MusicPlot: Interactive Self-Similarity Matrix for Music Structure Visualization

Wilson Louie*
Massachusetts Institute of Technology

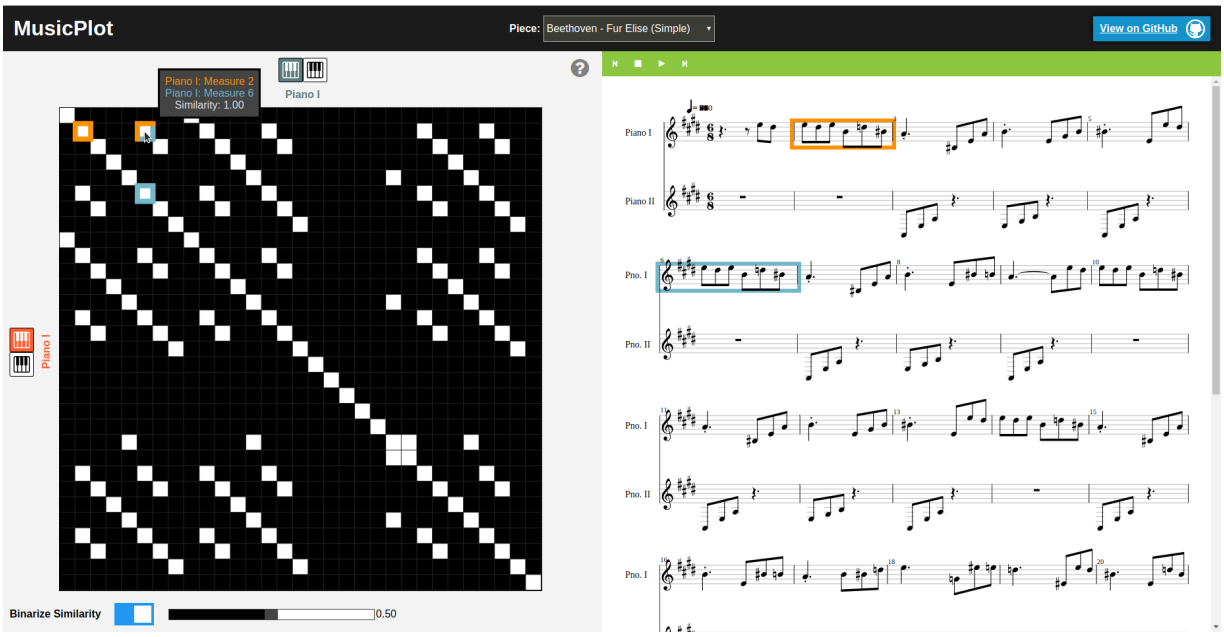

Figure 1: Example view of the MusicPlot interface. Users start by choosing the piece of music they wish to visualize via the top drop-down menu. The left panel shows a configurable self-similarity matrix representation between two instrumental parts of the piece, and the right panel shows the conventional music sheet representation with audio playback functionality. The two views are in sync, such that user interactions on one panel would reflect accordingly in the other panel.

Abstract

Self-similarity matrix representations of music has been shown to enable identification of structural and rhythmic elements by visual inspection. However, previous efforts in visualizing symbolically encoded music this way aim to visualize all, or most, instrumental parts of the opus as a single image, by overlaying multiple similarity matrices and introducing a myriad of color encoding. The result is something that can be difficult to decipher. Furthermore, from just the standalone matrix image, it becomes difficult to pinpoint the exact locations in the original piece a particular matrix cell is meant to represent. Here, we propose an interactive music visualization system that addresses these shortcomings. We anticipate that this visualization system would enable even laymen who cannot read musical notation to track and appreciate the structure of a piece of music.

1 Introduction

Although music is inherently an auditory phenomenon, there has been considerable interest in converting music into a visual medium. Previous efforts include artistic attempts of realizing images from sound [1], quantitative approaches via rendering of time and/or frequency content of an audio signal using methods such as the oscillograph and sound spectrograph [5, 7, 9], and visualization of symbolically encoded music, typically from MIDI note events and files [6, 10]. The motivation to build these systems range from purely entertainment or educational, to the potential applications to music categorization and automated retrieval systems [3, 4].

