree encoding

Successively send bits describing higher energy coefficients to greater accuracy.

Additional bits describe most important coefficients with increasing accuracy and less important coefficients when their magnitude becomes significant.

Variations:

- SPHIT algorithm of Said and Pearlman. Optimally trade off rate and distortion. Lower complexity, better performance.
- CREW algorithm of RICOH. Similar to SPHIT, but uses integer arithmetic and codes to lossless.
- Others: better handling of space/frequency correlation or more careful modeling/adaptation, but more complex: Xiong, Ramchandran, Orchard, Joshi, Jafarkani, Kasner, Fischer, Farvardin, Marcellin, Bamberger, Lopresto

Alternative: Stack-run coding (Villasenor) Simple runlength coding (Golomb-Rice codes) of separate subbands. Slightly inferior performance to SPHIT, but better than Shapiro and much lower complexity.
See Villasenor's web site for comparisons.

(a) Original LENA

(b) Rate = 0.5 bpp, PSNR = 37.2 dB

(c) Rate = 0.25 bpp, PSNR = 34.1 dB

(d) Rate = 0.15 bpp, PSNR = 31.9

SPHIT Examples

SPHIT Lena with single bit error

Packetizable zerotree coder,
53-byte packets, 10PSNR=27.7dB
(P. Cosman, UCSD)

"Second Generation" Codes

Model-based coding. Enormous compression reported, but need a Cray to compress. Likely to become more important.

Idea: Typically segment or identify objects/textures/background within image, apply custom compression to each.
EPFL, T. Reed (UCD), X. Wu

Model-based VQ: LPC

Very different methods used for low-rate speech. Many are based on classic LPC or Parcor speech compression

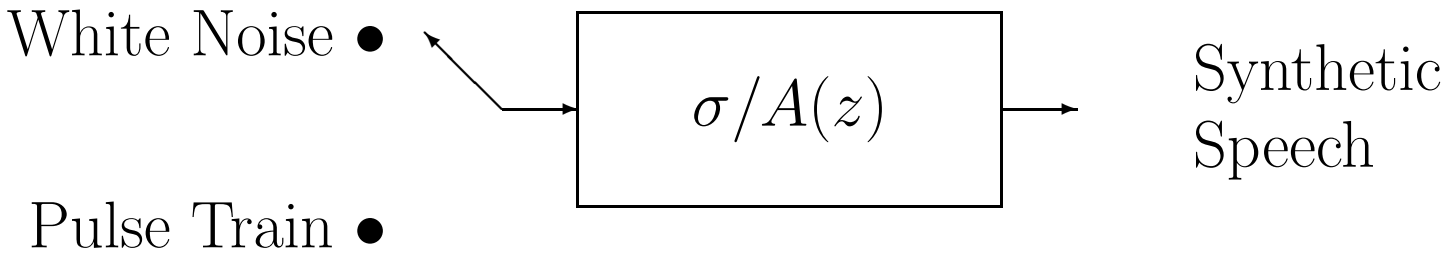
Itakura and Saito, Atal and Shroeder, Markel and (A.H.) Gray

First very low bit rate speech coding based on sophisticated SP, implementable in DSP chips.
Speak and Spell, TI
NTT, Bell, TI and Signal Technology

Model short speech segment $\mathbf{x}$ ($\approx 25 - 50$ms) by autoregressive process (all-pole) $\sigma/A(z)$ with

$$A(z) = 1 + \sum_{k=1}^{M} a_k z^{-k}.$$

Reproduction synthesized by driving filter with noise or periodic pulse train.



Without quantization, filter is determined by linear prediction arguments or maximum likelihood arguments $\Rightarrow$
Levinson-Durbin algorithm on sample autocorrelation.

Mathematically equivalent to a minimum disortion selection of A and σ to match sample autocorrelation of input segment (Itakura-Saito distortion)

Parameters quantized as scalars or vectors (LPC-VQ).
Several parameter vectors possible:

- reflection coefficients
- autocorrelation coefficients
- inverse filter coefficients
- cepstrum
- line spectral pairs (LSP)

Mathematically equivalent when no quantization,
but quantization errors effect each differently.

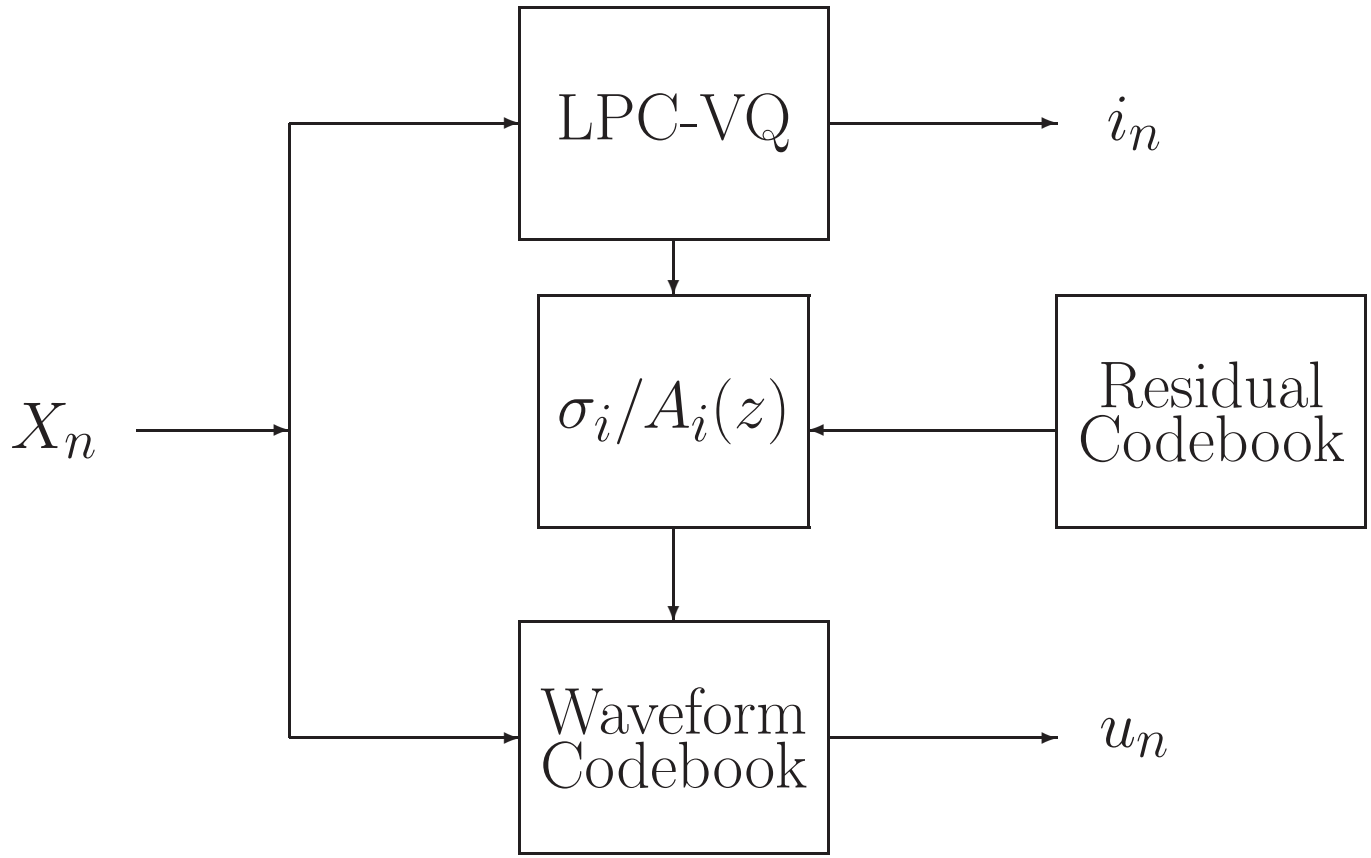
Hybrid Model + Closed Loop Residual CELP

