Corpus Studies and Feature Extraction Using jSymbolic

Part 2

Cory McKay
Marianopolis College and CIRMMT

Julie Cumming
McGill University and CIRMMT

NEH Institute / American Musicological Society
New York City, U.S.A.
June 20, 2026.

Topics

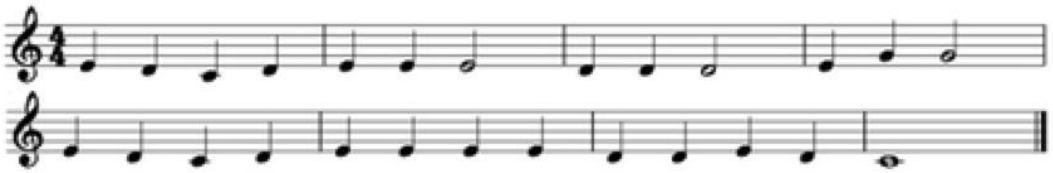
- Introduction to features
- The jSymbolic software
- Examples of research performed with jSymbolic
  - *Sidebar:* Avoiding encoding bias
- Hands-on jSymbolic workshop
- Wrap-up discussion

What is a “feature”?

- A piece of information that measures a characteristic of something (e.g., a piece of music) in a simple, consistent and precisely-defined way
- Represented using numbers
  - 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
- Usually extracted from pieces in their entirety
  - But can also be extracted from segments of pieces

Example: A simple feature

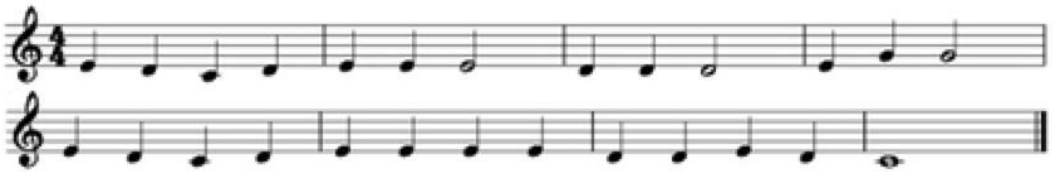
- Range (1-D): Difference in semitones between the lowest and highest pitches



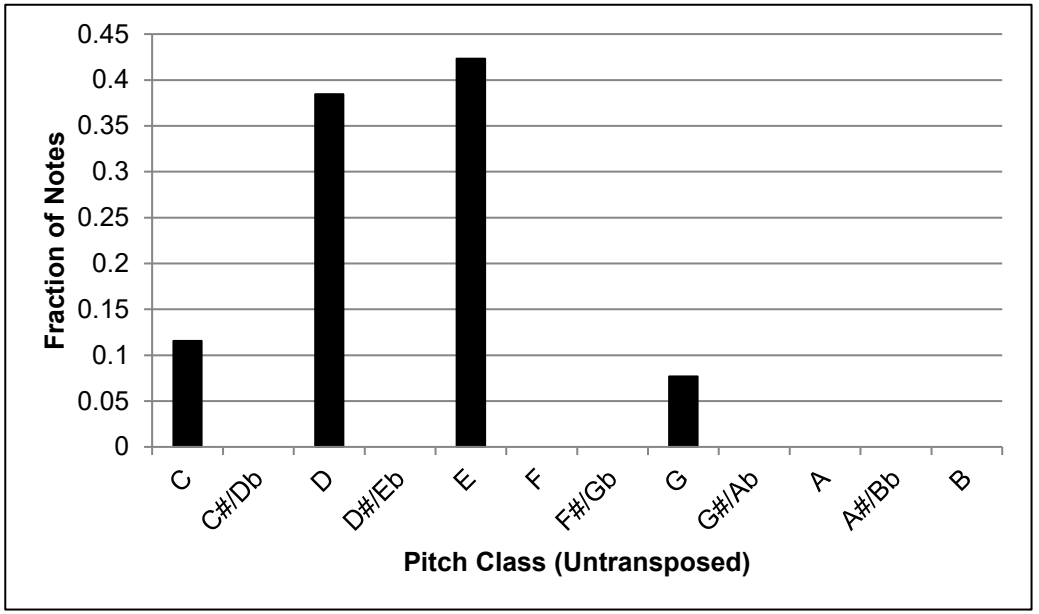
- Value of this feature: 7
  - G - C = 7 semitones

Example: A histogram feature

- Pitch Class Histogram: Consists of 12 values, each representing the fraction of all notes belonging to an enharmonic pitch class
  - e.g., all C notes are grouped together, regardless of octave

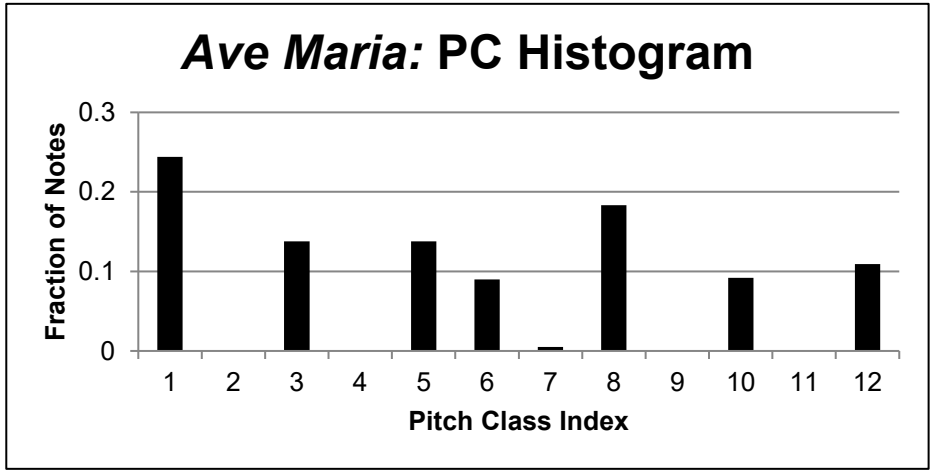


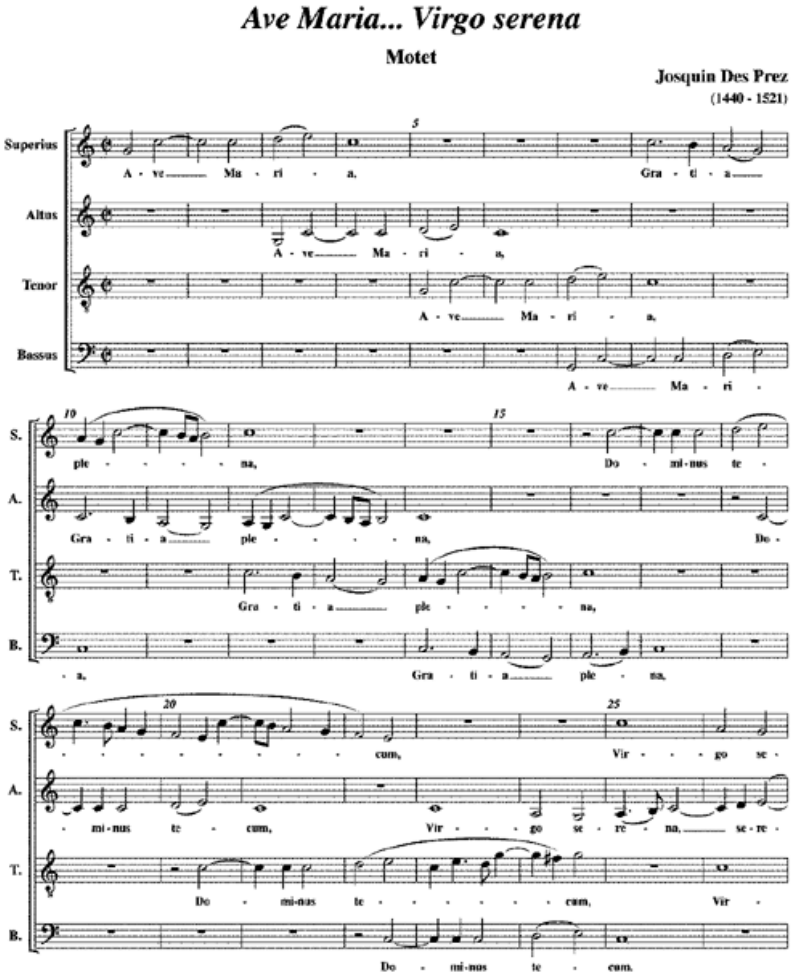
- Histogram graph on right shows feature values
- Pitch class counts:
  - C: 3, D: 10, E: 11, G: 2
- Most common note is E:
  - 11/26 notes
  - Corresponds to a feature value of 0.423 for E



