

Slicing It All Ways: Mathematical Models for Tonal Induction, Approximation, and Segmentation Using the Spiral Array

Elaine Chew

Epstein Department of Industrial and Systems Engineering, University of Southern California Viterbi School of Engineering, 3715 McClintock Avenue, GER240, Los Angeles, California 90089-0193, USA, echew@usc.edu

This paper presents the spiral array model and its associated algorithms for tonal induction, approximation, and segmentation. The spiral array is a geometric model for tonality that clusters perceptually similar tonal entities. The model summarizes music information as interior points inside an array of spirals. Distances in the spiral array space are used to quantify tonal similarity. The paper traces the evolution, and presents general forms, of the existing algorithms for key finding, pitch spelling, and segmentation, and proposes a new $O(n)$ algorithm, Argus, for tonal segmentation. The proposed algorithm computes a value that quantifies the discrepancy between the local contexts in the future and past at each point in time. Discrepancy values exceeding control thresholds are shown to mark the segmentation boundaries of the test set that concur with expert analyses. A number of window sizes and threshold settings are investigated. The algorithm is demonstrated using Edward McDowell’s “To A Wild Rose” and tested on Franz Schubert’s *Allegretto* from Moment Musical D780 No. 6 and *Thema* from Impromptu D935 No. 4. The algorithm accurately locates tonal boundaries in all three case studies.

Key words: computational music cognition; automated tonal analysis; real-time algorithms

History: Accepted by Mark Steedman, Associate Guest Editor for the Special Cluster on Computation in Music; received March 2004; revised August 2004; accepted January 2005.

1. Introduction

This paper reviews algorithms for tonal induction, approximation and segmentation using the spiral array model and proposes an $O(n)$ algorithm for tonal segmentation. The spiral array model is a geometric representation of the hierarchical entities that form the framework of pitch relations known as tonality. This paper first presents an overview of existing algorithms for computational music analysis using the spiral array, including the center of effect

generator (CEG) algorithm for key finding, three pitch-spelling algorithms, and a boundary search algorithm. The issue of tonal segmentation is a common thread that runs through these different approaches to problems in music perception and cognition. Based on insights gleaned from these earlier algorithms and computational experiments, I propose a real-time algorithm for tonal segmentation, presenting and analyzing the results of this real-time algorithm for a range of parameter values when tested on “To A Wild Rose,” a composition by American composer Edward McDowell (1896). Two examples by Franz Schubert (1828), *Allegretto* from Moment Musical D780 No. 6 and *Thema* from Impromptu D935 No. 4, further demonstrate the efficacy of this algorithm.

Systematic and computational research applied to problems in music perception and cognition has gained rapid momentum in the past decade, motivated by the increased technological interest in creating, manipulating and accessing digital music. Central to machine models of music cognition or human understanding of music is the concept of *tonality*, the system of pitch relations that frame most of the music we hear. Humans exhibit numerous cognitive abilities associated with the recognition and comprehension of tonal structure including key finding (also known as tonal induction) and the perception of key changes (*modulations*). Machine models that mimic such musical intuition will advance the state of the art in automated music transcription, music information retrieval, and interactive music systems.

A number of approaches have been proposed for tonal induction. Longuet-Higgins (1962) observed that members of the same key form compact shapes on the harmonic network (a lattice of pitch classes arranged so that neighboring pitches are five scale steps apart in the horizontal axes and three major scale steps apart in the vertical axes). In Longuet-Higgins and Steedman (1971), these distinctive shapes were used to identify the key of the subjects of Bach’s “Well-Tempered Clavier.” The CEG algorithm revisited in this paper uses the spiral orientation of the harmonic network and its interior space to determine key. This interior-point approach summarizes music information, representing the aggregate information as a spatial point, and does not require perfect adherence to the pitch membership of the key for accurate identification.

A different approach was taken by Krumhansl and Schmuckler, described in Krumhansl (1990). Using experimental data describing human perception of how well different tones fit in a prescribed context, Krumhansl used multidimensional scaling to deduce pitch profiles for each key. The Krumhansl and Schmuckler (K-S) method for key finding first creates

a histogram of pitch frequencies (or durations) and matches these against the pitch profile templates. The assumption is that pitch classes that are perceived to be more stable should occur more frequently. Krumhansl's multidimensional scaling results and the K-S algorithm are popular choices for applications requiring key finding, for example, Gomez's (xxxx) method for determining the tonality of polyphonic audio, Pickens et al.'s (2003) approach to retrieval by harmonic content, and Toiviainen and Krumhansl's (2003) study on the perception of tonal regions in music.

One of the oft quoted uses of key-finding algorithms is in automated transcription of music, cited by both Krumhansl (1990) as well as Longuet-Higgins (1976). Knowing the key allows one to spell musical tones using the contextually correct pitch names (essential for notating music), and having the correct spelling of all notes allows one to determine the key more accurately. Pitch spelling occurs because a pitch represented by a number (such as in MIDI, the Musical Instrument Digital Interface, format), can be spelled in a number of different ways. A simple example is that MIDI number 58 can be A \sharp or B \flat . The coupling of the problems of key finding and pitch spelling makes it difficult to evaluate and analyze the results of the algorithms. Recently, several researchers have proposed methods for pitch spelling that do not require the key to be pre-determined (including Temperley 2001, Cambouropoulos 2003, Chew and Chen 2003ab, 2005, and Meredith 2003, 2004).

Windows of information that capture the context are essential to the pitch-spelling approaches. Both Meredith's (2003, 2004) and Cambouropoulos' (2003) pitch-spelling algorithms use sliding windows to capture the tonal context. Temperley's approach assigns pitch names one note at a time. The two-phase boot-strapping approach to determining context described in Chew and Chen (2003b, 2005) will be outlined in this paper. Meredith's (2003, 2004) ps13 algorithm prescribes a spelling based on frequency counts that reflect the likely tonal context of the window. Cambouropoulos' (2003) interval-optimization approach selects spellings that maximize the occurrence of intervals in the diatonic scale. Temperley's (2001) algorithm selects the nearest neighbor along the line of fifths.

Apart from pitch spelling and key finding, a related problem is determining local key context. The problem other researchers have faced in applying the K-S algorithm to a frame of notes, advancing forward one unit at a time, is that the key selected varies a fair bit from one frame to another. Two researchers have proposed solutions to the problem of stabilizing the key choice of the K-S algorithm. Temperley (1999) imposed a penalty for selecting a key different from the one in the prior window. Shmulevich and Yli-Harja (2000) proposed a

smoothing technique that, in a given window, selects the key with the shortest total distance (using Krumhansl’s 1990 model of key distances) to all other keys in that window.

The problem of segmentation emerges as one that is pervasive in, and critical to, machine models for tonal comprehension. What is the local context? When does the key change? The answers to these questions affect the accuracy of the algorithms. So far, most researchers used a simple sliding window to capture the local context (see, for example, Temperley 1999, Shmulevich and Yli-Harja 2000, Cambouropoulos 2003, and Meredith 2003, 2004). The single sliding window was found to be insufficient for catching sudden and drastic tonal changes for pitch spelling, as described in Chew and Chen (2003b). The unmitigated smoothing effect of a single sliding window is less than ideal for capturing contextual changes. Chew and Chen (2003ab, 2005) explored ways to approximate the local context constrained by a global estimate using both local and cumulative windows.

