Digital Musicology via jSymbolic and Machine Learning

Cory McKay

Marianopolis College and CIRMMT

March 3, 2020

Brandeis University, Waltham, USA

Topics

- Context: Computation and musicology
- Intro to features and machine learning
- jSymbolic
- Sample research with jSymbolic
  - Sidebar: Avoiding encoding bias
- jMIR, SIMSSA and MIRAI

- Demo of jSymbolic and Weka

Software and statistics

- Automated software tools and statistical analysis techniques allow us to:
  - Study huge quantities of music very quickly
    - More than any human could reasonably look at
  - Empirically validate (or repudiate) our theoretical predictions
  - Do purely exploratory studies of music
  - See music from fresh perspectives

Human involvement is crucial

- Of course, computers certainly cannot replace the expertise and insight of musicologists and theorists
  - Computers instead serve as powerful tools and assistants that allow us to greatly expand the scope and empirical supportability of our work
- Computers do not understand or experience music in ways at all similar to humans
  - We must pose the research questions for them to investigate
  - We must interpret the results they present us with
- Music is, after all, defined by human experience, not some "objective" external truth

Big questions to think about

- What existing needs of music scholars can be addressed by computational approaches?
- What new, different opportunities for scholarship do computational approaches present?
- What challenges and pitfalls do computational approaches pose?
- How can we stimulate discussions and collaborations between domain experts (e.g. musicologists and data scientists)?

What is a “feature”?

- A piece of information that measures a characteristic of something (e.g. a piece of music) in a simple and consistent way
- Represented as simple number(s)
  - Can be a single value, or can be a set of related values (e.g. a histogram)
- Provides a summary description of the characteristic being measured
  - Usually macro, rather than local
- Can be extracted from pieces in their entirety, or from segments of pieces

Example: A basic feature

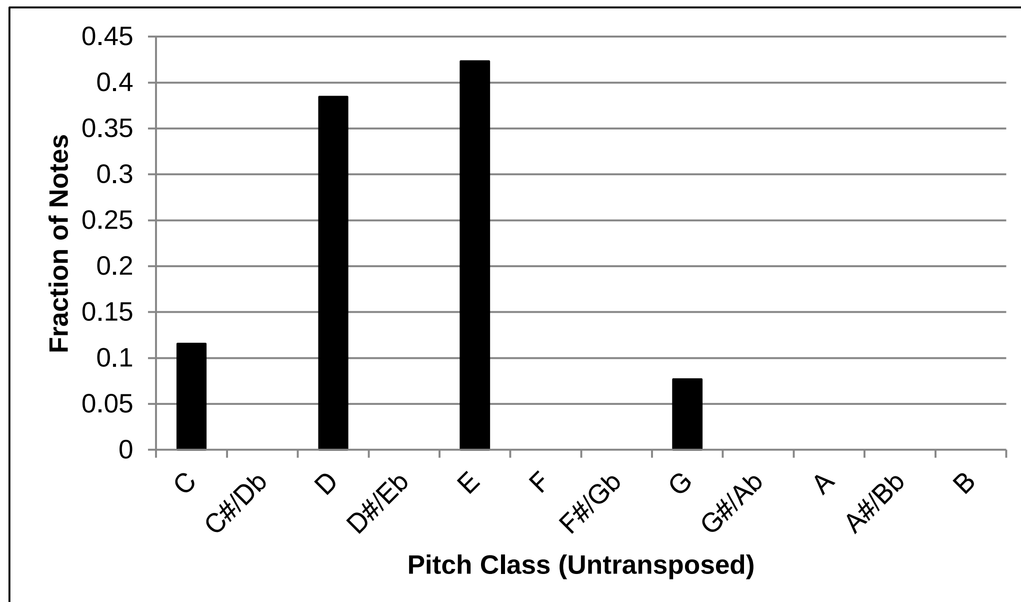
- Range (1-D): Difference in semitones between the highest and lowest pitches

- Value of this feature: 7
  - G - C = 7 semitones

Example: A histogram feature

- Pitch Class Histogram: Consists of 12 values, each representing the fraction of all notes belonging to an enharmonic pitch class

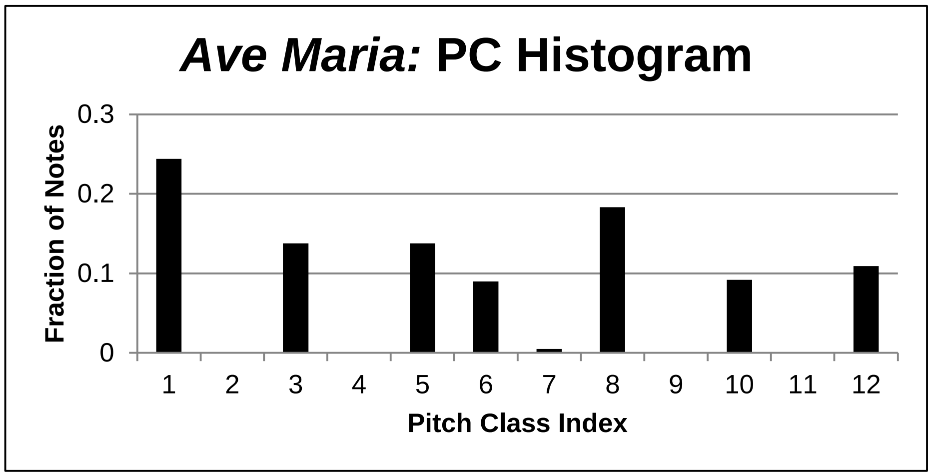
- Graph on right shows feature values
- Pitch class counts:
  - C: 3, D: 10, E: 11, G: 2
- Most common note is E:
  - 11/26 notes
  - Corresponds to a feature value of 0.423 for E



jMIR

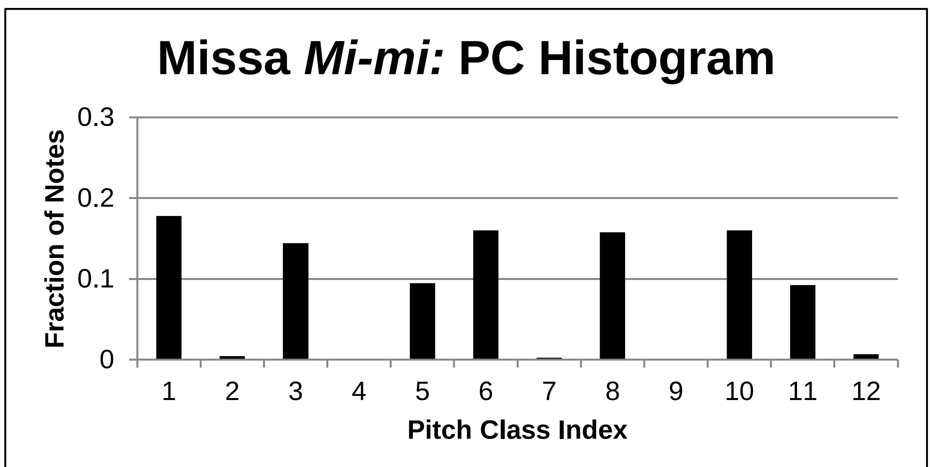
Josquin's *Ave Maria . . . virgo serena*

- Range: 34 (semitones)
- Repeated notes: 0.181 (18.1%)
- Vertical perfect 4[ths]: 0.070 (7.0%)
- Rhythmic variability: 0.032
- Parallel motion: 0.039 (3.9%)



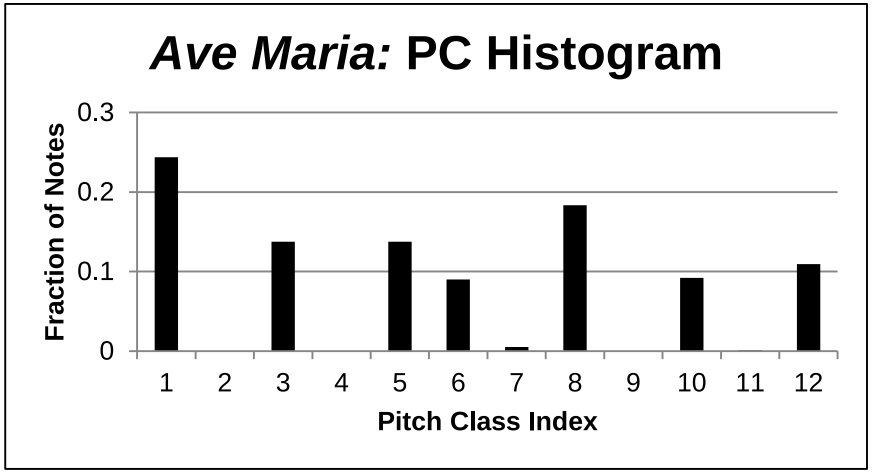
Ockeghem’s Missa *Mi-mi* (Kyrie)

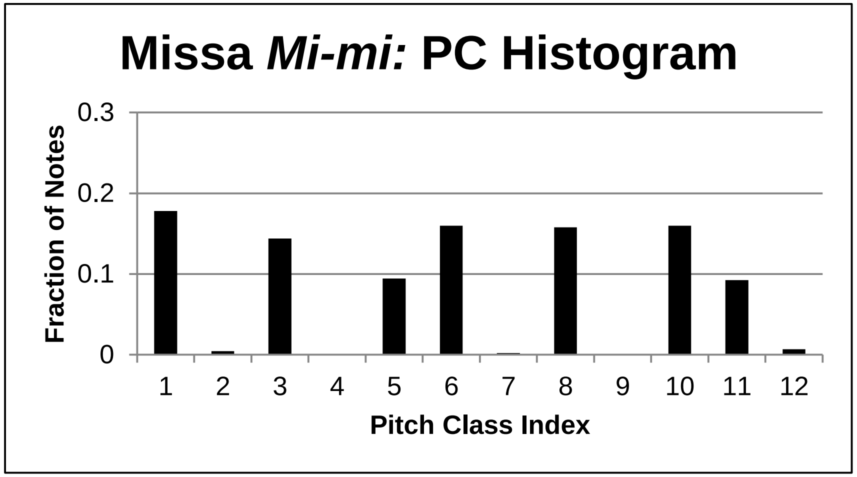
- Range: 26 (semitones)
- Repeated notes: 0.084 (8.4%)
- Vertical perfect 4$^{\text{ths}}$: 0.109 (10.9%)
- Rhythmic variability: 0.042
- Parallel motion: 0.076 (7.6%)



Feature value comparison

Feature	Ave Maria	Missa *Mi-mi*
Range	34	26
Repeated notes	0.181	0.084
Vertical perfect 4ths	0.070	0.109
Rhythmic variability	0.032	0.042
Parallel motion	0.039	0.076





Comparing features

- Comparing pairs of pieces like this in terms of features can be very revealing
  - Especially when that comparison involves hundreds or thousands of features, not just six
- Things get even more interesting, however, when comparisons are made between hundreds or thousands of pieces, not just two
  - Especially when the music is aggregated into groups, which can then be contrasted collectively
  - e.g. comparing composers, genres, regions, time periods, etc.

