INTRODUCTION TO COMPUTER MUSIC

# AUDIO DATA COMPRESSION

Roger B. Dannenberg

Professor of Computer Science, Art, and Music

## General Compression Techniques

- Coding Redundancy
  - optimize the coding of symbols or samples
  - quantize in lossy coding
  - e.g. variable length coding
- Intersample Redundancy
  - represent runs of pixels, silence in audio
  - frequency domain representations
  - object descriptions
- Psycho-Perceptual Redundancy
  - masking effects
  - noise and distortion thresholds

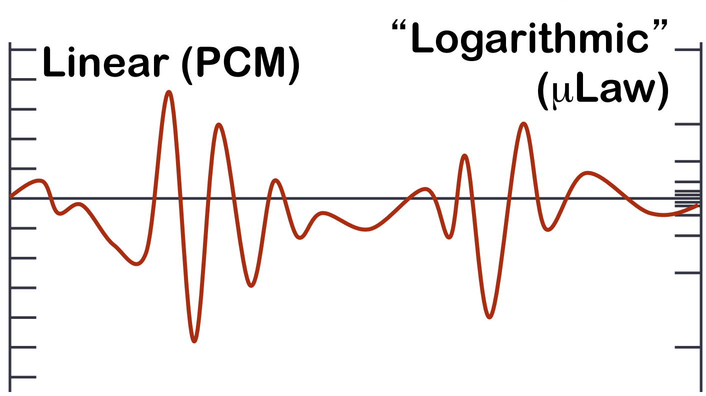
## Coding Redundancy

- PCM – Pulse Code Modulation
  - linear encoding of sampled audio
  - this is the reference “uncompressed” representation
- U-law – logarithmic encoding to 8 bits:
  - sign bit, 3 exponent bits, 4 mantissa bits
  - scaled to span 13 bit dynamic range
- A-law
  - similar, but 12 bit dynamic range





## μLaw vs PCM



## Implementation

- Encoding and decoding by table lookup.
  - Encoding table is lossy, many-to-one mapping.
  - Decoding (256 values) is one-to-one.
- Common in telephony
- What about more bits for music?
  - Studies show that 16-bit PCM is better.
  - Other techniques (perceptual coding) *much* better if you can't afford 16-bit PCM.

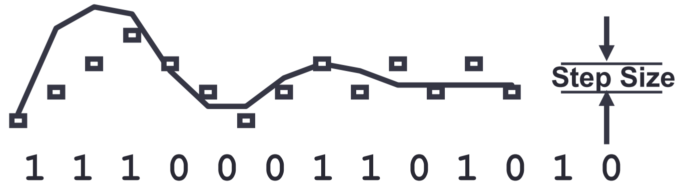
  

## Intersample Redundancy Examples

- DPCM – delta PCM
  - encode sample differences
  - about 6dB improvement over PCM for speech
- ADPCM – Adaptive Delta PCM
  - adapt quantization size
  - about 10-11dB improvement over PCM for speech

