Computer-Aided Analysis Across the Tonal Divide: Cross-Stylistic Applications of the Discrete Fourier Transform

Jennifer Diane Harding
Florida State University
jdharding@fsu.edu

Abstract

The discrete Fourier transform is a mathematically robust way of modeling various musical phenomena. I use the music21 Python module to interpret the pitch classes of an encoded musical score through the discrete Fourier transform (DFT). This methodology offers a broad view of the backgrounded scales and pitch-class collections of a piece. I have selected two excerpts in which the composers are very frugal with their pitch class collections—one in a tonal idiom, the other atonal. These constrained vocabularies are well suited for introducing the DFT's methodological strengths as they pertain to score analysis.

Introduction

The discrete Fourier transform (DFT) has recently gained traction in the music theory community as a mathematically robust way of modeling various musical phenomena. Theorists have used the DFT to model harmonic motion [1], set class similarity [2], meter [3], and the analysis of larger musical excerpts [4]. The work presented in this article falls into this last category. I use the music21 Python module to interpret the pitch classes of an encoded musical score through the discrete Fourier transform. This methodology offers a broad view of the backgrounded scales and pitch-class collections of a piece, what Dmitri Tymoczko refers to as "macroharmony"[5]. I have selected two excerpts in which the composers are very frugal with their pitch class collections—one in a tonal idiom, the other atonal. The exposition of the first movement of Mozart's String Quartet No. 4 in C Major, K. 157 is almost entirely diatonic. The theme from Messiaen's *Theme and Variations for Violin and Piano* adheres strictly to two of his modes of limited transposition. These constrained and highly symmetrical (in the case of Messiaen) harmonic vocabularies are well suited for introducing the DFT's methodological strengths as they pertain to score analysis.

The discrete Fourier transform and pitch class collections

The Fourier transform (FT) is a mathematical function that decomposes an input signal into its constituent sinusoidal components. In contrast, the *discrete* Fourier transform uses an array of discrete numbers, as opposed to a continuous signal, as its input. This makes it possible to apply the DFT to the pitch-class content of musical scores: the counts of pitch classes become the input array. The results of the DFT provide information about the aural saliency of the input, which roughly correlate with the qualia of the familiar interval cycles [6]: is it chromatically clustered? Is it more "whole-tone" sounding? Is it "fifthy?" Applying the DFT to macroharmonies allows us to see a broad overview of a composition's sound.

The output of the DFT comes in the form of six non-trivial Fourier components, denoted f_1, f_2, f_3, ...f_6.[1] Each component can be imagined as a circle with twelve positions or nodes where pitch classes are conceptually located, which I refer to as the *component circle*.[2] The coefficient of each component reflects how far adjacent

1 The f_0 component is always equal to the cardinality of the set. Components f_7–f_{11} are mirror images of components f_1–f_5 and are therefore redundant.

2 The number 12 here comes from examining the twelve pitch classes. In music with quarter-tones, we would use 24 nodes. If we were examining meter, we might use four nodes for the four beats of a quadruple meter, or 16 nodes if we were looking at every sixteenth-note subdivision. The number of nodes is dependent upon the length of the input array.

integers (indicating pitch classes) are separated: on the f_1 component circle, each pitch class is one node apart from its neighbor; on the f_2 component circle, pitch classes are placed every two nodes, and so on. Figure 1 shows all six of the non-trivial Fourier components as component circles.