How can we use features? (1/3)

- Manual analysis to look for patterns
- Applying statistical analysis and visualization tools to study features extracted from large collections of music
  - Highlight patterns
  - Measure how similar various types of music are
  - Study the relative musical importance of various features
  - Observe unexpected new things in the music
- Perform sophisticated content-based searches of large musical databases
  - e.g. find all pieces with less than X amount of chromaticism and more than Y amount of contrary motion
  - e.g. the SIMSSA DB

How can we use features? (2/3)

- Use supervised machine learning to classify music
  - Done by training models on pre-labelled data
  - Can study music using whatever categories ("classes") one is interested in
    - e.g. composer, genre, style, time period, culture, region, etc.
  - Sample applications we have already explored:
    - Identify the composers of unattributed musical pieces
    - Explore the stylistic origins of genres (e.g. madrigals)
    - Delineate regional styles (e.g. Iberian vs. Franco-Flemish)

How can we use features? (3/3)

- Use unsupervised machine learning to cluster music
  - Done by training on unlabelled data
  - Can study how the model groups pieces based on statistical similarity
    - And then see if we can find meaning in these groups

Tools for examining features

- Manually:
  - Text editors
  - Spreadsheets
- With automatic assistance:
  - Statistical analysis software
    - e.g. SPSS, SAS, etc.
  - Machine learning and data mining software
    - e.g. Weka, Orange, etc.
- Many of these tools can produce helpful visualizations

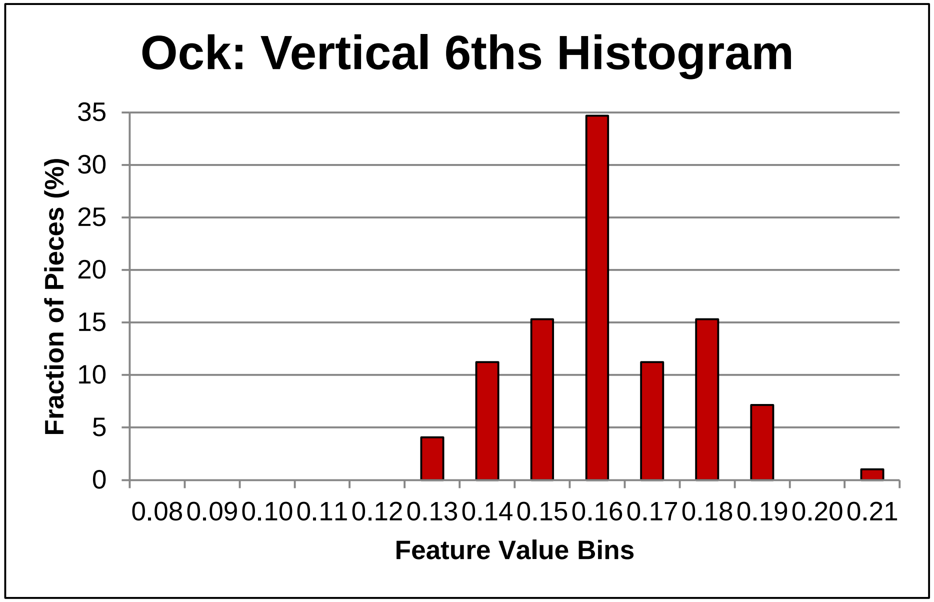
Feature visualization: Histograms (1/6)

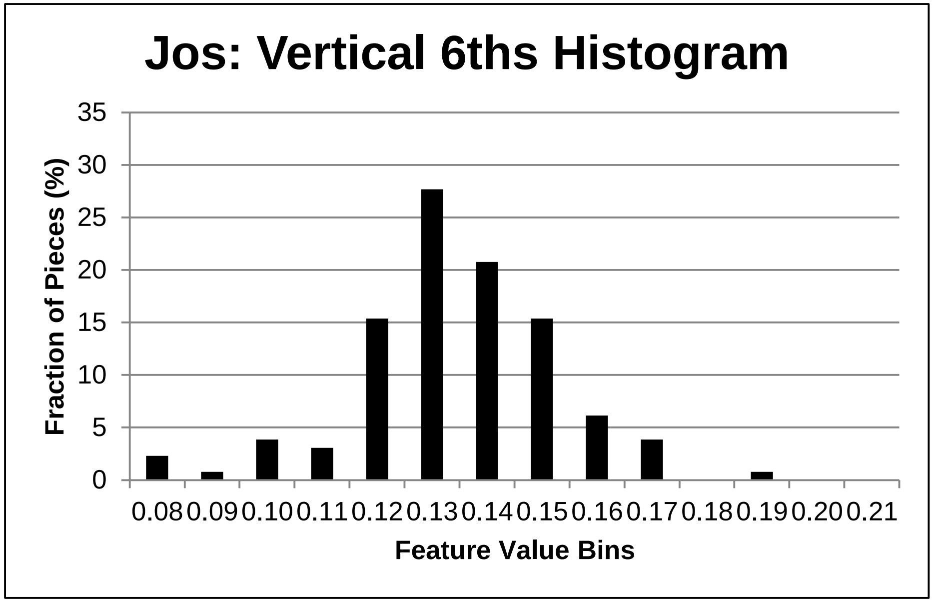
- Histograms offer a good way to visualize how the values of a feature are distributed across a corpus as a whole
  - As opposed to focusing on individual pieces
- The x-axis corresponds to a series of bins, with each corresponding to a range of values for a given feature
  - e.g. the first bin could correspond to Parallel Motion feature values between 0 and 0.1, the next bin to Parallel Motion values between 0.1 and 0.2, etc.
- The y-axis indicates the fraction of all pieces that have a feature value within the range of each given bin
  - e.g. if 30% of pieces in the corpus have Parallel Motion values between 0.1 and 0.2, then this bin (0.1 to 0.2) will have a y-coordinate of 30% (or, equivalently, 0.3)

Feature visualization: Histograms (2/6)

- In other words:
  - Each bar on a histogram represents the fraction of pieces in a corpus with a feature value falling in that bar's range of feature values
- Clarification: I am speaking here about a way to visualize a 1-dimensional feature as it is distributed across a corpus of interest
  - This is distinct from the multi-dimensional histogram features discussed earlier
    - e.g. Pitch Class Histograms
  - Although both are equally histograms, of course

Feature visualization: Histograms (3/6)

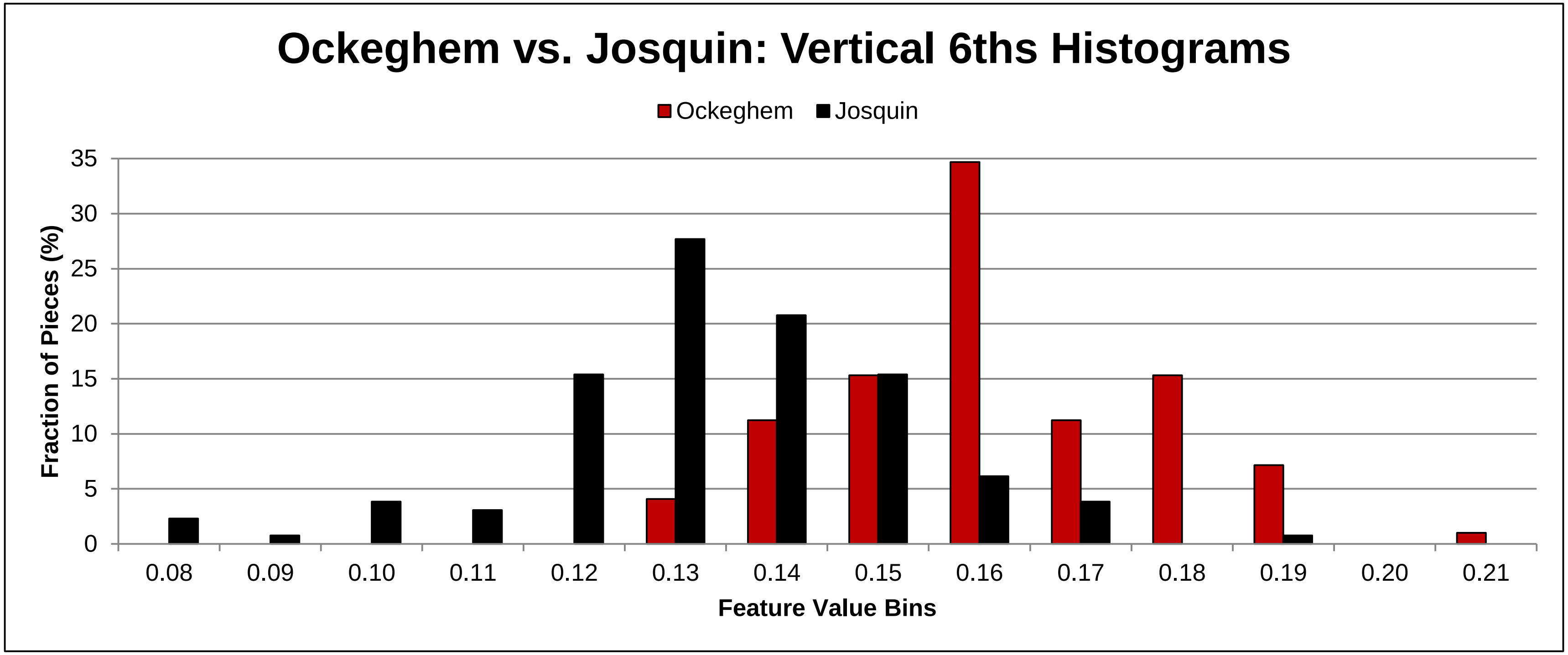




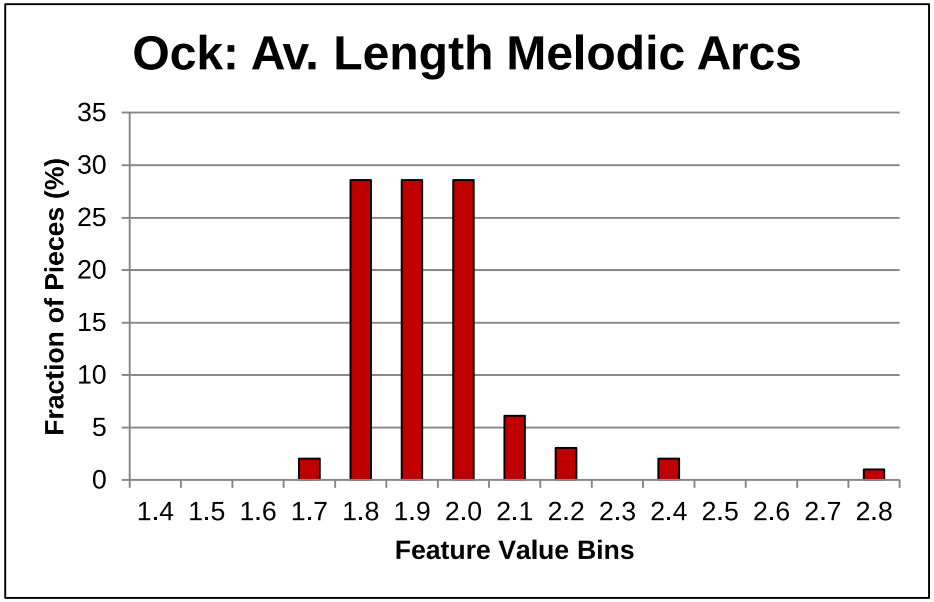
- These histograms show that Ockeghem tends to have more vertical 6ths (between all pairs of voices) than Josquin
  - Ockeghem peaks in the 0.16 to 0.17 bin, at nearly 35%
  - Josquin peaks in the 0.13 to 0.14 bin, at about 28%
- Of course, there are also clearly many exceptions
  - This feature is helpful, but is limited if only considered alone

