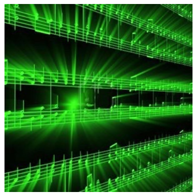
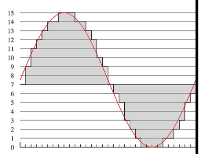
Penn Engineering

ESE150 Spring 2022

ESE





Lecture #12 – Psychoacoustic Compression Algorithm

ESE 150 – DIGITAL AUDIO BASICS

Based on slides © 2009—2022 DeHon, Koditschek
Additional Material © 2014 Farmer

1

1

ESE150 Spring 2022

LECTURE TOPICS

- **Review and preclass**
- **How do we take advantage of psychoacoustics in MP3 Achieve this 6—12x reduction from CD Audio**
  - Review Tricks
  - Formulate Optimization
  - Algorithm for Adaptation
- **Midterm**
- **References**

2

2

ESE150 Spring 2022

REVIEW AND PRECLASS SETUP/REMINDER

3

3

ESE150 Spring 2022

KNOBS WE CAN TURN

- **Amplitude quantization**
  - Per band
- **Frequency quantization**
  - Per band?
- **Frequencies kept (per critical band)**
  - Per band
- **…and can perform lossless compression**

4

4

ESE150 Spring 2022

PRECLASS 1

- **44,300 samples/s**
- **16b**
- **26ms window**
- **a) How many bits?**
- 128Kb/s stereo → 64Kb/s per audio channel
- **b) How many bits per 26ms window?**
- **c) ratio?**

5

5

ESE150 Spring 2022

MP3 ENCODING PROCESS

- **All MP3 files broken into "Frames"**
  - Each frame stores 1152 Audio Samples
  - Lasts for 26 ms
  - Frame also divided further into 2 "granuels"
    - Each granuel contains 576 samples

6

6

Optimization Problem

- **How fit in the resource constraints (128Kb/s) while maximizing goodness (sound quality)?**
- **Quantify bits used:** $\sum_{bands} \sum_{f \in freqs} Bits(f)$
  - Cannot exceed 128Kb/s
  - = 1,704 b / 26ms frame / channel
- **Quantify goodness: minimize** $\sum_{f \in freqs} Error(f) \times W(f)$

Preclass 2

- 576 frequencies
- 16b amplitude
- 1704b budget
- (frequency,amplitude) pairs to represent
- **How many frequencies can we keep?**
- **Conclude:** cannot keep all frequencies and hit budget

Preclass 3

- **Bits if only have 3 non-masked, non-zero frequencies?**
- **Conclude:** some frames won't use all their bits

Preclass 4

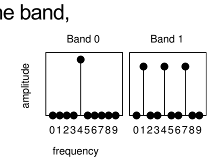
- If all frequencies equally likely, how many bits to represent 80 non-zero/non-masked frequencies?

Preclass 5

- 60dB, 42dB, 30dB each are 25% of amplitudes
- Other magnitudes remaining 25%
  - Equally likely
- How many bits to represent each of the 3 25% cases?
- How many bits to represent other cases?
- On average many bits to represent 80 non-zero/non-masked frequencies?
- **Conclude:** number of frequencies can keep depends on compressibility

Seen

- **May want to do something smarter than**
  - Allocating fixed number of frequencies per band
  - Allocating fixed quantization to a band
- **Like to adapt our encoding to the data**
  - If more Huffman compressible, we get more frequencies
  - If fewer frequencies suffice for one band,
    - Allow more frequencies for another
    - …or allocate less quantization



Optimization Problem

- How fit in the resource constraints (128Kb/s) while maximizing goodness (sound quality)?
- Optimization problems central to engineering

Optimization Problem

- How fit in the resource constraints (128Kb/s) while maximizing goodness (sound quality)?
- Quantify bits used: $\sum_{bands} \sum_{f \in freqs} Bits(f)$
  - Cannot exceed 128Kb/s
  - = 1,704 b / 26ms frame / channel
- Quantify goodness: minimize $\sum_{f \in freqs} Error(f) \times W(f)$