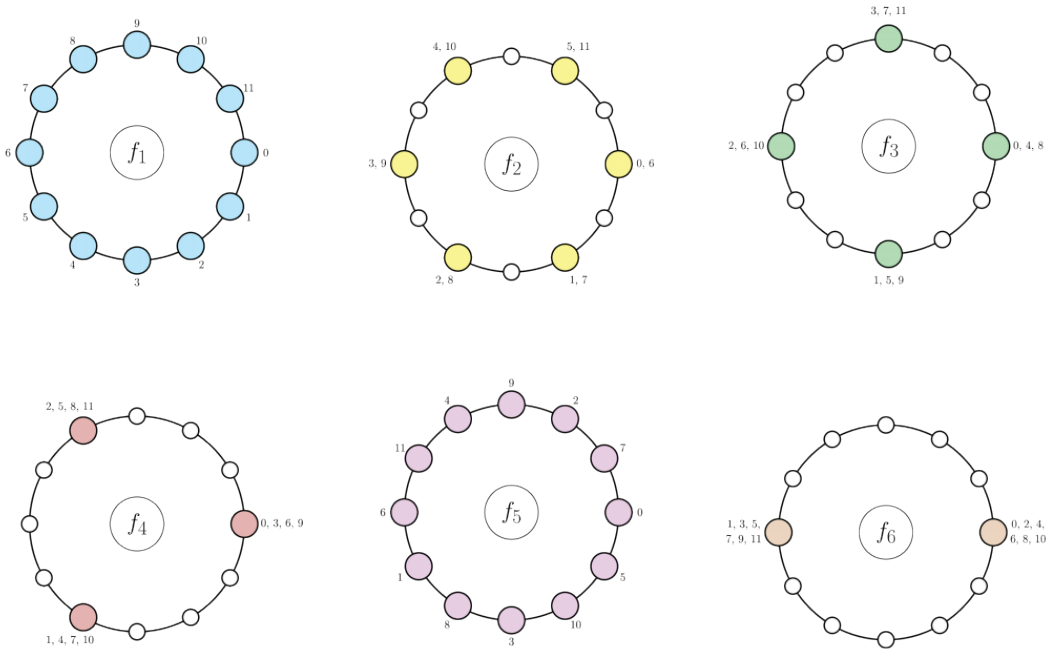
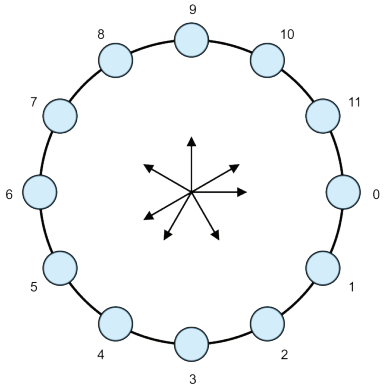


Figure 1: The six non-trivial Fourier components.

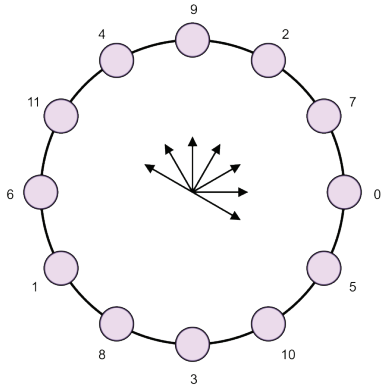
The values from a pitch-class array can be plotted on a component circle as vectors, each with a length or *magnitude* (equal to the corresponding value of the array) and a direction or *phase* (expressed as an angle).[3] The pitch class's position on the component circle determines the vector's phase. The vectors are added together by positioning them head-to-tail, resulting in a new vector from the origin of the circle to the end of the chain.[4] The magnitude and phase of this resultant vector for every Fourier component is the output of the DFT. This process is shown in Figure 2, mapping the C-major diatonic collection onto the f_1 and f_5 component circles. The resultant magnitude of the f_1 component indicates how chromatic the collection is. From the very low magnitude of 0.27, we can say that the diatonic collection is not very chromatically clustered (indeed we know that it is minimally chromatic for a septachord). In contrast, the f_5 component indicates how "fifthy" a collection is, and from the very high magnitude of 3.73, we can say that the diatonic collection is very fifthy (indeed, maximally so).

3 0° is positioned due East, as on a polar coordinate system.

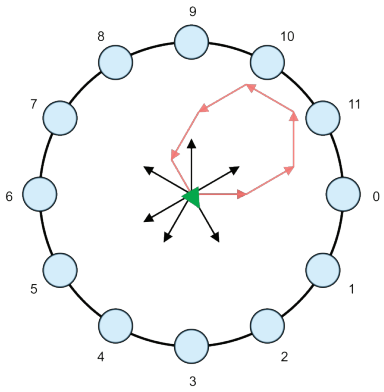
4 Since vector addition is both commutative and associative, the actual order in which the vectors are added does not matter.



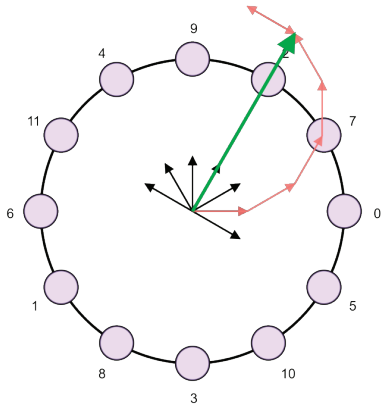
(a) C Major collection on the f_1 component circle



(b) C Major collection on the f_5 component circle



(c) Resultant vector has a magnitude of 0.27 and phase of -30°



(d) Resultant vector has a magnitude of 3.73 and phase of 60°

Figure 2: The C Major diatonic collection mapped onto the f_1 and f_5 component circles

Of note is the fact that on the f_5 component circle, the resultant vector for the C-major diatonic collection does not pass through pc0, the tonic (Figure 2(d)). Instead, it passes exactly through pc2 (60°), the point of symmetry for the collection. Of course, in music we rarely see exactly one instance of each pitch class together like this. Rather, some pitch classes will be omitted while others are duplicated. If we account for the number of times each pitch class is present in a passage by using multisets—sets that allow for multiple instances of each element—the phase of the resultant vector will probably not pass quite so cleanly through pc2, but rather somewhere in its vicinity. A quantizing function snaps the phase to the nearest node, making the data more comprehensible. On the f_5 component, a diatonic collection will quantize to its scale-degree 2, meaning that we can use the phase as an indicator of the key of a largely diatonic passage. Or, more accurately, we can use the phase to identify the implied key signature.