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

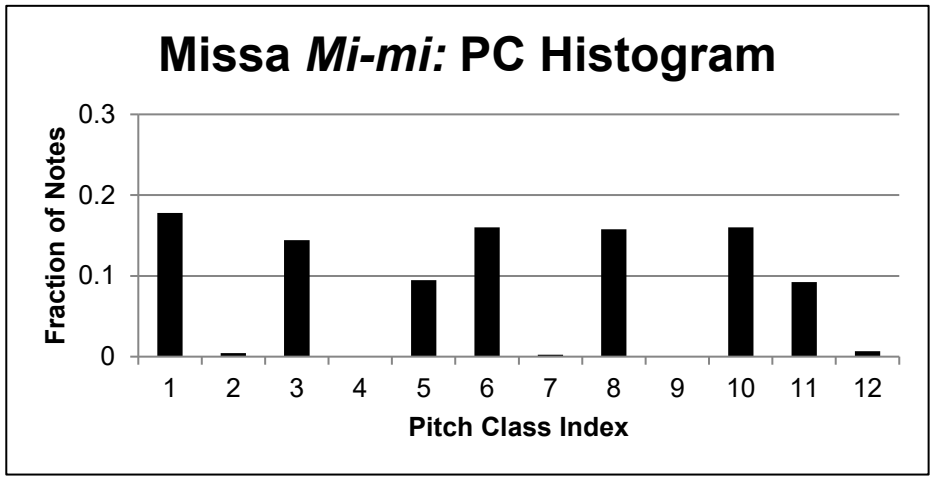
- Range: 34 (semitones)
- Repeated notes: 0.181 (18.1%)
- Vertical perfect 4^{ths}: 0.070 (7.0%)
  - Between all pairs of voices
- Rhythmic variability: 0.032
- Parallel motion: 0.039 (3.9%)

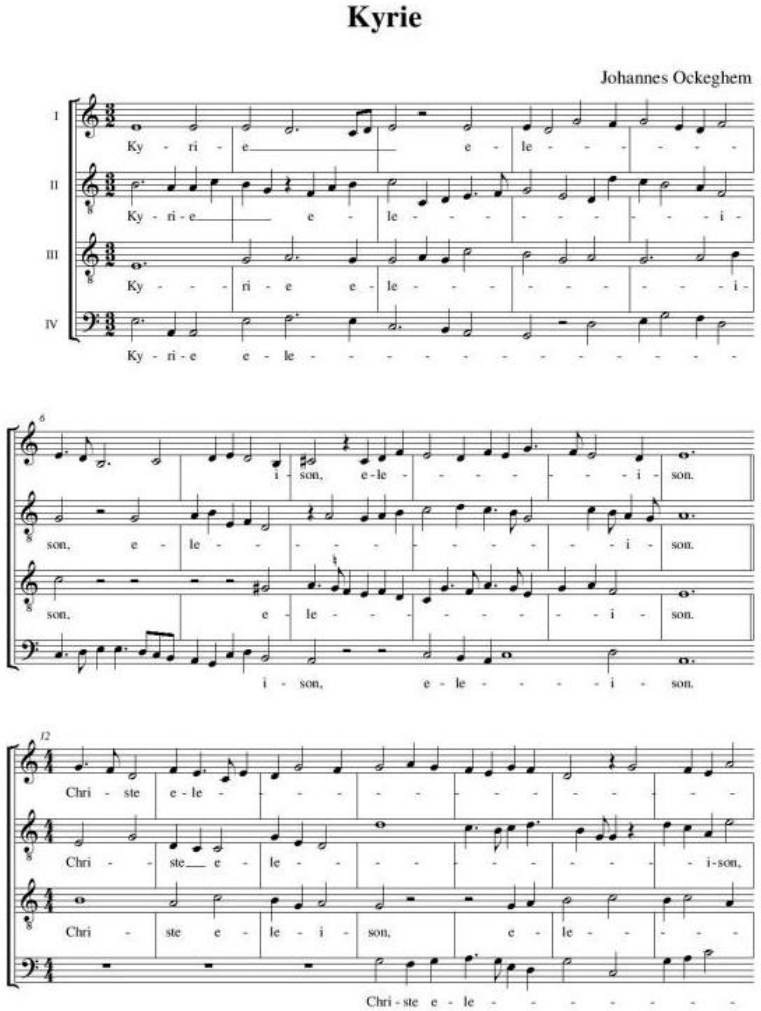




Ockeghem's Missa *Mi-mi* (Kyrie)

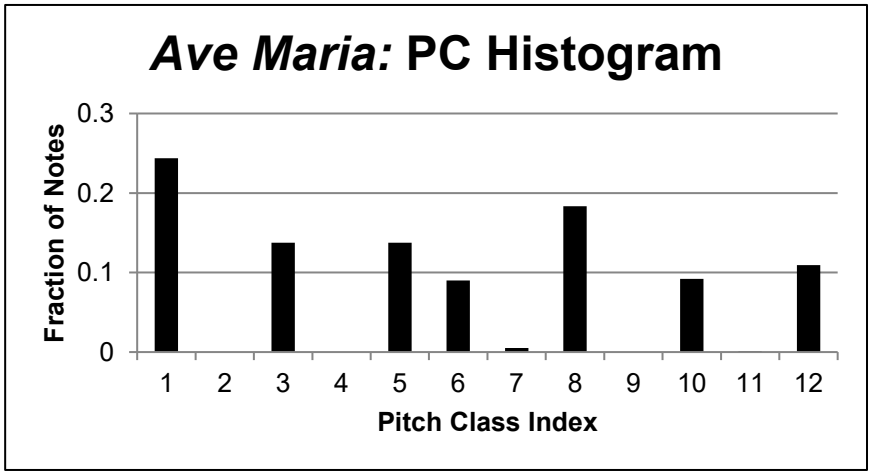
- Range: 26 (semitones)
- Repeated notes: 0.084 (8.4%)
- Vertical perfect 4^{ths}: 0.109 (10.9%)
- Rhythmic variability: 0.042
- Parallel motion: 0.076 (7.6%)

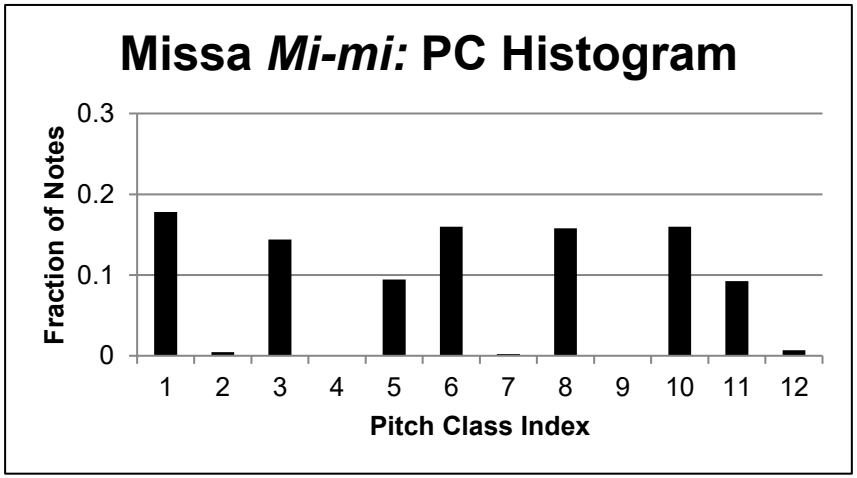




Feature value comparison

Feature	Ave Maria	Missa *Mi-mi*
Range	34	26
Repeated notes	0.181	0.084
Vertical perfect 4^{ths}	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 or contested pieces
    - Explore the stylistic origins of genres (e.g., madrigals)
    - Delineate regional styles (e.g., Iberian vs. Franco-Flemish)
    - Genre classification of popular music

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

We will NOT be using LLMs today

- The relatively simple types machine learning that I will discuss today (e.g., support vector machines, or SVMs) are quite different from more sophisticated approaches like LLMs (e.g., ChatGPT)
- The approaches we discuss are much less computationally intensive, and focus on using specialized and interpretable hand-engineered features
  - Rather than the broader, more general input to LLM-based approaches (e.g., raw piano roll representations)

Tools for examining features

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

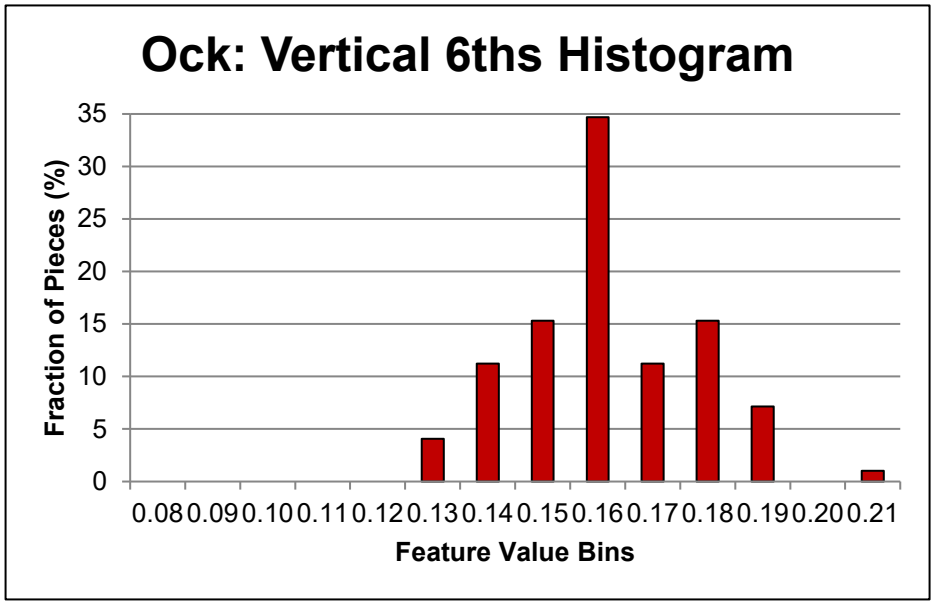
Feature visualization: Histograms (1/6)

