Introduction to Audio Compression and Representation

Perry R. Cook
Princeton Computer Science
(also Music)

Audio Compression Overview

- Compression in General
- Waveform Sampling, Storage, etc.
- Limits of Human Audio Perception
- Sound and Music Representation
- Audio Compression Techniques
- Two Contrasting Compressors
- References and Resources
Compression in General: Why Compress?

So Many Bits, So Little Time (Space)

- CD audio rate: $2 \times 2 \times 8 \times 44100 = 1,411,200$ bps
- CD audio storage: 10,584,000 bytes / minute
- A CD holds only about 70 minutes of audio
- An ISDN line can only carry 128,000 bps

Security: Best compressor removes all that is recognizable about the original sound

Graphics people eat up all the space

Compression in General

Classical Data Compression View:

Take advantage of

- Redundancy/Correlation
- Statistics (Local / Global)
- Assumptions / Models

Problem: Much of this doesn’t work directly on sound waveform data
Waveform Sampling and Playback

- **Sample and Hold**
  
  *Sample Rate vs. Aliasing*

- **Quantize**
  
  *Word Size vs. Quantization Noise*

- **Reconstruct: Hold and Smooth (filter)**
  
  *Filter Order vs. Error and Latency*

---

Waveform Sampling:
Quantization

**Quantization**

*Introduces Noise*

*Examples: 16, 12, 8, 6, 4 bit music*

*16, 12, 8, 6, 4 bit speech*
Audio Compression

*Limits of Human Perception*
- Time, Frequency, Amplitude, Masking, etc.

*Survey of Audio Compression Techniques*
- Perception-Based Compression
- Production-Based Compression
- (Event-Based Compression)

*Two Specific Compression Algorithms*
- Production Model-Based Speech Coder
- Frequency Transform (Subband) Coder

Views of Sound

- Sound is Perceived: Perception-Based
  Psychoacoustically Motivated Compression
- Sound is Produced: Production-Based
  Physics/Source Model Motivated Compression
- Music(Sound) is Performed/Published/Represented:
  Event-Based Compression
- Sound is a Waveform / Statistical Distribution / etc.
  (these are not very good ideas in general, unless we get lucky (LPC))
Psychoacoustics

Limits of Human Hearing

- Time Domain Considerations
- Frequency Domain (Spectral) Considerations
- Amplitude vs. Power
- Masking in Time and Frequency Domains
- Sampling Rate and Signal Bandwidth

Limits of Human Hearing

Time and Frequency

Events longer than 0.03 seconds are resolvable in time
shorter events are perceived as features in frequency

20 Hz. < Human Hearing < 20 KHz.
(for those under 15 or so)

“Pitch” is PERCEPTION related to FREQUENCY
Human Pitch Resolution is about 40 - 4000 Hz.
Limits of Human Hearing

**Amplitude or Power??**

- “Loudness” is PERCEPTION related to POWER, not AMPLITUDE.
- Power is proportional to (integrated) square of signal.
- Human Loudness perception range is about 120 dB, where +10 dB = 10 x power = 20 x amplitude.
- Waveform shape is of little consequence. Energy at each frequency, and how that changes in time, is the most important feature of a sound.

Limits of Human Hearing

**Waveshape or Frequency Content??**

- Here are two waveforms with identical power spectra, and which are (nearly) perceptually identical:

<table>
<thead>
<tr>
<th>Wave 1</th>
<th>Wave 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Magnitude Spectrum of Either</td>
<td>Magnitude Spectrum of Either</td>
</tr>
</tbody>
</table>
**Limits of Human Hearing**

**Masking in Amplitude, Time, and Frequency**

– Masking in Amplitude: Loud sounds ‘mask’ soft ones.  
   Example: Quantization Noise

– Masking in time: A soft sound just before a louder sound is more likely to be heard than if it is just after. 
   Example (and reason): Reverb vs. “Preverb”

– Masking in Frequency: Loud ‘neighbor’ frequency masks soft spectral components. Low sounds mask higher ones more than high masking low.