Methodology

I built my computational apparatus in Python using music21,[5] a Python module developed by Michael Cuthbert and Christopher Ariza at MIT [7]. Music21 parses many different symbolic music file formats: MusicXML, MIDI, MEI, Humdrum, and Lilypond, to name a few. The file is converted to a *stream*, the fundamental object in music21. Streams store pieces of score information (note objects, articulations, barlines, etc.) in a hierarchical structure reminiscent of XML. Each object in the stream is located at a particular *offset*—or point in time from the beginning of the piece—as measured in quarter-note units.

To process the score, I create a window of a constant number of beats. As the window slides across the score, incrementing by beat, the program performs the DFT on the pitch content contained within that window.[6] This continues until the end of the score, creating a series of overlapping windows in much the same way that a camera takes multiple pictures and stitches them together to create a panoramic photo .

For some pieces of music, the process of sliding the window by a beat is computationally trivial because the meter is both constant and symmetrical. To slide the window forward by a beat, the beginning and end of the window offsets can simply be increased by the length of the beat. However, music with asymmetrical or changing meters pose more of a problem and require a much more flexible system. The solution was to use the music21 method `getContextByClass()` to retrieve the meter of every measure regardless of whether the `<time>` tag (used to indicate a meter signature) is present in the XML. Then, using the music21 object `MeterSequence` and its various partition methods, I create a list of the offsets that correspond with beat location which becomes the basis for incrementing the window. This helps to ensure that the portion of music being examined is always a meaningful unit.

The program has three possible strategies for how to count pitch classes in each window: *onset*, *duration*, and a *flat* set. The onset and duration options both return multisets. Onset counts how many times a pitch class is attacked within the window. If a note is initiated once and sustained, it will only be counted once, whereas if it is repeated, it is counted again for every repetition. The duration option returns the total duration of every pitch class within the window, even if a note was initiated before the window began. The flat set option counts each pitch class only once, regardless of how many times it appears on the musical surface. In essence it asks the binary question: "is this pitch class present?" These three options provide different ways of interpreting the pitch class data of the score, with each representing a different potential hearing of the music based on which parameters the listeners attend to.

Figure 3 shows Cipriano de Rore's madrigal "Calami sonum ferentes" processed with each of these three approaches. The top two examples, using the onset and duration options, look fairly similar. This is a result of the voices tending to articulate each note just once as part of a melody rather than retaining the same pitch for multiple syllables. The f_5 component is quite jagged, indicating that while the diatonic collection is the primary background collection, chromaticism is suppressing its saliency. The graph showing the flat option confirms this, at times dropping to 0 in all components, which indicates that the entire chromatic aggregate is present in that window.

5 http://web.mit.edu/music21

6 The idea of combining the DFT with a sliding window was first proposed by Matthew Chiu at the 2019 Annual Conference of Music Theory Midwest. I am grateful for his correspondences on the topic.