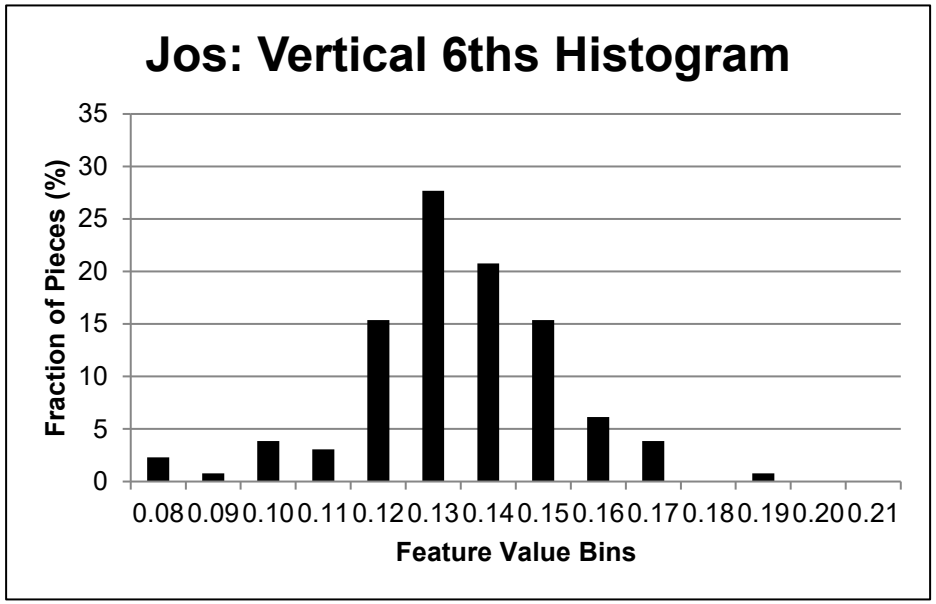
- Histograms offer a good way to visualize how the values of a feature are distributed across a corpus as a whole
  - As opposed to focusing on individual pieces
- The x-axis corresponds to a series of bins, with each corresponding to a range of values for a given feature
  - e.g., the first bin could 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 a quite different way of using histograms than the multi-dimensional histogram features discussed earlier like Pitch Class Histograms
  - Although both are equally histograms, of course

Feature visualization: Histograms (3/6)

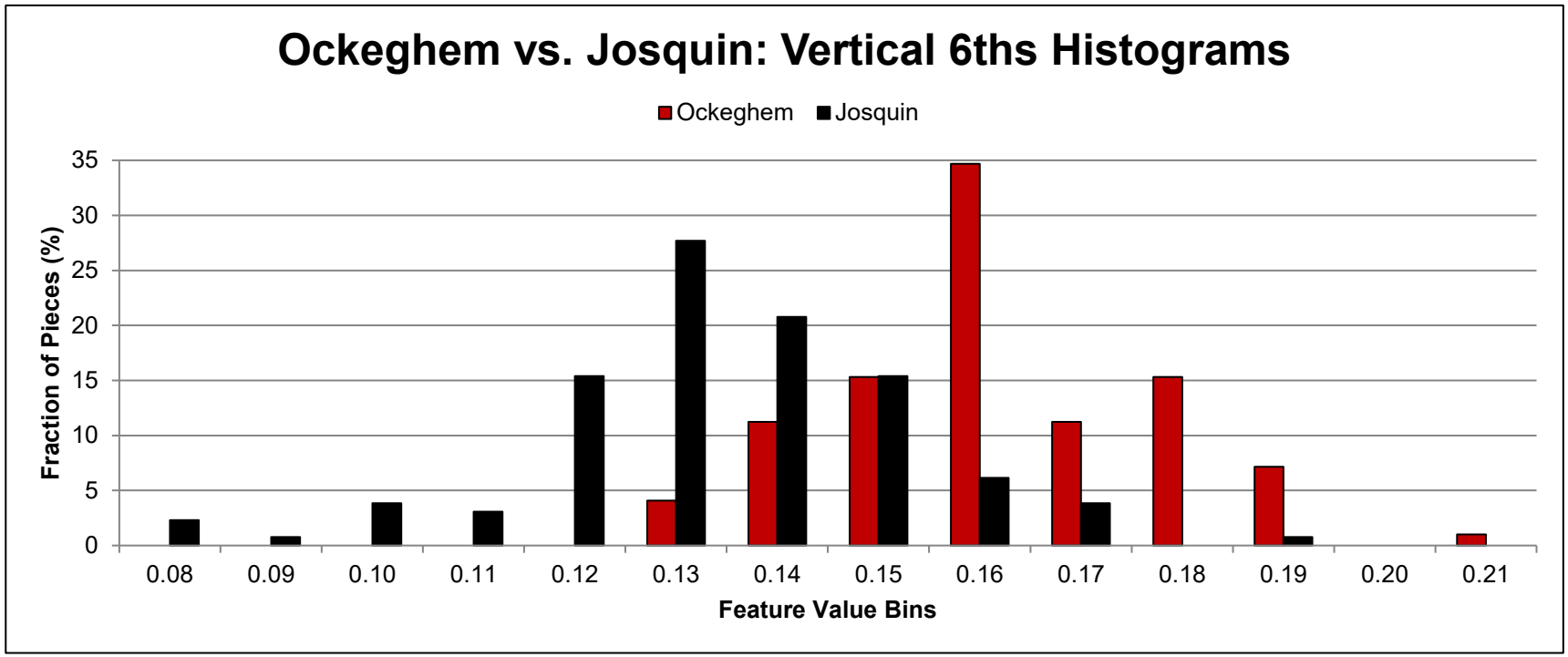




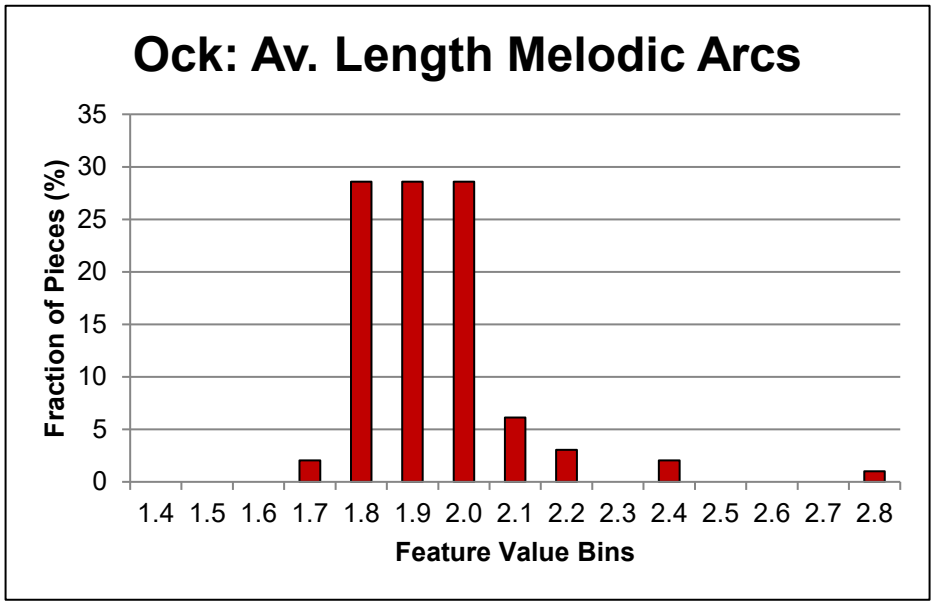
- These histograms show that Ockeghem tends to have more vertical 6^{ths} (between all pairs of voices) than Josquin
  - Ockeghem peaks in the 0.16 to 0.17 bin, at nearly 35%
  - Josquin peaks in the 0.13 to 0.14 bin, at about 28%
- Of course, there are also clearly many exceptions
  - This feature is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