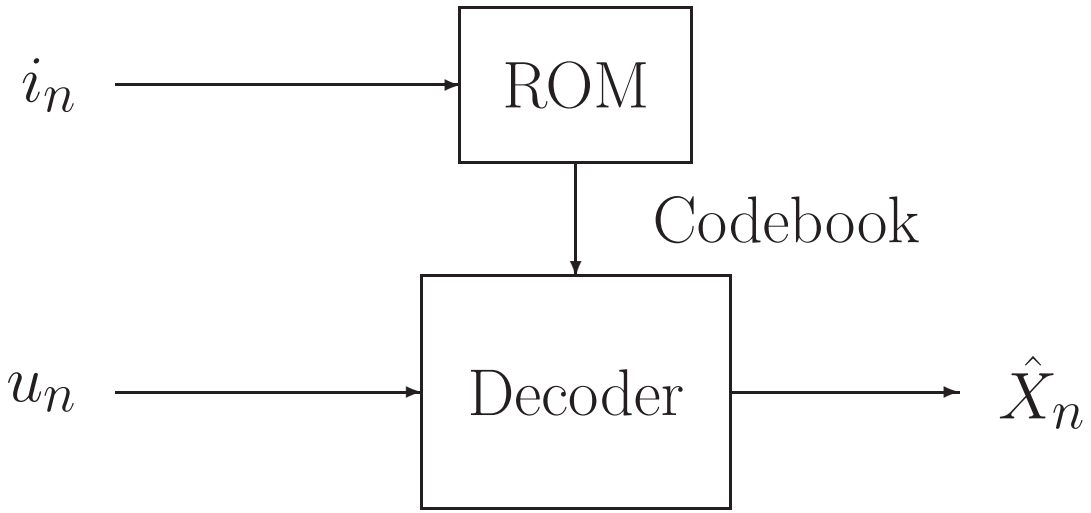
(Inverted RELP, modern version of analysis-by-synthesis)

Codebook for residuals. Given input vector, find residual codeword that produces best closed loop match out of LPC (or LPC-VQ) to original input.

Effectively produces a waveform codebook for searching by passing residual codewords through model linear filter.

Stewart et al. (1981, 1982), Shroeder and Atal (1984, 1985)


LPC-VQ
i_n
X_n
$\sigma_i/A_i(z)$
Residual Codebook
Waveform Codebook
u_n
Encoder



Decoder

Atal, Gersho & others made practicable by incorporating perceptual weighting filters, i.e., perceptually meaningful distortion measures in search.

Measuring Quality in

Lossy Compressed Images

Digitization, digital acquisition, compression, and DSP (even "enhancement") change content.
Allegedly for the better, but how *prove* quality or utility equivalent or superior?

SNR Can be useful for screening among algorithms and tells something about relative performance.
Possibly sufficient for "substantial equivalence" of minor variation.

Subjective Ratings Ask users to rate the "quality" of the image on a scale of 1 — 5.

Diagnostic Accuracy Simulate clinical application: screening or diagnosis.

Statistical analysis to test hypothesis that "image type A as good as or better than image type B" for specific tasks.

Selected Standards

- ITU (International Telecommunications Union)
  - telecommunications (telephony, teleconferencing, . . .)
  - low-delay two-way communications
  - recommendations; nations are members
- ISO (International Standards Organization)
  - computer information technology
  - asymmetric one-way communications
  - standards; standards bodies are members

ITU "p× 64" recommendations

(1990)

- for video telephony and video conferencing at ISDN rates (p× 64 kb/s)
  - H.261 video coding at p× 64 kb/s
  - G.711 audio coding at p× 64 kb/s (8kHz μ-law PCM)
  - H.221 multiplexing, H.242 control
  - H.320 capability negotiation

ITU, ISO 'JBIG (Joint Bi-level Experts Group)

(1993)

Transmission storage of binary images at multiple spatial resolutions, multiple amplitude resolutions

Context adaptive arithmetic coding

ITU "LBC" recommendations

(1997)

- for video telephony at modem rates

  (14.4 – 28.8 Kb/s and higher)