Goodness/Sound Quality

- Error(Amp) = |Orig Amplitude – Encoded|
  - Whole OrigAmplitude if dropped
  - |Orig Amplitude-Quantize(OrigAmplitude,bits)| if quantized
- W(freq)
  - 0 if below hearing threshold
  - 0 if masked
  - Value between 0 and 5 if partially masked in critical band
  - Really depend on what already encoded $\sum_{f \in freqs} Error(f) \times W(f)$

Example Weight Function W(f)

- W(f)=CBWeight*Mask
- Mask = 0 if MaxAmp-FreqAmp>3
  - 1 otherwise

Penn Engineering ESE

Part 2

Encodingn Algorithms

Greedy Incremental Frequency Selection

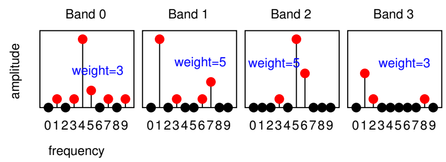
- EncodeFreq=0
- UnencodedFreqs=AllFreqs
- Error=ErrorCalc(AllFreqs)
- While((EncodeFreq<NumFreqs) && Error>0)
  - dErr,newFreq=MaxDeltaError(UnencodedFreqs)
  - FrameFreq.add(newFreq)
  - UnencodedFreqs.remove(newFreq)
  - Error-=dErr
  - numFreqs++

Approach

$$\sum_{bands}\sum_{f\in freqs} Bits(f)$$

- Start with nothing
- Start with bitbudget

$$\sum_{f\in freqs} Error(f)\times W(f)$$



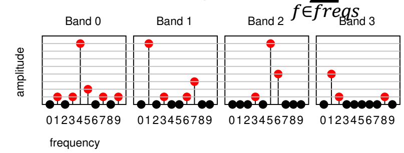
19

Approach

$$\sum_{bands}\sum_{f\in freqs} Bits(f)$$

- Start with nothing
- Start with bitbudget

$$\sum_{f\in freqs} Error(f)\times W(f)$$



Bits = 0
Error = 190

	Band 0	Band 1	Band 2	Band 3
	11 8 2 11	8 1 1 3	1 8 4	41 1
sum	14	13	13	6
weight	3	5	5	3
weighted	42	65	65	18

20

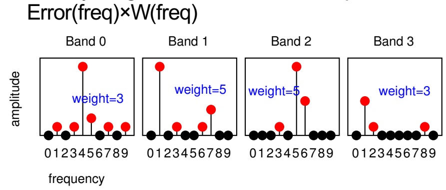
Approach

$$\sum_{bands}\sum_{f\in freqs} Bits(f)$$

- Start with nothing
- Start with bitbudget
- While(bitbudget>0)
  - Identify Largest Error reduction component: Error(freq)×W(freq)

$$\sum_{f\in freqs} Error(f)\times W(f)$$

Bits = 0
Error = 190



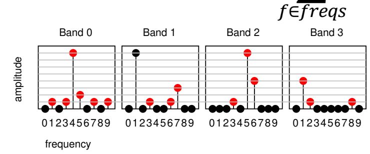
21

Approach

$$\sum_{bands}\sum_{f\in freqs} Bits(f)$$

- Start with nothing
- Start with bitbudget

$$\sum_{f\in freqs} Error(f)\times W(f)$$



Bits = 3+6=9
Error = 125

	Band 0	Band 1	Band 2	Band 3
	11 8 2 11	0 0 0 0	1 8 4	41 1
sum	14		13	6
weight	3	5	5	3
weighted	42	0	65	18

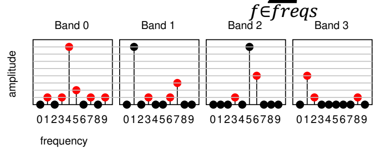
22

Approach

$$\sum_{bands}\sum_{f\in freqs} Bits(f)$$

- Start with nothing
- Start with bitbudget

$$\sum_{f\in freqs} Error(f)\times W(f)$$



Bits = 18
Error = 60

	Band 0	Band 1	Band 2	Band 3
	11 8 2 11	0 0 0 0	0 0 0	41 1
sum	14			6
weight	3	5	5	3
weighted	42	0	0	18