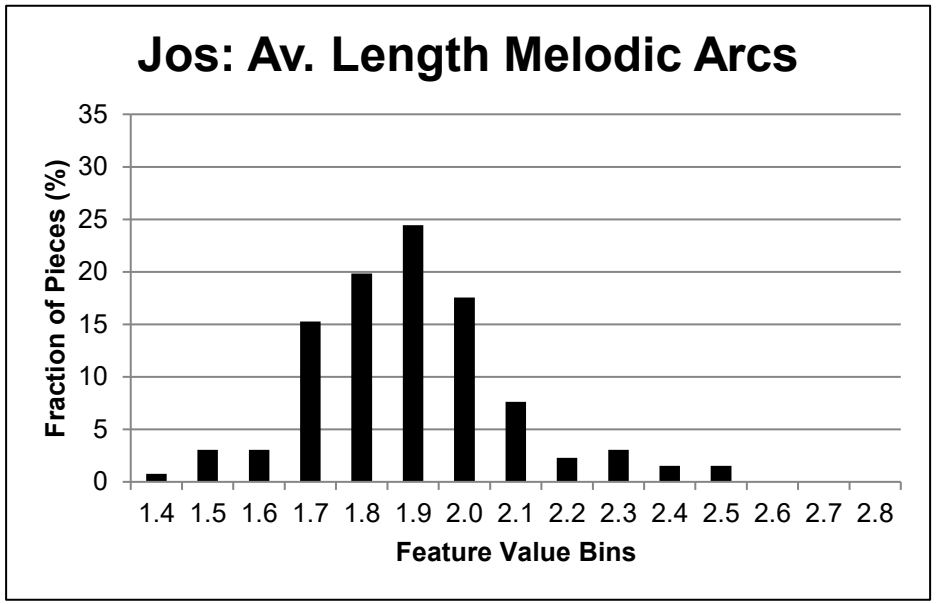
Feature visualization: Histograms (4/6)

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



Feature visualization: Histograms (5/6)

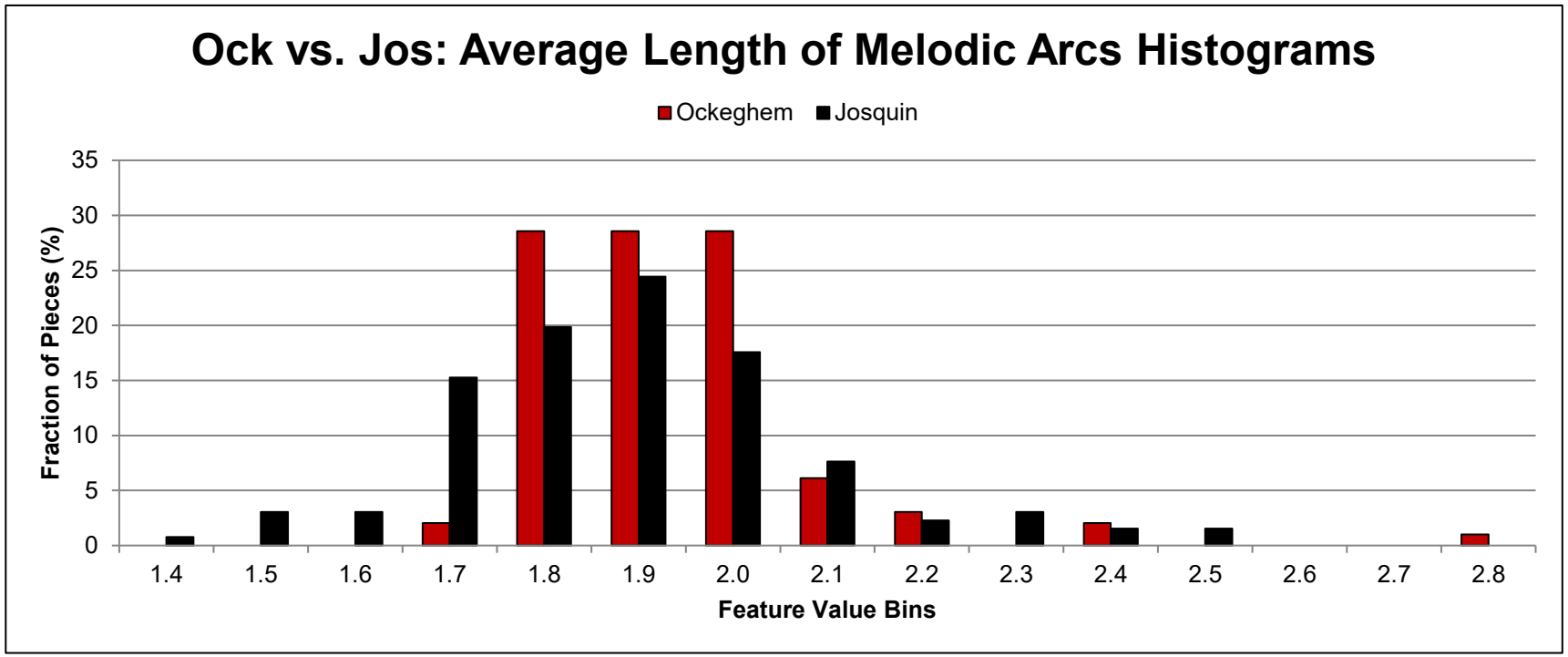




- These histograms show that Ockeghem tends to have longer melodic arcs (average number of notes separating peaks & troughs, regardless of note value)
  - Both peak in the 1.9 to 2.0 bin
  - However, Josquin's histogram is (slightly) more skewed to the far left
- Of course, there are once again clearly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

Feature visualization: Histograms (6/6)

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

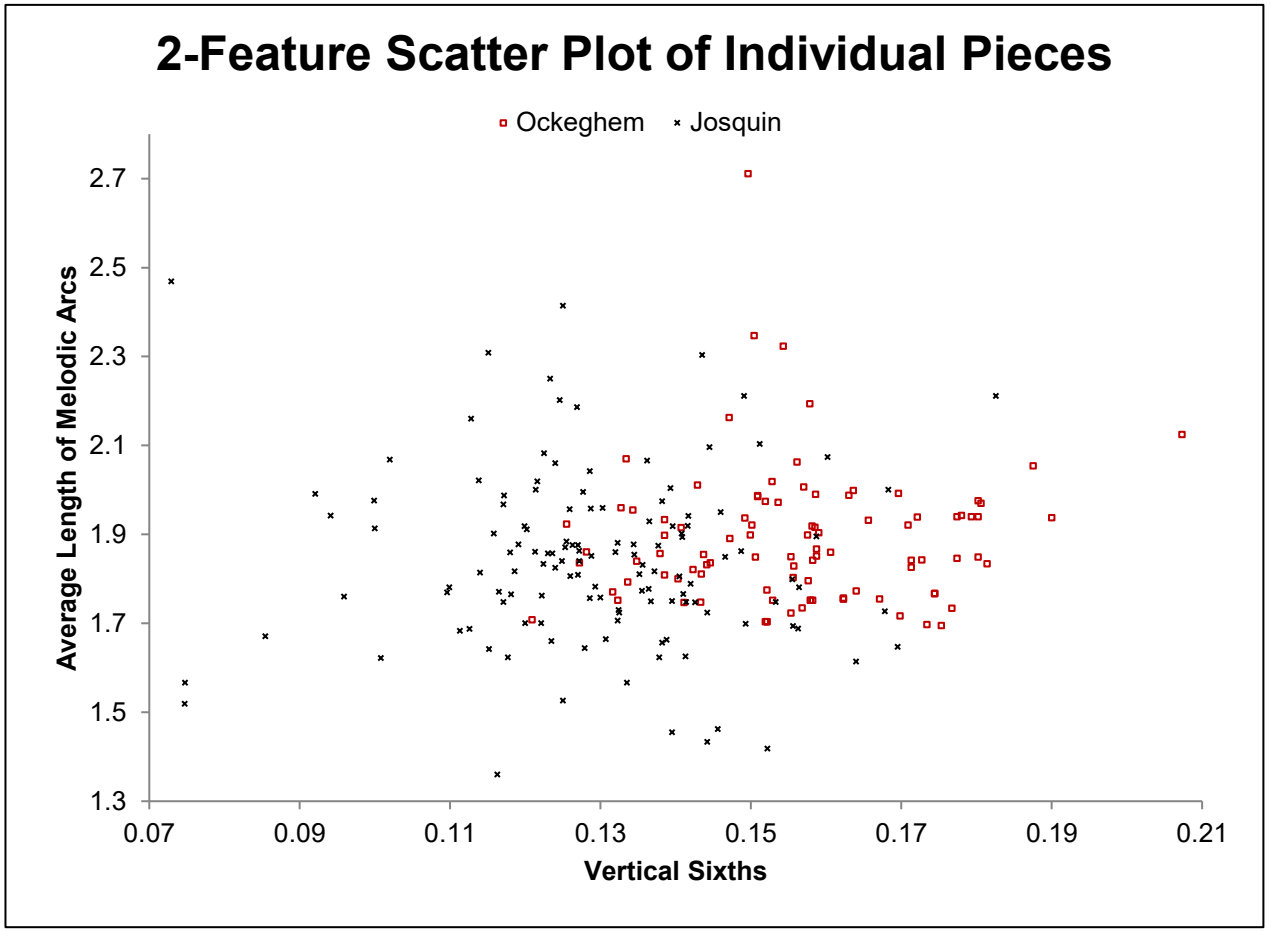


Feature visualization: Scatter plots (1/6)

