z

Pattern Recognition and Visualization of Pre-Classical Monophonic and Polyphonic Music

Aaron Carter-Énylì, PhD, CAGO
Department of Music, *Morehouse College*, aaron.carterenyi@morehouse.edu

About EMViz

Currently implemented in MATLAB and deployed as a standalone executable (using MATLAB Runtime), *EMViz* is now available to the members of SMT's Early Music Analysis Interest Group. The project is funded by the American Council of Learned Societies (ACLS).

EMViz (Early Music Visualization) provides built-in pattern recognition for symbolic music based on a contour recursion algorithm by Carter-Enyi (2016) producing visualizations of musical form using arc diagrams, as proposed by Wattenberg (2002). The algorithm brings together contour theory (Morris 1987, Quinn 1996, Schultz 2013) with studies of melodic accent (Thomassen 1982, Huron 2006). Symbolic music data (.midi, .xml) from various sources (including ELVIS at McGill and the Yale Classical Archives Corpus) may be imported, analyzed and visualized in a matter of minutes. Arc diagram visualizations in the supplemental materials include music from the *Liber Usualis*, Josquin des Prez and J. S. Bach.

For more information on the contour algorithm:

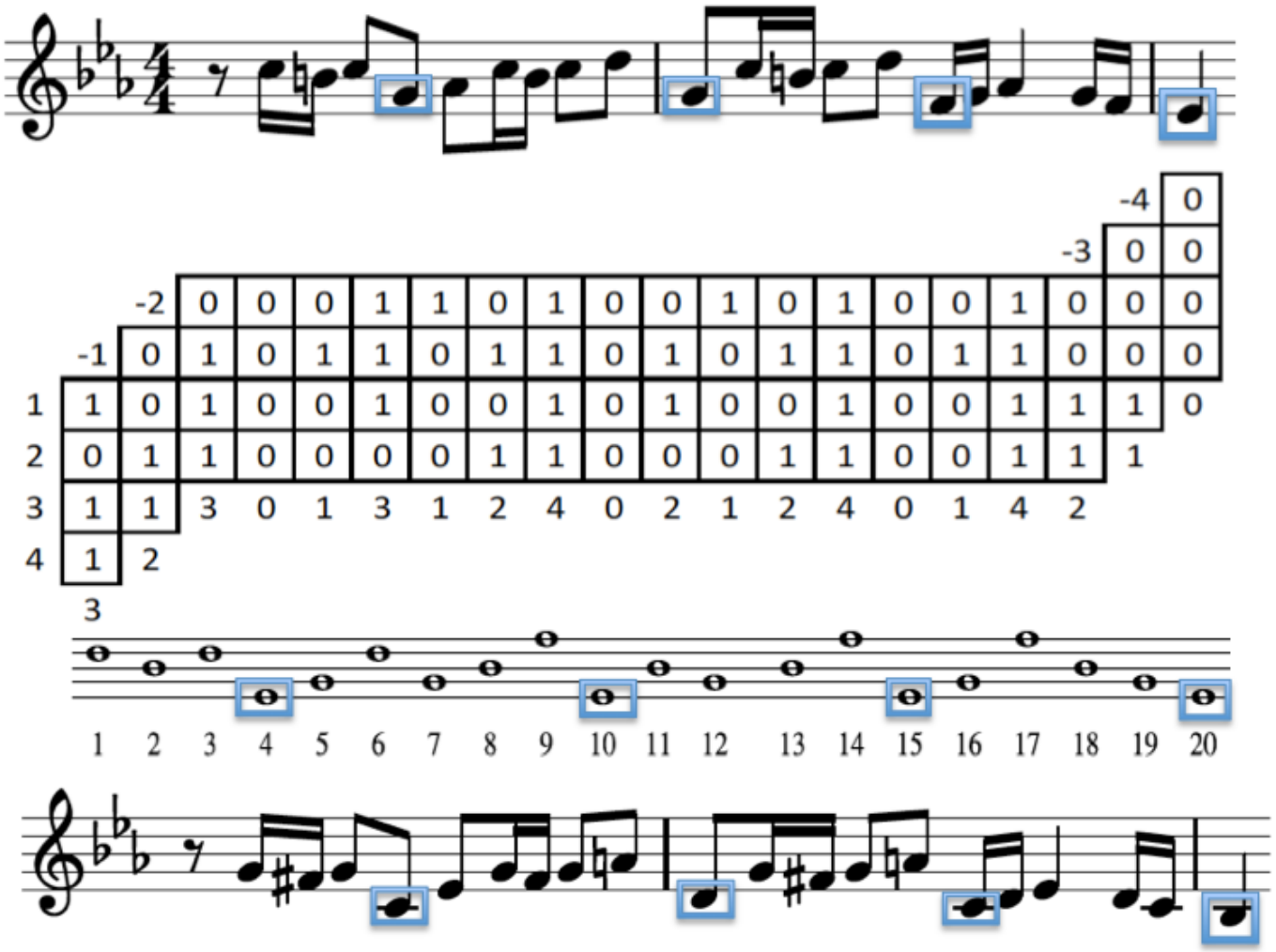
- Carter-Ényì, A. (2016). Contour Recursion and Auto-Segmentation. *Music Theory Online, 22*(1).