---

**Limits of Human Hearing**

**Masking in Amplitude**

Intuitively, a soft sound will not be heard if there is a competing loud sound. Reasons:

- Gain controls in the ear
  
  *stapedes reflex and more*

- Interaction (inhibition) in the cochlea

- Other mechanisms at higher levels
Limits of Human Hearing

Masking in Time

- In the time range of a few milliseconds:
  - A soft event following a louder event tends to be grouped perceptually as part of that louder event
  - If the soft event precedes the louder event, it might be heard as a separate event (become audible)

Limits of Human Hearing

Masking in Frequency

Only one component in this spectrum is audible because of frequency masking
Sampling Rates

For Cheap Compression, Look at Lowering the Sampling Rate First

44.1kHz 16 bit = CD Quality
8kHz 8 bit MuLaw = Phone Quality

Examples:

Music: 44.1, 32, 22.05, 16, 11.025kHz
Speech: 44.1, 32, 22.05, 16, 11.025, 8kHz

Views of Sound (revisited)

Two (mainstream) views of sound and their implications for compression

1) Sound is Perceived
The auditory system doesn’t hear everything present
– Bandwidth is limited
– Time resolution is limited
– Masking in all domains

2) Sound is Produced
– “Perfect” model could provide perfect compression
Perceptual Models

*Exploit masking, etc., to discard perceptually irrelevant information.*

- Example: Quantize soft sounds more accurately, loud sounds less accurately

**Benefits:**
Generic, does not require assumptions about what produced the sound

**Drawbacks:**
Highest compression is difficult to achieve

Production Models

*Build a model of the sound production system, then fit the parameters*

- Example: If signal is speech, then a well-parameterized vocal model can yield highest quality and compression ratio

**Benefits:**
Highest possible compression

**Drawbacks:**
Signal source(s) must be assumed, known, or identified
MIDI and Other ‘Event’ Models

**Musical Instrument Digital Interface**

Represents Music as Notes and Events
and uses a synthesis engine to “render” it.

An Edit Decision List (EDL) is another example.

A history of source materials, transformations, and processing steps is kept. Operations can be undone or recreated easily. Intermediate non-parametric files are not saved.

Event Based Compression

**MIDI and Other Scorefiles**

- A Musical Score is a very compact representation of music
- Even the score itself can be compressed further

**Benefits:** Highest possible compression

**Drawbacks:** Cannot guarantee the “performance”

Cannot assure the quality of the sounds

Cannot make arbitrary sounds
Event Based Compression

*Enter General MIDI*

- Guarantees a base set of instrument sounds,
- and a means for addressing them,
- but doesn’t guarantee any quality

*Better Yet, Downloadable Sounds*

- Download samples for instruments
  
  **Benefits:** Does more to guarantee quality  
  **Drawbacks:** Samples aren’t reality

---

Event Based Compression

*Downloadable Algorithms*

- Specify the algorithm,  
  the synthesis engine runs it,  
  and we just send parameter changes
  
  **Part of “Structured Audio” (MPEG4)**

  **Benefits:** Can upgrade algorithms later  
  Can implement scalable synthesis

  **Drawbacks:** Different algorithm for each class of sounds  
  (but can always fall back on samples)
Back to Waveforms

Time Domain Waveform Compression

- $\mu$ – Law: Non-linear amplitude quantization
- ADPCM: Adaptive quantization level of changes (deltas) in signal

Time Domain Log Amplitude

$\mu/a$-Law: More accuracy in low amplitudes, less in higher amplitudes. Decreases perceived quantization noise.