Leveraging the observation that structure and repetition is a general feature of nearly all music, others attempt to employ self-similarity matrices (SSM) to visualize music [2, 11]. In these approaches, the music signal is first converted into a suitable feature sequence. Each element of the feature sequence is then compared with all other elements in the sequence, and the results are organized in a matrix where each cell at positions i and j expresses the similarity between elements i and j of the music feature sequence. This can then be converted into an image by encoding the similarity values in each cell by color. Because similar segments of feature vectors tend to show up as paths of high similarity along and off diagonals of the SSM [8], musical structures and patterns can be elucidated visually. However, especially for long musical pieces that have correspondingly large SSMs, it can be difficult for one to corroborate the exact segments in the original piece a particular matrix cell is meant to represent. Moreover, a composition usually consists of several separated channels of musical information, but previous efforts aim to visualize them as a single image, by overlaying multiple similarity matrices and utilizing a large number of color encoding [11]. The resulting image can thus become hard to interpret due to the clutter. Another issue is ease of access, as previous work either just present

*e-mail: wilsonlouie1@gmail.com

the technique without a public implementation, or the implementation is no longer hosted for public consumption. An interactive web-based SSM interface that allows the user to selectively explore musical channels and to associate with the original music would alleviate these issues.

In this paper, we present MusicPlot, a web-based interface featuring SSMs for interactive exploration of music structure (Fig. 1). This system takes as a MIDI file as input, and generates a SSM view that is synced with a music sheet and audio player view. Rather than presenting a single potentially complex SSM representing the entire piece with all its musical channels, MusicPlot features configurable toggle and filter controls to show similarities between pairs of musical channels at a time. The SSM is also enriched with single cell resolution interactions that are synced with the sheet music view, which preserves reference to the original musical context. Based on generally positive reviews from seven people coming from a range of musical backgrounds, we anticipate this system to be insightful and enjoyable by laymen and musical experts alike.

2 Related Work

The application of SSMs on music visualization was initially described by Foote [2]. This approach takes as input a raw music recording, and converts the signal into a feature sequence using cepstral analysis. The definition of similarity between a pair of feature elements is based on vector autocorrelation. A SSM can then be constructed, by considering all pairwise similarities between elements of the feature sequence and laying them out in a grid. Through several case studies, the author demonstrated that structural and rhythmic patterns of the music become apparent in the resulting SSM, and alluded to applications of such a technique in music segmentation, analysis, and tempo and structure extraction.

Inspired by this work, Wolkowicz et al. [11] proposed a very similar technique, but for visualizing symbolically encoded music stored in MIDI files. While Foote's method uses a single channel recording as input data, Wolkowicz et al. argue that symbolically encoded music would produce more informative SSM visualizations, as performance-dependent features are hidden and logical musical channels are made explicit. Starting with an input MIDI file, their system first extracts the musical tracks. For each musical track, it extracts a series of notes representing the melody of each track. The series of notes is then converted into a feature sequence of relative pitch and relative duration between two consecutive notes. For every pair of musical tracks, a SSM is generated from their feature sequences, and the SSMs are overlayed as a single image. They also alluded to options to select only a subset of the tracks to be visualized, and an audio playback feature.

Our proposed MusicPlot builds off of ideas from both of these works. Rather than returning static SSM images given musical inputs as did previous works, our system provides SSM interactivity that enables easier understanding, and grounds the user to the original musical context by providing a synced music sheet and audio player view.

3 Methods

There are two major views in MusicPlot: a SSM panel and a reference music panel. The following sections describe the specifics of the two views, the SSM construction method, and the interactivity one can perform in MusicPlot.