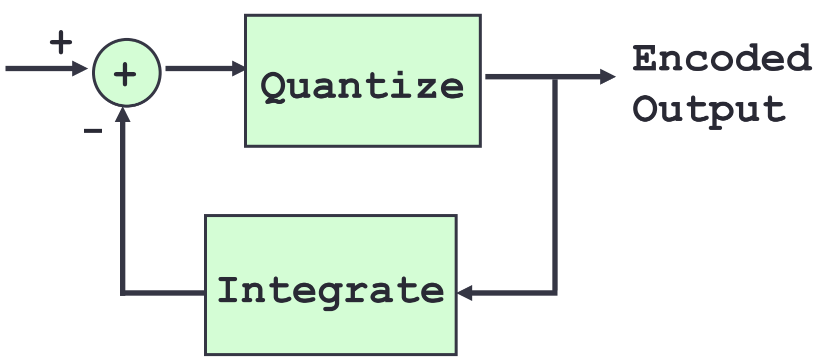
## DPCM


Step Size
1 1 1 0 0 0 1 1 0 1 0 1 0





## DPCM

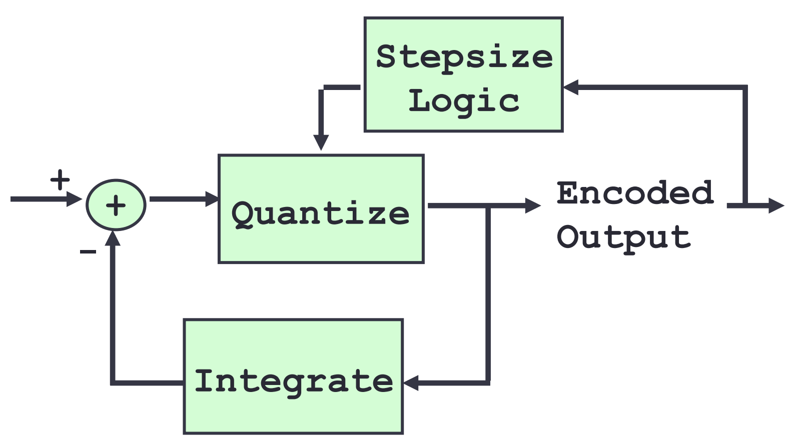

+
+
–
Quantize
Encoded Output
Integrate

# ADPCM






# ADPCM Coder



## Adaptive Prediction

- Signals are redundant: future samples are correlated with previous samples
- DPCM uses 1st order predictor, e.g. the previous sample
- Nth order predictor forms weighted sum of past N samples
- Need to adapt the weights → adaptive prediction
- Gain is about 3 or 4 dB, little gain beyond 4th or 5th order





## Why not store delta instead of PCM?

- Delta can be big
  - Cannot assume smaller samples
  - Therefore no compression (without loss)
- For music, experiments show PCM is better.

