

# Week 14 – Music Understanding and Classification

Roger B. Dannenberg

Professor of Computer Science, Music & Art  
Carnegie Mellon University

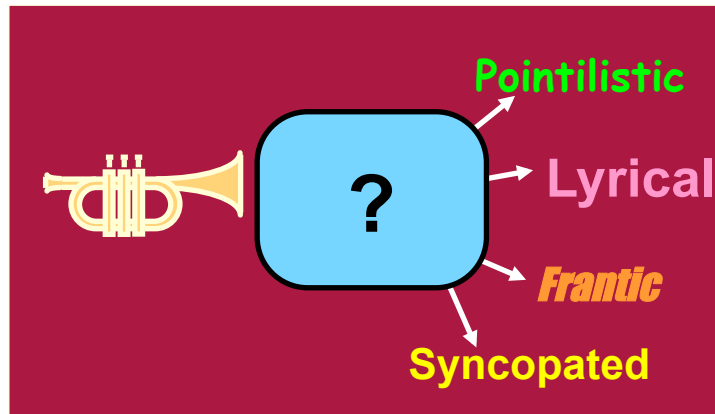


## Overview

---

- Music Style Classification
  - What's a classifier?
  - Naïve Bayesian Classifiers
  - Style Recognition for Improvisation
  - Genre Classification
  - Emotion Classification
- Beat Tracking
- Key Finding
- Harmonic Analysis (Chord Labeling)

## Music Style Classification

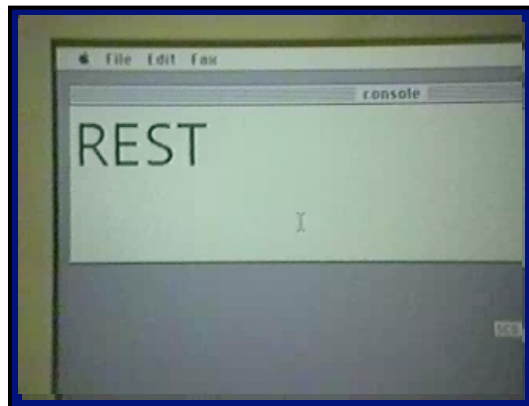


3

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## Video



4

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## What Is a Classifier?

- What is the class of a given object?
  - Image: water, land, sky
  - Printer: people, nature, text, graphics
  - Tones: A, A#, B, C, C#, ...
  - Broadcast: speech or music, program or ad
- In every case, objects have *features*:
  - RGB color
  - RGB Histogram
  - Spectrum
  - **Autocorrelation**
  - **Zero crossings/second**
  - **Width of spectral peaks**

5

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## What Is a Classifier? (2)

- Training data
  - Objects with (manually) assigned classes
  - Assume to be representative sample
- Test data
  - Separate from training data
  - Also labeled with classes
  - But labels are not known to the classifier
- Evaluation:
  - Percentage of correctly labeled test data

6

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## Game Plan

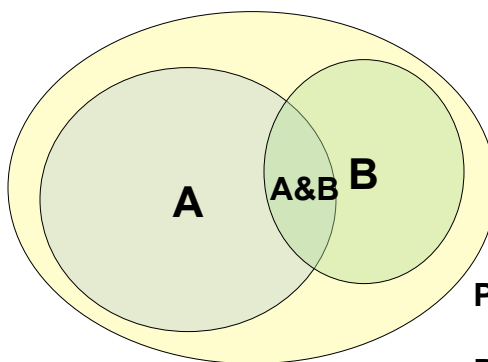
- We can look at training data to figure out typical features from classes
- How do we get classes from features?
  - → Bayes' Theorem
- We'll need to estimate  $P(\text{features}|\text{class})$
- Put it all together