For more information on the arc diagram visualization:

- Wattenberg, M. (2002). *Arc diagrams: Visualizing structure in strings.* Paper presented at the Information Visualization, 2002. INFOVIS 2002. IEEE Symposium on.

How it works:

The contour algorithm behind EMViz uses local contour comparisons within a 2-degree radius of each focus pitch. This makes use of Quinn's (1997) binary C+ comparison, where 0 is equal or below and 1 is above the reference pitch. In the examples below, the subject and tonal answer of Bach's C-minor Fugue (BWV 847) are shown to be equivalent strings using this method (they both produce the same "continuous" matrix).



How to install EMViz:

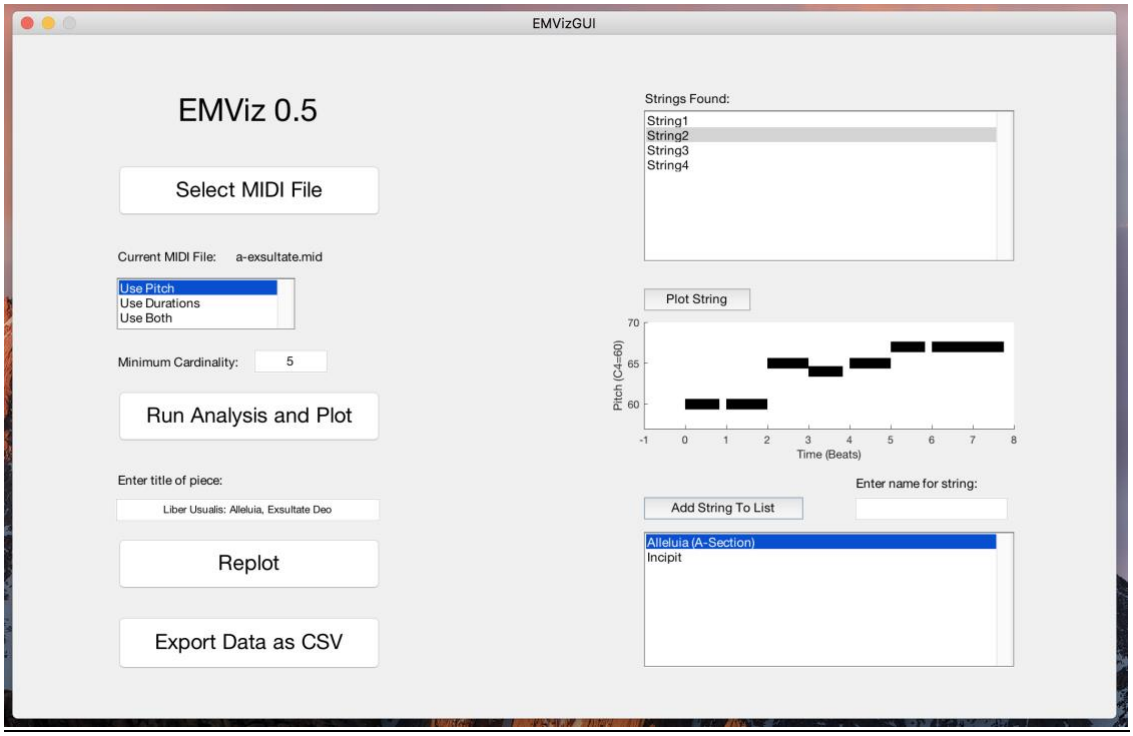
Mac OSX version:
https://morehousecollege.box.com/s/3ziy19h8n36l54oza5fcesx2f9z7y2yg