Actual 8 bit $\mu$-law uses 1 sign bit, 3 exponent bits, and 4 linear mantissa bits. The common claim is that this scheme yields 4 bits of compression, $12:8 = 1.5:1$
Adaptive Resolution: ADPCM

*Like Log-Compressor, but bit resolution changes as a result of recent signal history*

*Signal differences are compressed rather than signal values*

*Adapting the differences (deltas) yields Adaptive Delta PCM coding, claimed to do in 4 bits what µ-law does in 8.*

---

The Frequency Domain

*Exploit spectral properties to:*

1) Remove redundancy in signal
   - *slowly varying nature of real-world signals*
   - *periodic nature of many signals*

2) “Manage” error so it is less perceptible
Transform (Subband) Coders

Split signal into frequency subbands, then allocate bits to regions adaptively

Lossless (variable bit rate & comp. ratio):
- Subbands use lower sampling rate (no advantage)
- Bands with less information use less bits
- Adaptive prediction inter/intra bands

Lossy (fixed rate and ratio):
- Fix bit rate, then put bits where ear is most sensitive

Transform (Subband) Coders

Filter Bank Decomposition And Processing Can be Performed in the

Frequency Domain (FFT, etc.) and/or

Time Domain (FIR Filterbank, Wavelets, etc.)
Transform coders

Can reduce perceived quantization noise:

- frequency domain information, plus
- frequency masking knowledge

Production Models

Build a parametric model of the production system, then either

Fit the parameters to a given signal

Use signal processing techniques to extract parameters

Drive the parameters directly (no encoder?)

Examples: Rule system to drive speech synthesizer
MIDI file to drive music synthesizer
Speech Coders (production)

Assume speech is produced by a source-filter system (vocal folds/noise + vocal tract tube)

Identify filter, type of source, then code parameters

Takes advantage of slowly varying nature of vocal tract shape and other speech parameters

Future: Multi-Model Parametric Compressors?

Analysis front end identifies source(s)

Audio is (separated and) sent to optimal model(s)

Benefits:
- High compression
- Other knowledge

Drawbacks:
- We don’t know how to do all this yet

Audio In  →  Select Algorithm  →  Voice  Winds  Strings  Samples  →  Coded Audio Out
Two Contrasting Compressors

A simple speech coder

- Assume input is 8kHz, 16 bit
- 18.5 : 1 Ratio
- 7000 bps

A simple transform coder

- Assume input is 22kHz, 16 bit
- 2 (or 4) : 1 Ratio
- 176,400 (or 88200) bps

An LPC Speech Coder

Ten pole Linear Predictive speech Coder

- Frame rate is 30 frames / second (@ 8K sampling rate)
- Frame size is 30 ms.
- Source is encoded as pulse train or white noise
- LPC coefficients: quantized to 2 bytes each (20 total)
- Source type: coded in 1 bit (pitched/noise) per frame.
- Source amplitude: stored in one float per frame.
- Source pitch: stored in one float per frame.
- Total transmission rate: 7000 bps (18.5:1 ratio)
A Cheap Transform Coder

*FHT-based Delta Block Adaptive Log Amplitude Transform Coder*

- 64 point (32 subbands) FHT Frame (3 ms @ 22kHz)
- Frame rate is 344 frames/second
- Deltas of signal are used
- 4 (or 8) bit logarithmic compression of each band
- Each block peak is detected and stored as a short int
- Compression is 2 (or 4) : 1 (plus silence)

References and Resources

*General Psychoacoustics Books*


**Critical Bands and Masking**

**Old Views**


**Newer Views**


---

**Mu-Law, ADPCM Coding**


References and Resources

Speech Models and Compression


References and Resources

Subband Coding, Wavelets, AC-2


References and Resources

**MPEG**


---

**MIDI and Music Representation**


### Source Code

- **Quantization Program** *(N bit)*
- **MuLaw Coder/Decoder** *(8 Bit)*
- **SigLaw Coder/Decoder** *(4 bit)*
- **ADPCM Coder/Decoder** *(4 bit)*
- **Xform Coder/Decoder** *(4 and 8 bit)*
- **LPC Speech Coder/Decoder**
- **Utilities**