7

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## Bayes' Theorem



$$P(A|B) = P(A\&B)/P(B)$$
$$P(B|A) = P(A\&B)/P(A)$$

$$P(A|B)P(B) = P(A\&B)$$
$$P(B|A)P(A) = P(A\&B)$$

$$P(A|B)P(B) = P(B|A)P(A)$$

$$P(A|B) = P(B|A)P(A)/P(B)$$

8

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

$$P(A|B) = P(B|A)P(A)/P(B)$$

- $P(\text{class} | \text{features}) = \frac{P(\text{features} | \text{class})P(\text{class})}{P(\text{features})}$
- Let's guess the most likely class
  - (maximum likelihood estimation, MLE)
- Find class that maximizes:  
 $P(\text{features} | \text{class})P(\text{class})/P(\text{features})$
- And since  $P(\text{features})$  independent of class, maximize  
 $P(\text{features} | \text{class})P(\text{class})$
- Or if classes are equally likely, maximize:  
 $P(\text{features} | \text{class})$

9

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## Bayesian Classifier

- The most likely class is the one for which the observed features are most likely.
- The most likely class:  
$$\underset{\text{class}}{\text{argmax}} P(\text{class} | \text{features})$$
- The class for which features are most likely:  
$$\underset{\text{class}}{\text{argmax}} P(\text{features} | \text{class})$$

10

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## Game Plan

- We can look at training data to figure out typical features from classes
- How do we get classes from features?
  - → Bayes' Theorem
- **We' ll need to estimate  $P(\text{features}|\text{class})$**
- Put it all together

11

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## Estimating $P(\text{features}|\text{class})$

- A word of caution: Machine learning involves the estimation of parameters. The size of training data should be much larger than the number of parameters to be learned. (But recent research suggests *many more* parameters than data can *also* learn and generalize well in certain cases.)
- Naïve Bayesian classifiers have relatively few parameters, so they tend to be estimated more reliably than parameters of more sophisticated classifiers, hence a good place to start.

12

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## What's P(features|class)?

- Let's make a *big* (and *wrong*) assumption:
  - $P(f_1, f_2, f_3, \dots, f_n | \text{class}) = P(f_1|\text{class})P(f_2|\text{class})P(f_3|\text{class})\dots P(f_n|\text{class})$
  - This is the *independence* assumption
- Let's also assume (also *wrong*)  $P(f_i | \text{class})$  is normally distributed
  - So it's characterized completely by:
    - mean
    - standard deviation
- Naive Bayesian Classifier: assumes features are independent and Gaussian

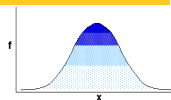
13

Carnegie Mellon University

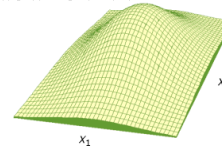
© 2019 by Roger B. Dannenberg

## Estimating P(features|class) (2)

- Assume the distribution is Normal (same as Gaussian, Bell Curve)
- Mean and variance are estimated by simple statistics on test set:
  - Classes partition test set into distinct sets
  - Collect mean and variance for each class
- Multiple features have a multivariate normal distribution:
- Intuition: Assuming independence,  $P(\text{features}|\text{class})$  is related to the distance from the peak (mean) to the feature



Copyright (c) Contingency Analysis, 2002



14

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

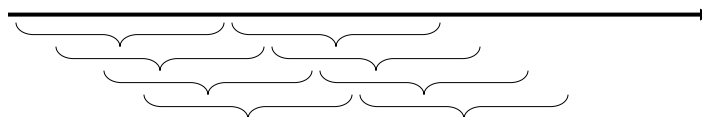
## Putting It All Together

- $F_i$  =  $i^{\text{th}}$  feature
- $C$  = class
- $\mu$  = mean
- $\sigma$  = standard deviation
- $\Delta_C$  = normalized distance from class
- Estimate mean and standard deviation just by computing statistics on training data
- Classifier computes  $\Delta_C$  for every class and picks the class ( $C$ ) with the smallest value.

$$\Delta_C = \sqrt{\sum_{i=1}^n \left( \frac{F_i - \mu_{C,i}}{\sigma_{C,i}} \right)^2},$$