  - H.263 video coding at p× 64 kb/s
  - G.723 audio coding at 6.3 kb/s

    (ACELP)
  - H.224 multiplexing, H.245 control
  - H.324 capability negotiation

ISO "MPEG-1" standard

(1992)

- for entertainment video ad CDROM rates

  ($\approx$ 1.5 Mb/s)

  - video coding at $\approx$ 1.2Mb/s
  - audio coding at $\approx$ 64Kb/s per channel (stereo)
  - systems (multiplexing, buffer management, synchronization, control, capability negotiation)

Frame sequence: I B B P B B P B B I

I= Intraframe coding (à la JPEG)

P= Predictive coding (motion estimation)

B= Bidirectional prediction (forward or backward)

Motion estimation: Predict macroblocks by matching luminance

for minimum MSE, code resulting residual by $\approx$ JPEG

ISO "MPEG-2" standard

(1994)

- for digital mastering and distribution of entertainment video at BISDN rates

  ($\approx$ 1.5 – 10 Mb/s and higher)

  - video coding at $\approx$ 1.5 – 10 Mb/s
  - audio coding at $\approx$ 64Kb/s per channel (up to 5 channels)
  - systems (multiplexing, buffer management, synchronization, control, capability negotiation)

ISO "MPEG-4" standard

(1997)

- for interactive information access from computers, set-top boxes, mobile units
  ($\approx$ 10 Kb/s – 1.5 Mb/s)

Intersection of computers (interactivity), entertainment (A/V data), and telecommunications (networks, including wireless).

Applications

- Interactive information access
  - home shopping, interactive instruction manuals, remote eductation, travel brochures
- Multiuser communication
  - conferencing with shared resources, chat rooms, games
  - Editing and recomposition; search

- Scalable to channel BW and decoder capabilities
- Error resilience
- Coding of arbitrarily shaped regions
- Composition of Visual Objects and Audio Objects

Part IV: Optimal Compression

A survey of some theory and implications and applications

General Goal: Given the signal and the channel, find an encoder and decoder which give the "best possible" reconstruction.

To formulate as precise problem, need

- probabilistic descriptions of signal and channel

  (parametric model or sample training data)

- possible structural constraints on form of codes

  (block, sliding block, recursive)

- quantifiable notion of what "good" or "bad" reconstruction is

  (MSE, P_e)

Mathematical: Quantify what "best" or optimal achievable performance is.

History

1948 Shannon published "A Mathematical Theory of Communication"

Followed in 1959 by "Coding theorems for a discrete source with a fidelity criterion"

Probability theory, random process theory, and ergodic theory $\Rightarrow$ fundamental theoretical limits to achievable performance:

Given probabilistic model of an *information source* and *channel* (for storage or transmission),

how reliably can the source signals be communicated to a receiver through a channel?

Channel characterized by *Shannon channel capacity* C which indicates how fast information can be reliably sent though it, possibly subject to constraints

Information source characterized by *Shannon distortion-rate function* $D(R)$, the minimum average distortion achievable when communicating at a rate R

Dual: *rate-distortion function* $R(D)$, the minimum average rate achievable when communicating at a rate D.

Both quantities are defined as optimizations involving information measures and distortion measures such as mean squared error or signal-to-noise ratios.

Shannon proved coding theorems showing these information theoretic quantities were equivalent to the corresponding operational quantities.

In particular: joint source/channel coding theorem:

Joint Source/Channel Coding Theorem

If you transmit information about a source with DRF $D(R)$ over a channel with capacity C, then you will have average distortion at least $D(C)$. (Negative theorem)

Furthermore, if complexity and cost and delay are no problem to you, then you can get as close to $D(C)$ as you would like by suitable coding (Positive theorem).

Hitch: Codes may be extremely complex with large delay.

Shannon's distortion rate theory: Fixed R, asymptotically large k.

Alternative: Bennett/Zador/Gersho asymptotic quantization theory; fixed k, asymptotically large R.

Bennett(1948), Zador (1963), Gersho (1979), Neuhoff and Na (1995)

What does theory have to do with the real world?

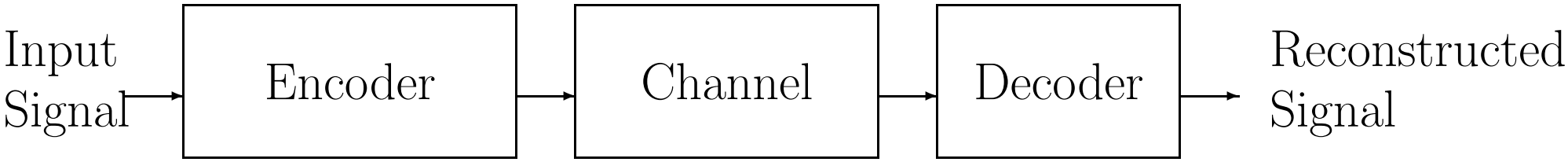
- Benchmark for comparison for real communication systems.
- Has led to specific design methods and code structures.

Impossible to do better than Shannon's bounds (unless you cheat or have inaccurate model).

Theory does not include many practically important issues, e.g.:

- Coding delay
- Complexity
- Robustness to source variations
- Effects of channel errors

Over the years, methods evolved to come close to these limits in many applications.



Famous Figure 1

Classic Shannon model of point-to-point communication system

General Goal: Given the signal and the channel, find encoder/decoder which give "best possible" reconstruction.