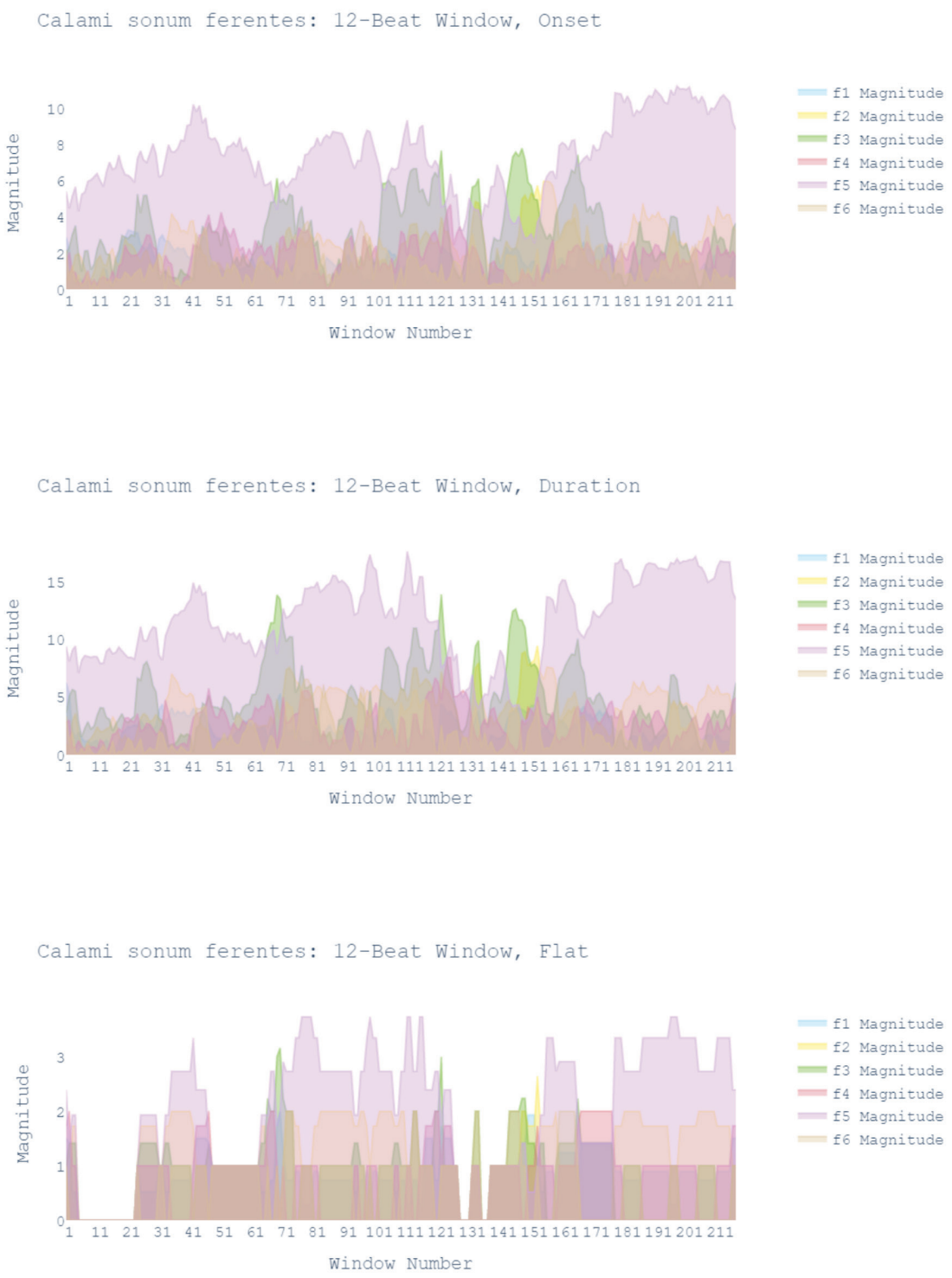


Figure 3: Three analyses of Cipriano de Rore's "Calami sonum ferentes."

Calculating the onsets is made complicated by the fact that in music21, the `Tie` class represents the visual and conceptual idea of tied notes. However, it retains the individual identity of the two different note objects rather than interpreting them as a single, long durational entity. Rather, music21 reads two individual notes. To com-

pensate for this, the onset option relies on the music21 method `stripTies()`, which replaces all tied notes with a single note having a duration equal to all the tied constituents. This ensures that only one instance of the note is counted.

The duration option uses the music21 method `sliceByBeat()`, which splits a note at the beat offsets that music21 determines based upon the local time signature. Since the window increments every beat, this ensures that when the window boundary intersects a held note, the duration of the note within the window is accounted for and the portion outside the window is omitted. The flat option processes in the same way to avoid "missing" any notes whose durations began before the beginning of the window.

Once all the pitch-class arrays are collected, I weight the data with a logarithmic scale. The weighting procedure helps to shape the data to match our cognitive experience with the music. As the frequency of an individual pitch class increases, the information value of any given instance of that pitch class decreases. Because of its uniqueness, a single instance of a pitch class will have a relatively high influence. Musically speaking, this is equivalent to saying that the lone appearance of scale-degree sharp-4 will be more salient than repeated occurrences of scale-degree 1. The general profile of the array will stay the same—the higher frequency of scale-degree 1 will still have a greater effect than the single sharp-4, but the higher values will be flattened out.

Finally, I apply the DFT to each of the arrays and store the data in Pandas (a Python library used for data manipulation and analysis) data frames.[7] These tables are then used for queries and generating visualizations of the data.

Analysis

The graph in Figure 4 shows the magnitudes of all six Fourier components in the exposition to the first movement of Mozart's String Quartet No. 4 in C Major, K. 157, generated with a 16-beat sliding window using the duration valuation of pitch classes. The mass of lavender represents the f_5 component, and its prevalence indicates that perfect fifths or *diatonicity* (i.e., sticking to only the notes that are native to a particular key signature) is the most salient feature of the harmonic landscape.