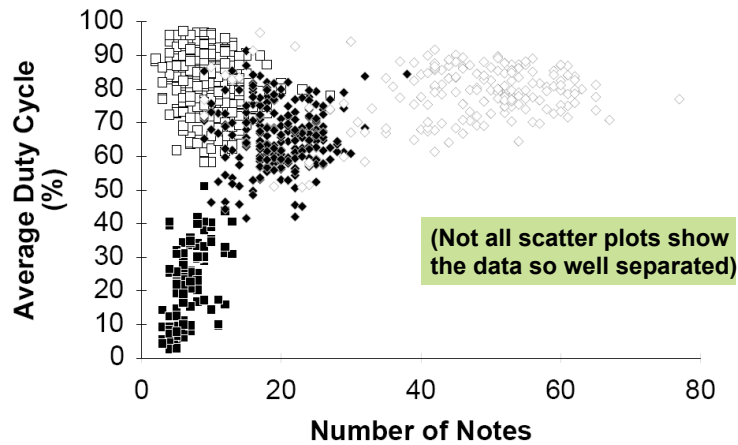
## Style Recognition for Improvisation

- Features are:
  - # of notes
  - Avg. midi key no
  - Std.Dev. of midi key no
  - Avg. duration
  - Std.Dev. of duration
  - Avg. duty factor
  - Std.Dev. of duty factor
  - No. of pitch bends
  - Avg. pitch
  - Std.Dev. of pitch
  - No. of volume controls
  - Avg. volume
  - Std.Dev. of volume
- Windowed MIDI Data:





## A Look At Some Data



17

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## Training

- Computer says what style to play
- Musician plays in that style until computer says stop
- Rest
- Play another style
- Note that collected data is “labeled” data

18

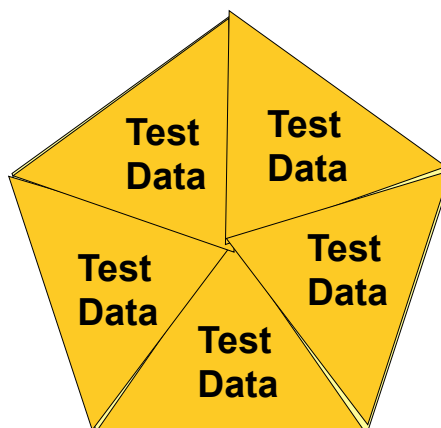
Carnegie Mellon University

© 2019 by Roger B. Dannenberg

# Results

- With 4 classes, 98.1% accuracy
  - Lyrical
  - Syncopated
  - Frantic
  - Pointillistic
- With 8 classes, 90.0% accuracy
  - Additional classes: blues, quote, high, low
- Results did *not* apply to real performance situation,
- but retraining in context helped

# Cross-Validation



## Other Types of Classifiers

- Linear Classifier
  - assumes normal distributions
  - but not independence
  - closed-form, very fast training (unless many features)
- Neural Networks – capable of learning when features are not normally distributed, e.g. bimodal distributions.
- kNN – k-Nearest Neighbors
  - Find k closest exemplars in training data
- SVM – support vector machines

21

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## In Practice: Classifier Software

- MATLAB – Neural Networks, others
- Weka – <http://www.cs.waikato.ac.nz/~ml/weka/>
  - Widely used
  - General data-mining toolset
- ACE – <http://coltrane.music.mcgill.ca/ACE/>
  - Especially made for music research
  - Handles classes organized as a hierarchical taxonomy
  - Includes sophisticated feature selection (note that sometimes classifiers get better with fewer features!)
- Machine learning packages in Matlab, PyTorch, TensorFlow

22

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## Genre Classification

- Popular task in Music Information Retrieval
- Usually applied to audio
- Features:
  - Spectrum (energy at different frequencies)
  - Spectral Centroid
  - Cepstrum coefficients (from speech recog.)
  - Noise vs. narrow spectral lines
  - Zero crossings
  - Estimates of “beat strength” and tempo
  - Statistics on these including variance or histograms

23

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## Typical Results

- Artist ID: 148 artists, 1800 files
- → 60-70% correct
- Genre: 10 classes:  
ambient, blues, classical, electronic, ethnic,  
folk, jazz, new\_age, punk, rock
- → ~80% correct
- Example: [http://www.youtube.com/watch?v=NDLhrc\\_WR5Q](http://www.youtube.com/watch?v=NDLhrc_WR5Q)

24

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## Summary

- Machine Classifiers are an effective and not-so-difficult way to process music data
- Convert low-level feature to high-level abstract concepts such as “style”
- Can be applied to many problems:
  - Genre
  - Emotion
  - Timbre
  - Speech/music discrimination
  - Snare/hi-hat/bass drum/cowbell/etc.