To formulate as precise problem, need

- probabilistic descriptions of signal and channel

  (parametric model or sample training data)

- possible structural constraints on form of codes

  (block, sliding block, recursive)

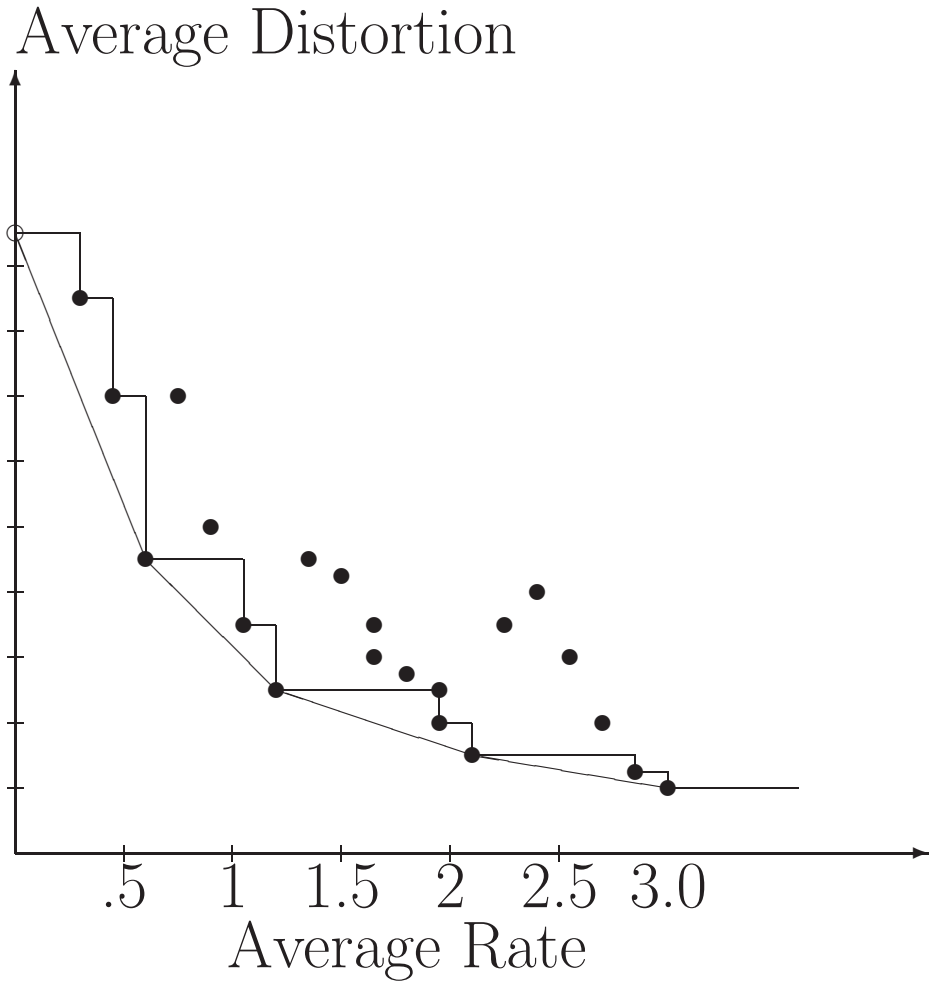
- quantifiable notion of "good" or "bad": distortion

Typical assumptions:

- Signal is discrete time or space (e.g., already sampled), modeled as a vector-valued random process $\{X_n\}$, common distribution P_X. Noiseless channel.
- Code maps blocks or vectors of input data into binary strings, i.e., a VQ $(\alpha, \mathcal{W}, \beta)$
- Performance measured by $D(\alpha, \beta) = E[d(X, \beta(\alpha(X))]$, $R(\alpha, \mathcal{W}) = E(r(X))$

(Theory can be extended to other structures, e.g., sliding-block codes, trellis and tree codes, finite-state codes)

⋆ Every code yields point in distortion/rate plane: (R, D).



Interested in undominated points in D-R plane: For given rate (distortion) want smallest possible distortion (rate) $\Leftrightarrow$ optimal codes

Optimization problem:

- Given R, what is smallest possible D?
- Given D, what is smallest possible R?
- $\star$ Lagrangian formulation: What is smallest possible $D + \lambda R$?

Make more precise and carefully describe the various optimization problems.

- Rate-distortion approach: Constrain $D(\alpha, \beta) \leq D$. Then optimal code $(\alpha, \mathcal{W}, \beta)$ minimizes $R(\alpha, \mathcal{W})$ over all allowed codes.

  operational rate-distortion function

$$\hat{R}(D) = \inf_{\alpha, \mathcal{W}, \beta : D(\alpha, \beta) \leq D} R(\alpha, \mathcal{W}).$$

- Distortion-rate approach: Constrain $R(\alpha, \mathcal{W}) \leq R$. Then optimal code $(\alpha, \mathcal{W}, \beta)$ minimizes $D(\alpha, \beta)$ over all allowed codes.

  operational distortion-rate function

$$\hat{D}(R) = \inf_{\alpha, \mathcal{W}, \beta : R(\alpha, \mathcal{W}) \leq R} D(\alpha, \beta).$$

- Lagrangian approach: Fix Lagrangian multiplier λ.

  Optimal code $(\alpha, \mathcal{W}, \beta)$ minimizes

$$J_\lambda(\alpha, \mathcal{W}, \beta) = D(\alpha, \beta) + \lambda R(\alpha, \mathcal{W})$$

  over all allowed codes.

  operational Lagrangian distortion function

$$\hat{J}_\lambda = \inf_{\alpha, \mathcal{W}, \beta} E\rho_\lambda(X, \beta(\alpha(X))) =$$

$$\inf_{\alpha, \mathcal{W}, \beta} [D(\alpha, \beta) + \lambda R(\alpha, \mathcal{W})].$$

First two problems are duals, all three are equivalent. E.g.,

Lagrangian approach yields R-D for some D or D-R for some R.

Lagrangian approach effectively uncostrained minimization of modified distortion $J_\lambda = E\rho_\lambda(X, \alpha(X))$ where

$$\rho_\lambda(X, \alpha(X)) = d(X, \beta(\alpha(X)) + \lambda l(\alpha(X))$$

Note: Lagrangian formulation has recently proved useful for rate control in video, optimize distortion + λ rate.

Usually wish to optimize over constrained subset of computationally reasonable codes, or implementable codes.

Example: fixed rate codes

Require all words in $\mathcal{W}$ = range space of $\mathcal{W}$ to have equal length.

Then $r(X)$ constant and problem simplified.

Eases buffering requirments when use fixed-rate transmission media, Eases effects of channel errors

Lagrangian approach $\Leftrightarrow$ distortion-rate approach for fixed R.