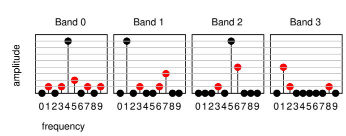
23

Approach

$$\sum_{bands}\sum_{f\in freqs} Bits(f)$$

- Start with nothing
- Start with bitbudget

$$\sum_{f\in freqs} Error(f)\times W(f)$$



Bits = 27
Error = 18

	Band 0	Band 1	Band 2	Band 3
		0 0 0 0	0 0 0	41 1
sum	0			6
weight	3	5	5	3
weighted	0	0	0	18

24

Also Quantization

$$\sum_{bands}\sum_{f \in freqs} Bits(f)$$

$$\sum_{f \in freqs} Error(f) \times W(f)$$

- **To keep simple, assumed fixed quant.**
- **Incrementally assign bits**
- **While(bitbudget>0)**
  - Identify Largest Error component: Error(freq)×W(freq)
  - Assign more bits to that frequency
    - Go from 0 bits to 1 bit
      - 1 bit to 2 bits

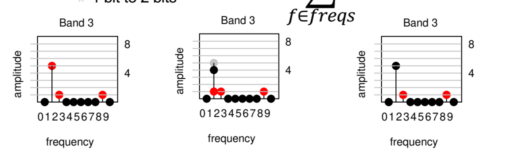
25

Also Quantization

$$\sum_{bands}\sum_{f \in freqs} Bits(f)$$

- **While(bitbudget>0)**
  - Identify Largest Error component: Error(freq)×W(freq)
  - Assign more bits to that frequency
    - Go from 0 bits to 1 bit
      - 1 bit to 2 bits

$$\sum_{f \in freqs} Error(f) \times W(f)$$



0b -- error=7

1b quant -- error=1 (1+6) encode

3b quant -- error=0 (3+9) encode

26

Approach (Greedy)

- **Start with nothing**
- **Start with bitbudget**
- **While(bitbudget>0)**
  - Identify Largest Error component
  - Allocate some bits to reduce error
    - Add frequency or Add quantization bits to band
    - Pick one to most reduce the error

27

Approach (Greedy)

- **Start with nothing**
- **Start with bitbudget**
- **While(bitbudget>0)**
  - Identify Largest Error component
  - Allocate some bits to reduce error
    - Add frequency or Add quantization bits to band
    - Pick one to most reduce the error
  - Re-Huffman encode and update bitbudget

28

Adaptive Refinement

- **Rediscovering where to allocate everything every time may be laborious**
  - Often same frequencies persist for more than 26ms
- **Maybe we can get close and adjust?**
  - Use critical band allocation from previous frames as a starting point guess
    - bits, frequencies, quantization
  - Try initial encoding with that

29

Approach (Adaptive)

- **Start with budget guess**
  - Quantization in bands
  - Frequencies to keep in each band
- **Encode, compress**

- **What can we do if takes up > 1704 bits?**
- **What can we do if takes up < 1704 bits?**

30

Finish with Adaptive Fine Tuning

- **<previous slide>**
- **FrameBits=Huffman(FrameFreq)**
- **While ((FrameBits>1704) | (FrameBits<(1704-26))**
  - If (FrameBits>1704)
    - NumFreqs—
    - FrameFreq←RemoveLeastImportant(FrameFreq)
  - Else if (FrameBits<(1704-26))
    - NumFreqs++
    - FrameFreq←AddNextImportant(FrameFreq)
  - FrameBits=Huffman(FrameFreq)

31

Finish with Adaptive Fine Tuning

Could add/subtract more than one freq at a time.
Could keep track of high and low encodings.