25

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## Summary (2)

- General Problem: map *feature vector* to *class*
- Bayes' Theorem tells us probability of class given feature vector is related to probability of feature vector given class
- We can estimate the latter from training data

26

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

# Beat Tracking

## The Problem

---

- The “foot tapping” problem
- Find the positions of beats in a song
- Related problem: estimate the tempo (without resolving beat locations)
- Two big assumptions:
  - Beats correspond to some acoustic feature(s)
  - Successive beats are spaced about equally (i.e. tempo varies slowly)

## Acoustic Features

- Can be local energy peaks
- Spectral flux: the change from one short-term spectrum to the next
- High Frequency Content: spectrum weighted toward high frequencies
- With MIDI data, you can use note onsets

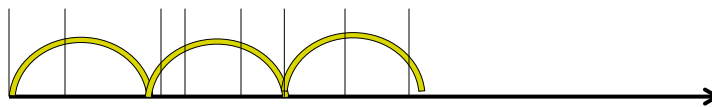
29

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## A Basic Beat Tracker

- Start with initial tempo and first beat (maybe the onset of the first note)
- Predict expected location of next beat
- If actual beat is in neighborhood, speed up or slow down according to error

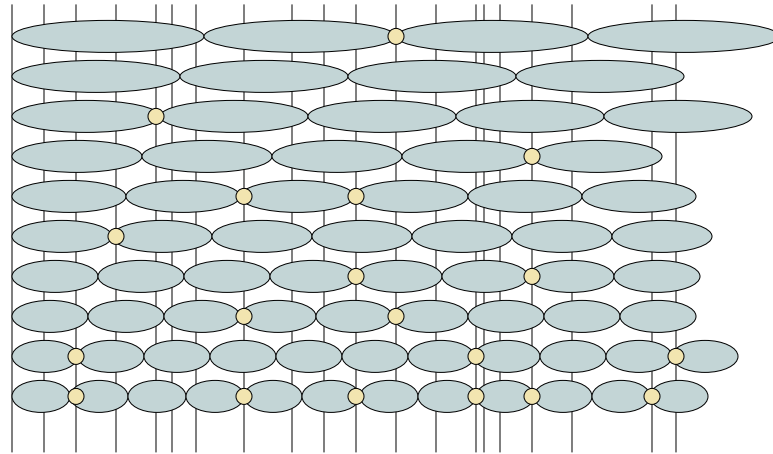


30

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## “Society of Agents” Model



31

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## Society of Agents (2)

- Each agent tries to find periodic beats much like the basic beat tracker, but with a limited range of tempi
- Agents report how well they are doing
- A “supervisor” picks the best agent and may arrange for “handoff” from one agent to another
- “Agent” is a bit overblown and anthropomorphic – it’s just a simple software object

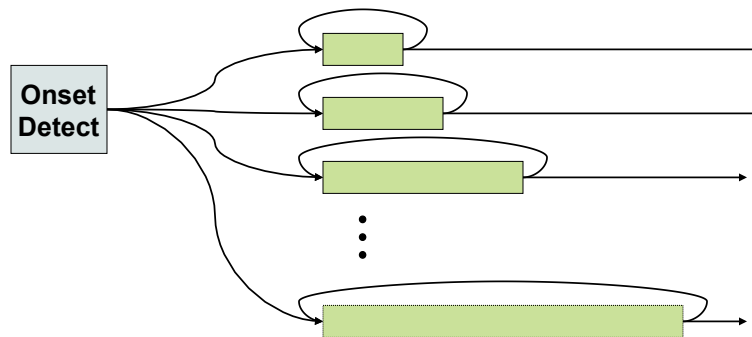
32

Carnegie Mellon University

© 2019 by Roger B. Dannenberg



## Filter Bank and Oscillator Models



33

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## Oscillators

- Some oscillator models (particularly in work by Ed Large) are inspired by actual neurons
- Oscillators maintain approximate frequency but phase can be adjusted