Windows version:
https://morehousecollege.box.com/s/cd2t5kb5xt4e3f7zsd0zcnbat2e9b1vw

Detailed instructions for installation:

1. *You will need a web connection for the entire installation* because MATLAB Runtime (also free) will also be downloaded and installed when you run the application installer
2. Download the entire folder at the appropriate link above
3. Find the "AppInstaller" folder or zip file and double-click it, then double-click "MyApplicationInstall_web" (.exe for Windows and .app for Mac); Mac will unzip revealing a "test" folder (which has the MyApplicationInstall_web.app inside); Windows may also ask you to expand the files before installing (if so, you will need to double-click the installer again after this unpacking)
4. *Because this software is not from an "App Store", you will likely need to override some security preferences after expanding/unzipping and clicking on the .app or .exe, to do this right-click or control-click and select "Open" on Mac or "Run as administrator" on Windows*
5. The actual installation process (which requires an internet connection) may take 5 – 20 minutes depending on the download speed of your internet connection (it is downloading MATLAB Runtime so your computer can interpret my code for the algorithm and GUI)

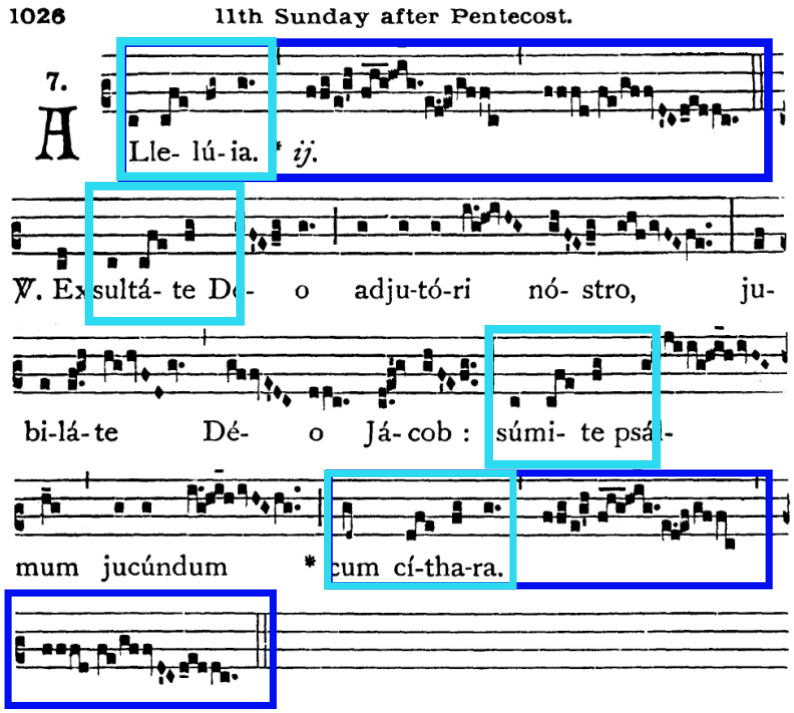
How to use EMViz:



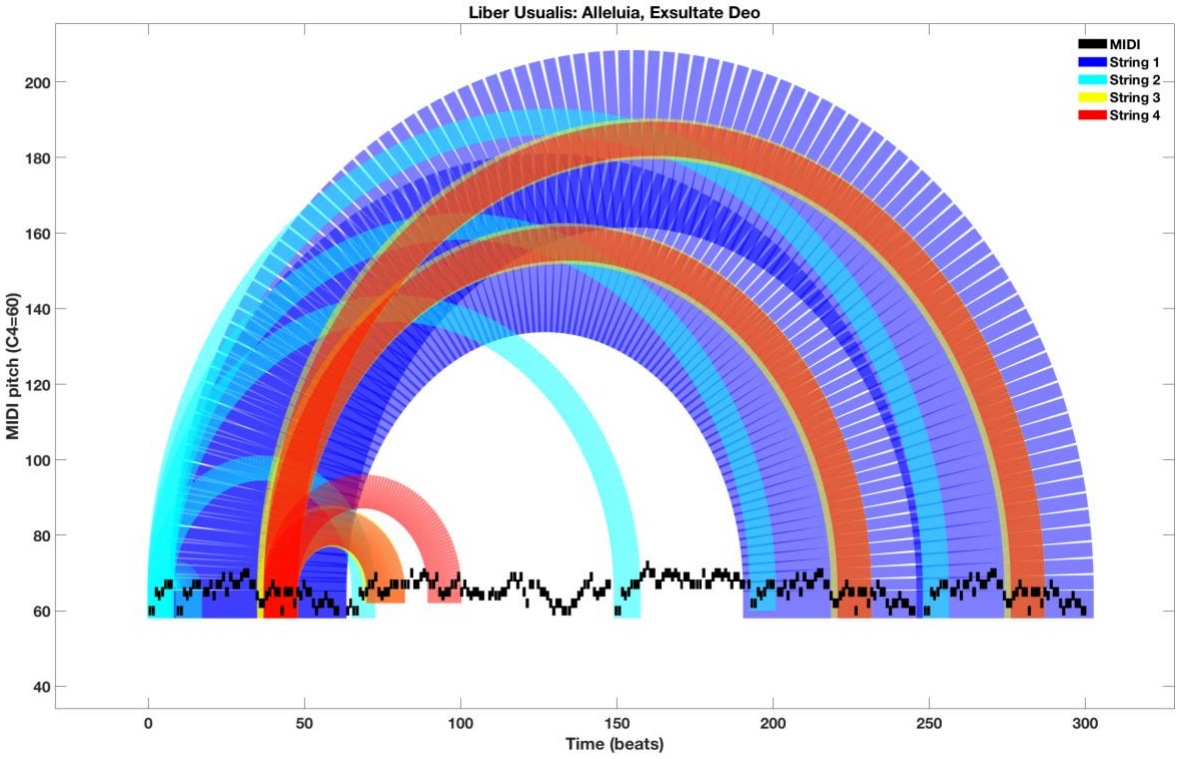
1. **Select MIDI File (Button)**: opens a finder window to select a MIDI file, you can search for ".mid" or for file type "MIDI Audio", or navigate to sample MIDI provided with the application
2. **Current MIDI File (Display Text)**: will display the filename for the MIDI file you have selected, in the above example, I selected "a-exsultate.mid" (provided in the MIDI folder)
3. **Minimum Cardinality (Text Entry Box)**: allows you to select the shortest string length you are interested in. Generally, five notes is a good minimum because three or four-note contours are too generic. If you know you are interested in longer strings (on the order of 10 or 20 events), then set a higher minimum for results closer to your analytical goals.
4. **Use Pitch; Use Durations; Use Both (Selection Box)**: allows you to select what type of pattern-recognition algorithm to run: (1) *Pitch* only makes binary contour comparisons of pitch height within a local window of two degrees of adjacency; (2) *Durations* only makes binary comparisons of note duration within a local window of two degrees of adjacency; (3) *Both* combines pitch contour and durational contour to find patterns.
5. **Run Analysis and Plot** (Button): triggers the algorithm you selected (Pitch, Durations, or Both) to run on the file listed as the **Current MIDI File**. This takes 20 – 60 seconds depending on the size of the file.
6. **Enter Title of Piece (Text Entry Box)**: allows you to name your diagram (this can be entered before **Run Analysis and Plot** or **Replot**)
7. **Replot** (Button): enables you to selected strings in the listbox in the bottom right corner of the GUI (see 9 and 11)
8. **Export Data as CSV** (Button): exports all strings found in a table format that can be opened in Excel or any text editor.
9. **Strings Found** (Selection List Box): select from all strings found for either 10 or 11
10. **Plot String** (Button): plots and plays the string selected in 9.
11. **Add String To List (Button) / Enter name for String (Text Entry Box)**: adds the string selected in 9 to a list of user-specified strings (for 7. Replot). If no name is entered, then it will be labeled String*X*.