In Chew (2002), I focussed on the problem of finding the boundaries that mark contextual changes. A static boundary search algorithm was proposed that determined the optimal segmentation for a given piece of music. This static boundary search algorithm required knowledge of the entire piece and the number of segmentation boundaries.

In this paper, I shall propose an $O(n)$ segmentation algorithm that evaluates in real time, using the spiral array, the likelihood of the presence of a segmentation boundary at every point. This dual window approach uses both a forward and back window to capture both the future and past context for tonal evaluation. Using the spiral array, the tonal discrepancy can be quantified and charted. Segmentation boundaries will be marked by large discrepancy values, while areas of tonal stability will give low discrepancy values.

I begin with a review of the spiral array model in Section 2, followed by descriptions of the CEG key-finding algorithm and three pitch-spelling algorithms in Section 3. Section 4 gives a review of the static boundary search algorithm before presenting the real-time segmentation algorithm, named Argus, followed by computational results when the algorithm was applied to Edward McDowell’s piano piece “To A Wild Rose.” Section 5 presents additional examples and computational results when the algorithm is applied to Franz Schubert’s *Allegretto* from Moment Musical D780 No. 6 and *Thema* from Impromptu D935 No. 4. The paper concludes with a discussion of the Argus algorithm and some applications of the tonal induction, approximation, and segmentation algorithms.

2. A Geometric Representation for Tonality

This section describes the spiral array model, a geometric representation for tonality. The outermost spiral, representing pitch classes, is the helical form of Longuet-Higgins' (1962) Harmonic Network. Distinct from the lattice representation of pitch relations, the spiral array combines discrete and continuous space to generate higher level constructs in the interior of the pitch class spiral. Musical information is summarized using convex combinations of pitches weighted by their time structures, which can be durations or metric weights. These interior points are called *centers of effect* (c.e.). Section 2.1 describes the geometry of the spiral array, Section 2.2 the calibration of the model, and Section 2.3 defines the c.e.

2.1. The Array of Spirals

As its name suggests, the spiral array consists of an array of spirals, each representing a musical object type. The types of tonal entities represented include *pitches*, *chords*, and *keys*. Pitches, fundamental building blocks in music, are sounds that can be placed on a scale of high or low depending on their frequencies. Chords result from the simultaneous sounding of three or more pitches, and keys consist of combinations of seven or more pitches that form the tonal framework for music compositions.

Pitches. The outermost spiral represents pitch classes as spatial points in three-dimensional space. Pitches separated by an *octave* (eight scale steps) are assumed to be equivalent and map to the same position; pitches a *perfect fifth* (five scale steps with the lower pitch as starting note of scale) apart are neighbors a quarter turn of a spiral apart, and vertically aligned pitches are a *major third* (three major scale steps with the lower pitch as starting note of scale) apart. Pitch representations, indexed by their number of perfect fifth steps from C are defined by

$$\mathbf{P}(k) \stackrel{def}{=} \begin{bmatrix} x_k \\ y_k \\ z_k \end{bmatrix} = \begin{bmatrix} r \sin \frac{k\pi}{2} \\ r \cos \frac{k\pi}{2} \\ kh \end{bmatrix}.$$

For example, C is represented by the point $[0, r, 0]^T$. The pitch F, which is a perfect fifth below C, is represented by the point $[-r, 0, -h]^T$. The left-most spiral in Figure 1 shows the configuration of pitch classes on the pitch class spiral.

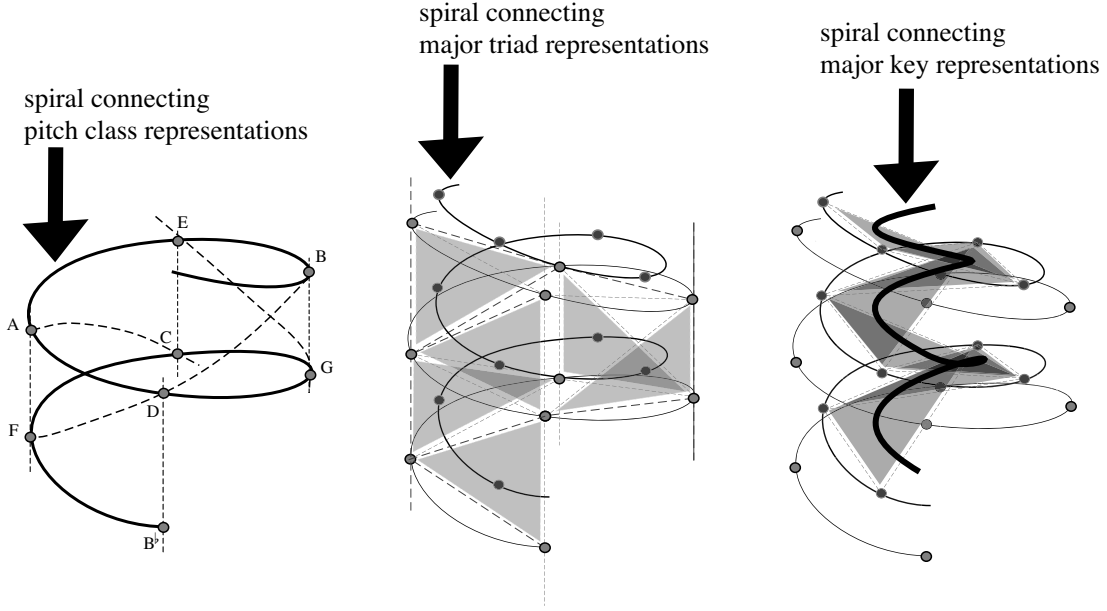


Figure 1: The Array of Spirals in the Spiral Array

Chords. The most basic chords are the major and minor triads. An *interval* is a measure of distance along a musical scale. A major triad consists of pitches separated by intervals of perfect fifth and major third with reference to a *root* pitch, which maps to a compact cluster of pitches forming a triangle with its base along the pitch class spiral. A minor triad consists of pitches separated by intervals of perfect fifth and minor third with reference to the root, which maps to the triangle that is the mirror inverse of the major triad. Each triad is represented by a spatial point inside its defining triangle. The mathematical definitions of the major and minor triads, respectively, are:

$$\mathbf{C}_M(k) \stackrel{def}{=} w_1\mathbf{P}(k) + w_2\mathbf{P}(k+1) + w_3\mathbf{P}(k+4),$$

where $w_1 \geq w_2 \geq w_3 > 0$ and $\sum_{i=1}^3 w_i = 1$.

$$\mathbf{C}_m(k) \stackrel{def}{=} u_1\mathbf{P}(k) + u_2\mathbf{P}(k+1) + u_3\mathbf{P}(k-3),$$

where $u_1 \geq u_2 \geq u_3 > 0$ and $\sum_{i=1}^3 u_i = 1$.

The double helix in the center of Figure 1 shows the outer pitch class spiral, the triangles covering the major triad pitches and the inner major triad spiral, a helical configuration of spatial points representing major triads. Each major triad representation resides in the interior of the triangle outlined by its component pitches.