34

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## Agents and Oscillators

- Note that “Agents” act like oscillators
  - Detect periodicity
  - “Tuned” to small range of tempi
- My opinion:
  - Music data is so noisy, you need to search within a narrow range of tempi
  - A wide-tempo-range tracker is likely to get lost
  - That’s why multiple agents/oscillators work
- State-of-the art uses machine learning to learn to find beats and downbeats, post-processing to look for periodicity

35

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## Key Finding

- Standard (or at least common) approach is based on *Krumhansl-Schmuckler Key-Finding Algorithm*
- In turn based on *key profile*: essentially a histogram of pitches observed in a given key.
- Key is estimated by:
  - Create a profile for a given work
  - Find the closest match among the Krumhansl-Schmuckler profiles

36

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## Variations on Key Finding

- Weighting profile by note duration
- Using exponential decay to give a more local estimate of key center
- Using spectrum rather than pitches when the data is audio
- Probably better results can be obtained with machine learning approaches and more features related to tonal harmony

37

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## Harmonic Analysis/Chord Labeling

- An under-constrained problem
- Goal is to give chord labels to music

The image shows a musical staff with a treble clef and a common time signature. The notes are C4, F4, and C5. A yellow oval labeled "Labeling #1" contains the text "C F C", indicating a C major chord. A purple oval labeled "Labeling #2" contains the text "C" and "F is a passing tone", indicating a C major chord with a passing tone. An arrow points from the text "F is a passing tone" to the F4 note on the staff.

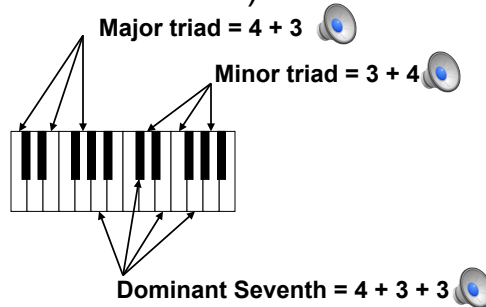
38

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

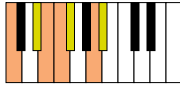
# Chords

- Conventionally, chords have 3 or 4 notes separated by major and minor thirds (intervals of 4 or 3 semitones)



39

# Chords Can Be Complex

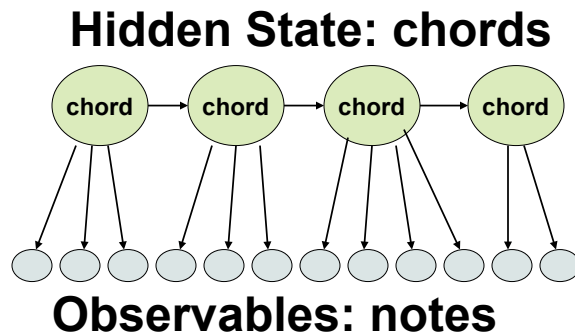
- Any configuration of notes has an associated chord type (which may be highly improbable):
- E.g.  = C dominant seventh with a flat-5, added sharp 9<sup>th</sup>, 11<sup>th</sup>, and 13<sup>th</sup>
- Chords can change at any time:



- Chords do not necessarily match all the notes (extra notes are called non-chord tones)

40

## Chords as “Hidden” Variables



41

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## How Can We Approach This Problem?

- Find a balance between
  - use relatively few chords
  - get good match between observed notes and chords (minimize non-chord tones)
- Create a scoring function to rate a chord labeling
  - Penalty for each new chord
  - Penalty for each non-chord tone
- Search for optimal labeling

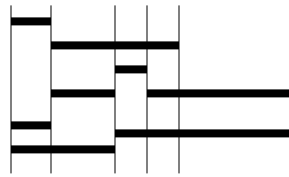
42

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## What Do We Label?