Examples of form visualizations by EMViz GUI: In all examples, notes appear as small rectangles at the bottom of the visualization, if multiple voices are present, they are color-coded. Iterations of repeated contour patterns are connected with arcs.

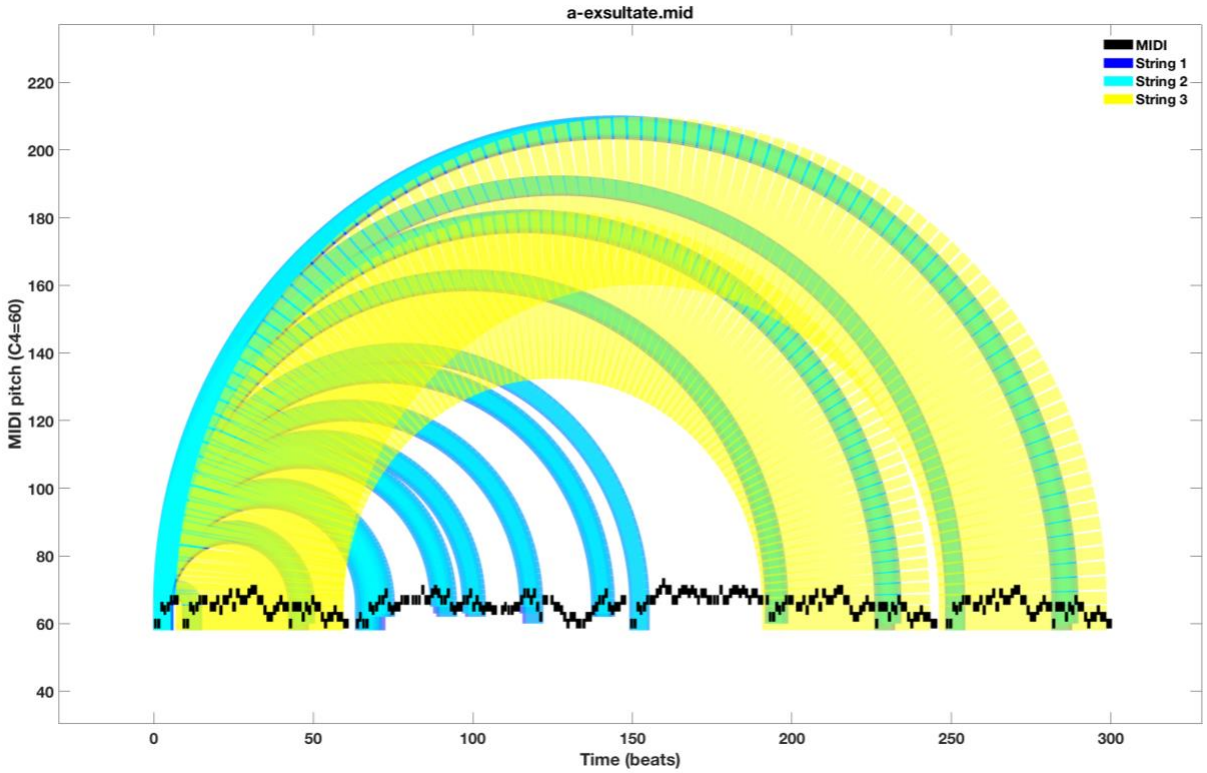
Example 1: *Liber Usualis: Alleluia – Exultate Deo*: Visualization of full performance (e.g. https://youtu.be/oa1XLOW55RE) with incipit at the beginning and repetition of "Alleluia" at the end.