3.1 Self-Similarity Matrix

Given an input MIDI file, MusicPlot generates SSMs that resembles Wolkowicz et al.'s system [11]. At any given time, a SSM for one pair of MIDI tracks (e.g. violin track vs. piano track) is shown. A given MIDI file may have multiple musical tracks, so the user needs to select a desired pair of tracks to compare using the available toggle buttons. In our case, a SSM is a square matrix where each

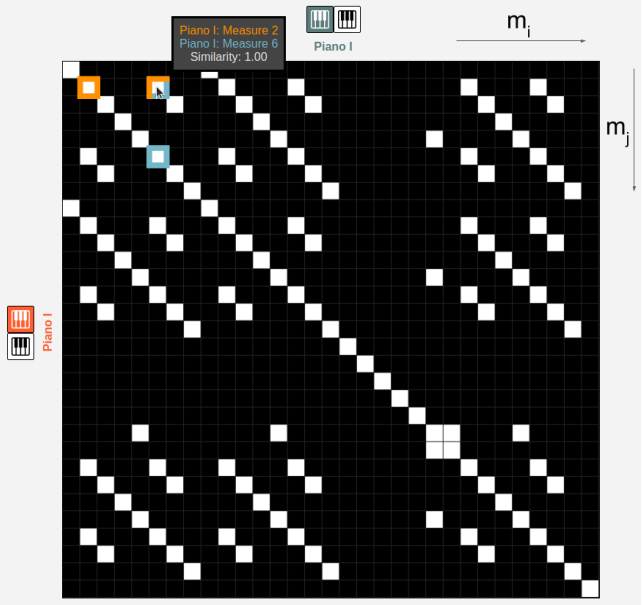

Figure 2: A MusicPlot SSM representation of Beethoven's Für Elise. Here, there are two piano tracks, and the SSM representation shown compares the first piano track with itself. There are 31 measures of music, so this SSM is a 31×31 matrix. The selected cell represents the comparison between measure 6 (m_6) of the top selected track and measure 2 (m_2) of the left selected track.

cell represents the similarity between two corresponding measures of a pair of tracks (Fig. 2). More formally, consider a MIDI file representing a composition with N measures of music. Each track in the MIDI would have N measures of music. For every MIDI track, we convert the N measures into N feature representations, yielding us a feature sequence of length N for each track. The corresponding SSM representation of two tracks (same or different) from the MIDI file would be an $N \times N$ matrix, where cell (i, j) represents the similarity between the feature representation of measure i (m_i) in the first track, and the feature representation of measure j (m_j) in the second track. White colored cells signify great similarities, whereas black colored cells signify little similarities. Time proceeds from the upper left corner to the lower right, e.g. the cell at the top left corner represents the similarity between the first measures of both tracks, and cells along the diagonal represent the comparison between measures of the same rank. The following sections describe our method of translating a MIDI track into a feature sequence of length N, and our similarity measure, in more details.

3.1.1 Feature Sequence from MIDI

Our feature sequence generation from MIDI tracks is derived from the notion of unigram as described by Wolkowicz et al. [11]. Given a MIDI track, we first extract a sequence of notes representing the melody of the track. Heuristically, we achieve this by retaining only the highest note at a time frame, which allows us to handle chords and concurrency in a single track. We then group these notes into measures. Measures are not explicitly encoded in MIDI, but we can unambiguously determine which notes belong to which measure from the time signature of the MIDI (which effectively gives us the number of quarter notes in a measure), the ticks per quarter note (which gives us the number of ticks in a measure), and the ticks duration of each MIDI note. To summarize, we have at this point a conversion of a MIDI track into a sequence of measures, where each measure consists of a sequence of MIDI notes, and each note has a certain pitch and duration. More formally, we

have a sequence of measures $[m_1, m_2, ..., m_N]$, where each measure $m_i = [(p_1, d_1), (p_2, d_2), ..., (p_k, d_k), ...]$ has some number of notes, each with a pitch p_k and duration d_k.

To make for more meaningful measure comparisons, we want our final feature sequence to take into account melody and rhythmic directions. To achieve this, we transform each measure in our sequence from a series of pitch and duration, to a series of relative pitch and relative duration. More formally, for each measure $m_i = [(p_1, d_1), ..., (p_k, d_k), ...]$, we transform it into $m'_i = [(p_2 - p_1, \frac{d_2}{d_1}), ...(p_{k+1} - p_k, \frac{d_{k+1}}{d_k}), ...]$. Our final feature sequence representation of the MIDI track becomes $[m'_1, m'_2, ..., m'_N]$.