Optimalility Properties and Quantizer Design

Extreme Points

Can actually find global optima for two extreme points:

$\lambda \to 0$: all of the emphasis on distortion

force 0 average distortion

Minimize rate only as an afterthought.

$(0, R)$ 0 distortion, corresponds to lossless codes

Shannon's noiseless coding theorem:

$$\frac{1}{k}H(X) \leq \hat{R}(0) < \frac{1}{k}H(X) + \frac{1}{k}$$

Huffman code achieves minimum

Note: $H = \infty$ if X continuous.

$\lambda \to \infty$

All emphasis on rate, force 0 average rate

minimize distortion as an afterthought.

$(D, 0)$: 0 rate, no bits communicated.

Optimal peformance:

$$\hat{D}(0) = \inf_{y \in \hat{A}} E[d(X, y)].$$

centroid

For example, $A = \hat{A} = \Re^k$, squared error distortion, i.e.,

$$d(x, y) = ||x - y||^2 = (x - y)^t(x - y)$$

Then

$$\hat{D}(0) = \inf_y E[||X - y||^2]$$

is achieved by

$$\min_y{}^{-1} E[||X - y||^2] = E(X)$$

General case: Generalized Lloyd algorithm:

- optimize the encoder for the decoder:

  use a minimum distortion mapping

- optimize the decoder for the encoder:

  assign conditional centroids as reproductions

- optimize the codeword assignment for the encoder:

  use a Huffman or arithmetic coding of the encoder outputs.

Extension of Lloyd's algorithm to vectors.

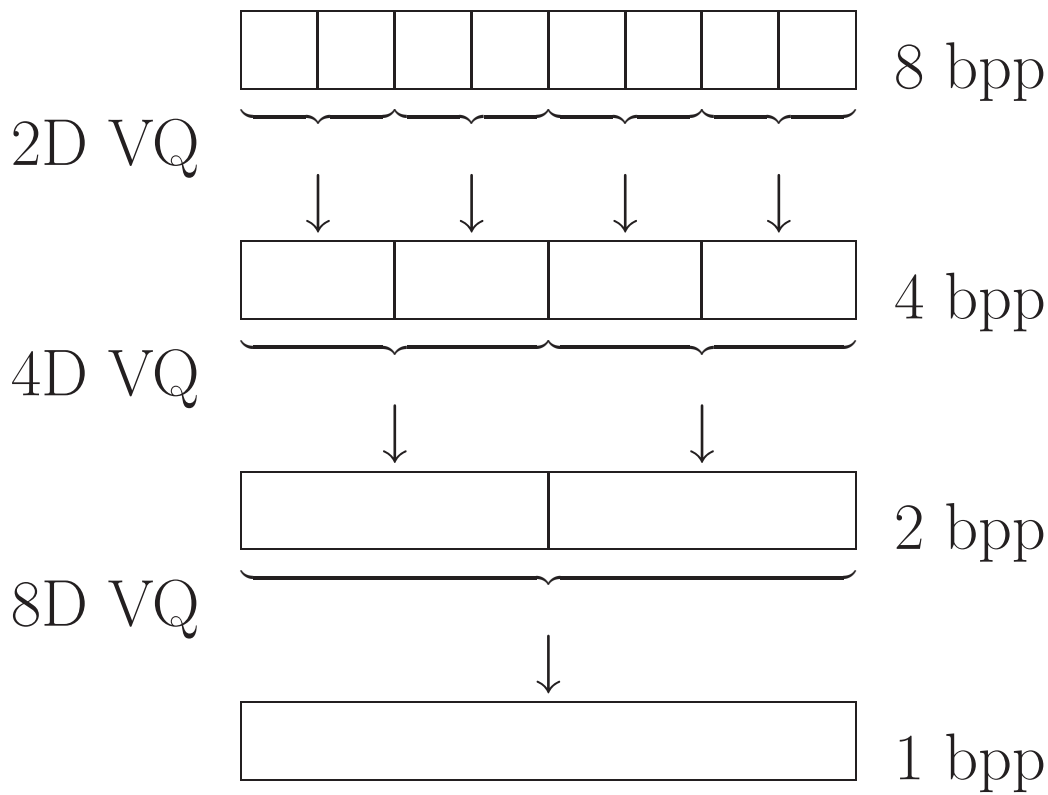
"Vanilla" vector quantization.

Will yield descent design algorithm for complete code

Hierarchical (Table Lookup) VQ

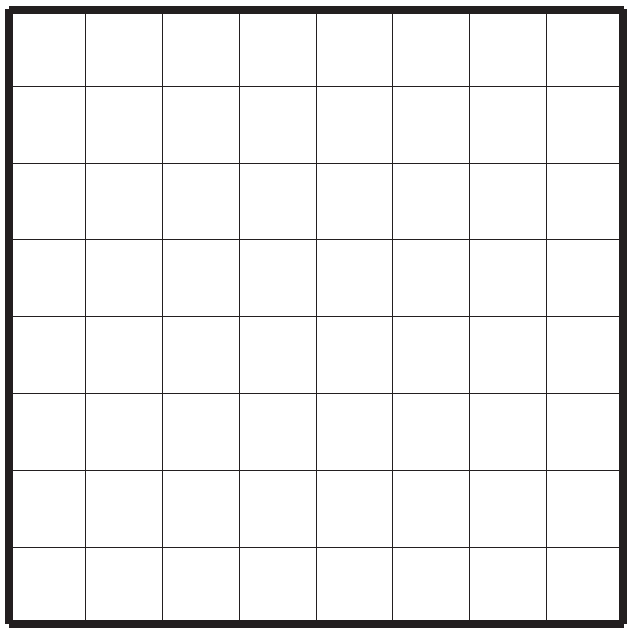
(Chang et al. (1985 Globecom)) Vector quantization can involve significant computation at encoder because of searching, but decoder is simple table lookup.