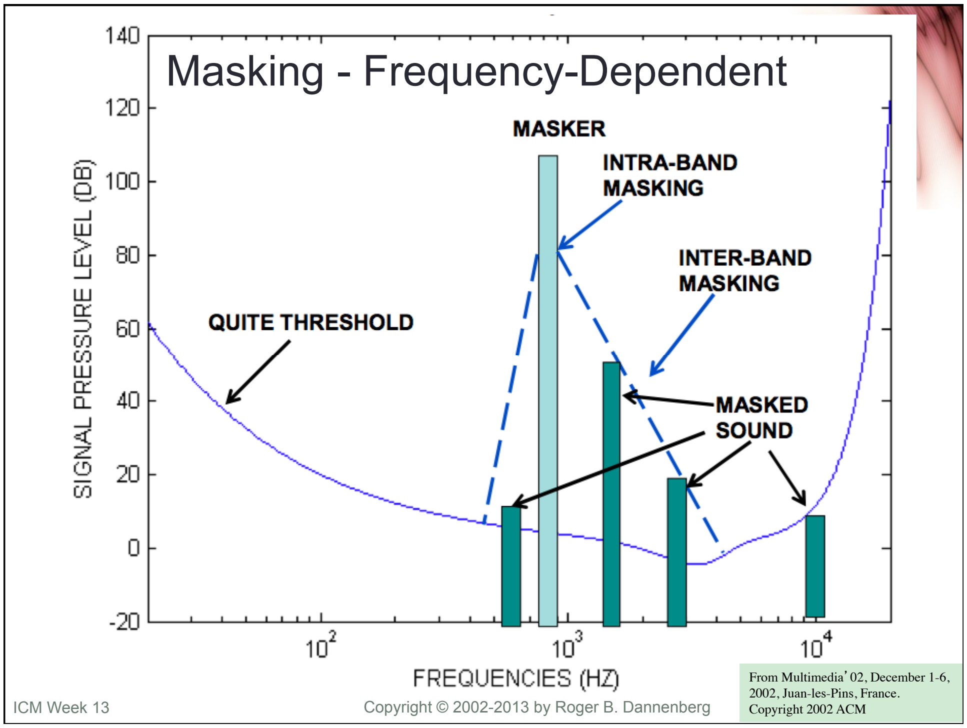
## Review

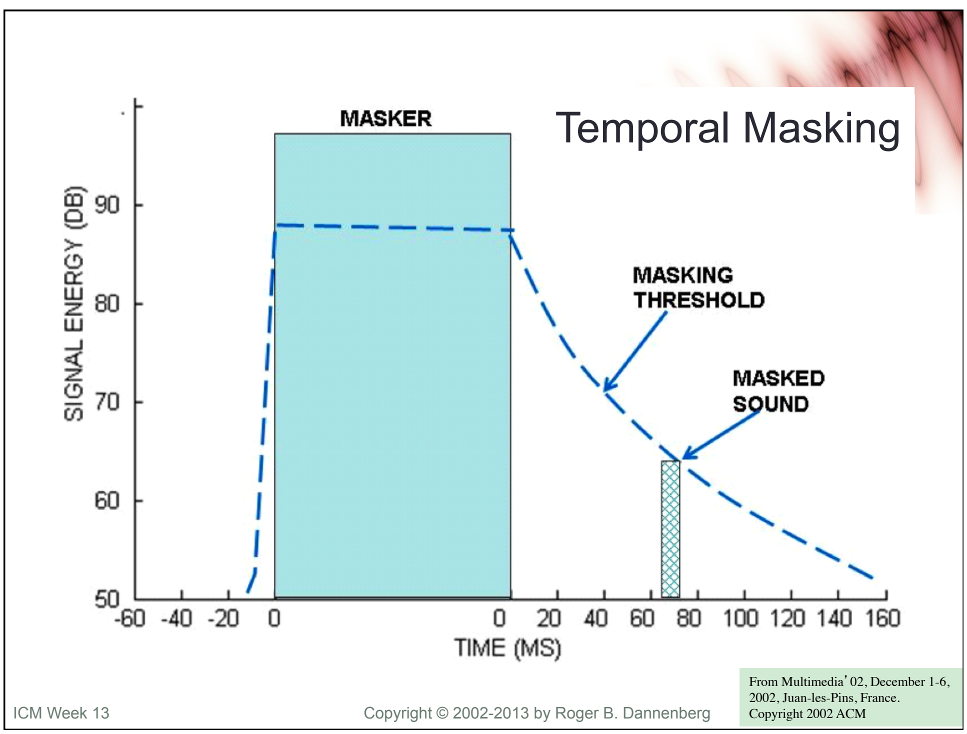
- Uncompressed audio = PCM
  - linear, independent integer samples
- Coding Redundancy
  - U-LAW, A-LAW
- Intersample Redundancy
  - DPCM
  - ADPCM
- Psycho-Perceptual Redundancy





## PSYCHO-PERCEPTUAL CODING AND MP3


Masking - Frequency-Dependent
MASKER
INTRA-BAND MASKING
INTER-BAND MASKING
QUITE THRESHOLD
MASKED SOUND
SIGNAL PRESSURE LEVEL (DB)
140
120
100
80
60
40
20
0
-20
$10^2$
$10^3$
$10^4$
FREQUENCIES (HZ)
ICM Week 13
Copyright © 2002-2013 by Roger B. Dannenberg
From Multimedia' 02, December 1-6, 2002, Juan-les-Pins, France. Copyright 2002 ACM


Temporal Masking
MASKER
MASKING THRESHOLD
MASKED SOUND
SIGNAL ENERGY (DB)
90
80
70
60
50
-60 -40 -20 0
0 20 40 60 80 100 120 140 160
TIME (MS)
ICM Week 13
Copyright © 2002-2013 by Roger B. Dannenberg
From Multimedia' 02, December 1-6, 2002, Juan-les-Pins, France. Copyright 2002 ACM

## Masking, Perceptual Coding

- Ear as filter bank
- Loud sound in a channel masks soft sounds in adjacent channels
- Quantize as much as possible – using masking to hide quantization effects
- Frequency shift bands and sample at minimum sample rate

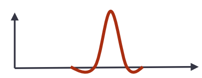
## Some Insight on Frequency Domain and Coding

- PCM is time domain – not so much sample-to-sample correlation. Changing all the time.
- Spectral data tends to be more static. The spectrum at time *t* is a good predictor for spectrum at time *t*+1.
- In tonal music, spectrum is relatively sparse: non-zero only where there are sinusoids. This is easier to encode efficiently.