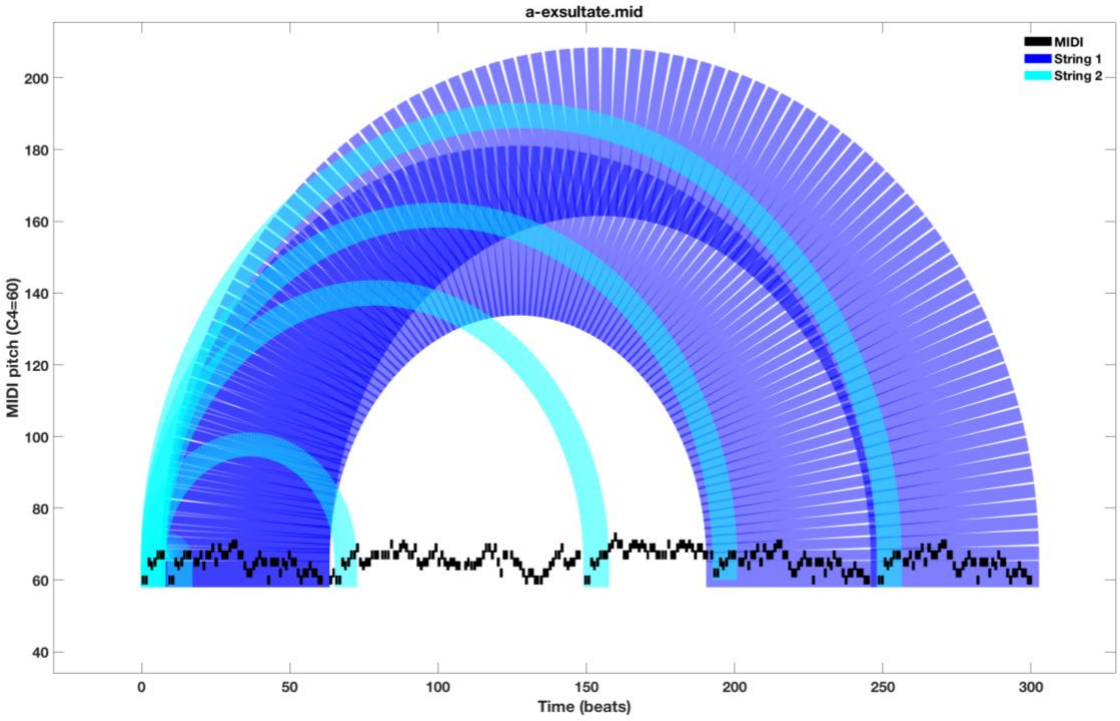
Example 1a (below): Plot of results from **Pitch** algorithm