- Scatter plots offer another 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)

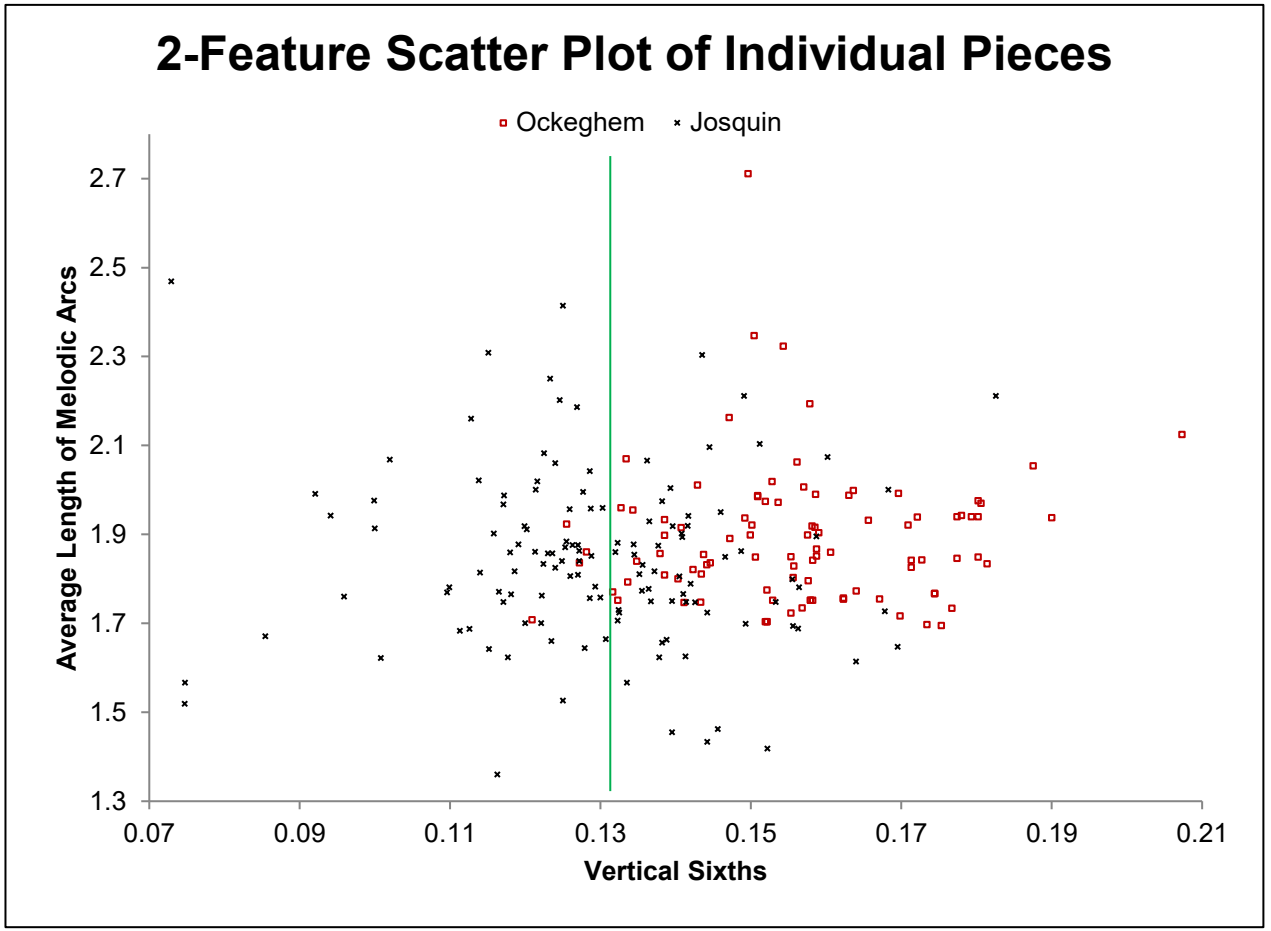
Feature visualization: Scatter plots (2/6)

- Josquin (black crosses) pieces tend to be left and low on this graph than Ockeghem pieces (red squares)



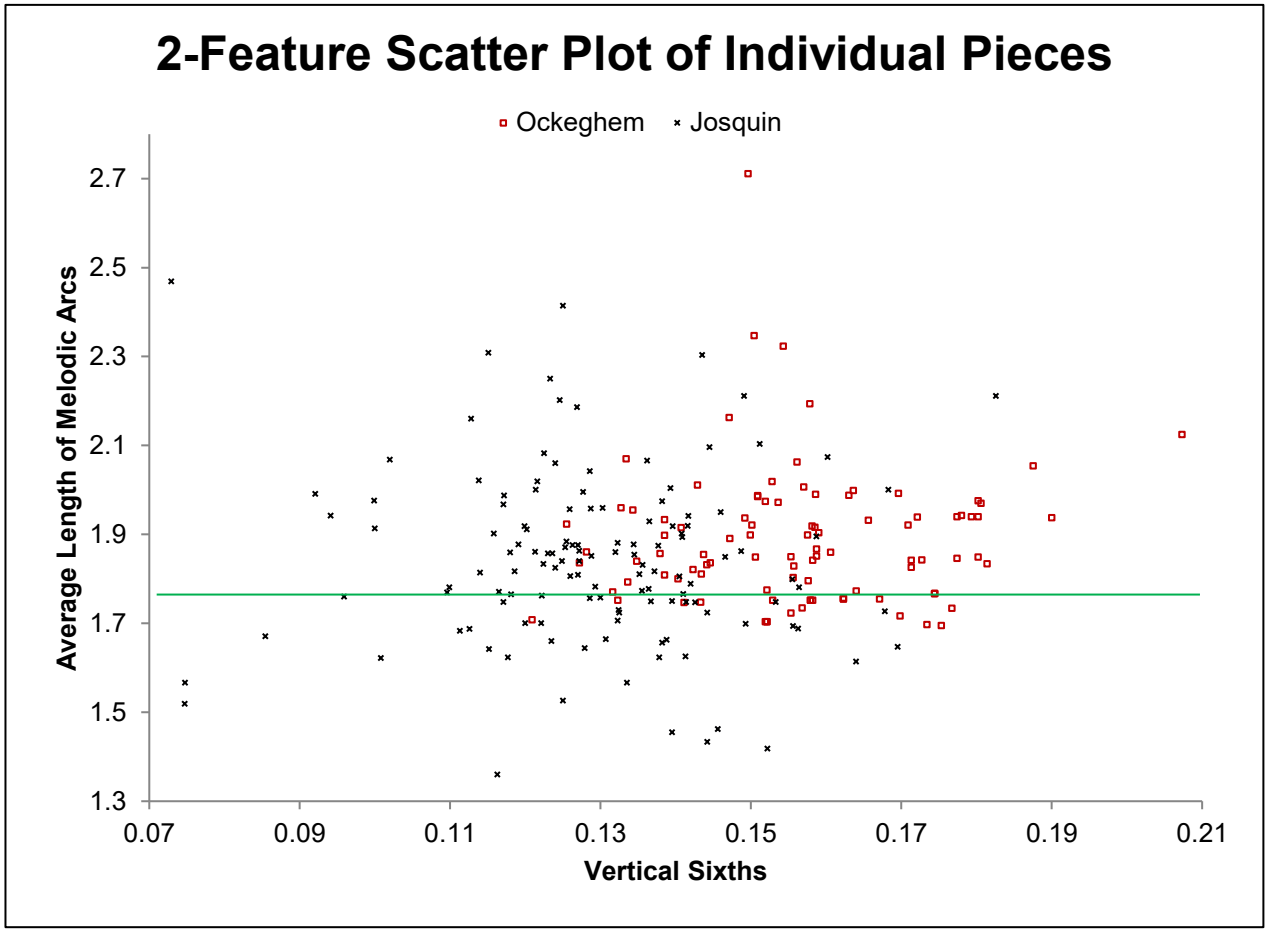
Feature visualization: Scatter plots (3/6)