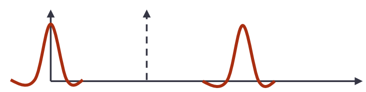
  

# A Note on Banded Processing

- You might think when a signal is separated into N bands, you' d get N x Original Rate
- Spectral view of a band:



- Recall that multiplication by a sinusoid will create shifted copies of original spectrum:



- Now, low-pass filter to get just the lower band:

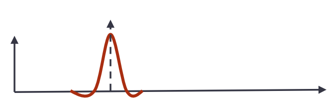
- In theory, you can encode the signal as N bands at 1/N the sample rate, so the total information is the same.




# Banded Processing, Continued

- To recover original band, start with encoded signal

- Multiply by sine to frequency shift



## MP3 – MPEG Audio Layer 3

- Part of broader video compression standard
- A standard for consistent decoding
- Encoding algorithm details not specified
- 6-to-1 encoding of 48kHz audio gives no perceptual difference in extensive tests
- Fraunhofer claims 12-to-1




## Step 1: Filter Bank

- Signal is filtered into 32 bands of equal width.
- Each band is sub-sampled by factor of 32.
- Some simplifications:
  - Bands have overlap
  - Bands are wider than sample-rate/32, so subsampling causes aliasing
  - Filter and inverse are slightly lossy

## Step 2: Psychoacoustic Model

- How much can each band be quantized?
- Take 1024 point FFT of audio, group spectrum into *critical bands*
- Identify *tonal components* (sinusoids) and assign tonal index to each critical band
  (Note: noise masking and tonal masking differ)
- From spectrum, compute masking threshold for each of 32 subbands.
- Compute signal-to-mask ratio for each of 32 subbands.




## Coding

- Each subband is transformed with modified discrete cosine transform (MDCT) of length 18 or 6 subband samples.
- Essentially, this is a short-term frequency representation of the subbands.
- This allows for more efficient coding, e.g. if only one sinusoid is in the subband, one MDCT coefficient should be high and others low.
- Quantize MDCT coefficients according to signal-to-masking ratio.

## Coding (2)

- There are 576 coefficients per frame (18 MDCT coefficients x 32 subbands)
- Order them by increasing frequency.
- Highest frequencies tend to be zeros (no bits expended here)
- Next is a run of -1, 0, and 1, encoded 4 at a time into alphabet of 81 symbols.
- Remaining values coded in pairs.




## Huffman Coding

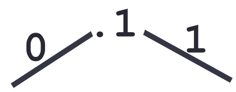
- Most popular technique for removing coding redundancy
- Gives smallest number of code symbols per source symbol
- Symbols are coded one at a time.
- Codes are variable length;
  chosen to minimize expected bit length