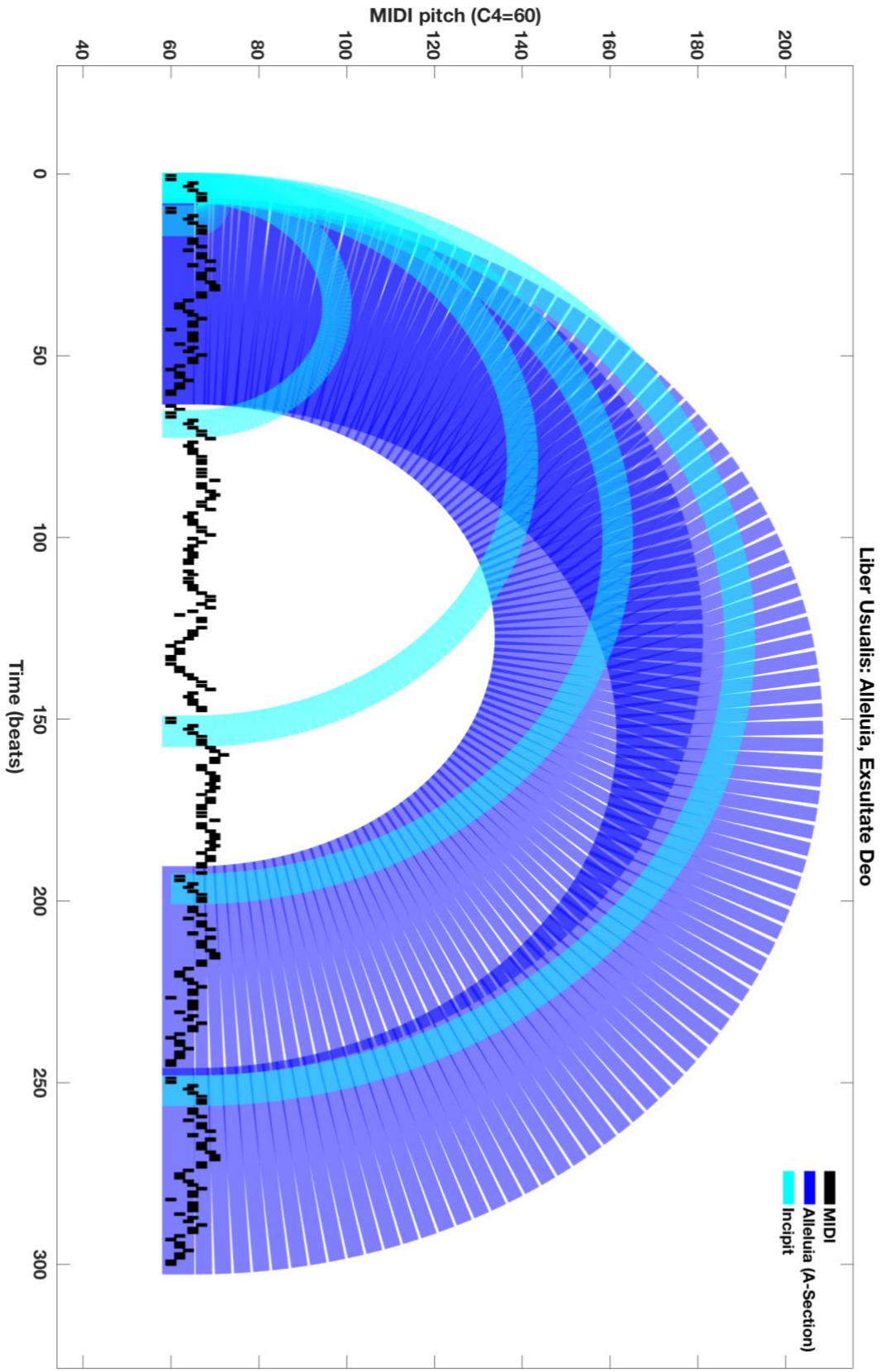
Example 1b (below): Plot of results from **Durations** algorithm



Example 1c (below): Plot of results from **Both** (Pitch and Durations) Algorithm