- Every place a note begins or ends, start a new segment (Pardo and Birmingham call this a concurrency)

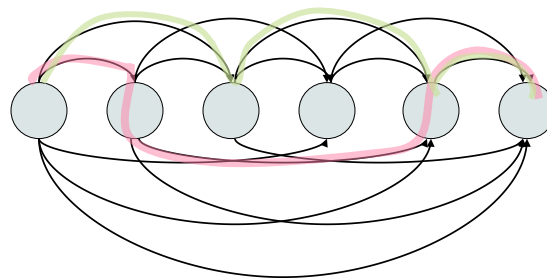


43

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## Chord Labeling as Graph Algorithm



**Nodes are concurrencies, arcs are the cost of consolidating concurrencies and labeling them as one chord.**

- Cost depends on some assumptions, but can be  $N^2$  using shortest path algorithm

44

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## Chord Recognition from Audio

- For the latest, most advanced techniques, see the literature (esp. ISMIR Proceedings)
- Another classification problem?
  - Given audio, classify into a chord type
  - Need to think about:
    - Labeled training data
    - Features
    - Training procedure

45

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## Chord Recognition: Training Data

- (1) Use hand-labeled audio
- (2) Create labels automatically from MIDI data; create audio by synthesizing MIDI
- (3) Create labels automatically from MIDI; align MIDI to "real" audio (we will talk about alignment later)
- Note: theoretically  $2^{12}$  chords, but typically stick to some subset of major, minor, dominant 7th, diminished, and augmented (each in all 12 transpositions)

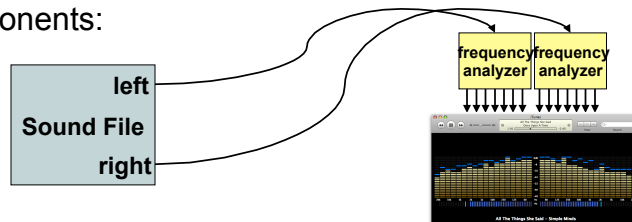
46

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## Features: A Diversion on FFT

- Audio analysis often begins with frequency content analysis.
  - Our ear is in some sense a frequency analyzer
  - Shape of the audio waveform is not really significant -- shifting the phase of one note can change wave shape completely, even if it "sounds the same"
- Every sound can be broken down into frequency components:



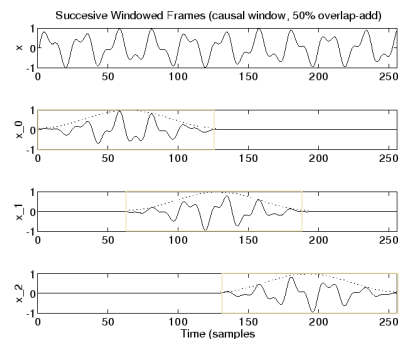
47

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## FFT

- Typically *many* more frequency "bins"
- Not continuous
- Divide signal into regions called frames (not to be confused with sample periods)
- Typical frame is 10 to 100ms
- Each frame analyzed separately
- 256 to 2048 frequency bins per frame