- 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



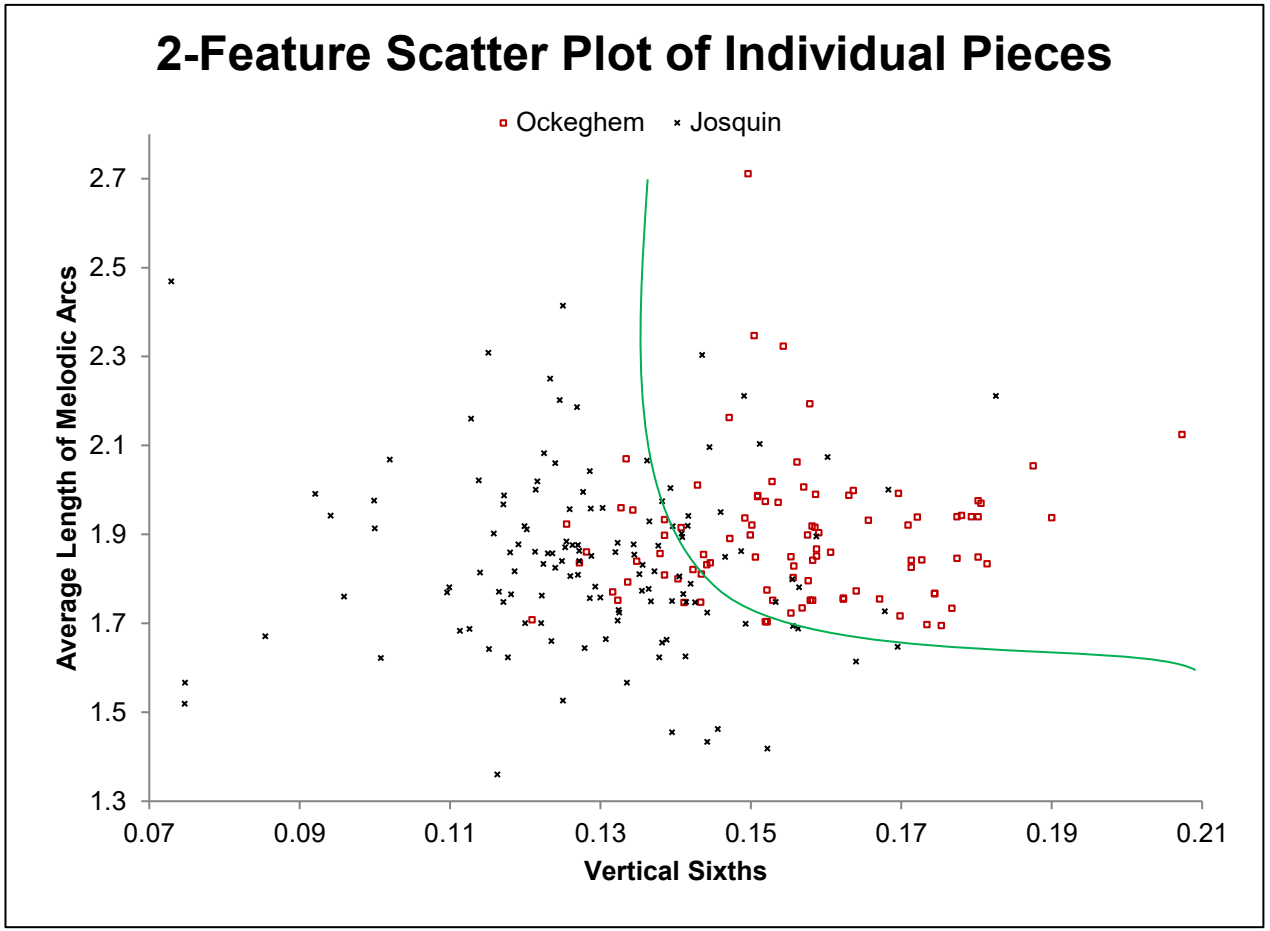
Feature visualization: Scatter plots (4/6)

- 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%)



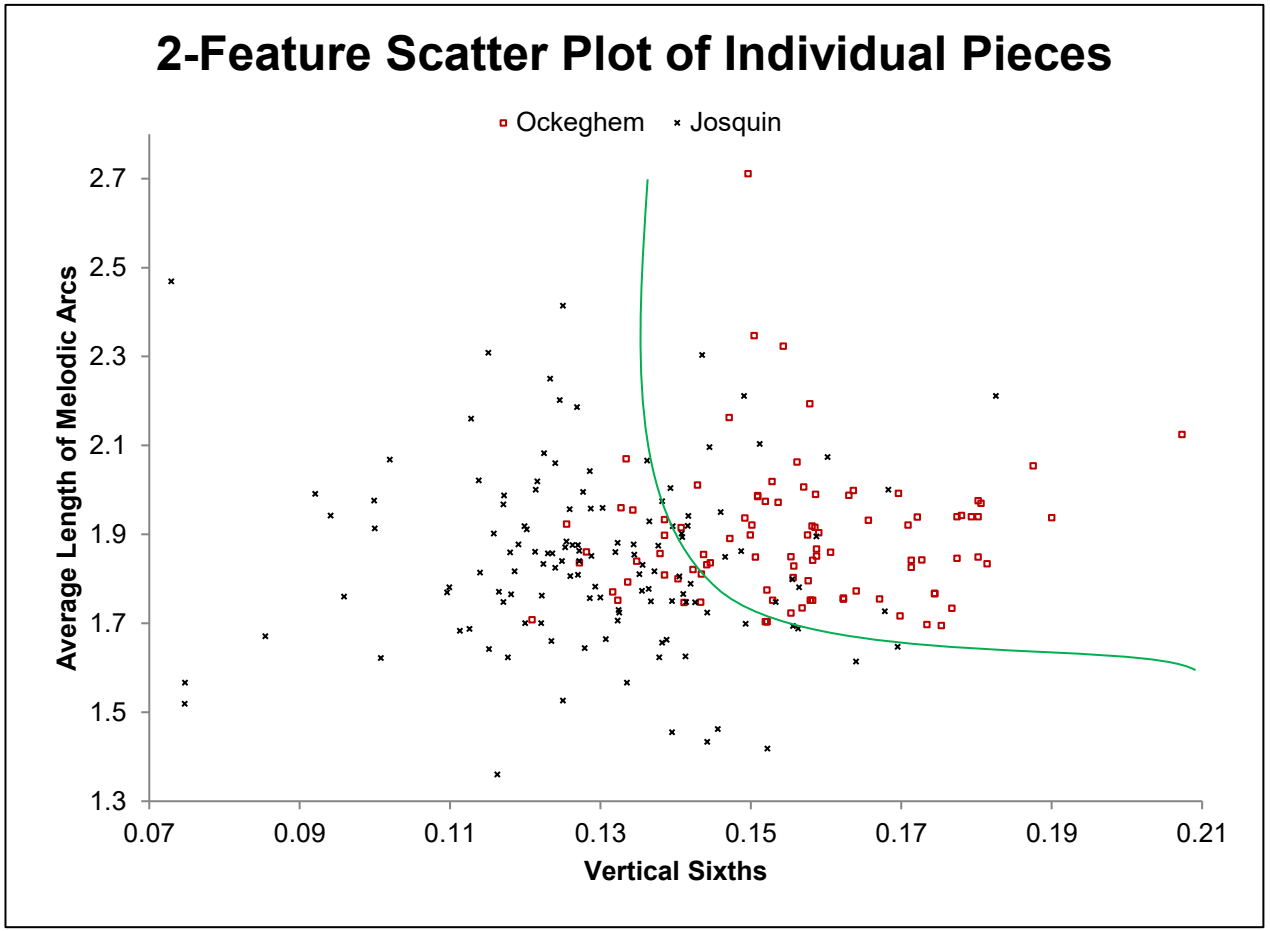
Feature visualization: Scatter plots (5/6)

- 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 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 than just two features



Benefits of features

- Provide an empirical basis for manual comparison by experts, machine learning and statistical analysis
- Permit fast studies involving huge quantities of music (thousands of pieces or more!)
- Can simultaneously explore a broad range of musical characteristics (thousands or more!) and their interrelationships
  - Including characteristics one might not have considered
  - Can statistically condense many features into more interpretable low-dimensional spaces when needed
- Can be applied to diverse types of music in consistent ways
- No need to formally specify specific queries or heuristics before beginning analyses,
  - But may do so if one wishes to, of course
  - Facilitates exploratory research
- Help to avoid or detect potentially incorrect ingrained assumptions and biases
  - But only if treated properly

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
  - Features facilitate challenging the canon and established assumptions by casting a wider net

But . . .

- 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

jMIR

Computers need us!

- A feature-based approach is useless without:
  - Human experts to ask important questions
  - Human experts to interpret results
  - 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 and potential bias

- But does a 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?