Keys. The union of pitch classes in three adjacent major triads in the spiral array uniquely define the pitch set of a major scale. The root of the central triad in the spatial model is the *tonic* of the major key. The parallel minor key definition uses the minor form of the tonic triad, and combinations of the major and minor versions of the neighboring triads. The mathematical definitions are as follows:

$$\mathbf{T}_M(k) \stackrel{def}{=} \omega_1 \mathbf{C}_M(k) + \omega_2 \mathbf{C}_M(k+1) + \omega_3 \mathbf{C}_M(k-1),$$

where $\omega_1 \geq \omega_2 \geq \omega_3 > 0$ and $\sum_{i=1}^3 \omega_i = 1$.

$$\mathbf{T}_m(k) \stackrel{def}{=} v_1 \mathbf{C}_m(k)$$

$$+ v_2 [\alpha \mathbf{C}_M(k+1) + (1-\alpha) \mathbf{C}_m(k+1)]$$

$$+ v_3 [\beta \mathbf{C}_m(k-1) + (1-\beta) \mathbf{C}_M(k-1)],$$

where $v_1 \geq v_2 \geq v_3 > 0$ and $v_1 + v_2 + v_3 = 1$,

and $0 \geq \alpha \geq 1, 0 \geq \beta \geq 1$.

The bottom right triple helix in Figure 1 shows the pitch class spiral, the major triad spiral, and the generation of the major key spiral, each major key representation being the convex combination of three adjacent major triads. Each set of three adjacent major triads forms the vertices of one triangle in the diagram. The representation of a major key resides in the interior of its defining triangle.

2.2. A Constraint-Satisfaction Problem

One of the design criteria of the spiral array space is that tonal entities that are perceived to be close (according to the conventions of western harmony) are spatially near one to another. The spatial configuration of pitch classes in the spiral array ensures partial satisfaction of this criterion. The spiral array uses constraints on the spatial distance among pitches to calibrate distances in the model. The task of determining appropriate parameters becomes one of constraint satisfaction over a nonlinear system of equations.

The selection of an appropriate aspect ratio for the pitch class spiral presents a straightforward illustration of the proximity principle. The following paragraphs describe one set of criteria governing the distances between pairs of pitch class representations. First, a quick review of terminology pertaining to distance between pitch classes. Aural distance between

pitches is measured by their number of steps apart on a major scale, with the lower pitch acting as the first pitch in the scale. Major intervals decreased by a half step become *minor intervals*. Fifths and octaves are considered *perfect intervals*. Major and augmented intervals increased by a half step become *augmented intervals*. Minor intervals decreased by a half step become *diminished intervals*.

In the spiral array, pitches that are a perfect fifth apart (or a perfect fourth, the same interval inverted) should be at least as close as pitches that are a major third (or a minor sixth) apart. Those a major third (or a minor sixth) apart should be at least as close as those a minor third (or a major sixth) apart, which should be at least as close as those a major second (or minor seventh) apart, which should be at least as close as those a minor second (or major seventh) apart, which should be at least as close as those a diminished fifth (or augmented fourth) apart. These conditions map to the mathematical constraints:

$$\begin{aligned} \|\mathbf{P}(k) - \mathbf{P}(k+1)\| &\leq \|\mathbf{P}(k) - \mathbf{P}(k+4)\|, \\ \|\mathbf{P}(k) - \mathbf{P}(k+4)\| &\leq \|\mathbf{P}(k) - \mathbf{P}(k-3)\|, \\ \|\mathbf{P}(k) - \mathbf{P}(k-3)\| &\leq \|\mathbf{P}(k) - \mathbf{P}(k+2)\|, \\ \|\mathbf{P}(k) - \mathbf{P}(k+2)\| &\leq \|\mathbf{P}(k) - \mathbf{P}(k+5)\|, \\ \|\mathbf{P}(k) - \mathbf{P}(k+5)\| &\leq \|\mathbf{P}(k) - \mathbf{P}(k+6)\|. \end{aligned}$$

This example has a closed-form solution given by

$$\sqrt{\frac{2}{15}} \leq \frac{h}{r} \leq \sqrt{\frac{2}{7}}.$$

The computational experiments presented in the latter part of this paper use $r = 1$ and $h = \sqrt{2/15}$.

Other conditions can involve more than one level of description in the spiral array. A condition involving pitches and triads is that the root of a major triad should be at least as close to the triad representation as the fifth, which in turn should be at least as close to the triad representation as the third:

$$\begin{aligned} \|\mathbf{C}_M(k) - \mathbf{P}(k)\| &\leq \|\mathbf{C}_M(k) - \mathbf{P}(k+1)\|, \\ \|\mathbf{C}_M(k) - \mathbf{P}(k+1)\| &\leq \|\mathbf{C}_M(k) - \mathbf{P}(k+4)\|. \end{aligned}$$

A condition involving pitches and keys is that any pitch should be closest to the major key representation having the same name:

$$\arg \min_{T \in \mathbf{T}} \|T - \mathbf{P}(k)\| = \mathbf{T}_M(k).$$

A condition involving pitches and keys is that the average of two pitches a minor second apart (a point representing the center of the interval) should be closest to the key representation of the higher pitch:

$$\arg \min_{T \in \mathbf{T}} \|T - \frac{1}{2}(\mathbf{P}(k) - \mathbf{P}(k + 5))\| = \mathbf{T}_M(k), \text{ and}$$

the average of two pitches a perfect fourth apart should be closest to the key representation of the higher pitch:

$$\arg \min_{T \in \mathbf{T}} \|T - \frac{1}{2}(\mathbf{P}(k) - \mathbf{P}(k + 1))\| = \mathbf{T}_M(k).$$

The challenges of finding parameter values that satisfy all the desired conditions are twofold; namely, the distance relations among pitches and minor keys and triads are not as clear-cut as that for major keys and triads, and that closed-form solutions do not always exist. Numerical solutions are derived using a flip-flop heuristic, described in Chew (2000), for streamlined versions of the system (for example, with the major and minor triad weights constrained to be the same). Finding solutions for the complete system and determining the appropriate constraints for minor keys and triads (if any) remains an open problem.

2.3. The Center of Effect

A major advantage of a spatial model is that any collection of notes can be summarized using a c.e., a convex combination of its pitch classes. Each pitch position can be weighted by the total duration of that pitch class, or by the sum of the metric weights of the corresponding notes, or by some combination of duration and metric weight. The following definitions show the case when only the duration is used. If

- n_j = number of active pitches in the j th time period,
- $\mathbf{p}_{i,j}$ = the spiral array position of the i th pitch in the j th time period, and
- $d_{i,j}$ = the duration of the i th pitch in the j th time period,

we define $\mathbf{c}_{a,b}$ to be the c.e. of all pitch events in the window from time units a through b :

$$\mathbf{c}_{a,b} \stackrel{def}{=} \sum_{j=a}^b \sum_{i=1}^{n_j} \frac{d_{i,j}}{D_{a,b}} \mathbf{p}_{i,j}, \quad \text{where } D_{a,b} = \sum_{j=a}^b \sum_{i=1}^{n_j} d_{i,j}.$$

Figure 2 demonstrates the generation of a c.e. for a segment of music between times a and b . The figure also outlines the convex hull of the pitch classes in the musical segment. The

concept of the center of effect is fundamental to the computational approaches to problems in music perception and cognition described in the remainder of this paper. For example, the c.e. has been shown to be an effective tool for determining the tonal context of a segment of music.