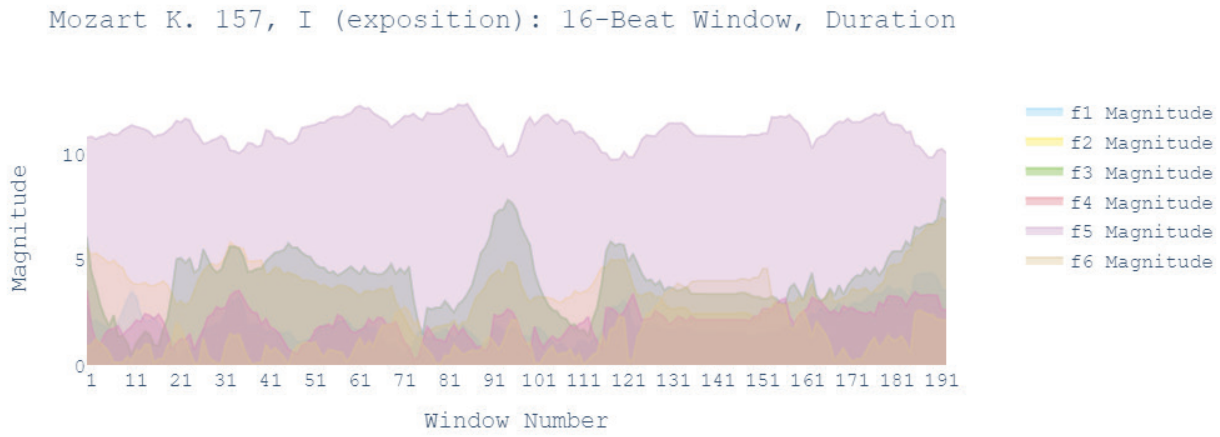


Figure 4: Magnitudes of the six Fourier components in the exposition of Mozart K. 157, I.

Figure 5 extracts the f_5 magnitude from Figure 4 and overlays both the raw phase data and the quantized phase. Since the phase of the f_5 component correlates with implied key signature, we can use this information to infer that the piece begins with 0 sharps or flats (indicated by the 60° position) and moves to 1 sharp (indi-

7 https://pandas.pydata.org

cated by the 90° position sometime around the 75th window and generally remains there for the rest of the exposition. In other words, we see evidence of the modulation to the dominant.

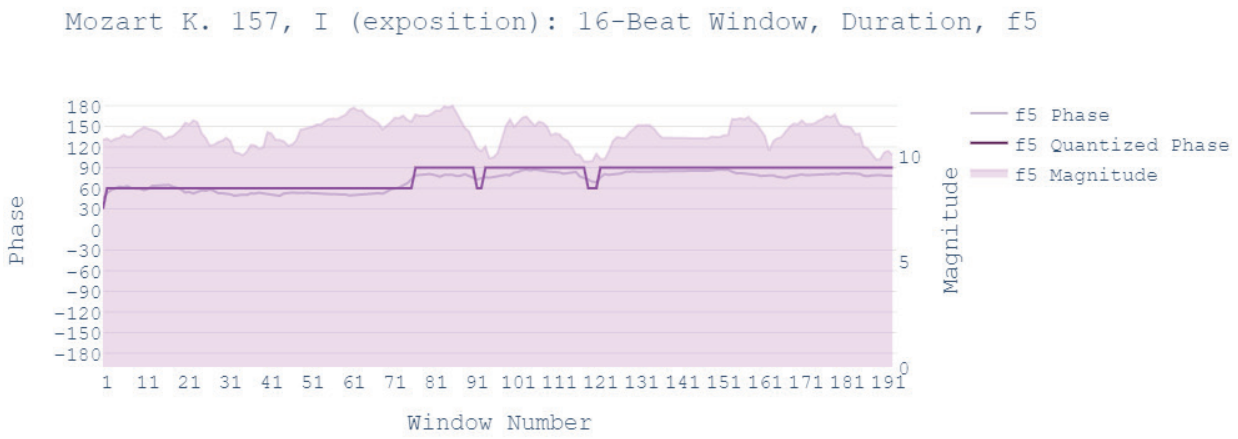


Figure 5: Magnitudes and phase of the f_5 component in the exposition of Mozart K. 157, I.

The theme from Messiaen's *Theme and Variations for Violin and Piano* adheres strictly to his second (octatonic) and third modes of limited transposition, a very different harmonic context than Mozart's. Figure 6 shows the magnitudes of the six Fourier components generated with a 16-beat sliding window using the onset valuation of pitch classes. The three-part construction of the theme is apparent based on which Fourier components are the most salient (have the highest magnitude). The first section is in *AA'* form, with the triple peaks of the f_6 component (in windows 1-11 and again around 31-41) showing the similarity of the beginnings of the two *A* repetitions most clearly.