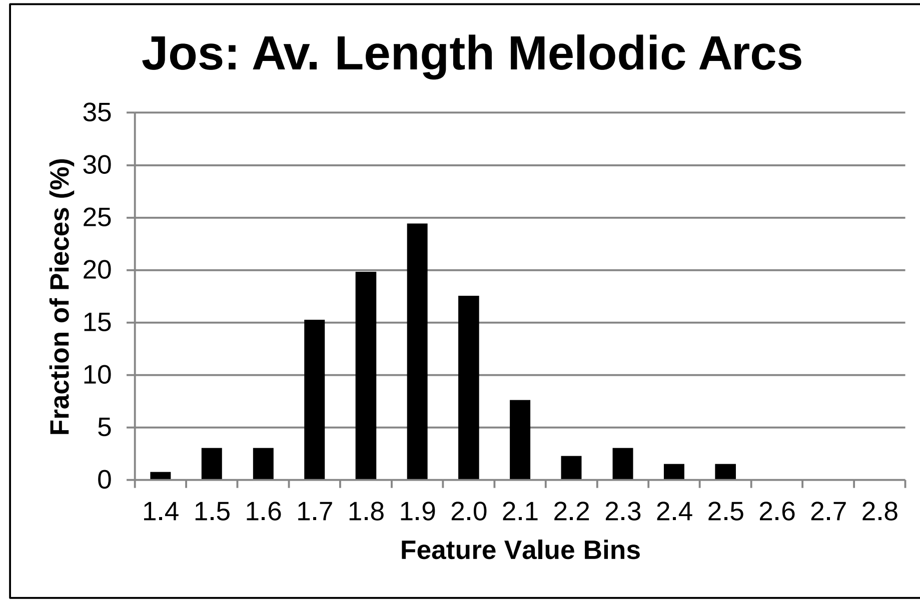
Feature visualization: Histograms (4/6)

- The histograms for both composers can be superimposed onto a single chart:



Feature visualization: Histograms (5/6)

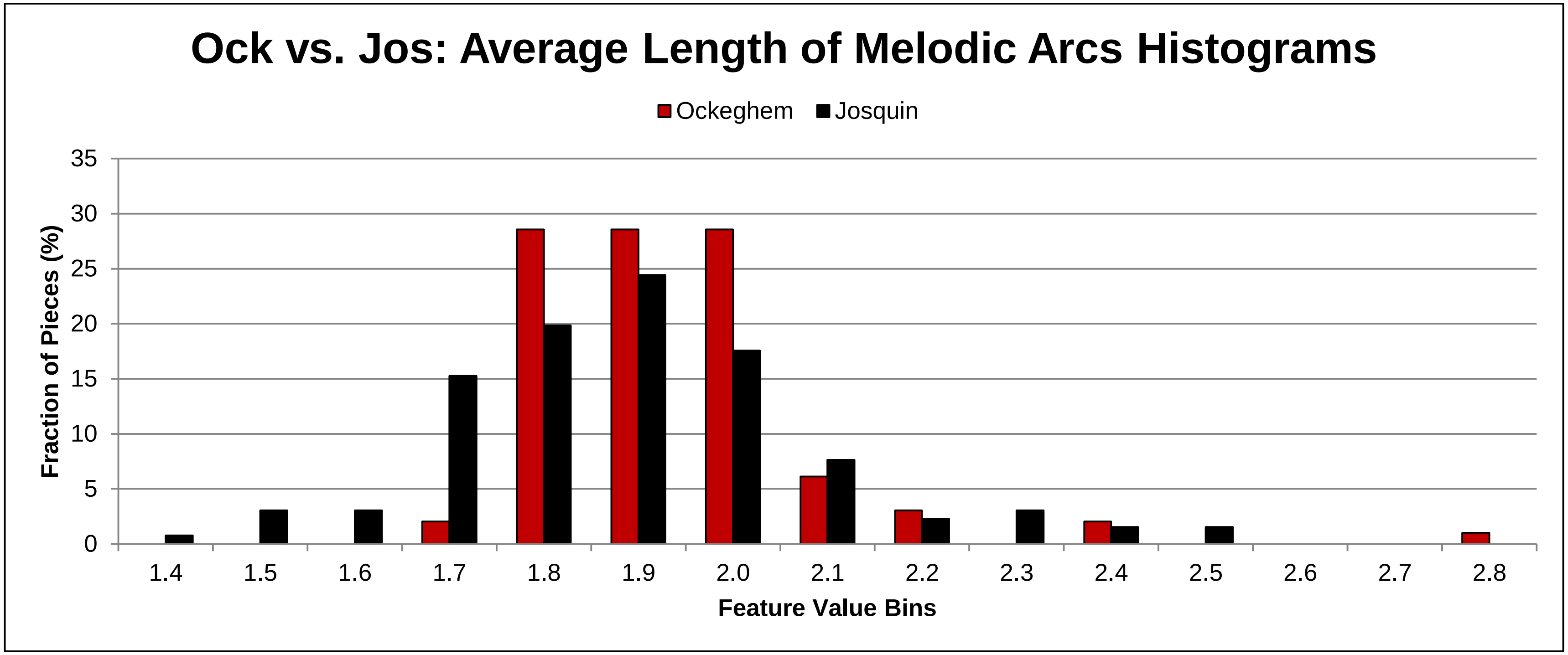




- These histograms show that Ockeghem tends to have longer melodic arcs (average number of notes separating peaks & troughs)
  - Both peak in the 1.9 to 2.0 bin
  - However, Josquin's histogram is (slightly) more skewed to the far left
- Of course, there are once again clearly many exceptions
  - This feature is also helpful, but also limited if considered alone

Feature visualization: Histograms (6/6)

- Once again, the histograms for both composers can be superimposed onto a single chart:

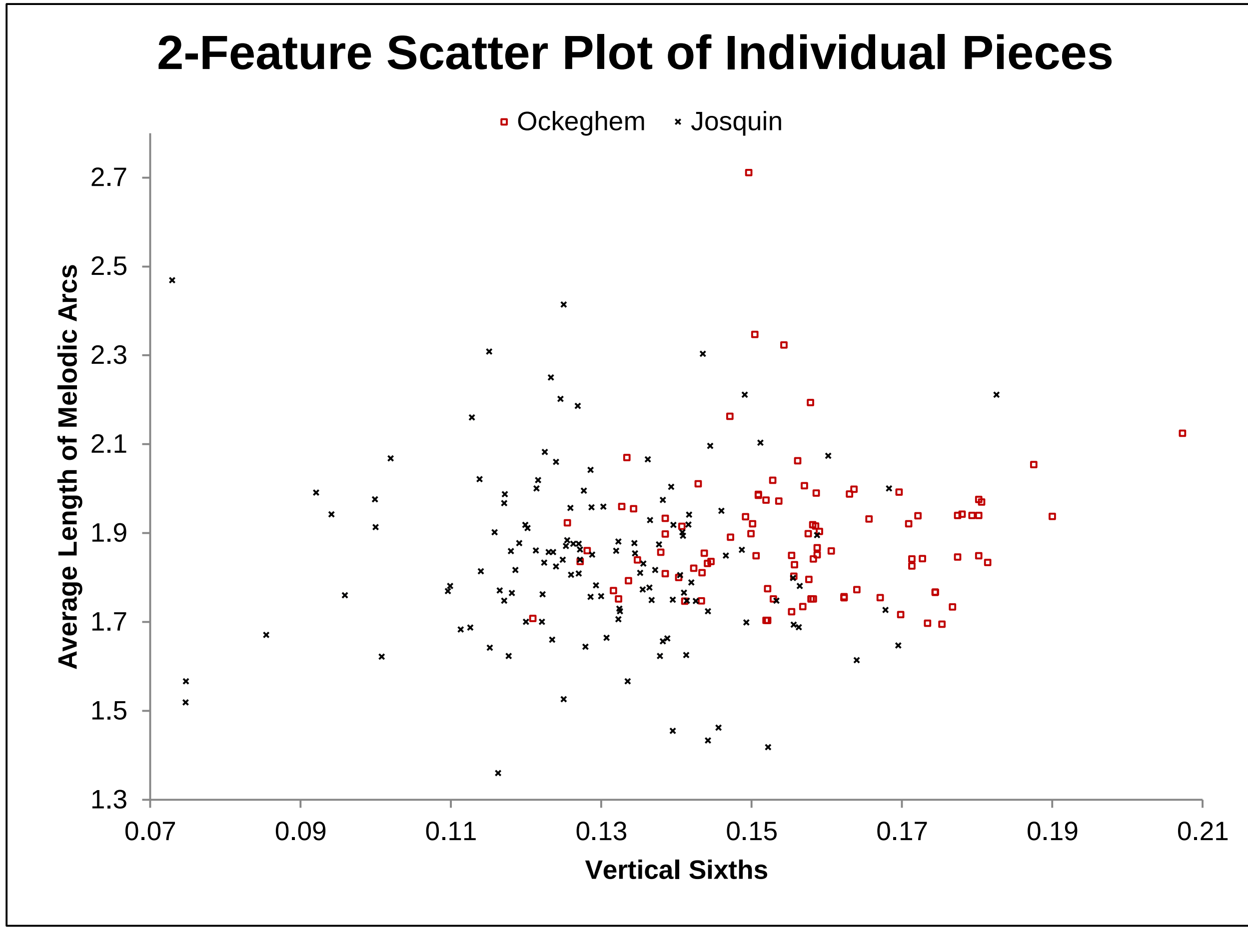


Feature visualization: Scatter plots (1/6)

- Scatter plots are another good way to visualize feature data
  - The x-axis represents one feature
  - The y-axis represents some other feature
  - Each point represents the values of these two features for a single piece
- Scatter plots let you see pieces individually, rather than aggregating them into bins (as histograms do)
  - Scatter plots also let you see more clearly how features jointly separate the different composers
- To make them easier to read, scatter plots typically have just 2 dimensions
  - Computer classifiers, in contrast, work with much larger n-dimensional scatterplots (one dimension per feature)