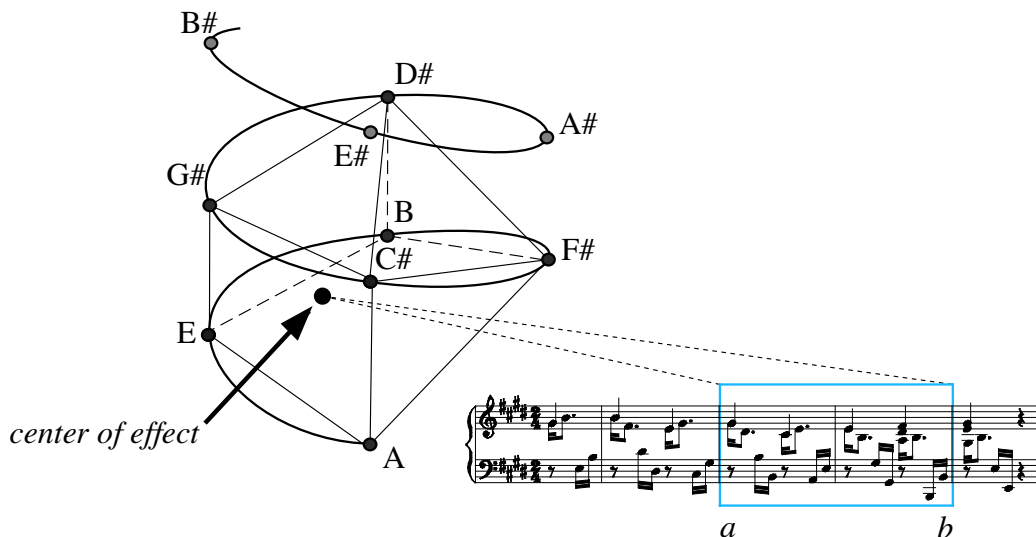


Figure 2: The Center of Effect

3. Tonal Induction and Pitch Spelling

The spiral array allows numerous problems in music perception and cognition to be modeled in quantitative ways that lend themselves readily to the design of computational solutions. This section outlines some computational solutions to problems in automated music analysis; in particular, the examples focus on the extraction of context and pitch structure.

3.1. Tonal Induction

A method, the *CEG* (center of effect generator) algorithm, was proposed in Chew (2000, 2001) for using the spiral array to compute the key context of a melody (*monophonic* music). As its name suggests, the CEG algorithm first generates a c.e. from a given segment of music, then determines the key context through a nearest neighbor search in the spiral array space for the closest key representation. In this section, I describe and present the generalized CEG algorithm using the c.e. defined in Section 2.3.

Key finding is the problem of determining the most stable pitch, the one that, when heard at the end of a phrase, makes the phrase sound complete, like it could end on this pitch. This most stable pitch is often referred to as the *tonal center* or “*doh*.” Apart from modeling music cognition, computational models for key finding are necessary for automated transcription, music information retrieval, and expression synthesis. Further details of the applications will be given in Section 6.2.

In Chew (2001), the c.e. was shown to gravitate quickly to the key representation of its tonal context as information accrues, and the CEG algorithm was shown to ascertain the correct key in fewer notes than existing methods when tested on the fugue subjects of Bach’s *Well-Tempered Clavier*, Book I. On average, the CEG algorithm required 3.75 notes to determine the correct key context, compared to Krumhansl and Schmuckler’s probe tone profile method (see Krumhansl 1990) which required 5.25 notes, and Longuet-Higgins and Steedman’s (1971) shape matching algorithm, which required 8.71 notes.

Because of the low density of notes in this data set, a cumulative window was used, i.e., the c.e. is calculated for $a = 0$ and $b = t$ at each point in time t . The tonal context of the melody is given by $T_{0,t} = \arg \min_{T \in \mathbf{T}} \|T - \mathbf{c}_{0,t}\|$, where \mathbf{T} is the set of all key representations, major and minor. In a melody, the number of notes at any given time j , $n_j \in \{0, 1\} \forall j$. The same formula for determining the key holds true for *polyphonic* (having multiple independent voices or instruments) or *homophonic* (single voice with accompaniment) music, where n_j can take on values greater than 1. The general formula for determining the tonal context of a segment of music between time a and b is $T_{a,b} = \arg \min_{T \in \mathbf{T}} \|T - \mathbf{c}_{a,b}\|$.

Figure 3 illustrates the technique for determining the key of a segment of music in $[a,b]$. As in Figure 2, the segment $[a,b]$ generates a c.e. $\mathbf{c}_{a,b}$ in the interior of the pitch class spiral. Using this c.e., the key context is identified through a nearest neighbor search for the closest key representation. For clarity, only the major key spiral is portrayed in the figure. In actuality, the minor key spiral is also taken into consideration.

The general form of the CEG algorithm allows a window of any size to be used to define the tonal context. The pitch-spelling algorithms in the next section use the idea of the sliding-window to approximate a local tonal context. As will be explained in the next section, the sliding window technique shows sensitivity to local changes, but lacks the ability to find sharp boundaries between key changes. The next section offers strategies for using local windows to generate a context-defining c.e. The search for segmentation boundaries between tonal contexts is the focus of Section 4.

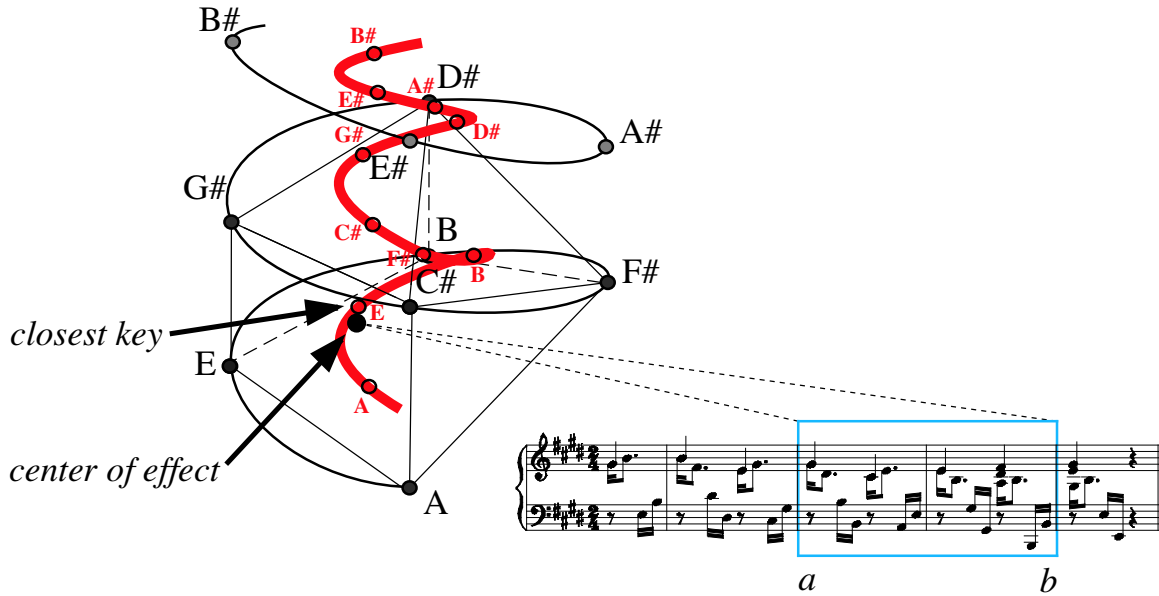


Figure 3: The CEG Algorithm: Nearest-Neighbor Search to Determine Key Context

3.2. Pitch Spelling

This section presents a bootstrapping pitch-spelling algorithm and its variants. An important concept in these algorithms is the determining of contextual windows for approximating the tonal context. In the previous section, the c.e. generated by a musical sample was shown to gravitate quickly to the key representation of the piece. As a result, the c.e. is used as a proxy for the key context in the pitch-spelling algorithm described in this section. Since the actual key does not need to be determined explicitly, we decouple the problems of key finding and pitch spelling.