Can also make encoder table lookup and hence trade computation for memory if use hierarchical table lookup encoder:

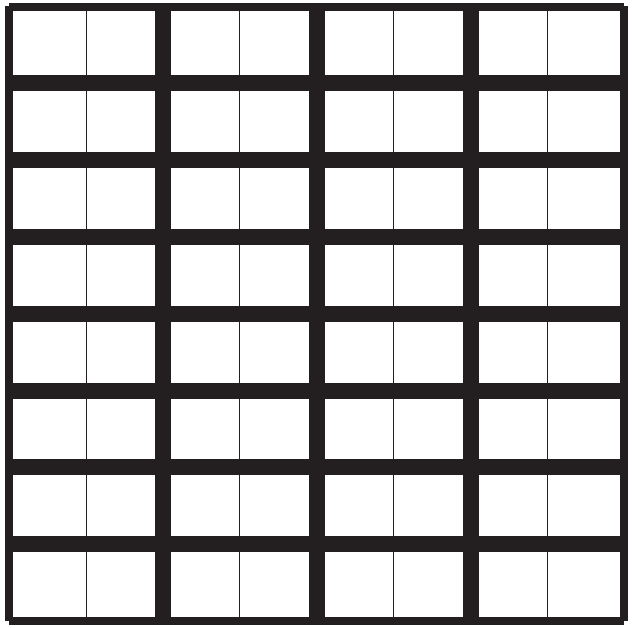


Each arrow is implemented as a table lookup having 65536 possible input indexes and 256 possible output indexes. The table is populated by an off-line minimum distortion search.

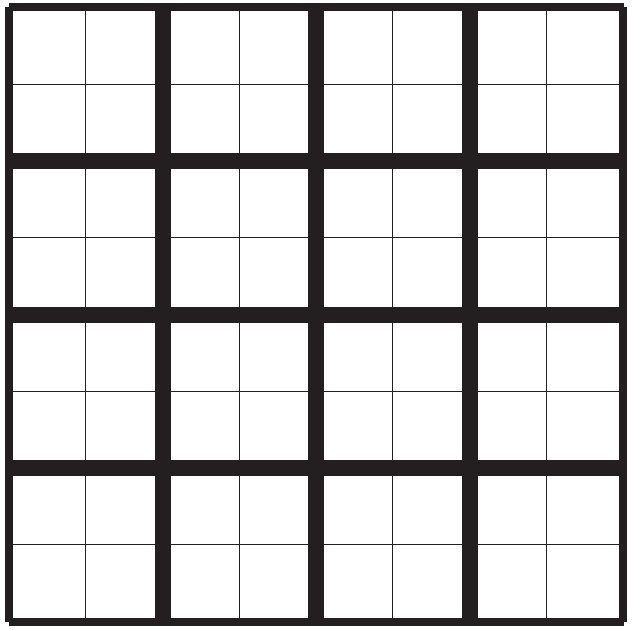
For 2D (image/video) compression: Each stage codes to smaller codewords of rate 8 bits/vector into single larger codeword of rate 8 bits/vector; $256^2 \rightarrow 256$ table lookup.