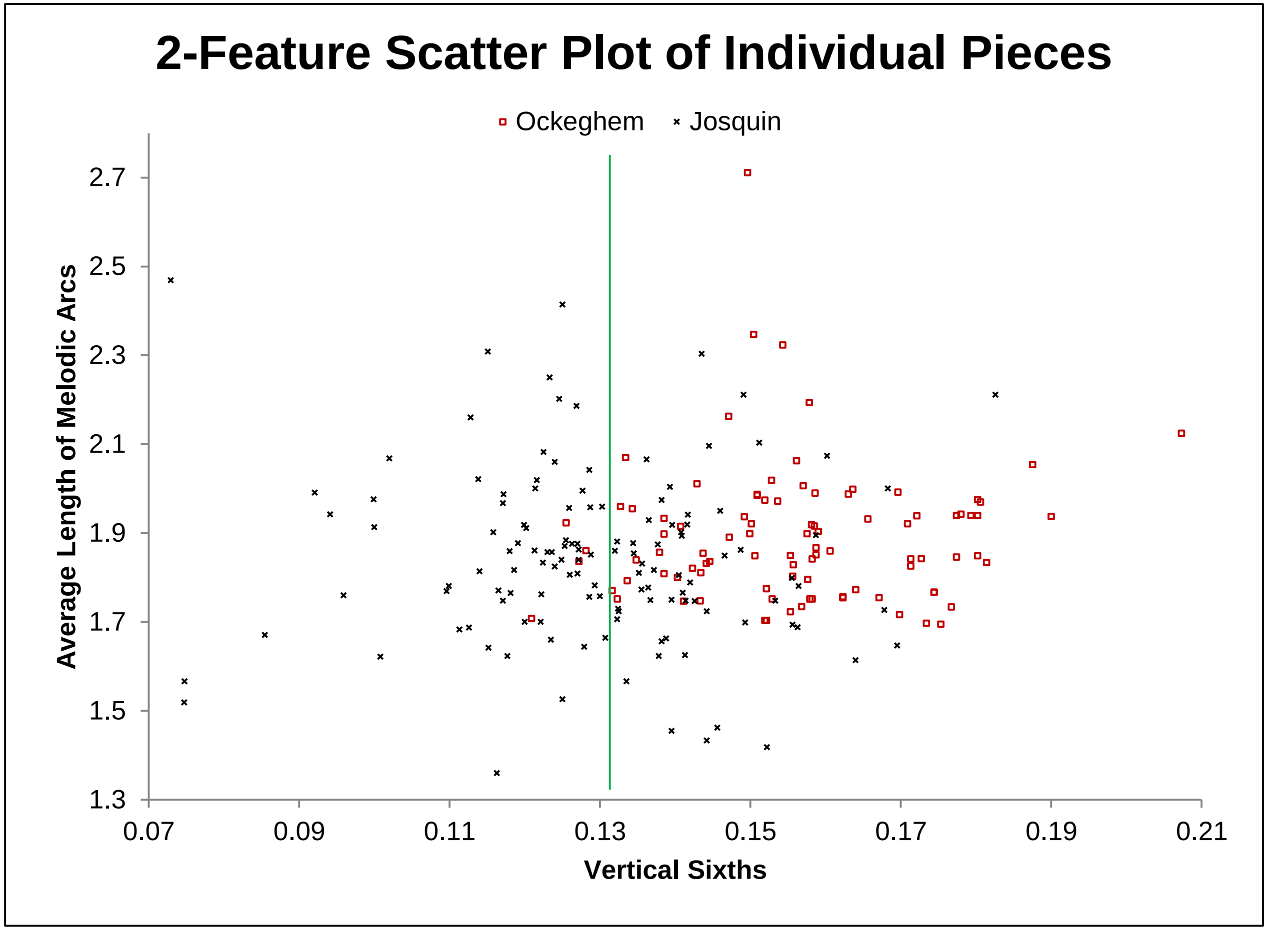
Feature visualization: Scatter plots (2/6)

- Josquin pieces tend to be left and low on this graph



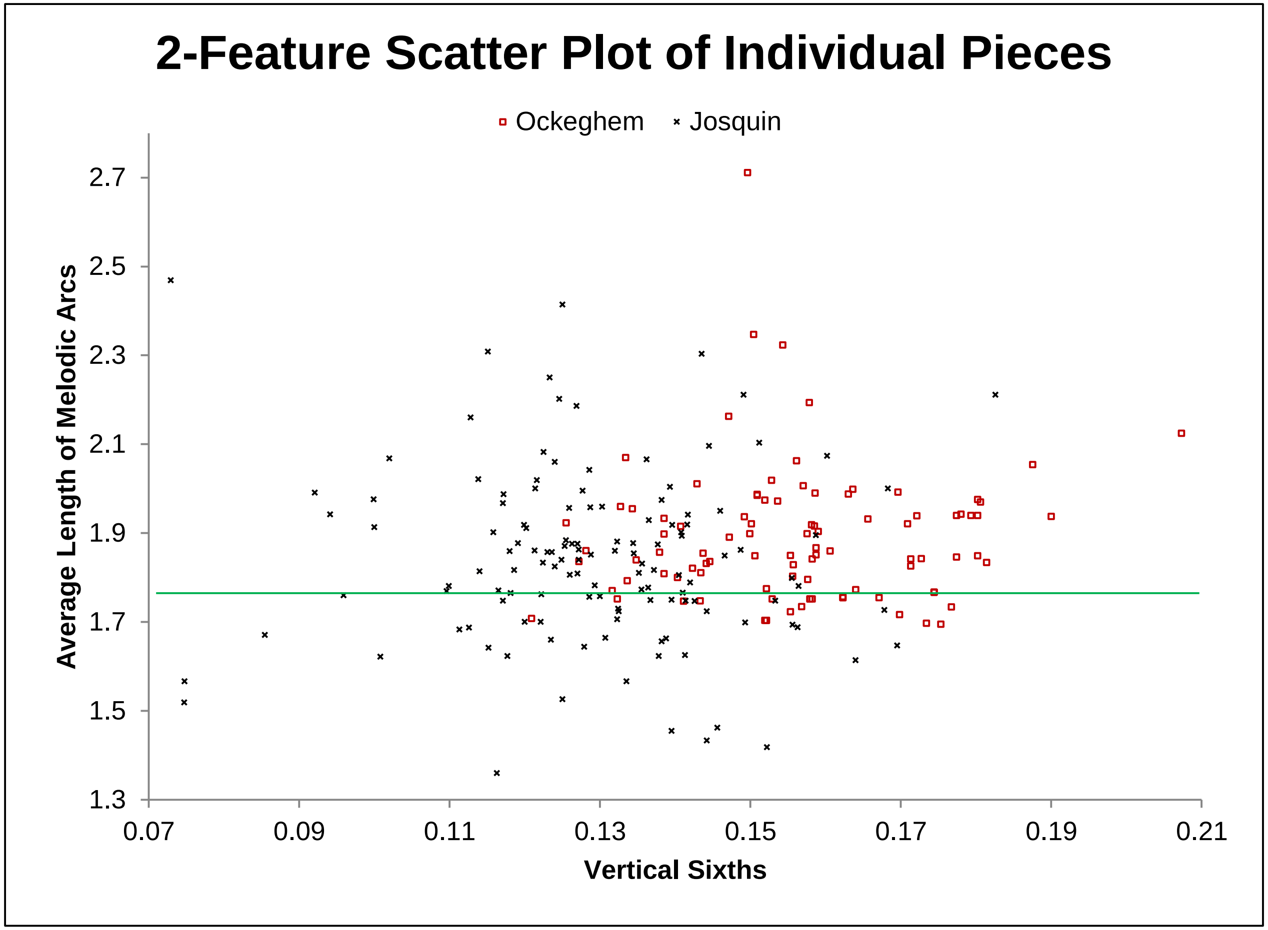
Feature visualization: Scatter plots (3/6)

- Simply drawing a single 1-D dividing line ("discriminant") results in a not entirely terrible classifier based only on Vertical Sixths
  - But many pieces would still be misclassified
  - Can get 62% classification accuracy using an SVM and just this one feature



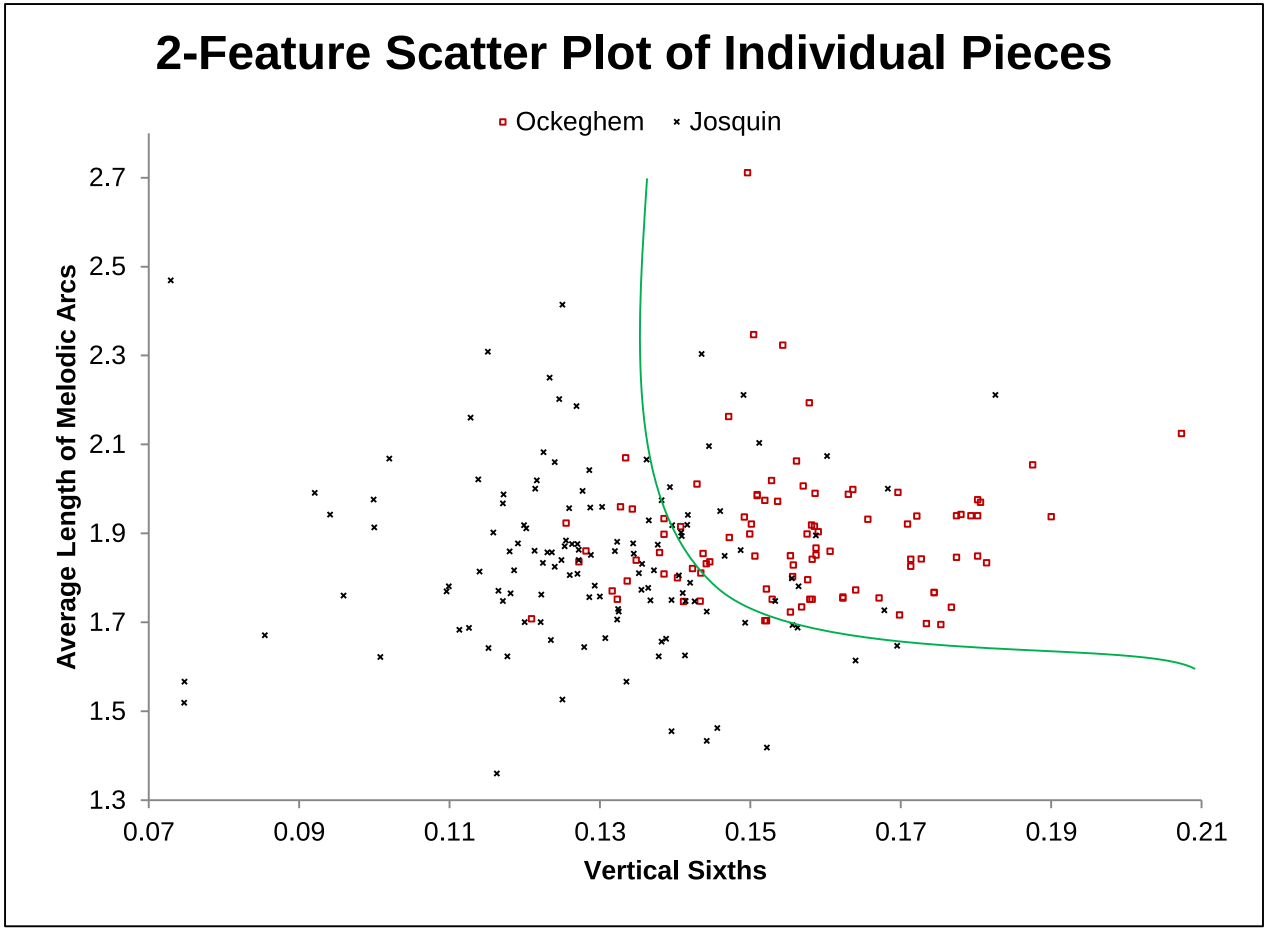
Feature visualization: Scatter plots (4/6)