<http://www.dsprelated.com/josimages/sasp/img1411.png>

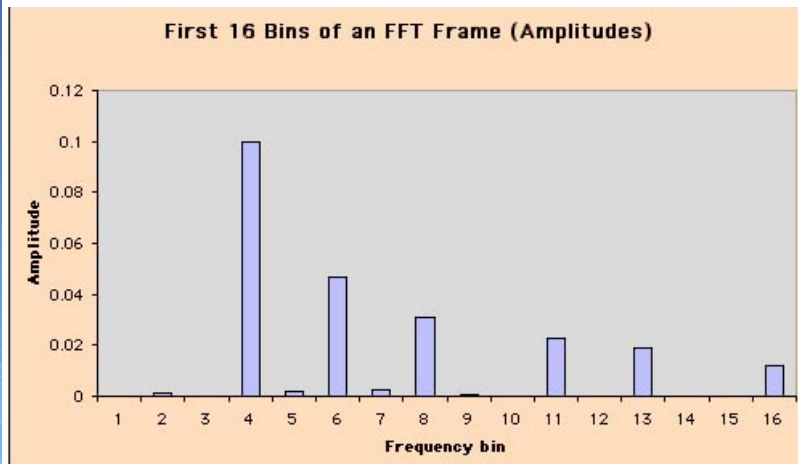
48

Carnegie Mellon University

© 2019 by Roger B. Dannenberg



## FFT Frames



49

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## FFT Parameters

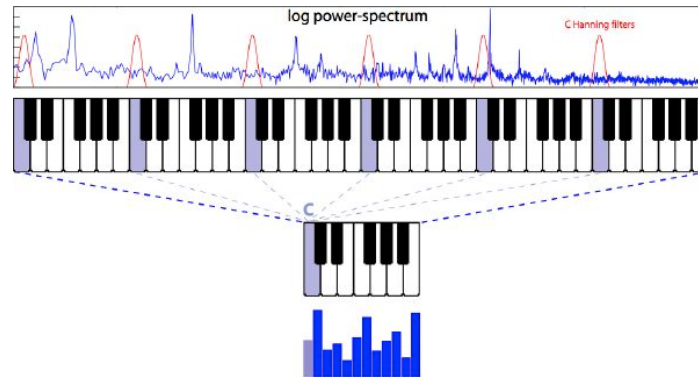
- Frequencies in audio range from 0 to half the sample rate
- An  $n$ -point FFT uses  $n$  samples, so it spans  $n/SR$  seconds
- There are  $n/2$  frequency bins, all same width over range from 0 to  $SR/2$ , so each bin is  $SR/n$  Hz wide.
- Example: 4096-point FFT and 44.1kHz sample rate
  - Bins are  $44.1k/4096 = 10.7\text{Hz}$  wide
  - Semitones (ratio of 1.059) are 10.7Hz wide at 181Hz
  - F3 in Hz is 175, F#3 in Hz is 185
- Larger FFT -> better frequency resolution
- Smaller FFT -> better time resolution

50

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

# Chroma Vector



Source: Tristan Jehan, PhD Thesis

51

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

# Chroma Vectors

- Note that any given tone will have overtones that contribute to many chroma bins:
  - 3rd harmonic is roughly 19 semitones
  - 5th harmonic is roughly 28 semitones
  - 6th harmonic is roughly 31 semitones
  - 7th harmonic is roughly 34 semitones
  - (none of these is a factor of 12)

52

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## Why Chroma Vector?

- Experience shows that chroma vectors capture harmonic and melodic information
- Chroma vectors do *not* capture timbral information (well)
  - C major on a piano looks like C major from string orchestra -- this is a good thing!
- Chroma vectors are typically normalized to eliminate any loudness information

53

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## Building a Simple Classifier

- Classes are chords
  - E.g. major/minor \* 12 gives 24 classes
  - Train classifier on labeled data
- Computation
  - For each FFT frame:
  - Compute chroma vector (12 features)
  - Run classifier
  - Output most likely chord label
- Example: <https://www.youtube.com/watch?v=kH8MgjKEFOU>

54

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## Using Context

- "Absolute" (*a priori*) information:
  - Chord probabilities: e.g.  $P(\text{major}) > P(\text{augmented})$
- Smoothing:
  - The sequence CCCC $\ominus$ CCCCC is likely all C's
  - Dynamic programming is a good way to optimize tradeoff between "cost" of transitions to new chords and likelihoods of chord choices
- Context
  - Chord sequences are not random
  - Hidden Markov Models often used to model chord sequences and prefer chords that are more likely due to context.

## Some References

- Robert Rowe: Machine Musicianship
- David Temperley: *The Cognition of Basic Musical Structures*
- Danny Sleator:  
<http://www.link.cs.cmu.edu/music-analysis/>  
(algorithms online)
- ISMIR Proceedings (all online)

## Summary and Conclusions

- Music involves communication
- Communication usually involves some conventions: syntax, phonemes, frequencies, selected/modulated to convey meaning
- In music, notes are the syntax; meaning is somewhere else
- Music Understanding attempts to get at these more abstract levels of meaning

57

Carnegie Mellon University

© 2019 by Roger B. Dannenberg

## Summary and Conclusions (2)

- Many of these techniques are for tonal music
  - It's rich with structure and convention
  - We understand it well enough to decide what's right and what's wrong (to some extent)
  - But it's not "what's happening" now in music
  - Or at least it's restricted to popular music
- Future work needs music theory, representations for time-based data, and sophisticated pattern recognition

58

Carnegie Mellon University

© 2019 by Roger B. Dannenberg