## Huffman Code Example

| A | B | C | D | E | F |
|---|---|---|---|---|---|
| .2 | .1 | .1 | .03 | .5 | .07 |

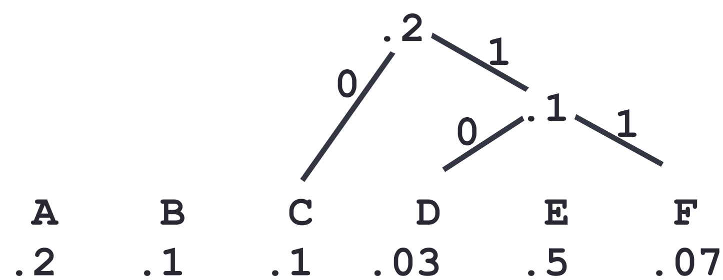



## Huffman Code Example




| A | B | C | D | E | F |
|---|---|---|---|---|---|
| .2 | .1 | .1 | .03 | .5 | .07 |

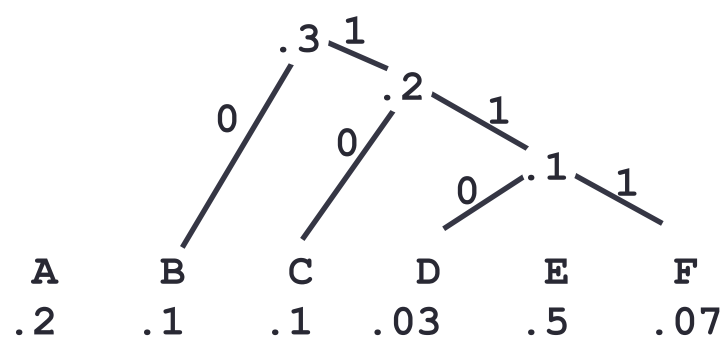
## Huffman Code Example



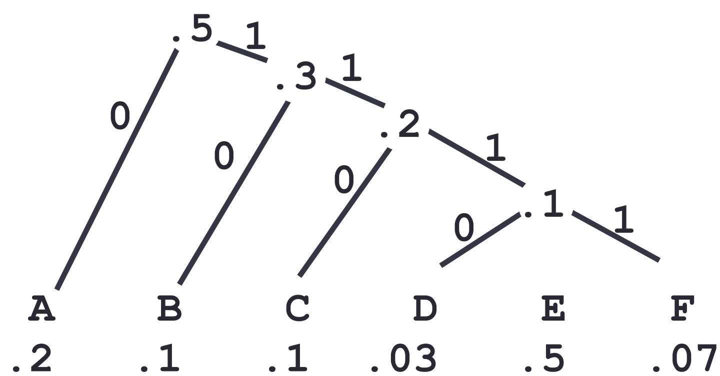




## Huffman Code Example



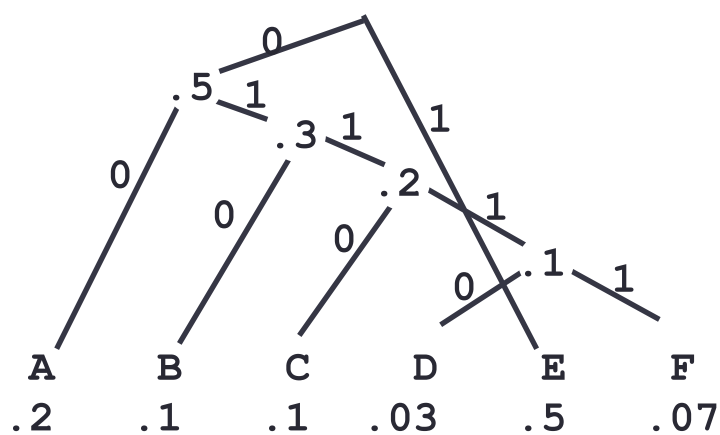
## Huffman Code Example




ICM Week 13



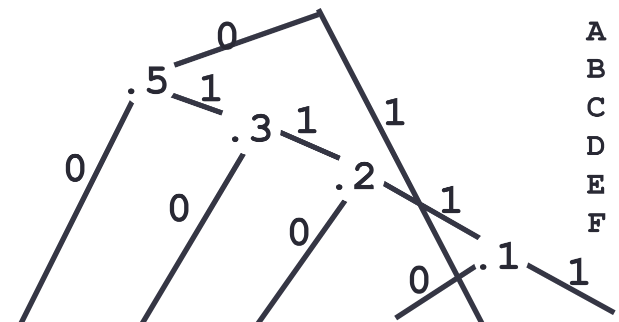
## Huffman Code Example




ICM Week 13

Copyright © 2002-2013 by Roger B. Dannenberg

# Huffman Code Example




| | |
|---|---|
| A | 00 |
| B | 010 |
| C | 0110 |
| D | 01110 |
| E | 1 |
| F | 01111 |

| A | B | C | D | E | F |
|---|---|---|---|---|---|
| .2 | .1 | .1 | .03 | .5 | .07 |




Huffman Code Example

**Average Length =**

2(.2)+3(.1)+4(.1)+
5(.03)+1(.5)+5(.07)

= 2.1 bits/symbol

| A | B | C | D | E | F |
|---|---|---|---|---|---|
| .2 | .1 | .1 | .03 | .5 | .07 |

## Tricky Concept

- Symbols (A, B, C, …) can represent numbers or vectors, e.g.
  A = [-1, -1, -1, -1]
  B = [-1, -1, -1, 0]
  C = [-1, -1, -1, 1]
  D = [-1, -1, 0, -1]
  E = [-1, -1, 0, 0]
  F = [-1, -1, 0, 1]
  etc.