3.1.2 Similarity Measure

Let $S_{i,j}$ be the value at cell (i, j) in the SSM, which represents the similarity between the feature representation m'_i and m'_j. We define $S_{i,j}$ to be the dice coefficient between the two:

$$S_{i,j} = \frac{2|m'_i \cap m'_j|}{|m'_i| + |m'_j|}$$

This definition constrains $S_{i,j}$ to be between 0 and 1, where 0 signifies the two measures are completely dissimilar, and 1 signifies the measures are identical in relative pitch and relative duration of notes.

3.1.3 Matrix Binarization

The resulting SSM is a matrix of similarity values, each a real value between 0 (0% similar) and 1 (100% similar). To convert this matrix into a visual image, we assign the color black to cells with similarity of 0, the color white to cells with similarity of 1, and shades of gray to similarity values in between. To make patterns more obvious, MusicPlot has the option of binarizing the similarity values of the matrix using a threshold, such that any cells with similarity values above the threshold is colored white, and black otherwise. This binarization feature is turned on by default, and can be configured using the controls below the SSM (Fig. 1).

3.2 MusicPlot Interactivity

Aside from the track selection toggles and the binarization threshold controls discussed in the sections above, users can hover over a specific cell (i, j) of the SSM and glean the tooltip for information about the measure numbers being compared, and the similarity value of the comparison (Fig. 1). To ground the user in the original musical context, the targeted measures are also highlighted in the music sheet view. The user can click to select a cell, and the corresponding measures of sheet music will be colored. The user can also use the player controls above the sheet music to play the associated audio.

3.3 Results

To demonstrate the utility of MusicPlot in revealing interesting musical structures and patterns from visual cues of SSM, we perform a case study using J.S. Bach's "Aria" BWV988 (Fig. 3).

3.3.1 Example: J.S. Bach - "Aria" BWV988

From visual inspection alone, one can see that there are hierarchical patterns within the piece. At the highest level, there seems to be two distinct melodies within the piece, demarcated by the red and blue regions. Within the first melody (red), there is a repetition of melodic phrases **a** and **b**, signified by the off diagonal copy **c**. Similarly this is the case for the second melody (blue) with regions **A** and **B** signified by **C**. Square blocks signify contiguous repeats of a certain pattern, represented by **D**. The other off-diagonal white colored cells indicate similarities between specific isolated measures. With the ability to also inspect the sheet music and access to the audio playback in MusicPlot, one can easily corroborate these visual cues with the relevant parts of the original music.

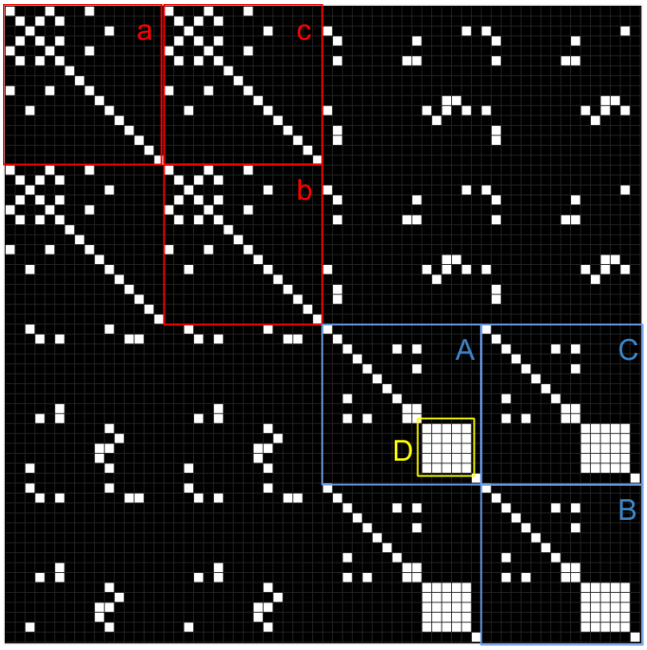

Figure 3: Annotated SSM of J.S. Bach - "Aria" BWV988. Just from the SSM, one can see that there are generally two distinct melodies in the piece (red and blue regions), and repeating patterns within these melodies.