The problem of pitch spelling arises because music notation has evolved in such a way as to encode not only the pitch height but also its tonal context and, less often, its voice-leading tendencies. Numeric representations of musical pitch, such as MIDI or pitch class numbers, are related only to the pitch height and not the context. For example, MIDI number 68 could map to $G\sharp$ or A_b . Pitch spelling is a necessary part of any automatic transcription (MIDI or audio-to-music notation) system. It is also a critical pre-processing step in key-finding algorithms. More accurate spelling leads naturally to more accurate tonal induction.

In Chew and Chen (2003ab, 2005), several algorithms were proposed that use the spiral array in pitch spelling. All three algorithms use the c.e. as a proxy for the key context and assign pitch names by a nearest neighbor search for the closest pitch class representation.

Figure 4 illustrates the pitch-name assignment process. In the example given, the pitch-name assignments are batched into beat-sized chunks. The spelled notes in the contextual window, covering the previous four beats, generate a c.e. Using this contextual c.e., each new spelling assignment is made through a nearest neighbor search on the pitch class spiral. In Figure 4, the next note to be assigned a name is MIDI number 68, which maps to $G\sharp$ or $A\flat$ as shown. The selected spelling is $G\sharp$, the option nearest the c.e.

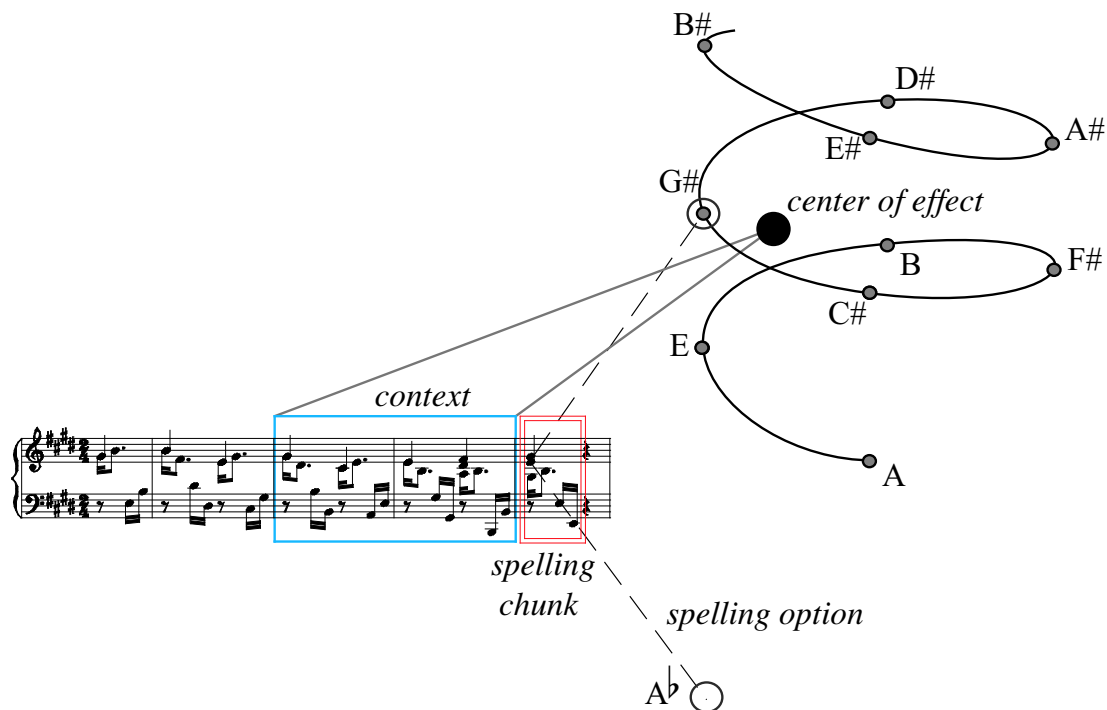


Figure 4: Pitch-Name Assignment Using the Spiral Array

The pitch-spelling algorithms differ only in the definition of the contextual window. Here, we show that all three variations can be modeled as instances of a general bootstrapping process. The bootstrapping process consists of a first phase that assigns pitch names to the spelling chunk (this can be viewed as a feasible but not-necessarily-optimal solution). The second phase revisits the spelling using a smaller local context window (allowing sudden contextual changes) in combination with a cumulative context window (so as to constrain the sudden changes to be within reasonable range of the global context). The formal description is as follows: at time j , where $0 \leq f \leq 1$,

Phase I. Define a sliding local window of size w_s . Compute a c.e. $\mathbf{c}'_j = \mathbf{c}_{j-w_s, j-1}$ and assign pitch names using \mathbf{c}'_j .

Phase II. Define a local window (including the new chunk) of size w_r . Compute a hybrid c.e. $\mathbf{c}_j'' = f\mathbf{c}_{j-w_r+1,j} + (1-f)\mathbf{c}_{1,j-1}$. Re-assign pitch names using \mathbf{c}_j'' .

The initialization procedure involves assigning pitch names with indices closest to 2 (ensuring the simplest spellings with the fewest number of sharps and flats), generating a c.e. with this first assignment, then revisiting the name assignments to make sure they are consistent with the contextual c.e.

Qualitatively, f indicates the relative importance placed on the local versus the global (approximated by the cumulative window) contexts. In general, the best values for f lie above 0.5 and close to 1. Note that a cumulative single-phase window algorithm or a sliding-window single-phase algorithm are special instances of the general bootstrapping algorithm. One gets the cumulative window algorithm when $w_s = 0$ and $f = 0$, and the sliding window algorithm when $w_r = 0$ and $f = 1$.

Phase I: *local context I* *spell*

Phase II: *local context II* *cumulative context* *spell again*

Figure 5: Two-Phase Assignment Method ($w_s = 4, w_r = 2$)

When tested on a moderately difficult data set, Beethoven’s Piano Sonata Op.79, third movement, the cumulative-window and sliding-window algorithms both had only one spelling error out of 1375 notes (99.93% correct rate). When tested on an extremely difficult data set, the first movement of Beethoven’s late Piano Sonata Op.109, having a total of 1516 notes, the cumulative window algorithm had 73 errors (95.18% correct), the sliding window 31 (97.96% correct, with $w_s = 4$), and the bootstrapping algorithm had 27 errors (98.22% correct, with $(w_s, w_r, f) \in \{(4,3,0.8), (4,3,0.7), (8,6,0.9)\}$). Further evaluations are underway to test the pitch-spelling algorithm on larger data sets against other algorithms.