- Could alternatively draw a 1-D discriminant dividing the pieces based only on the Average Length of Melodic Arcs
  - Get 57% classification accuracy using an SVM and just this one feature
  - Not as good as the Vertical Sixths discriminant (62%)



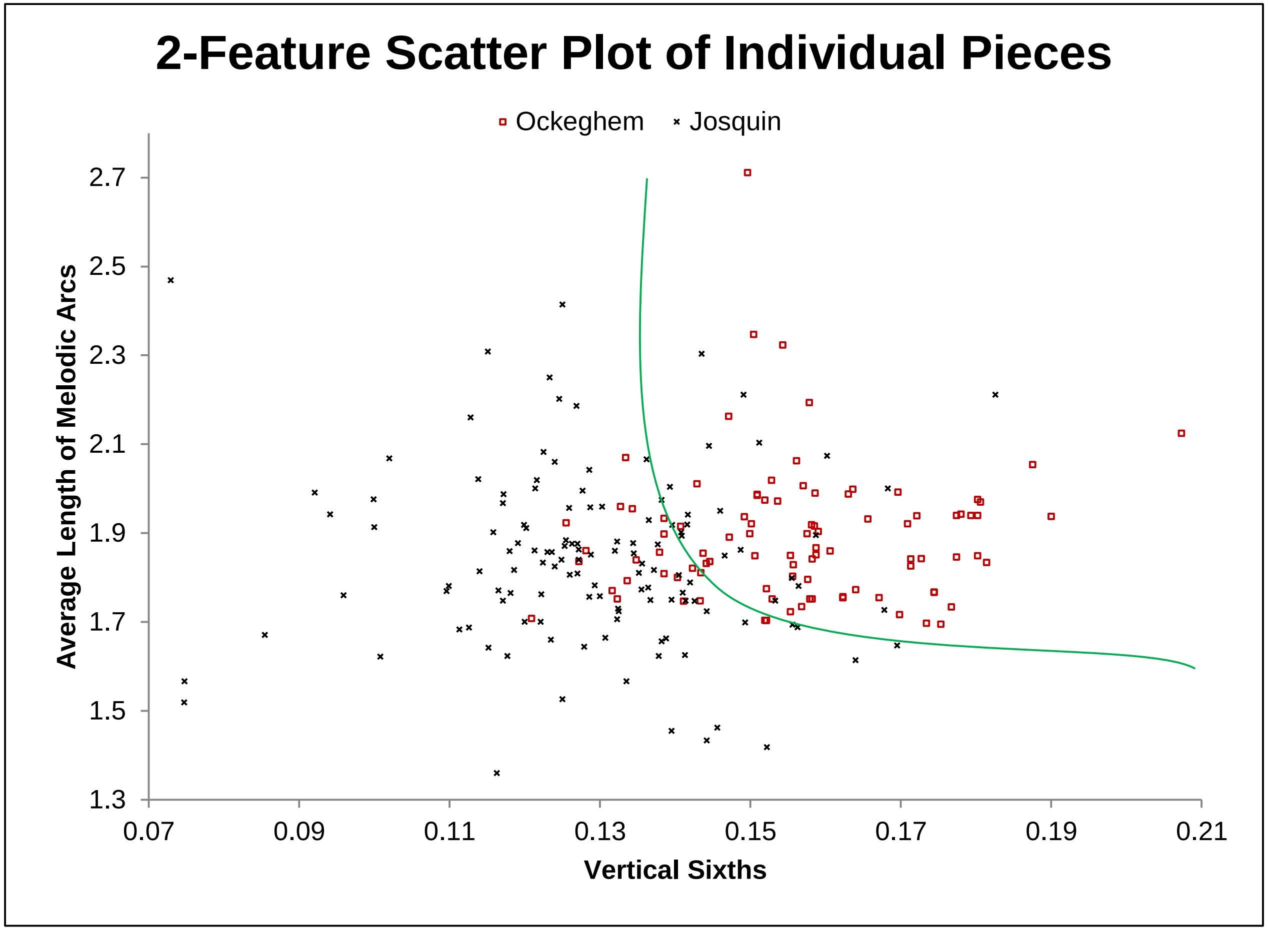
Feature visualization: Scatter plots (5/6)

- Drawing a curve (another kind of discriminant) divides the composers still better than either of the previous discriminants
  - Get 80% accuracy using an SVM and just these 2 features!
- More than 2 features are clearly needed to improve performance



Feature visualization: Scatter plots (6/6)

- In fact, many (but not all) types of machine learning in effect simply learn where to place these kinds of discriminants as they train
- But typically with many more then just two features, of course



Benefits of features

- Can quickly perform consistent empirical studies involving huge quantities of music
- Can be applied to diverse types of music in consistent ways
- Permit simultaneous consideration of thousands of features and their interrelationships
  - One can statistically condense many features into more interpretable low-dimensional spaces when needed
- No need to formally specify any queries or heuristics before beginning analyses
  - But one may if one wishes to, of course
- Help to avoid potentially incorrect ingrained assumptions and biases

Salience

- Two fundamental differences between traditional and feature-based approaches to analysis are linked to:
  - (Perceived) salience of particular pieces
  - (Perceived) salience of particular musical characteristics
- Human experts know (or assume they know?) what is important to look at
  - Due to time constraints, experts tend to focus primarily on the pieces (or excerpts) and the musical characteristics they expect to be important
  - This means that, in many research projects, the significant majority of a given repertoire is left unstudied, and many musical characteristics are left unexplored
  - The selected pieces or characteristics may not be representative
- Computers, in contrast, have no expectations as to what is important, and time is much less of a constraint for them
  - So they can look at everything we let them look at

But . . .

- Does a computational feature-based approach really avoid bias?
  - What if the makeup of the research corpus computers are provided with is limited or biased?
  - What if the encoding of the music is biased?
    - A particular problem if files with inconsistent encodings (and editorial decisions) are compared
  - What if the particular features that are implemented are limited or biased?

Missing feature types

- Certain essential areas of insight are left uninvestigated by content-based symbolic features (at least so far)
  - Qualities that are difficult to precisely define and measure consistently
    - e.g. amount and types of imitation
  - Text
    - Although text mining methodologies can be used
  - Historical evidence

Computers need us!

- Remember that a feature-based approach is useless without:
  - Human experts to ask important questions
  - Human experts to interpret results musically
  - Human experts to place feature values in the larger context
- Automatically extracted features are a tool that expert musicologists and theorists can add to their already rich toolbox
  - Features are a great tool that opens up many new possibilities, but a tool that this is of very limited utility by itself

Choosing features to implement

- Which features do we need?
  - The ones that are relevant to the kinds of music under consideration
  - Including the ones we already know or suspect are important
  - Including the ones that are important, but we do not know it yet
- So, we need a lot of diverse features!
  - So we can deal with many types of music
  - So we can address the interests of many different researchers
  - So we encourage unexpected but important results
  - So we are less likely to miss out on important insights
- The same can be said for data
  - The more music there is and the more varied it is the better!
  - We'll return briefly to data in a bit, but let's focus on features for the moment . . .

jSymbolic: Introduction

- jSymbolic is a software platform for extracting features from symbolic music
  - Part of the much larger (multimodal) jMIR package
- Compatible with Macs, PCs and Linux computers
- Free and open-source

What does jSymbolic do?

- (Version 2.2) extracts 246 unique features
- Some of these are multi-dimensional histograms, including:
  - Pitch and pitch class histograms
  - Melodic interval histograms
  - Vertical interval histograms
  - Chord types histograms
  - Rhythmic value histograms
  - Beat histograms
  - Instrument histograms
- In all, (version 2.2) extracts a total of 1497 separate values

jSymbolic: Feature types (1/3)

- Pitch Statistics:
  - What are the occurrence rates of different pitches and pitch classes?
  - How tonal is the piece?
  - How much variety in pitch is there?
- Melody / horizontal intervals:
  - What kinds of melodic intervals are present?
  - How much melodic variation is there?
  - What kinds of melodic contours are used?
- Chords / vertical intervals:
  - What vertical intervals are present?
  - What types of chords do they combine to make?
  - How much harmonic movement is there?

jSymbolic: Feature types (2/3)

- Texture:
  - How many independent voices are there and how do they interact (e.g. moving in parallel, crossing voices, etc.)?
- Rhythm:
  - Rhythmic values of notes
  - Intervals between the attacks of different notes
  - Use of rests
  - What kinds of meter is used?
  - Rubato?
- Instrumentation:
  - What types of instruments are present and which are given particular importance relative to others?
- Dynamics:
  - How loud are notes and what kinds of dynamic variations occur?

jSymbolic: Feature types (3/3)

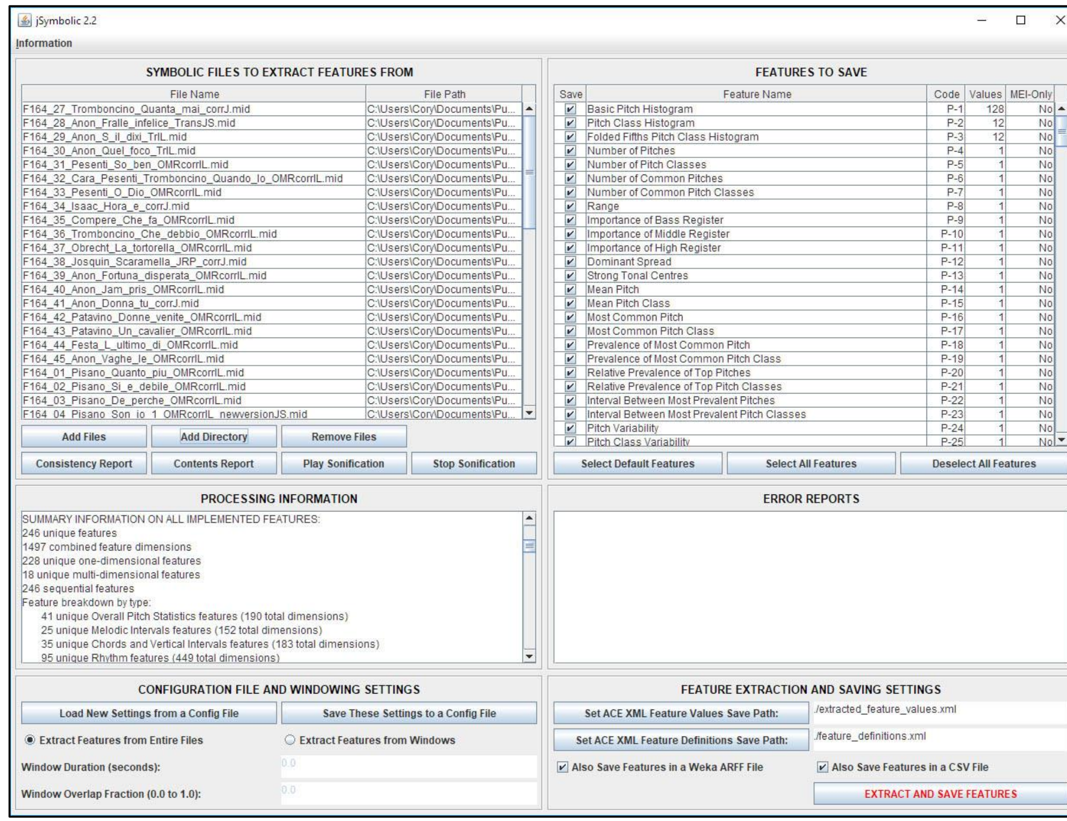
- jSymbolic only (for now) extracts features associated with musical content
- There are thus no features associated with:
  - Text
  - Historical evidence
- This is partly a disadvantage:
  - Obviously these kinds of information can be essential
  - Researchers using jSymbolic features must of course use their expertise to consider extracted features in the larger context
- It is also partly an advantage, however:
  - It allows us to (temporarily) focus only on the music, so that we can find insights there that we might otherwise have missed