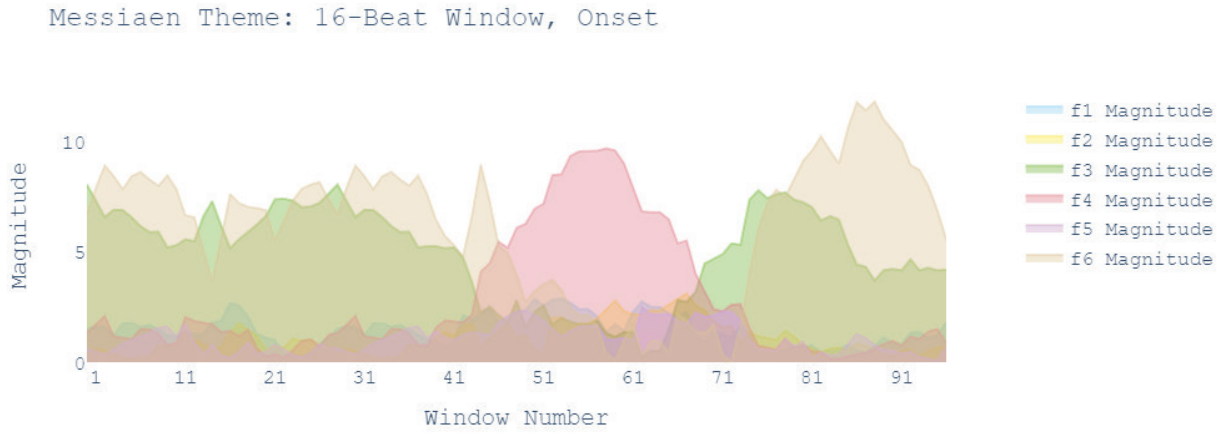


Figure 6: Magnitudes of the six Fourier components in the Theme from Messiaen's Theme and Variations for Violin and Piano.

The magnitude of the f_4 component is extracted in Figure 7 and shown along with the phase and quantized phase. The high magnitudes in the f_4 component and the phase at -60° indicate the presence of the Octatonic 01 collection (Messiaen's second mode of limited transposition), in much the same way that the position of phase can be used to determine the implied key signature when the f_5 component reaches a high magnitude.

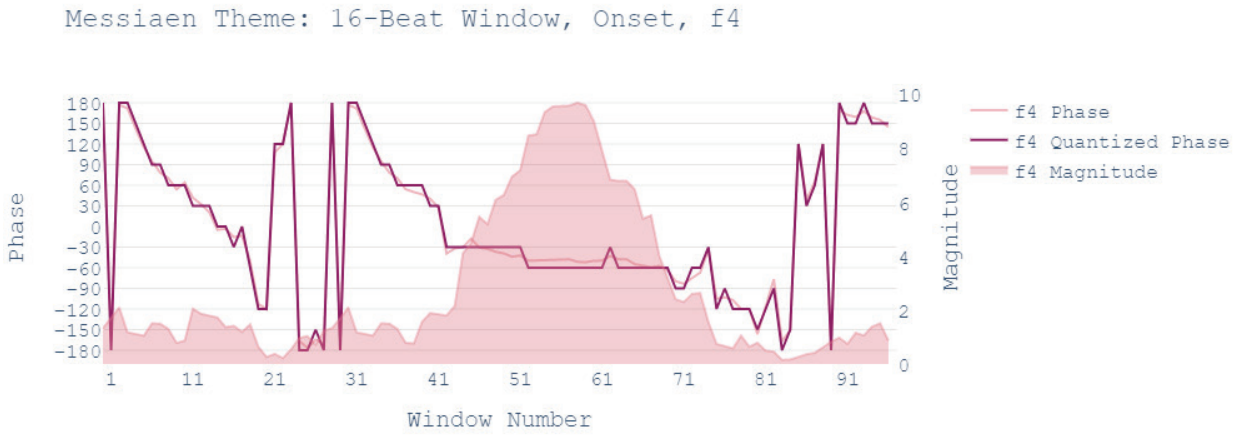


Figure 7: Magnitude and phase of the f_4 component in Messiaen's Theme.