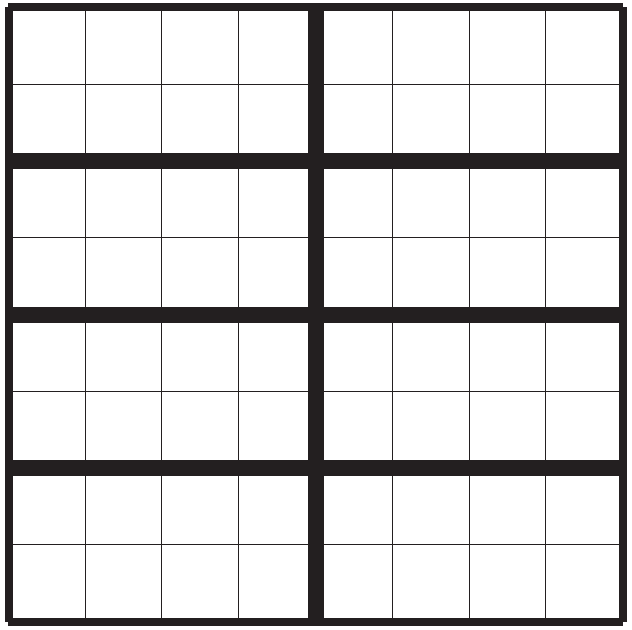
1D
8 bpp



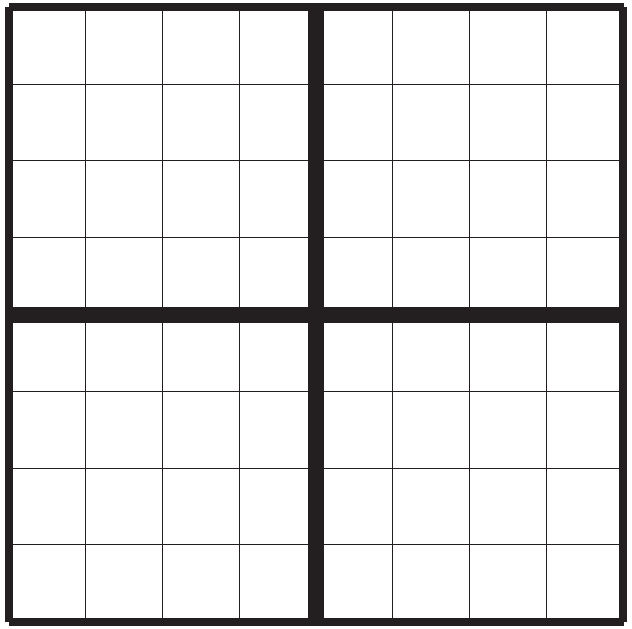
2D
4 bpp



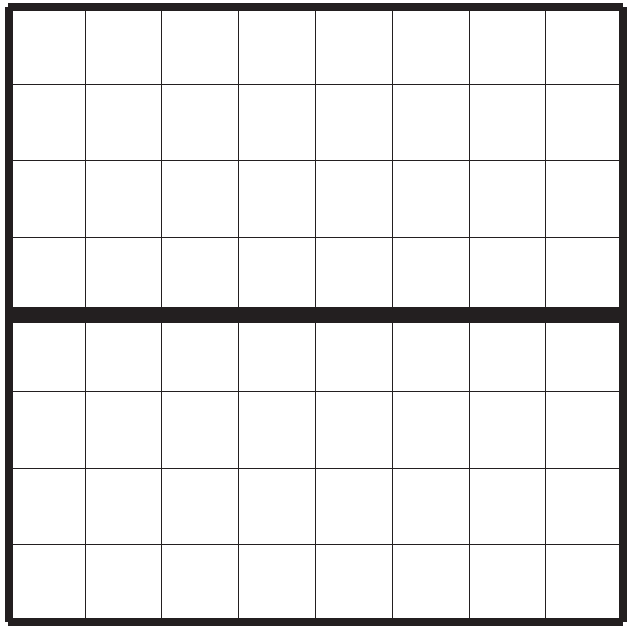
4D
2 bpp



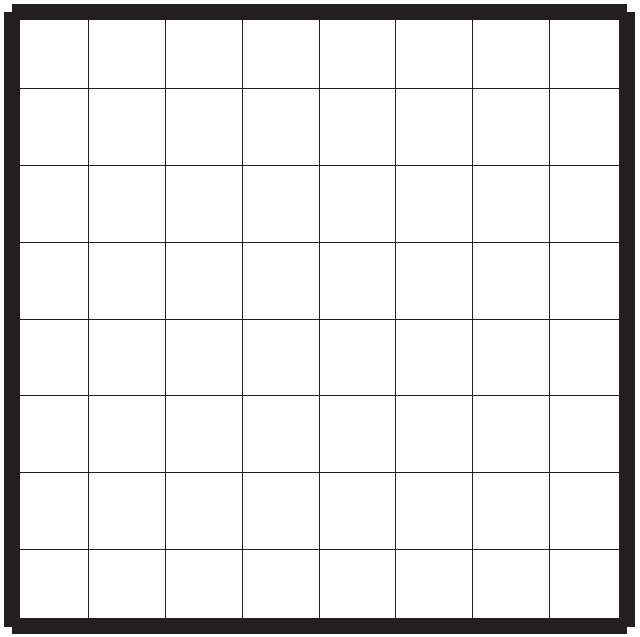
8D
1 bpp



16D
0.5 bpp



32D
0.25 bpp



64D
0.125 bpp

Can combine with transform coding and perceptual distortion measures to improve.

(Chou, Vishwanath, Chaddha)

Yields very fast nearly computation-free compression.

Increasing Dimension

$X = X^k$, k allowed to vary.

Can define optima for increasing dimensions:

$\hat{D}_k(R)$, $\hat{R}_k(D)$, and $\hat{J}_{\lambda,k}$

Quantities are subadditive & can define asymptotic optimal performance

$$\bar{\hat{D}}(R) = \inf_k \hat{D}_k(R) = \lim_{k\to\infty} \hat{D}_k(R)$$
$$\bar{\hat{R}}(D) = \inf_k \hat{R}_k(D) = \lim_{k\to\infty} \hat{R}_k(D)$$

$$\bar{\hat{J}}_\lambda = \inf_k \hat{J}_{\lambda,k} = \lim_{k\to\infty} \hat{J}_{\lambda,k}.$$

Shannon coding theorems relate these operationally optimal performances (impossible to compute) to information theoretic minimizations.

$$\bar{\hat{D}}(R) = \bar{D}(R), \quad \bar{\hat{R}}(D) = \bar{R}(D)$$

i.e., operational DRF (RDF) = Shannon DRF (RDF)

Need some definitions to define Shannon DRF.

Average mutual information between two discrete random variables X and Y is

$$I(X;Y) = H(X) + H(Y) - H(X,Y) =$$

$$\sum_{x,y} \Pr(X = x, Y = y) \log_2 \frac{\Pr(X = x, Y = y)}{\Pr(X = x) \Pr(Y = y)}$$

Definition extends to continuous alphabets by maximizing over all quantized versions:

$$I(X;Y) = \sup_{\alpha_1, \alpha_2} I(\alpha_1(X); \alpha_2(Y))$$

Shannon channel capacity: Channel described by family of conditional probability distributions $P_{Y^k|X^k}$, $k = 1, 2, \ldots$

$$C = \lim_{k\to\infty} \sup_{P_{X^k}} \frac{1}{k} I(X^k; Y^k)$$

Shannon distortion-rate function: Source described by family of source probability distributions P_{X^k}, $k = 1, 2, \ldots$

$$D(R) = \lim_{k\to\infty} \inf_{P_{Y^k|X^k}: I(X^k;Y^k)\le kR} \frac{1}{k} E[d_k(X^k, Y^k)]$$

Lagrangian a bit more complicated, equals $D + \lambda R$, where $D = \bar{D}(R)$ at point where λ is the magnitude of the slope of the DRF.

Shannon's distortion rate-theory is asymptotic in that its positive results are for a fixed rate R and asymptotically large block size k (and hence large coding delay)

Another approach: Bennett/Zador/Gersho asymptotic quantization theory.

Fixed k, asymptotically large R.

High rate or low distortion theory.

Theories consistent when both R and k asymptotically large.

But keep in mind: the real world is not asymptopia!

High Rate (Asymptotic) Quantization Theory

Recall basics (with slighlty different notation):

X, $f_X(x)$, MSE

VQ Q or $(\tilde{\alpha}, \psi, \tilde{\beta})$: Reproduction codebook

$\mathcal{C} = \{\beta(i);\ i = 1, \ldots, N\}$

Encoder partition $\mathcal{S} = \{S_i;\ i = 1, \ldots, N\}$

$S_i = \{x : Q(x) = \beta(i)\}$, $P_X(S_i) = \Pr(X \in S_i)$

Average distortion:

$$\begin{aligned} D(Q) &= \frac{1}{k} E[||X - Q(X)||^2] \\ &= \frac{1}{k} \sum_{i=1}^{N} \int_{S_i} ||x - \beta(i)||^2 f_X(x)\, dx \end{aligned}$$

Average rate: Fixed rate code: $R(Q) = k^{-1} \log N$

Variable rate code:

$R(Q) = H_k(X) = -\frac{1}{k} \sum_{i=1}^{N} P_X(S_i) \log P_X(S_i)$

As in ECVQ, constraining entropy effectively constraining average length.