Other music research software

- jSymbolic is intrinsically different from other software used in empirical symbolic music research
  - e.g. music21 (includes a port of the original jSymbolic features)
  - e.g. Humdrum
  - e.g. VIS
- This other software is excellent for finding exactly where specific things one is searching for happen
  - Perfect for very targeted research based on specific searches
- jSymbolic, in contrast, allows one to acquire large amounts of summary information about music with or without *a priori* expectations of what one is looking for
  - Good for general annotation of symbolic databases
  - Good for statistical analysis and machine learning
  - Good for free exploratory research
  - Good for large-scale validation of theoretical models

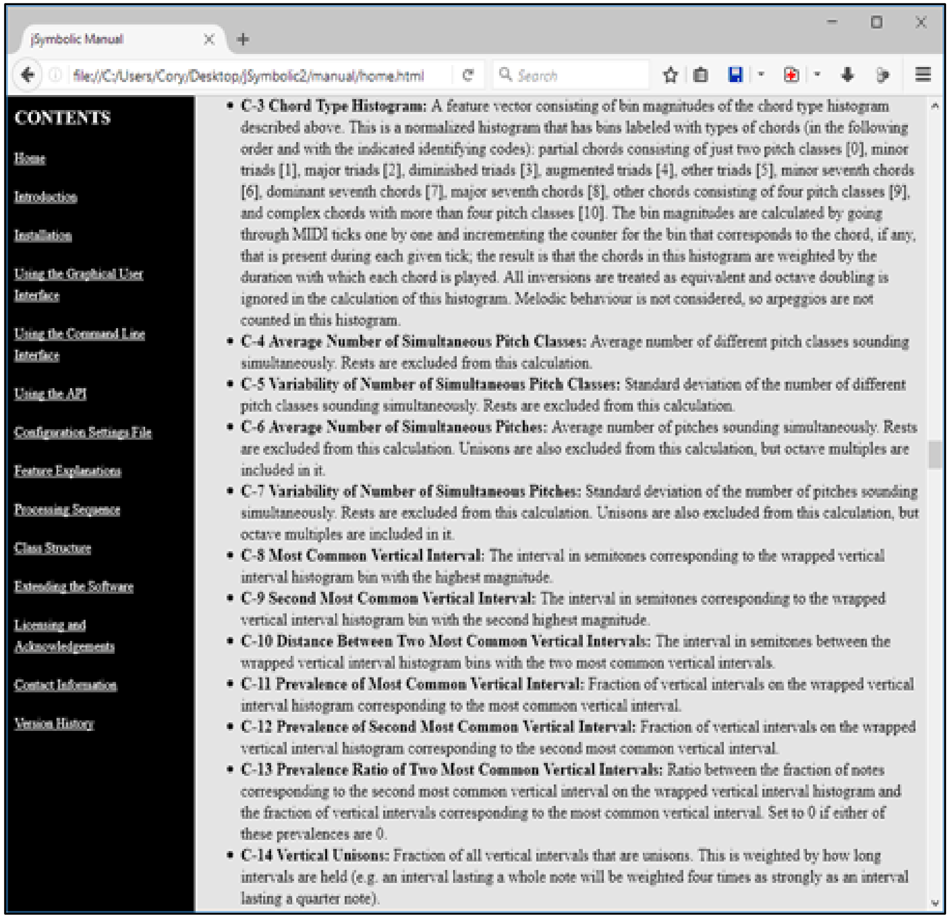
jSymbolic: User interfaces

- Graphical user interface
- Command line interface
- Java API
- Rodan workflow for distributed processing



jSymbolic: Manual

- Extensive manual includes:
  - Detailed feature descriptions
  - Detailed instructions on installation and use
- There is also a step-by-step tutorial with worked examples



jSymbolic Manual

file:///C:/Users/Cory/Desktop/jSymbolic2/manual/home.html

CONTENTS

- **C-3 Chord Type Histogram:** A feature vector consisting of bin magnitudes of the chord type histogram described above. This is a normalized histogram that has bins labeled with types of chords (in the following order and with the indicated identifying codes): partial chords consisting of just two pitch classes [0], minor triads [1], major triads [2], diminished triads [3], augmented triads [4], other triads [5], minor seventh chords [6], dominant seventh chords [7], major seventh chords [8], other chords consisting of four pitch classes [9], and complex chords with more than four pitch classes [10]. The bin magnitudes are calculated by going through MIDI ticks one by one and incrementing the counter for the bin that corresponds to the chord, if any, that is present during each given tick; the result is that the chords in this histogram are weighted by the duration with which each chord is played. All inversions are treated as equivalent and octave doubling is ignored in the calculation of this histogram. Melodic behaviour is not considered, so arpeggios are not counted in this histogram.
- **C-4 Average Number of Simultaneous Pitch Classes:** Average number of different pitch classes sounding simultaneously. Rests are excluded from this calculation.
- **C-5 Variability of Number of Simultaneous Pitch Classes:** Standard deviation of the number of different pitch classes sounding simultaneously. Rests are excluded from this calculation.
- **C-6 Average Number of Simultaneous Pitches:** Average number of pitches sounding simultaneously. Rests are excluded from this calculation. Unisons are also excluded from this calculation, but octave multiples are included in it.
- **C-7 Variability of Number of Simultaneous Pitches:** Standard deviation of the number of pitches sounding simultaneously. Rests are excluded from this calculation. Unisons are also excluded from this calculation, but octave multiples are included in it.
- **C-8 Most Common Vertical Interval:** The interval in semitones corresponding to the wrapped vertical interval histogram bin with the highest magnitude.
- **C-9 Second Most Common Vertical Interval:** The interval in semitones corresponding to the wrapped vertical interval histogram bin with the second highest magnitude.
- **C-10 Distance Between Two Most Common Vertical Intervals:** The interval in semitones between the wrapped vertical interval histogram bins with the two most common vertical intervals.
- **C-11 Prevalence of Most Common Vertical Interval:** Fraction of vertical intervals on the wrapped vertical interval histogram corresponding to the most common vertical interval.
- **C-12 Prevalence of Second Most Common Vertical Interval:** Fraction of vertical intervals on the wrapped vertical interval histogram corresponding to the second most common vertical interval.
- **C-13 Prevalence Ratio of Two Most Common Vertical Intervals:** Ratio between the fraction of notes corresponding to the second most common vertical interval on the wrapped vertical interval histogram and the fraction of vertical intervals corresponding to the most common vertical interval. Set to 0 if either of these prevalences are 0.
- **C-14 Vertical Unisons:** Fraction of all vertical intervals that are unisons. This is weighted by how long intervals are held (e.g. an interval lasting a whole note will be weighted four times as strongly as an interval lasting a quarter note).

jSymbolic: File formats

- Input:
  - MIDI
  - MEI
  - MusicXML (after conversion)
- Output:
  - CSV
  - ACE XML
  - Weka ARFF

Why MIDI?

- jSymbolic's features have been designed to deal most natively with MIDI
  - As opposed to alternatives like MusicXML and MEI
- But MIDI has serious problems for music analysis:
  - e.g. Cannot distinguish enharmonic equivalents
    - Pitch is encoded in semitone steps
  - e.g. Can have problems with rhythmic synchronization of "simultaneous" note attacks
    - Some MIDI encodings are real-time performance captures, so there may be slight time offsets
    - Some score editing software artificially creates such offsets to make music playback sound more natural

Benefits of MIDI (1/2)

- MIDI is better than general symbolic alternative file formats at representing non-Western or live musical traditions
  - e.g. Can encode microtones precisely
  - e.g. Can encode complex rhythms difficult to annotate using Western notation
  - e.g. Can be used to symbolically record performances directly
- Far more (and more diverse) music has been encoded in MIDI than any symbolic alternative

Benefits of MIDI (2/2)

- MIDI is a stable, mature format
  - MIDI encoders and decoders are widely available
  - MIDI is compatible with almost all symbolic software
  - MIDI files are reliably easy and consistent to parse
    - Unlike alternatives like MEI which, despite its many advantages, can be very difficult to write a stable parser for, given its in-flux specification and free-wheeling encoding culture
- MIDI can be easily and directly sonified
  - Almost all symbolic alternatives must be first converted to MIDI to be listened to
- MIDI does not allow ambiguity, it forces encoders to commit
  - Alternatives like MEI purposely (and appropriately for archiving) allow ambiguous encodings
  - While good for the purposes of archiving, such ambiguity is highly problematic when performing automatic analysis

jSymbolic: Miscellany

- Windowed feature extraction
  - Including overlapping windows
- Configuration files
  - Pre-set feature choices
  - Pre-set input and output choices
  - More
  - Useful for saving specific feature extraction jobs
- Can combine jSymbolic with other jMIR components to perform multimodal research
  - i.e. combine symbolic features with other features extracted from audio, lyrics and cultural data
  - This improves results substantially! (McKay et al. 2010)

jSymbolic: Extensibility

- jSymbolic is specifically designed such that music scholars can design their own features and work with programmers to then very easily add these features to the jSymbolic infrastructure
  - Fully open source
  - Modular plug-in feature design
  - Automatically handles feature dependencies and scheduling
  - Very well-documented code

To come in jSymbolic 3.0

- Many miscellaneous usability improvements
  - Including expanded multilingual support
- Many new features
  - 533 unique features and 2040 feature values as of March 3, 2020, in total
  - Including features base on note onset slices
  - Including features base on n-grams

Research involving jSymbolic

- I will now briefly highlight several research projects that have been carried out based on jSymbolic features
  - To give you an idea of what is possible
- I put special emphasis on a study comparing Renaissance composers
  - It is particularly illustrative
- Several other studies will also be discussed
  - In less detail

Composer identification study

- Related paper: MedRen 2017
- Used jSymbolic features to automatically classify pieces of Renaissance music by composer
  - As an example of the kinds of things that can be done with jSymbolic
  - As a meaningful research project in its own right

RenComp7 dataset

- Began by constructing the "RenComp7" dataset:
  - 1584 MIDI files
  - By 7 Renaissance composers
- Combines:
  - Top right: Music drawn from the Josquin Research Project (Rodin, Sapp and Bokulich)
  - Bottom right: Music by Palestrina (Miller 2004) and Victoria (Sigler, Wild and Handelman 2015)