Messiaen's third mode of limited transposition is a bit more subtle to tease out. Mode 3 can be thought of as a scale built of repeated <half-half-whole> step patterns or, more productively for our use, as a combination of three of the four augmented triads (interval-class 4-cycles, represented by f_3). This further means that it includes one of the two whole-tone scales (2-cycles, represented by f_6) and half of the other. Looking at the first and third sections of the piece (Figure 6), Fourier components f_3 and f_6 are the most salient. While the magnitude is high, the phase of f_6 (shown in Figure 8) stays at 0°, the angle indicating the even whole-tone collection. However, if this were just a whole-tone collection, the vectors in the f_3 component would cancel each other out (Figure 10), effectively removing f_3 from the graph, and the magnitude of f_6 would be even higher than it is. Clearly this is not the case as the magnitude for the f_3 component is relatively high, close to that of f_6. The f_3 phase hovers around 90° (Figure 9), the location of three pitch classes (3, 7, and 11) that are not included in the even whole-tone collection as shown in Figure 10. The presence of these three pitch classes dramatically bolsters the f_3 component, but has the opposite effect on the f_6 component, substantially lowering its magnitude.

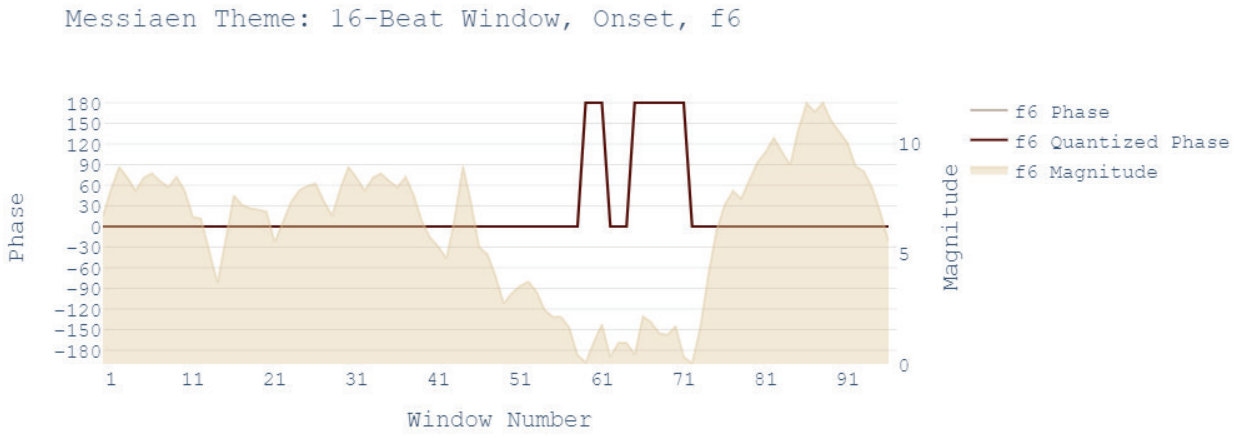


Figure 8: Magnitude and phase of the f_6 component in Messiaen's Theme.

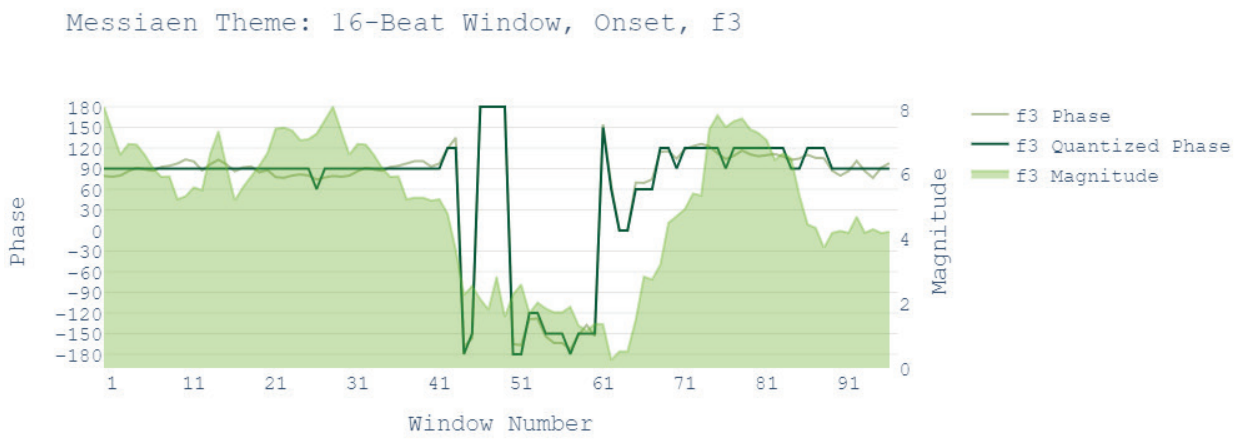


Figure 9: Magnitude and phase of the f_3 component in Messiaen's Theme.