4. Tonal Segmentation

Accurate analyses result from the ability to segment a piece into contextually similar sections. This section describes algorithms for tonal segmentation. Section 4.1 describes a static method for tonal segmentation that requires the number of boundaries to be pre-specified. Next, an $O(n)$ algorithm using a pair of sliding windows is proposed for tonal segmentation. The method, which computes discrepancy values between the forward and back window c.e.'s in real-time, is described in Section 4.2 and demonstrated using Edward McDowell's (1896) "To A Wild Rose," the first piece in his "Woodland Sketches," Op.51. The highest peak of each island of points above the thresholds is shown to correspond to a segmentation boundary. The computational results for McDowell's piece are presented in Section 4.3. The algorithm is further tested on two pieces by Franz Schubert (1828). Application of the segmentation algorithm to Schubert's *Allegretto* from Moment Musical D780 No.6 is discussed in Section 5.1, and that to his *Thema* from Impromptu D935 No.4 is presented in Section 5.2.

A challenge of verifying segmentation algorithms is that even musical experts may not agree on the "correct" segmentation. Other researchers compared their results to that of music theorists (see Shmulevich and Yli-Harja 2000, and Temperley 1999.) Each discussion of the results will be preceded by a pre-computation tonal analysis of the example. For the McDowell and the second Schubert example, I analyzed the piece manually to create one plausible segmentation of the piece before running it through the algorithm. In the first Schubert example, I used the key changes indicated by key signature change boundaries in the score.

4.1. Boundary Search Algorithm

A solution process for determining key boundaries (commonly known as points of *modulation*) using the spiral array was proposed in Chew (2002). The insight behind the proposed approach is that, when given the best segmentation, each segment consists of the notes that best reflect their tonal context. These notes would then generate a c.e. whose distance to their closest key representation is at a minimum.

Figure 6 illustrates the concept behind the boundary search algorithm using McDowell's "To A Wild Rose." Music segmentation is far from an exact science, and this ambiguous piece has more than one plausible segmentation. The piece is in the key of A major, with a

brief interlude that strongly suggests E major but never fully committing to this neighboring key. One interpretation is to hear bars 1-20 as being in the key of E, bars 21-24 (shaded gray in the figure) as being in the key of E major, and bars 25-51 as being in the key of A. In the middle section, the D \sharp 's in the left hand in bars 22 and 24 act as notes leading to E, and the right-hand motif can be heard as a direct transposition of the bar 19 right-hand motif to the new key. Bars 25 through 28 then serve as a preparation for the return of the main theme and A major key.

Assuming that the proposed segmentation is a correct one, the segments of music (bars 1-20, 21-24, and 25-51) will generate c.e.'s that are as close as possible to their key representations, thereby minimizing the sum of the c.e.'s distances to their closest key representations. In Figure 6, these distances from c.e.'s to closest key representations are highlighted by thick lines connecting each pair of entities. When the boundaries divide the segments so that they best match the actual key boundaries, these distances marked by the thick lines will be minimized. One can of course contrive pathological (and atonal) examples where minimal c.e.-to-key distances do not correspond to key segmentations. However, tonal music is well behaved and the spiral array exploits the properties of tonal music so that members of the same key will cluster in space.

The process of determining the segmentation boundaries in this way can be modeled mathematically. Suppose that m boundaries have been chosen, given by (B_1, \dots, B_m) that divide the passage into $m + 1$ time epochs, and let time 0 (the beginning of the piece) be B_0 and the end of the passage be $B_{m+1} = L$. Let the distance from $\mathbf{c}_{B_i, B_{i+1}}$ to the most likely key be $d_{B_i, B_{i+1}}^{\min}$. The best boundaries will then minimize the distances $d_{B_i, B_{i+1}}^{\min}, i = 0, \dots, m,$. In addition, common sense requires that neighboring keys be different from each other, which can be modeled as a constraint. The optimal boundaries are given by

$$\begin{aligned}
\min \quad & \sum_{i=0}^m d_{B_i, B_{i+1}}^{\min} \\
\text{s.t.} \quad & d_{B_i, B_{i+1}}^{\min} = \min_{T \in \mathbf{T}} \|\mathbf{c}_{B_i, B_{i+1}} - T\|, \quad i = 0, \dots, m, \\
& \arg \min_{T \in \mathbf{T}} \|\mathbf{c}_{B_{i-1}, B_i} - T\| \neq \arg \min_{T \in \mathbf{T}} \|\mathbf{c}_{B_i, B_{i+1}} - T\|, \quad i = 1, \dots, m - 1, \\
\text{and} \quad & B_i < B_{i+1}, \quad i = 0, \dots, m
\end{aligned}$$

Assuming that the given piece has n notes, then the worst-case computational complexity of this approach is $O(n^m)$. In practice, the number of key changes is typically small in relation

to the number of notes ($m \ll n$). In smaller-scale compositions, the pieces can most often be segmented into three key areas. In Chew (2002), this boundary search algorithm was demonstrated to perform well on two examples from Bach’s “A Little Notebook for Anna Magdalena.”