Composer	Files
Busnoys	69
Josquin *(only includes the 2 most secure Jesse Rodin groups)*	131
La Rue	197
Martini	123
Ockeghem	98

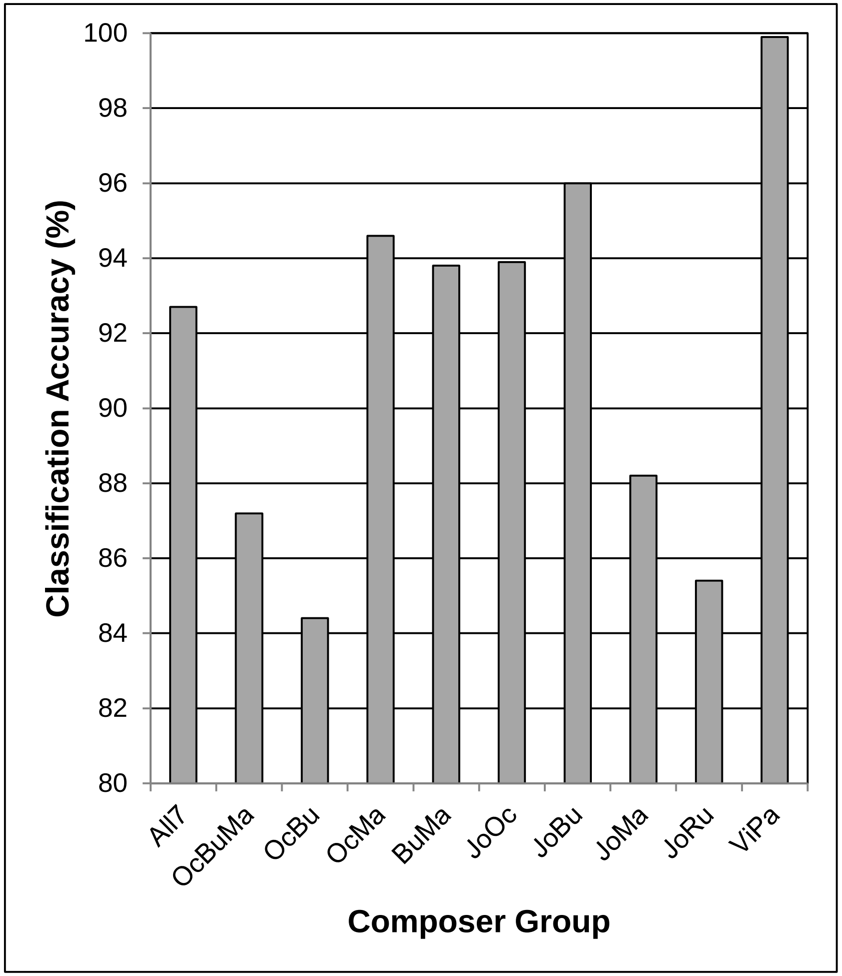
Composer	Files
Palestrina	705
Victoria	261

Methodology

- Extracted 721 feature values from each of the 1584 RenComp7 files using jSymbolic 2.0
- Used machine learning to teach a (SVM) classifier to automatically distinguish the music of the composers
  - Based on the jSymbolic features
- Used statistical analysis to gain insight into relative compositional styles
- Performed several versions of this study
  - Classifying amongst all 7 composers
  - Focusing only on smaller subsets of composers
    - Some more similar, some less similar

Classification results

Composer Group	Classification Accuracy
All 7	92.7%
Ockeghem / Busnoys / Martini	87.2%
Ockeghem / Busnoys	84.4%
Ockeghem / Martini	94.6%
Busnoys / Martini	93.8%
Josquin / Ockeghem	93.9%
Josquin / Busnoys	96.0%
Josquin / Martini	88.2%
Josquin / La Rue	85.4%
Victoria / Palestrina	99.9%



Direct applications of such work

- Validating existing suspected but uncertain attributions
- Helping to resolve conflicting attributions
- Suggesting possible attributions of currently entirely unattributed scores

How do the composers differ?

- Some very interesting questions:
  - What musical insights can we learn from the jSymbolic feature data itself?
  - In particular, what can we learn about how the music of different composers differs?
- Chose to focus on two particular cases:
  - Josquin vs. Ockeghem: Relatively different
  - Josquin vs. La Rue: Relatively similar

A priori expectations (1/2)

- What might an expert musicologist expect to differentiate the composers?
  - Before actually examining the feature values
- Once formulating these expectations, we can then see if the feature data confirms or repudiates these expectations
  - Both are useful!
- We can also see if the feature data reveals unexpected insights
- I consulted one musicologist (Julie Cumming) and one theorist (Peter Schubert), both experts in the period . . .

A priori expectations (2/2)

- Josquin vs. Ockeghem: Ockeghem may have . . .
  - Slightly more large leaps (larger than a 5th)
  - Less stepwise motion in some voices
  - More notes at the bottom of the range
  - Slightly more chords (or simultaneities) without a third
  - Slightly more dissonance
  - A lot more triple meter
  - More varied rhythmic note values
  - More 3-voice music
  - Less music for more than 4 voices
- Josquin vs. La Rue: La Rue may have . . . Hard to say!
  - Maybe more compressed ranges?

Were our expectations correct?

- Josquin vs. Ockeghem: Ockeghem may have . . .
  - OPPOSITE: Slightly more large leaps (larger than a 5th)
  - SAME: Less stepwise motion in some voices
  - SAME: More notes at the bottom of the range
  - SAME: Slightly more chords (or simultaneities) without a third
  - OPPOSITE: Slightly more dissonance
  - YES: A lot more triple meter
  - SAME: More varied rhythmic note values
  - YES: More 3-voice music
  - YES: Less music for more than 4 voices
- Josquin vs. La Rue: La Rue may have . . .
  - SAME: Maybe more compressed ranges?

Importance of empiricism

- These results show that even some of the most highly informed experts in the field can have a number of inaccurate assumptions
  - And so, it is certain, do we all
- These results highlight the important need for empirical validation in general in musicology and music theory
  - There are very likely a range of widely held beliefs and theoretical models that will in fact turn out to be incorrect when they are subjected to exhaustive and rigorous empirical examination

(Free) diving into the feature values

- There are a variety of statistical techniques for attempting to evaluate which features are likely to be effective in distinguishing between types of music
- We used seven of these statistical techniques to find:
  - The features and feature subsets most consistently statistically predicted to be effective at distinguishing composers
- We then manually examined these feature subsets to find the features likely to be the most musicologically meaningful
- IMPORTANT NOTE: exploratory studies like this ultimately need confirmatory studies on a different dataset in order to properly show statistical significance

Novel insights revealed (1/2)

- Josquin vs. Ockeghem (93.9%):
  - Rhythm-related features are particularly important
    - Josquin tends to have greater rhythmic variety
      - Especially in terms of both especially short and long notes
    - Ockeghem tends to have more triple meter
      - As expected
    - Features derived from beat histograms also have good discriminatory power
  - Ockeghem tends to have more vertical sixths
  - Ockeghem tends to have more diminished triads
  - Ockeghems tends to have longer melodic arcs

Novel insights revealed (2/2)

- Josquin vs. La Rue (85.4%):
  - Pitch-related features are particularly important
    - Josquin tends to have more vertical unisons and thirds
    - La Rue tends to have more vertical fourths and octaves
    - Josquin tends to have more melodic octaves

Research potential (1/2)

- The results above are the product of an initial accurate but relatively simple analysis
- There is substantial potential to expand this study
  - Apply more sophisticated and detailed statistical analysis techniques
  - Perform a more detailed manual exploration of the feature data
  - Implement new specialized features
  - Look at more and different composer groups

Research potential (2/2)

- Composer attribution is just one small example of the many musicological and theoretical research domains to which features and jSymbolic2 can be applied

Tools used

- All machine learning and feature selection/weighting was performed using the Weka machine learning framework
  - Free and open-source
  - Surprisingly (relatively) easy to use for such technical software

Excluded features

- Only 721 of the available 1230 jSymbolic 2.0 features were used in order to avoid bias
  - Some excluded features were irrelevant to the data under consideration
  - Some excluded features were correlated with the source of the data
- This primarily meant removing features linked to instrumentation, dynamics and tempo

Sidebar: Avoiding encoding bias (1/2)

- If music from multiple different sources is included in a study, then one must be careful to avoid making conclusions based on the source of the music rather than the underlying music itself
  - As this could corrupt the results
- Problems can occur when inconsistent editorial decisions are present. To be careful of in early music:
  - Inconsistent additions of accidentals (*musica ficta*)
  - Choice of different rhythmic note values to denote the beat
  - Differing metrical interpretations of mensuration signs
  - Transposition to different keys
- Inconsistent encoding practices can also have an effect
  - e.g. if one set of files has precise tempo markings but another is arbitrarily annotated at 120 BPM

Sidebar: Avoiding encoding bias (2/2)

- How to avoid corrupted feature-based results associated with the kinds of corpus inconsistencies and biases described above:
  - Ideally, use music files that were all consistently generated using the same methodology
    - All editorial decisions (e.g. *musica ficta*) should be applied consistently and should be documented
  - If this is not possible, then exclude all features that are sensitive to the particular biases present
- jSymbolic includes functionality that can help detect and identify these kinds of problems

Building valid digital symbolic music research corpora

- Related publication: ISMIR 2018
- Presents techniques and workflows for building large collections of symbolic digital music that avoid bias and facilitate statistically valid large-scale empirical studies
- Presents a corpus of Renaissance duos as a sample of how this can be done
  - Includes experiments with jSymbolic 2.2 features empirically demonstrating the negative effects that improper methodologies can produce

Josquin attribution study (1/3)

- Related publication: ISMIR 2017
- We also did a second composer-related study using the Josquin Research Project data
  - This one investigated the attribution of pieces suspected to be by Josquin

Josquin attribution study (2/3)

- Jesse Rodin has broken Josquin's music into 6 levels of attribution certainty
  - Based on historical sources, not musical content
- We used the jSymbolic 2.0 features to train a 2-class SVM classifier
  - First class: Josquin
    - The Josquin music in the 2 most secure Rodin levels
  - Second class: NotJosquin
    - All the JRP music available from 21 other Renaissance composers similar to Josquin
- This model was then used to classify the Josquin music in the remaining 4 Jesse Rodin levels

Josquin attribution study (3/3)

- It turns out that, the more insecure a piece is according to Rodin's classification, the less likely it was to be classified as being by Josquin by our classifier
- This demonstrates some good empirical support for Rodin's categorizations
  - This is a great example of how features extracted by a computer and human expert knowledge can complement each other

Rodin Certainty Level	% Classified as Josquin
Level 3 "Tricky"	48.6%
Level 4 "Questionable"	17.2%
Level 5 "Doubtful"	14.0%
Level 6 "Very doubtful"	5.5%

Origins of the Italian madrigal (1/3)