- **<previous slide>**
- **FrameBits=Huffman(FrameFreq)**
- **While ((FrameBits>1704) | (FrameBits<(1704-26))**
  - If (FrameBits>1704)
    - NumFreqs—
    - FrameFreq←RemoveLeastImportant(FrameFreq)
  - Else if (FrameBits<(1704-26))
    - NumFreqs++
    - FrameFreq←AddNextImportant(FrameFreq)
  - FrameBits=Huffman(FrameFreq)

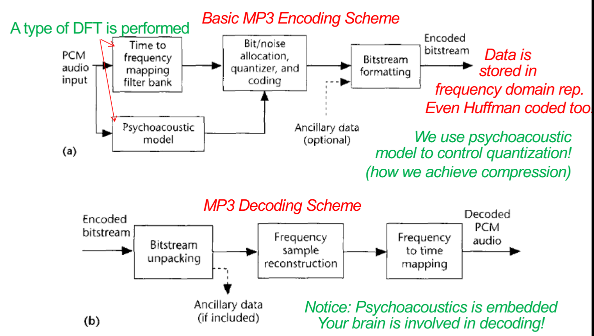
32

Roundup

PERCEPTUAL CODING & MP3

33

MP3 Flow Chart (Encoding/Decoding)



34

Midterm – Monday in Lecture

- **Closed book, notes**
- **Calculators allowed**
- **50 minutes**
  - Shorter than previous years
- **5% of grade**
  - prepare for final
- **Last 4 year's midterm and answers**
  - on 2018, 2019, 2020 syllabus
    - Were all in-person, closed book (75 minutes)
  - 2021 was online, open book, also on syllabus
    - (120 minutes)

Topics

- Data representation in bits
- Sounds waves
- Sampling
- Quantization
- Nyquist
- Lossy/lossless compression
- Common case
- Frequency domain
- Psychoacoustics
- Perceptual coding

35

Compare to Lab

- **Lecture 11 & 12 (this week)**
  - Help understand more what real MP3 encoding looks like
  - Thinking about fixed rate
    - And adaptation for variable rate encoding from Huffman
  - Formulating masking explicitly
    - But simplistic
  - Illustrating Optimization Approaches
- **Lab 6**
  - Capture spirit of reducing frequencies
  - Simplified – only taking loudest fraction in each band
    - Rather than being rigorous about masking
    - Or trying to hit some fixed rate

36

Big Ideas

- **Can use pyschoacoustics to compress audio**
  - Eliminate portions of signal that human's don't notice
- **Optimization**
  - Identify Design Space (knobs)
  - Identify Costs and Constraints
  - Formulate quantitatively
  - Algorithms to approach
  - Iterative/adaptive approach
    - Deal with effects that aren't completely predictable

37

Learn More

- **Optimization –**
  - continuous mathematical optimization ESE204, ESE504, ESE605
  - discrete optimization CIS121, CIS320
- **Signal processing – ESE224**

38

Coming Up

- **Feedback**
- **Lab 6 start today**
  - 2 week lab
  - Brings together first half of course
  - Formal report
- **Midterm on Monday**
- **Lab 6 continue next Wednesday**

39

References

- Tutorials on Psychoacoustic Coding (in increasing order of abstraction and generality)
  - D. Pan, M. Inc, and I. L. Schaumburg. A tutorial on MPEG/audio compression. IEEE multimedia, 2(2):60–74, 1995.
  - Nikil Jayant, James Johnston, and Robert Safranek. Signal compression based on models of human perception. Proceedings of the IEEE, 81(10):1385–1422, 1993.
  - V. K. Goyal. Theoretical foundations of transform coding. IEEE Signal Processing Magazine, 18(5):9–21, 2001.
- Lightweight Overview of MP3
  - Rassol Raissi. The theory behind mp3. Technical report, MP3' Tech, December 2002.
- Scientific Basis of MP3 Coding Standard
  - J. D. Johnston. Transform coding of audio signals using perceptual noise criteria. IEEE Journal on selected areas in communications, 6(2):314–323, 1988.

40