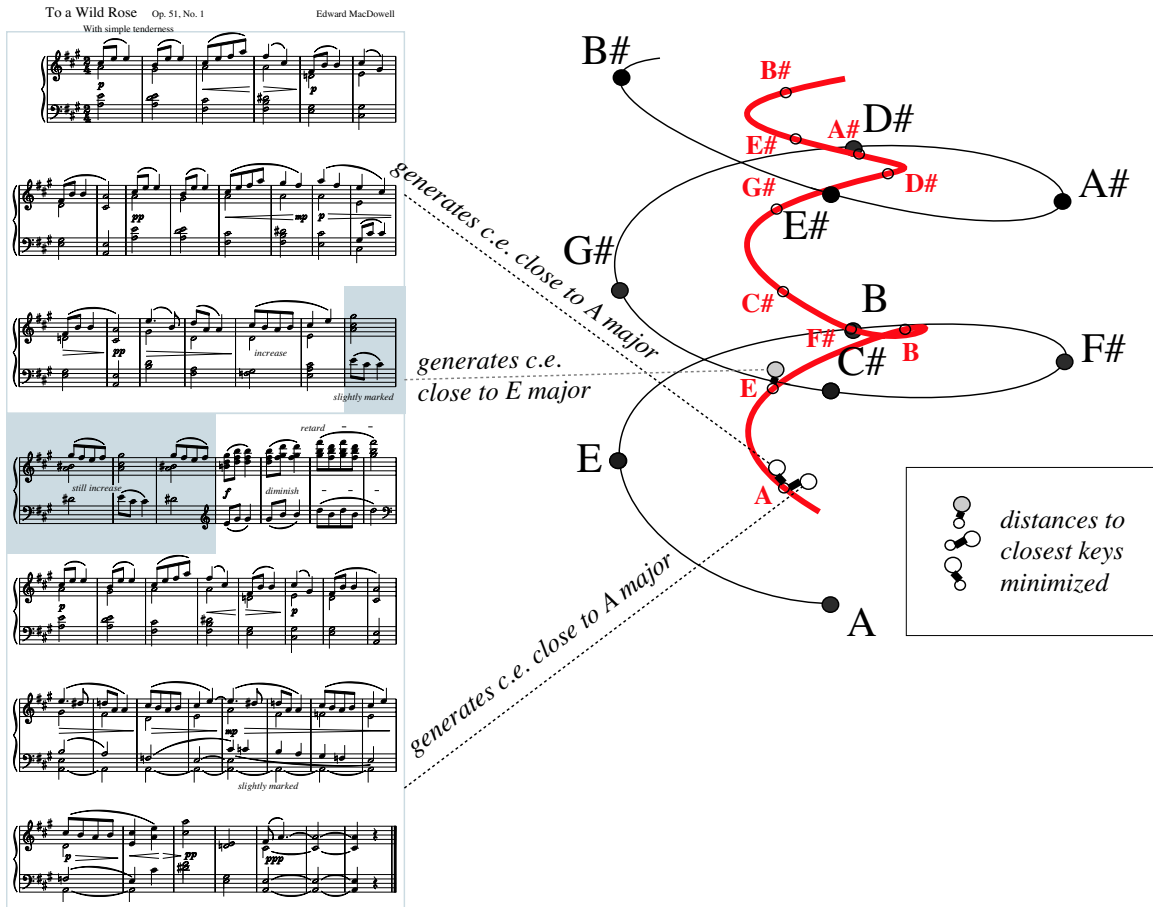


Figure 6: Insight to Off-Line Segmentation Procedure

There are two disadvantages to this boundary search approach. First, it is a static algorithm that requires data from the entire piece to be known. Second, the algorithm requires that the number of segments be known. Trying all possible m 's is a plausible but not particularly elegant solution to the second problem. The next section proposes and presents results from a real-time segmentation algorithm.

4.2. Segmentation in Real Time: the Argus Approach

This section presents an $O(n)$ algorithm for segmentation inspired by the results of previous research on tonal induction and boundary search algorithms. The main idea behind this approach is to keep track of both the c.e. of the music ahead as well as the c.e. of the music that has passed. I call this the *Argus* approach to segmentation. (Argus is the watchman of Greek mythology who, according to some accounts, has two eyes in front and two in the back.) The advantage of a real-time algorithm is that the algorithm can naturally follow and analyze the music as it unfolds over time.

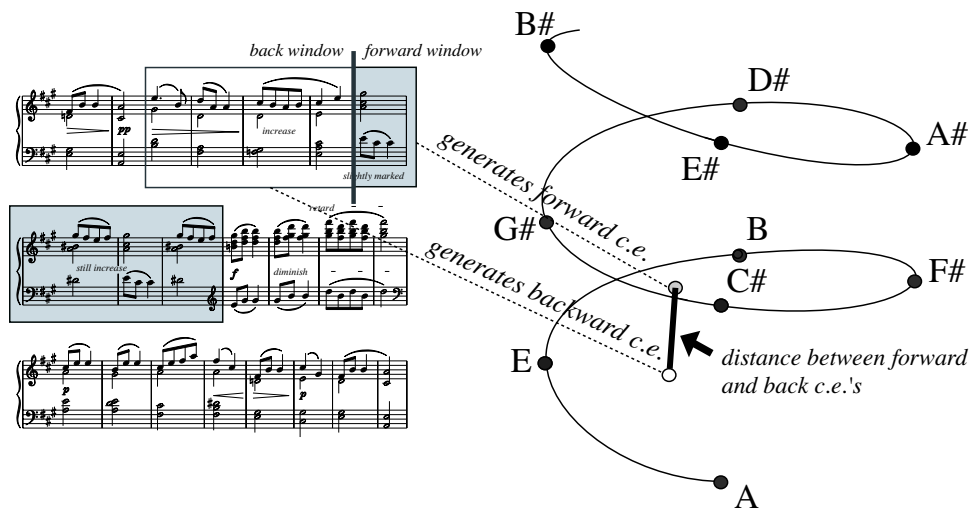


Figure 7: Measuring the Distance Between the Forward and Back c.e.'s

The insight behind the real-time segmentation algorithm is that at any point that can be considered a tonal boundary, the music before and after the boundary would generate c.e.'s that are far away from each other. Figure 7 depicts the generating of the forward window (indicated by the gray box) c.e. and back window (signified by the clear box) c.e. and the span of the distance between them. The forward and back windows, in this case, are each four bars or sixteen eighth notes long. The distance between the forward and back c.e.'s reaches a peak value at each segmentation boundary as pictorially in Figure 8, thus providing a way to identify these boundaries.

As with the experiments with pitch-spelling algorithms, the size of the contextual window affects the accuracy of the results. The experiments in the next section use McDowell's "To A Wild Rose" to illustrate and validate this new approach using a variety of window sizes.

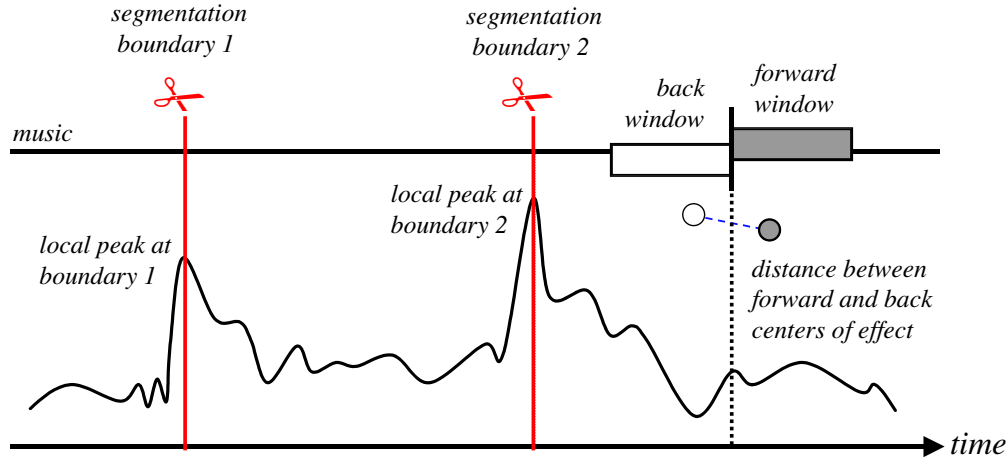


Figure 8: Insight: Local Maxima Correspond to Segmentation Boundaries

4.3. Computational Results

The piece was encoded from the score in text format such that, at each point in time, a list of active pitches is given. To calculate the c.e. between a window $[a, b]$, each pitch at each point in time is mapped to its spiral array position and the positions are averaged over all pitches in all time slices in the given interval. The algorithm requires the following parameters to be specified: the forward window size w_f , and the back window size w_b . At any point in time b , the algorithm calculates the c.e. of a forward window $[b, b + w_f)$ and a back window $[b - w_b, b)$. The distance between the two c.e.'s is calculated as b advances in time.