- Related paper: MedRen 2018
- Where did the madrigal come from?
  - The frottola (Einstein 1949)?
  - The chanson and motet in Florence (Fenlon and Haar 1988)?
  - The Florentine carnival song, villotta, and improvised solo song (A. Cummings 2004)?
- How could we analyze the music to help us decide?
  - Extracted jSymbolic 2.2 features
  - Applied machine learning and feature analysis techniques
    - As we did with composers in the MedRen 2017 study
- Constructed the "3RenGenres" corpus: MIDI files derived from Florence BNC 164-167 (c. 1520)
  - Madrigals (27 files)
  - Motets (12 files)
  - Frottole & Villotte (19 files)

Origins of the Italian madrigal (2/3)

- Madrigals and motets are the most dissimilar genres (from an empirical content-based perspective)
  - Because they can be easily distinguished with features and machine learning
- Frottole / Villotte and madrigals are the most similar genres
  - Because they are harder to tell apart
- Frottole / Villotte and motets are in between

Genre Group	Classification Accuracy
Frottole / Villotte vs. Madrigals	64.6%
Frottole / Villotte vs. Motets	84.8%
Madrigals vs. Motets	99.1%

Origins of the Italian madrigal (3/3)

- Expert *a priori* prediction results:
  - Half of the predictions were correct
  - Half were partly or completely incorrect
- Exploratory feature analysis results:
  - Features related to rhythm and (to a lesser extent) texture were by far the most important
  - Pitch-related features were almost irrelevant (relatively speaking) in distinguishing the genres
- Opens very promising avenues for future research

Iberian vs. Franco-Flemish music (1/4)

- Related paper: Anatomy of Polyphonic Music around 1500 Conference (2018)
- Research question:
  - Is Iberian Renaissance music demonstrably stylistically distinct from Franco-Flemish music of the time?
- Investigated empirically:
  - Extracted jSymbolic 2.2 features from a dataset of Iberian and Franco-Flemish masses and motets
  - Trained machine learning models that could distinguish between Iberian and Franco-Flemish music
    - Based on these features
  - Tested expert predictions to see if they match the actual musical data
  - Used statistical analysis techniques to find those features that strongly (statistically) distinguish Iberian and Franco-Flemish music

Dataset used

- Used the “FraFle/Iber” dataset provided by the Anatomy project's team
- Consists of masses and motets
- 467 MIDI files total
- IMPORTANT CAVEAT:
  - This dataset was prepared for initial rough exploration
  - It was no yet fully cleaned, so it (and the results about to be presented) may be subject to a certain amount of encoding bias

Region	Composers	Files
Franco-Flemish Mass movements	3	286
Franco-Flemish Motets	3	59
Iberian Mass movements	7	79
Iberian Motets	10	43

Iberian vs. Franco-Flemish music (3/4)

- Performed three versions of this study, where the music was classified by region:
  - Iberian masses and motets vs. Franco-Flemish masses and motets: 97.9%
  - Iberian masses vs. Franco-Flemish masses: 99.6%
  - Iberian motets vs. Franco-Flemish motets: 87.7%
- So, the Iberian music stylistically is distinct from the Franco-Flemish music, especially the masses
  - Because the classifier could tell the musics apart so easily

Iberian vs. Franco-Flemish music (4/4)

- Comparing expert *a priori* predictions (submitted anonymously) with empirical data:
  - Expert predictions matched the data very well for motets, but less well for masses
- Analysis of statistically most predictive features:
  - Matched four of the features highlighted by experts
  - Revealed three features not highlighted by experts
- Highlights important new areas where more research could be very revealing

Genre classification study (1/4)

- Related paper: unpublished 2017
- Classified music according to a variety of genres using jSymbolic 2.0 features
  - Including popular music
- Used our SLAC dataset to do this
  - Composed of 250 pieces
- Each piece in SLAC has a matching:
  - MIDI transcription
  - Text file containing lyrics (if any)
  - Audio recording
  - Metadata mined from search engines
    - Containing "cultural" information

Genre classification study (2/4)

- SLAC is divided among 10 genres
  - 25 pieces of music per genre
- These 10 genres can be grouped into 5 pairs of similar genres
  - This permits both 5-genre and 10-genre experiments
- The genres are:
  - Blues: Modern Blues and Traditional Blues
  - Classical: Baroque and Romantic
  - Jazz: Bop and Swing
  - Rap: Hardcore Rap and Pop Rap
  - Rock: Alternative Rock and Metal

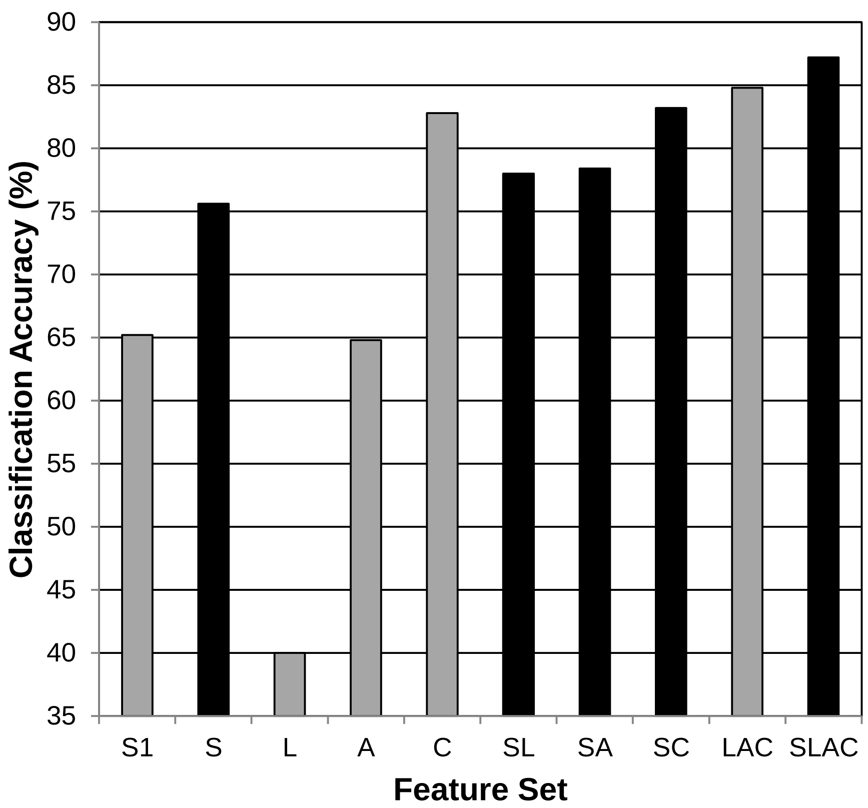
Genre classification study (3/4)

- Using just the MIDI files, the jSymbolic 2.0 features were able to classify among the 10 genres 75.6% of the time
- Experiments were also performed with other types of features, alone and in various combinations . . .

Genre classification study (4/4)

- S1 = jSymbolic 1.0
- S = jSymbolic 2.0
- L = jLyrics
- A = jAudio
- C = jWebMiner

- Combining different feature groups substantially improved performance:
  - 87.2% among 10 classes
- This offers support for multimodal research
  - i.e. research involving different types of data



A few more samples of research involving jSymbolic

- Using features to generate style-specific music
  - Melomics, 2012 …
- Analyzing and generating fado music
  - Gonzaga Videira, 2015
- Content-based searches of symbolic music databases
  - McKay et al. 2017
- Comparing compositional styles of La Rue and Peñalosa
  - Cuenca, 2018
- Patterns in Dutch folk music
  - Ret et al., 2018
- Overview and comparison of jSymbolic 2.2
  - McKay et al. 2018
- Exploring anonymous and doubtfully attributed Coimbra masses
  - Cuenca and McKay 2019

Overview of jMIR

- jSymbolic is actually part of my larger jMIR toolset
  - Designed specifically for multimodal music research
- Primary tasks performed:
  - Feature extraction
  - Machine learning
  - Data storage file formats
  - Dataset management
    - Acquiring, correcting and organizing metadata

Characteristics of jMIR

- Has a separate software component to address each important aspect of automatic music classification
  - Each component can be used independently
  - Can also be used as an integrated whole
- Free and open source
  - http://jmir.sourceforge.net
- Architectural emphasis on providing an extensible platform for iteratively developing new techniques and algorithms
- Interfaces designed for both technical and non-technical users
- Facilitates multimodal research

jMIR components

- jSymbolic: Feature extraction from MIDI files
- jAudio: Audio feature extraction
- jWebMiner: Cultural feature extraction
- jLyrics: Extracts features from lyrical transcriptions
- ACE: Meta-learning classification engine
- ACE XML: File formats
  - Features, feature metadata, instance metadata and ontologies
- lyricFetcher: Lyric mining
- Codaich, Bodhidharma MIDI and SLAC: datasets
- jMusicMetaManager: Metadata management
- jSongMiner: Metadata harvesting
- jProductionCritic: Detecting mixing and editing errors
- jMIRUtilities: Infrastructure for conducting experiments

SIMSSA and MIRAI context (1/2)

- Much of the work I have presented is part of the multi-institutional SIMSSA and MIRAI projects
- These projects aim to make the huge number of digitized scores held at libraries and other institutions around the world accessible and searchable to the public
  - Using optical music recognition (OMR) to transform images of scores into digital symbolic formats
  - Annotating music with pre-extracted jSymbolic features
  - And much more . . .

SIMSSA and MIRAI context (2/2)

- Not only will this allow music researchers to query scores in relatively traditional ways (e.g. using textual metadata or melodic segments); it will also allow content-based searches based on feature values and ranges
  - A researcher could thus filter results based on the amount of chromaticism in a piece, for example, or the amount of parallel motion between voices
- Can use statistical analysis to build multidimensional combinations of features that allow sophisticated searches
  - e.g. the level of tonality of a piece, where this is estimated based on the values of several existing features
- Can use features to train classification models for directly assisting research by music scholars
  - e.g. identifying composers of pieces with unknown attribution

Acknowledgements

- Thanks to my colleagues and the students in the SIMSSA and MIRAI research groups, especially:
  - Julie Cumming
  - Ichiro Fujinaga
  - Tristano Tenaglia
  - Rían Adamian
- Thanks to the Fonds de recherche du Québec - Société et culture (FRQSC) and the Social Sciences and Humanities Research Council of Canada (SSHRC) the for their generous funding

jSymbolic demo

- Tutorial:
  - jmir.sourceforge.net/manuals/jSymbolic_tutorial/home.html
- Manual:
  - jmir.sourceforge.net/manuals/jSymbolic_manual/home.html
- jSymbolic download:
  - sourceforge.net/projects/jmir/files/jSymbolic/

Thanks for your attention!

- **jSymbolic:** http://jmir.sourceforge.net
- **E-mail:** cory.mckay@mail.mcgill.ca

McGill

Schulich School of Music
École de musique Schulich

CIRMMT Centre for Interdisciplinary Research in Music Media and Technology

Social Sciences and Humanities Research Council of Canada

Conseil de recherches en sciences humaines du Canada

Canada

Marianopolis College

Fonds de recherche sur la société et la culture Québec

DDMAL Distributed Digital Music Archives & Libraries Lab

SIMSSA Single Interface for Music Score Searching and Analysis

jMIR