Operational distortion-rate functions:

$$\hat{D}_{k,f}(R) = \inf_{Q \in \mathcal{Q}_f : R(Q) \leq R} D(Q)$$
$$\hat{D}_{k,v}(R) = \inf_{Q \in \mathcal{Q}_v : R(Q) \leq R} D(Q)$$

where $\mathcal{Q}_f$ and $\mathcal{Q}_v$ are the sets of all fixed and variable length quantizers, respectively.

Bennett assumptions:

- N is very large
- f_X is smooth (so that Riemann sums approach Riemann integrals and mean value theorem of calculus applies)
- The total overload distortion is negligible.
- The volumes of all bounded cells are tiny.
- The reproduction codewords are the Lloyd centroids of their cell.

Assumptions are combined with calculus and approximations to develop asymptotic (large N) expressions of the form

$$D \approx E[\frac{m(X)}{\lambda(X)^{2/k}}]2^{-2R},$$

where λ is a *quantizer point density function* and m is an *inertial profile* of the quantizer.

If VQ is a uniform lattice with cell edge length Δ, this becomes famous $D \approx \frac{\Delta^2}{12}$ commonly used (and misused) in quantization analysis.

Result has many applications:

Can use to evaluate scalar and vector quantizers, transform coders, tree-structured quantizers. Can provide bounds.

Can show, for example, that in the variable rate case, entropy coded lattice vector quantizers are nearly optimal when the rate is large.

Used to derive a variety of approximations in compression, e.g., bit allocation and "optimal" transform codes.

Approximations imply good code properties:

- For the fixed rate case, cells should have roughly equal partial distortion.
- For the variable rate case, cells should have roughly equal volume.

In neither case should you try to make cells have equal probability (maximum entropy).

Asymptotic theory implies that for iid sources, high rates (low distortion)

$$\frac{\hat{D}_{1,v}(R)}{\hat{D}_{k,v}(R)} \leq 1.533dB$$

Or, equivalently, for low distortion

$$\hat{R}_{1,v}(D) - \hat{R}_{k,v}(D) \leq 0.254 \text{ bits}$$

famous "quarter bit" result.

Suggests at high rates there may be little to be gained by using vector quantization (but still need to use vectors to do entropy coding!)

Bennett theory can be used to "model" the quantization noise when the conditions hold: the quantization noise under suitable assumptions is approximately uncorrelated with the input and approximately white.

This led to the classic "additive white noise model" for quantization noise popularized by Widrow.

Problems have resulted from applying this approximation when the Bennett assumptions do not hold.

Final Observations

Lossless Coding is well understood and there are many excellent algorithms. Thought near Shannon limit in many applications.

Research:

- off-line coding
- transparent incorporation into other systems
- hardware implementations, esp. parallel
- ever better prediction

Lossy Coding

Wavelets + smart coding generally considered best approach in terms of trading off complexity, fidelity, and rate, but DCT Transform coding still being tweaked & variations stay competive. Inertia resists change.

Uniform quantizers and lattice VQ dominate quantization step, and nearly optimal for high rate variable rate systems.

Rate-distortion ideas can be used to improve standards-compliant and wavelet codes. (e.g., Ramchandran, Vetterli, Orchard)

Research:

- Joint frequency/time(space) quantization.
- Segmentation coding (MPEG 4 structure): different compression algorithms for different local behavior.
- Multimode/multimedia compression
- Multiple distortion measures (include other signal processing) E.g., MSE and P_e in classification/detection.
- Compression on networks
- Joint source and channel coding.

- Universal lossy coding (Ziv, Chou, Effros)
- Supporting theory
- Perceptual coding
- Combining compression with other signal processing, e.g., classification/detection or regression/estimation
- Quality evaluation of compressed images: distortion measures vs. human evaluators

Research vs. Standards

- Standards have application targets

  Research should have some farther targets.

- Research should advance understanding, improve component technology.

- Research within standards, e.g., MPEG 4.

- Standards are less important with downloadable code.

Acknowledgements

These notes have benefited from conversations with many individuals, comments from students, and presentations of colleagues. Particular thanks go to Phil Chou, , Eve Riskin, Pamela Cosman, Sheila Hemami, Khalid Sayood, and Michelle Effros.

Some Reading on Data Compression

Introduction to Data Compression, K. Sayood. Morgan Kauffman, 1996.

M. Rabbani and P. Jones, *Digital Image Compression*, Tutorial Texts in Optical Engineering, SPIE Publications, 1991.

A. N. Netravali and B. G. Haskell, *Digital Pictures: Representation and Compression*,

Plenum Press, 1988.

Timothy C. Bell, John G. Cleary, Ian H. Witten. *Text compression*, Prentice Hall, Englewood Cliffs, N.J., 1990.

R. J. Clarke, *Transform Coding of Images*, Academic Press, 1985.

Vector Quantization and Signal Compression, A. Gersho and R.M. Gray. Kluwer Academic Press, 1992.

N. S. Jayant and P. Noll, *Digital Coding of Waveforms*, Prentice-Hall, 1984.

William B. Pennebaker and Joan L. Mitchell. *JPEG still image data compression standard*, New York : Van Nostrand Reinhold, 1992.

J. Storer, *Data Compression*, Computer Science Press, 1988.

T. J. Lynch, *Data Compression: Techniques and Applications*, Lifetime Learning, Wadsworth, 1985.

P.P. Vaidyanathan, *Multirate Systems & Filter Banks*, 1993

Historical Collections

N. S. Jayant, Ed., *Waveform Quantization and Coding* IEEE Press, 1976.

L. D. Davisson and R. M. Gray, Ed's., *Data Compression*, Benchmark Papers in Electrical Engineering and Computer Science, Dowden, Hutchinson,& Ross, 1976.