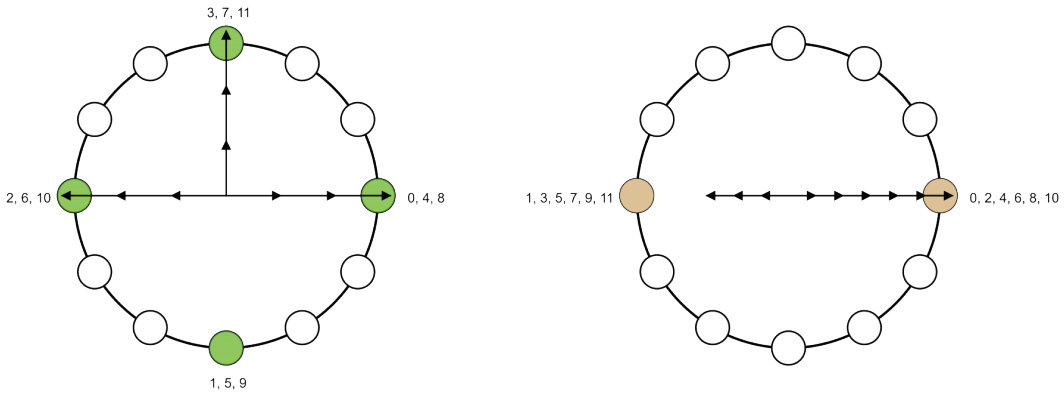


Figure 10: Third mode of limited transposition mapped onto the f_3 and f_6 component circles**.**

Discussion and conclusion

Examining macroharmonies through the DFT clearly displays a broad overview of a piece's sonic landscape, and it does so in a completely quantifiable and mathematically precise way. These two brief analyses offer a limited view into this methodology and its capabilities; many other variants exist. The overview can be as blurred or as granular as the analyst wishes by adjusting the size of the sliding window. Music based in other collections that equally divide the octave is evaluated with as much ease as our 12-note chromatic world, opening the door to examine other understudied repertoires simply by changing the length of the input array. Different modes of listening can be modeled by using the different pitch class evaluation strategies (onset, duration, and flat set).

The DFT can also function as a key-finding algorithm—a long-studied topic in both music cognition and computation. It is typically not enough to examine just the f_5 component to determine a key, as in the simple[MB1] Mozart example here. Instead, components f_2 and f_3 also play strong roles in tonal music, fluctuating based

on sounding harmony while f_5 remains stable [8]. The DFT provides a mathematically robust way to model harmonic and tonal activity by tracking motion through $f_{2/3/5}$ space [1].

Perhaps most significantly, this is a single methodology that is equally applicable to music from a wide variety of genres, time periods, and styles. As long as the music can be encoded with discrete pitches and rhythms, this approach will have something to say about it. Many music-analytical techniques are bound to a particular repertoire. The DFT is stylistically and historically agnostic, allowing for a more unified understanding of music writ large.

Works cited

[1] Yust, Jason. "Harmonic Qualities in Debussy's 'Les sons et les parfums tournent dans l'air du soir'" *Journal of Mathematics and Music* 11, nos. 2-3 (2017), 155-73.

[2] Tymoczko, Dmitri. "Set-Class Similarity, Voice Leading, and the Fourier Transform." Journal of Music Theory 52, no. 2 (2008), 251-72.

[3] Amiot, Emmanuel. *Music through Fourier Space: Discrete Fourier Transform in Music Theory*. Cham, Switzerland: Springer International Publishing, 2016.

[4] Chiu, Matthew. "A systematic approach to macroharmonic progressions: Durufle's Requiem through Fourier Space" presented at the Thirtieth Annual Conference of Music Theory Midwest, Cincinnati, OH, May 10-11, 2019.

[5] Tymoczko, Dmitri. *A Geometry of Music: Harmony and Counterpoint in the Extended Common Practice.* Oxford: Oxford University Press, 2011.

[6] Quinn, Ian. "General Equal-tempered Harmony: Parts 2 and 3" Perspectives of New Music 44, no. 2 (2006), 114-58.

[7] Cuthbert, Michael Scott, and Christopher Ariza. "Music21: A Toolkit for Computer-Aided Musicology and Symbolic Music Data" in *Proceedings of the International Society for Music Information Retrieval* (ISMIR 2010), 637-42.

[8] Yust, Jason. "Probing Questions about Keys: Tonal Distributions through the DFT" in *Mathematics and Computation in Music, Proceedings of the Sixth International Conference* (MCM 2017). Vol. 10527 of *Lecture Notes in Artificial Intelligence*, eds. Octavio A Agustin-Aquino, Emilio Lluis-Puebla, and Mariana Montiel. Cham, Switzerland: Springer International Publishing, 167-79.