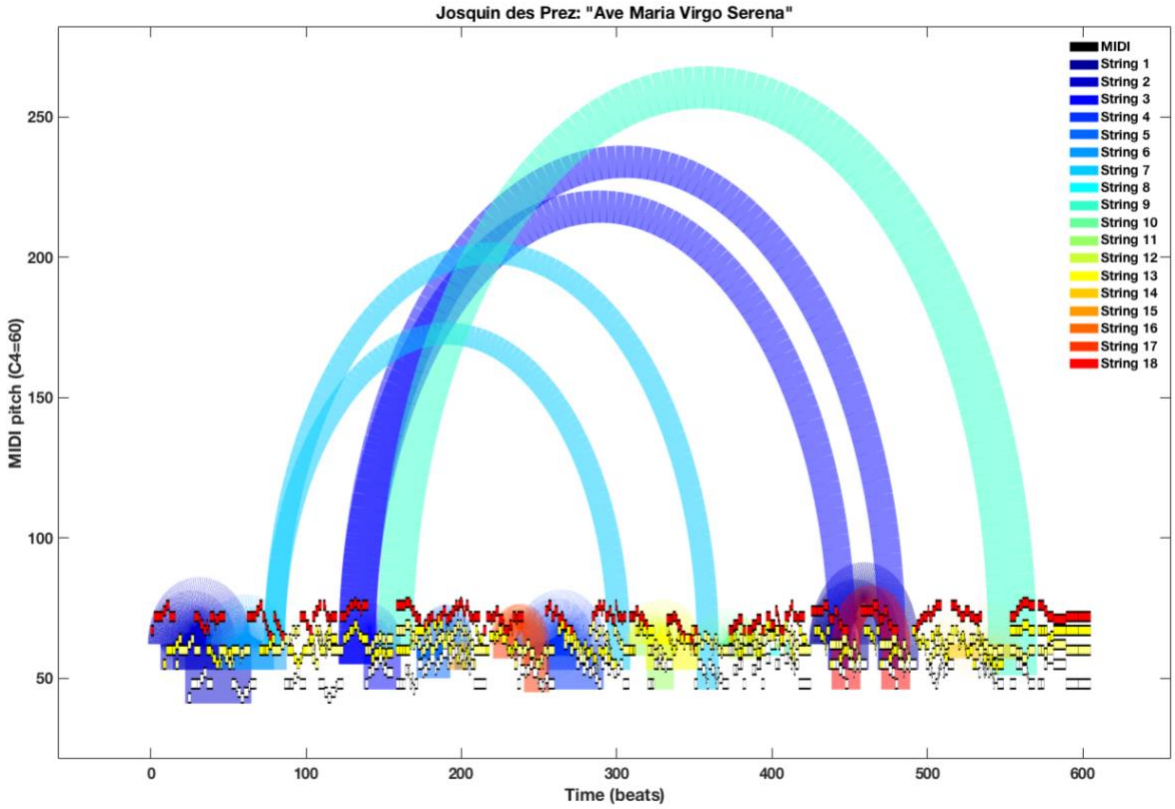
Example 1d (below): replot of selected strings (from **Pitch** algorithm) with labels added for legend (top right)



Exported CSV for **Pitch** algorithm analysis of "Alleluia, Exsultate Deo" (EXAMPLE 1):

String	Cardinality	Channel	Onset	Pitch
1	39	1	9	60
1	39	1	192.8	62
1	39	1	248.3	60
2	7	1	0	60
2	7	1	9	60
2	7	1	65.3	60
2	7	1	149.3	60
2	7	1	192.8	62
2	7	1	248.3	60
3	10	1	35.3	62
3	10	1	69.3	65
3	10	1	219.1	62
3	10	1	274.6	62
4	9	1	37.3	64
4	9	1	70.3	67
4	9	1	89.8	67
4	9	1	221.1	64
4	9	1	276.6	64

Example 2: Josquin des Prez “Ave Maria Virgo Serena” with all strings found shown



Example 3: Johann Sebastian Bach “Fugue in c minor” (BWV 847) with three strings selected (String 1: Subject; String 2: Countersubject 1; String 3: Countersubject 2)