Choosing features to implement

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

jSymbolic: Introduction

- jSymbolic is a software platform I created 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 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 are used?
- 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)

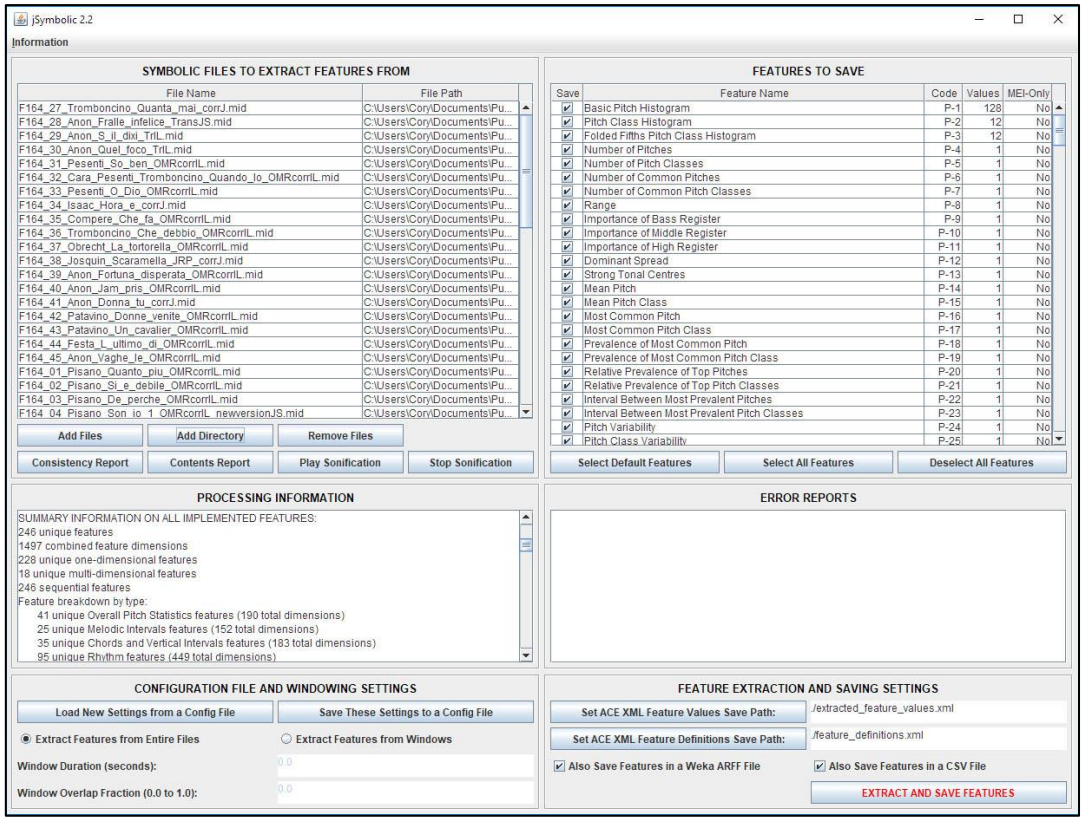
- jSymbolic only (for now) extracts features associated with musical content
- There are thus no features associated with:
  - Text
  - Historical evidence
- This is partly a disadvantage:
  - Obviously these kinds of information can be essential
  - Researchers using jSymbolic features must of course use their expertise to consider extracted features in the larger context
- It is also an advantage, however:
  - It allows us to (temporarily) focus only on the music, so that we can find unexpected insights 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., CRIM
  - e.g., music21 (includes a port of the original jSymbolic features)
  - e.g., Humdrum
  - e.g., VIS
- This other software is excellent for finding where and how specific things one is searching for happen
  - Focuses on local rather than macro musical content
  - Perfect for 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 statistical analysis and machine learning
  - Good for free exploratory research
  - Good for large-scale validation of theoretical models
  - Good for general annotation of symbolic databases

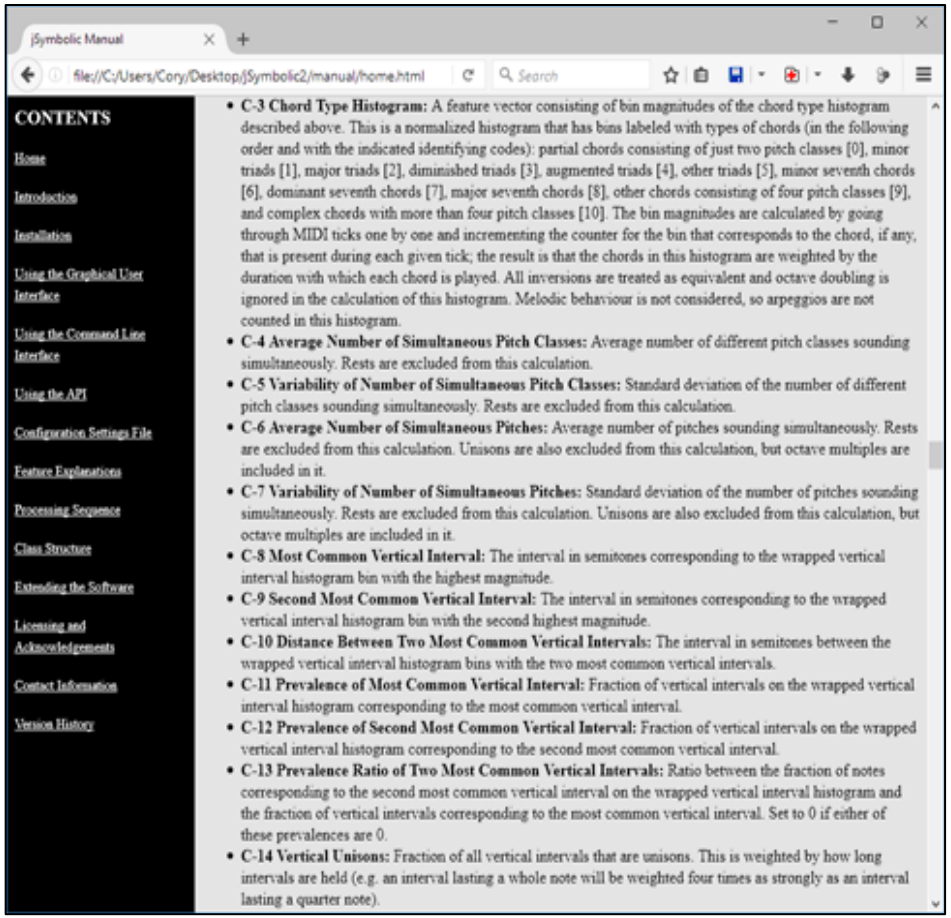
jSymbolic: User interfaces

- 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: File formats

- Input:
  - MIDI
  - MEI (or at least earlier vanilla versions)
  - MusicXML (after conversion to MIDI)
- 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
- MIDI has serious problems for music analysis:
  - e.g., Cannot distinguish enharmonic equivalents
    - Pitch is encoded as 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 music 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 (including dynamics)
- 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 specialized parser for
- MIDI can be easily and directly sonified
  - Almost all symbolic alternatives must be first converted to MIDI to be listened to
- MIDI largely 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 can be problematic when performing automatic analysis

jSymbolic: Miscellany

- Windowed feature extraction
  - i.e. automatically breaking the music into small successive segments from which features are extracted individually
  - Including overlapping windows
- Configuration files
  - Pre-set feature choices
  - Pre-set input and output choices
  - More
- Can combine jSymbolic with other jMIR components to perform multimodal research
  - e.g., combine symbolic features with other features extracted from audio, lyrics and cultural data
  - This can improve results substantially!
    - Vatolkin and McKay 2022; 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

Important software principles

- As Frans Wiering has wisely pointed out a number of times, those of us who produce research software must be careful to give musicologists what they want and need
  - Rather than trying to impose choices on them
- This emphasizes the importance of establishing an on-going dialog
  - Software designers should find out from musicologists what will be valuable to them
  - Software designers can also present musicologists with the possibility of options that they would not necessarily have though of, or thought possible
- So, please let me know what you need or want!

To come in jSymbolic 3.0

- Many miscellaneous usability improvements
  - Including expanded multilingual support
- Many new features
  - 533 unique features and 2040 features, as of June 2026
  - Including features based on note onset slices
  - Including features based on n-grams

Research involving jSymbolic

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

Composer identification study

- Related papers: MedRen 2017, ISMIR 2018
- 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