## Bit Reservoir

- If a frame is encoded with fewer bits than the compressed data rate, e.g. 64kbps, allows, bits are "donated" to "bit reservoir."
- Later, encoder can use bits from "bit reservoir", temporarily exceeding maximum data rate, to do the best job of encoding audio.

# Bits Allocation

- Bits are allocated to bands where quantization noise exceeds masking threshold.
- Iterative allocation of bits followed by recalculation of noise.
- This makes encoding slow.




# Summary

- Masking reduces our ability to hear "everything"
- Including quantization noise
- MP3 descriptions are highly quantized (but these are frequency domain descriptions)
- After quantization, use run-length coding and Huffman coding to encode descriptions efficiently

# MORE AUDIO COMPRESSION

LPC
Physical Models – Analysis/Synthesis
Music Notation

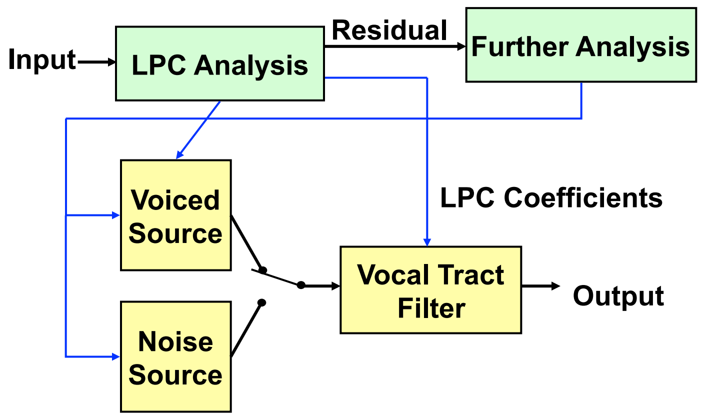




## More Audio Compression

- LPC: Linear Predictive Coding
  - voice is source (voiced/unvoiced) + filter
  - voice pitch and filter coefficients change slowly
  - 1-2K bits/s (Sambur), typical 2400-7200 (Rabiner & Schafer)
- Another example of intersample redundancy
  - Here, we have an "object model" that is used to estimate samples

# Linear Predictive Coding in Practice







# Physical Models / Speech Synthesis

- Future speech compression technology
- Muscles control is low bandwidth
- Speech sounds highly constrained
- Idea: transmit control signals for speech then simulate acoustics of speech production
- May need one-time transmission of parameters to characterize speaker

## Audio Compression Examples

*Note: This slide is not in the video lecture. If you would like to hear some speech compression examples, there is a link to them in the online course page containing the video lecture.*

- Reference (uncompressed, 16-bit)
- Downsampled to 11kHz, 16-bit
- 8-bit uLaw
- IMA ADPCM
- TrueSpeech 8.5 (8.5 Kbits/sec, proprietary coder)




## Music

- Music Notation
  - compact, symbolic representation
  - does not capture performance information
  - expressive "performance" not fully automated
- Performance Information
  - MIDI bandwidth is 3KB/s, or 180KB/min
  - More typical: 3KB/minute, 180KB/hour
    - Complete Scott Joplin: 1MB
    - Output of 50 Composers (400 days of music): 500MB (1 CD-ROM)
  - synthesis of acoustic instruments is a problem

## Summary

- Three kinds of redundancy:
  - Coding
  - Intersample
  - Psycho-Perceptual
- uLaw, ADPCM, etc. – simple, fast, but not high quality or high compression
- MP3 and related schemes – general, high quality and compression, fairly high computation
- Analysis to Model and Resynthesis – special purpose (e.g. speech, musical instruments), potentially very high quality and compression, but still research