4 Discussion

We have presented MusicPlot, an interactive music visualization system centered around SSMs. This system differs from previous works mainly in that it is designed to make corroboration between matrix cells and the original music explicit and easy. It is also a web-based application with no installation requirements, which lowers the barrier of access for many users.

We also presented an early version of this system to seven university students of varying musical background, and received generally positive feedback. The general consensus among laymen is that the interface is friendly and fun, and the entire visualization concept is interesting and intriguing. They really like that the SSM view is synced with the audio player and sheet music view, which encourages them to explore more. Those who are more musically inclined are pleasantly surprised to see this unconventional way to explore music, and draws parallel with how they think of musical structural similarity to the SSM representation. The only negative feedback is that there is a slight initial learning curve in understanding and interpreting SSMs, but the tool becomes entertaining and didactic thereafter. Thus, we expect MusicPlot to be insightful and enjoyable by other laymen and musical experts alike.

5 Future Work

MusicPlot has many areas that can be expanded upon. For one, custom inputs would be useful for users. At the time of writing, MusicPlot preloads a list of MIDI files users can choose from, but does not allow users to upload their own MIDI files for visualization. We would like to implement some sort of custom input functionality in the future. Secondly, MusicPlot currently only supports MIDI files, but since the system applies a general approach to visualizing symbolically encoded music, other forms of music encodings such as ABC notation files and MusicXML can also be supported. If we apply the data processing technique described by Foote [2], we can in theory also support raw audio recording with limited functionality on the sheet music view. Another area of improvement

is to add features useful for navigating the SSM. For example, adding zoom interactions would be useful for investigating especially large SSMs. The ability to multi-select cells, or to automatically select connected blocks of cells on click, would also be useful. Another useful extension is to add an easy to understand walk-through or tutorial of interpreting SSMs and the general MusicPlot interface, as that is something our small group of reviewers suggested would be helpful for newcomers.

REFERENCES

[1] Disney. *Fantasia*. Buena Vista Pictures Distribution, Inc., 1940.

[2] J. Foote. Visualizing music and audio using self-similarity. In *Proceedings of the seventh ACM international conference on Multimedia (Part 1)*, pp. 77–80, 1999.

[3] S. Jun and E. Hwang. Music segmentation and summarization based on self-similarity matrix. In *Proceedings of the 7th international conference on ubiquitous information management and communication*, pp. 1–4, 2013.

[4] S. Jun, S. Rho, and E. Hwang. Music structure analysis using self-similarity matrix and two-stage categorization. *Multimedia Tools and Applications*, 74(1):287–302, 2015.

[5] W. Koenig, H. Dunn, and L. Lacy. The sound spectrograph. *The Journal of the Acoustical Society of America*, 18(1):19–49, 1946.

[6] S. Malinowski and L. Turetsky. Music animation machine. *Music Worth Watching"," http://www. musanim. com*, 2011.

[7] W. Moritz. Mary ellen bute: Seeing sound. *Animation World*, 1(2):29–32, 1996.

[8] M. Müller and M. Clausen. Transposition-invariant self-similarity matrices. In *ISMIR*, pp. 47–50, 2007.

[9] K. Potter Ralph, G. A. Kopp, and H. C. Green. Visible speech, 1947.

[10] S. M. Smith and G. N. Williams. A visualization of music. In *Proceedings. Visualization'97 (Cat. No. 97CB36155)*, pp. 499–503. IEEE, 1997.

[11] J. Wolkowicz, S. Brooks, and V. Kešelj. Midivis: Visualizing music pieces structure via similarity matrices. In *Proceedings of the 2009 International Computer Music Conference, ICMC'09*, pp. 53–6, 2009.