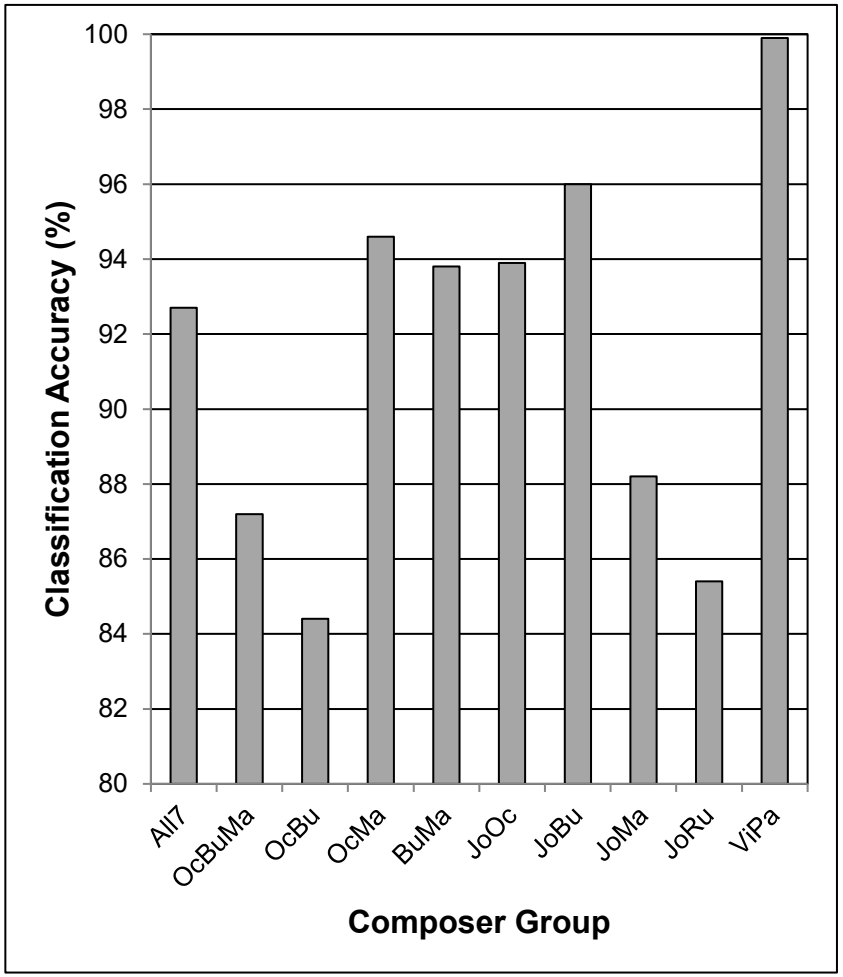
Composer	Files
Palestrina	705
Victoria	261

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 (1/3)

- 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

A priori expectations (2/3)

- What do you think might distinguish the composers?
  - Josquin vs. Ockeghem?
  - Josquin vs. La Rue?
- I consulted one musicologist (Julie E. Cumming) and one theorist (Peter Schubert), both experts in the period . . .

A priori expectations (3/3)

- 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
  - Ockeghem 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
    - e.g., features measuring dynamics, instrumentation and tempo
  - Some excluded features were correlated with the source of the data

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:
  - Cumming, J., C. McKay, J. Stuchbery, and I. Fujinaga. 2018. Methodologies for creating symbolic corpora of Western music before 1600. *Proceedings of the International Society for Music Information Retrieval Conference*. 491–8.
- 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)

- We also carried out another composer-related study using the Josquin Research Project data
  - This one investigated the attribution of pieces suspected to be by Josquin

- Related publications: ISMIR 2017, MedRen 2024
- Upcoming expanded version: Chapter in upcoming book *Josquin: A New Approach*
  - eds. C. Bokulich, J. Rodin, E. Zazulia

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 presentation
- Upcoming expanded version: MedRen 2026 joint session in July

- 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 did we investigate this?
  - Constructed the "3RenGenres" corpus: MIDI files derived from Florence BNC 164-167 (c. 1520)
    - Madrigals (27 files)
    - Motets (12 files)
    - Frottole & Villotte (19 files)
  - Extracted jSymbolic 2.2 features
  - Applied machine learning and feature analysis techniques

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

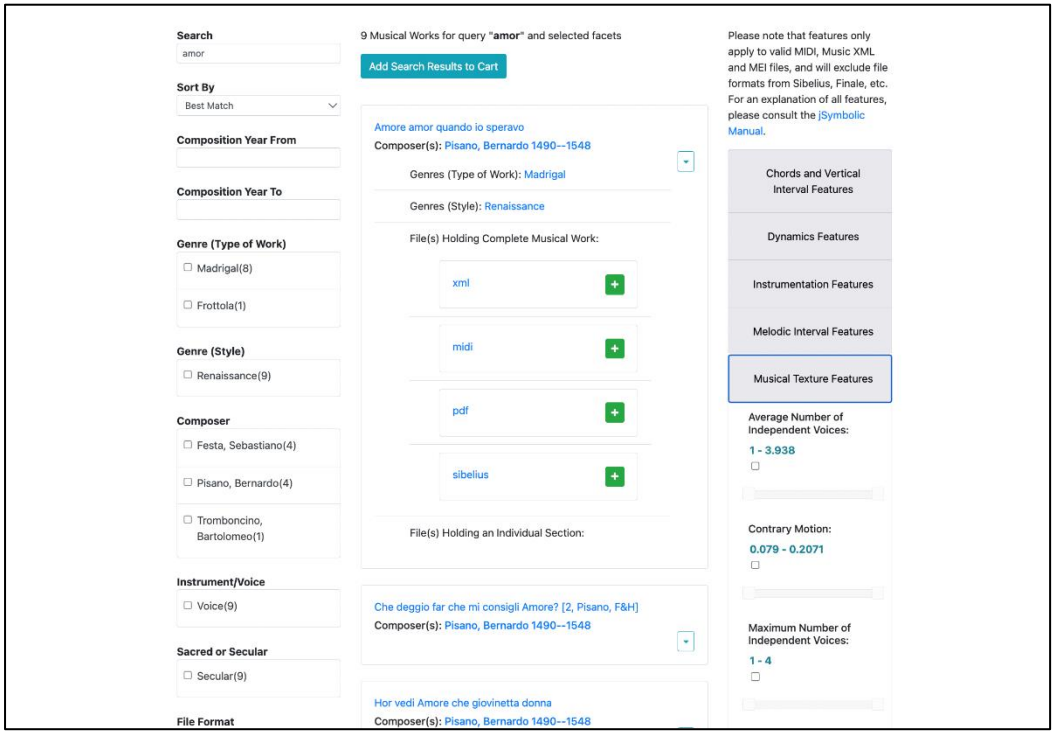
Genre Group	Classification Accuracy
Frottole / Villotte vs. Madrigals	64.6%
Frottole / Villotte vs. Motets	84.8%
Madrigals vs. Motets	99.1%

Origins of the Italian madrigal (3/3)

- Expert *a priori* meaningful feature 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
- Opened very promising avenues for future research

SIMSSA DB

- SIMSSA DB is a collaborative database prototype infrastructure for holding and accessing symbolic music files, associated auto-extracted content-based feature values, and musicologically-focused metadata
- Content-based queries can be performed based on feature values
  - As well as standard metadata-based queries



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
- Genre classification of popular music
  - McKay et al., 2018
- Comparing La Rue and Josquin
  - Cumming and McKay, 2018
- Was Iberian Renaissance music stylistically distinct from Franco-Flemish music of the time?
  - McKay, 2018
- Influences on the masses and motets of Cristóbal de Morales and Francisco Guerrero
  - Cuenca and McKay, 2021
- Attribution of Iberian motets
  - Rodriguez-Garcia and McKay, 2021
- Exploring anonymous and doubtfully attributed Coimbra masses
  - Cuenca and McKay, 2021
- Attribution of *Ave festiva ferculis*
  - Rodriguez-Garcia and McKay, 2021
- Influences on the masses of Pedro Fernández Buch
  - Cuenca and McKay, 2022
- Stylistic origins of 16th century masses transcribed by Siro Cisilino
  - Cuenca and McKay, 2023

Accessing jSymbolic 2.2

- jSymbolic 2.2 home page:
  - https://jmir.sourceforge.net/index_jSymbolic.html
- jSymbolic 2.2 tutorial:
  - https://jmir.sourceforge.net/manuals/jSymbolic_tutorial/home.html
- jSymbolic 2.2 manual:
  - https://jmir.sourceforge.net/manuals/jSymbolic_manual/home.html
- jSymbolic 2.2 download:
  - https://sourceforge.net/projects/jmir/files/jSymbolic/

- These slides are available via this session's Google Doc linked to on the Institute's home document

Acknowledgements

- Thanks to my colleagues and the students involved in the LinkedMusic, SIMSSA and MIRAI projects, especially:
  - *Colleagues:* Julie Cumming + Ichiro Fujinaga
  - *Student RAs:* Tristano Tenaglia + Rían Adamian + Gustavo Polins Pedro + Rebecca Mizrahi + Hong Van Pham
- 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 2.2 Afternoon session

- 13:30 to 15:30 Hands-on workshop
  - Practice extracting features (using jSymbolic) and studying music with them (using Weka)
    - Using provided musical datasets
  - Instructions available here:
    - https://jmir.sourceforge.net/manuals/jSymbolic_tutorial/home.html
  - Work alone or in groups, as you prefer
    - If you have not already installed Java, jSymbolic and Weka on your computer, then you may wish to work with someone who has
  - Please feel free to take a coffee break if and when you need it
- 15:30 to 16:00 Wrap-up
  - Extended time for hands-on workshop (if needed)
  - Open discussion
    - What kinds of research could you imagine using features to conduct?
    - What kinds of corpus would you need for this research?
    - What would you like to know more about?

Thanks for taking part!

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