The distance values are charted in Figure 9 for window sizes $w_f = w_b = 8, 12, 16, 20,$ and 24 , which correspond to 2, 3, 4, 5, and 6 bars of music. The horizontal axes mark time according to the number of eighth notes (the smallest time unit in the piece), while the vertical axes measure the distance between each pair of forward and back window c.e.'s. The vertical grid lines correspond to barlines in the music. Table 1 provides further statistics on the distance values corresponding to the charts in Figure 9. Each chart in Figure 9 also shows the 95% and the 97.5% threshold for the distances as thick dotted horizontal lines. Each peak above the 97.5% threshold is marked by a thick solid vertical line, while each peak that lies between the 95% and 97.5% threshold is marked by a thick dotted vertical line.

Compare the charts with the segmentation shown in Figure 6. Recall the human analysis of the piece outlined in Section 4.1. The analysis segments the composition into three sections according to key regions as shown in Figure 6. This manual analysis was completed before

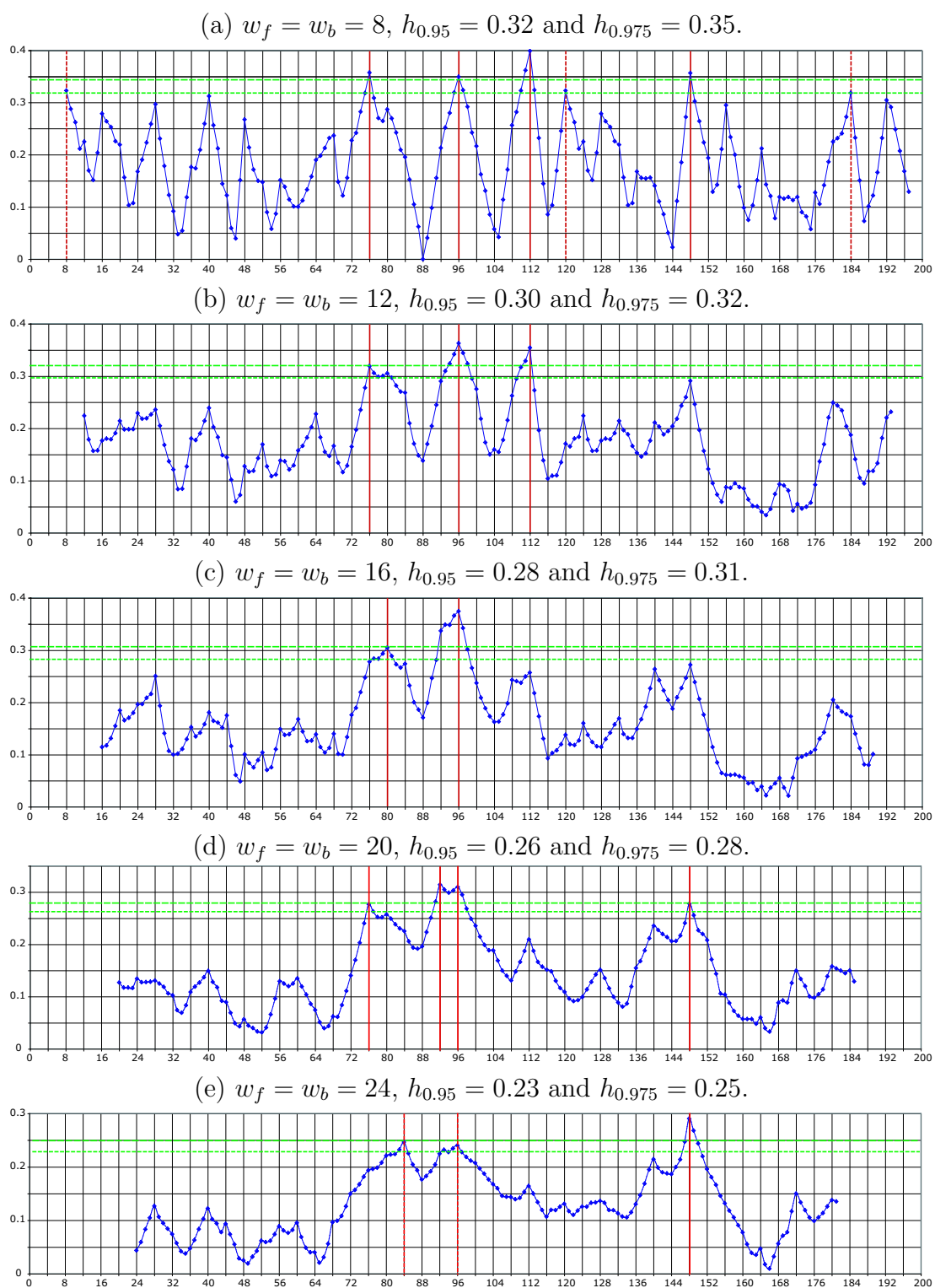


Figure 9: Charting Tonal Discrepancies in McDowell's "To A Wild Rose"

Table 1: Tonal Discrepancy Distance Statistics

<i>forward</i> <i>window, w_f</i>	<i>back</i> <i>window, w_b</i>	<i>average</i> <i>distance, μ_d</i>	<i>stddev,</i> <i>σ_d</i>	<i>95% thresh, $h_{.95}$</i> <i>$= \mu_d + 1.645\sigma_d$</i>	<i>97.5% thresh, $h_{.975}$</i> <i>$= \mu_d + 1.96\sigma_d$</i>
8	8	0.1859	0.0813	0.3197	0.3453
12	12	0.1769	0.0733	0.2975	0.3206
16	16	0.1610	0.0751	0.2846	0.3083
20	20	0.1446	0.0695	0.2590	0.2809
24	24	0.1293	0.0633	0.2335	0.2534

the computational experiments took place; the boundaries dividing the central E major area from the neighboring A major regions occur at times 80 and 96 (counting eighth notes, the same time units as the charts in Figure 9). This is precisely the answer shown in Figure 9(c), obtained by setting $w_f = w_b = 16$. At both peaks, the distance between the forward and back c.e.’s both exceed the 97.5% threshold, $h_{0.975}$. The boundary at 80 is also shown in Figure 7; the boundary at 96 is at the rightmost edge of the forward (gray) window in the same figure.

The distance values in Figure 9(a), charting the results for window sizes $w_f = w_b = 8$, shows many more pronounced peaks than the other plots. This is because the small window size tracks local changes over a small window of time. At this myopic level of description, the c.e. distance exceeds $h_{0.975}$ at times 76, 96, 112, and 148. McDowell plays with the boundary between E major and A major throughout the piece. There is only one pitch difference between A major and E major: A major has a D natural while E major has a D sharp. This short-range push and pull between A major and E major finally appears to lean strongly toward E major at time 80 (bar 21), only to be sharply pulled away to a nebulous diminished seventh chord after 96 (beginning bar 25), which resolves back to the A major at time 112 (bar 29).

At time 148, the D \sharp just before the boundary in bar 37 leads the algorithm to assume that there is a hint of E major in the back window, while the F \natural in bar 39 leads the algorithm to assume that the context is tending to a region with fewer sharps, thus resulting in the peak at 148. This is the first time the composer uses such chromatic motion in the piece. In both cases, the accidentals are the result of stepwise motion to fill in the space between the structural pitches so as to form smooth lines that the ear can follow. An analysis of the larger-scale organization of the piece shows that time 148 corresponds approximately to the

beginning of the coda (at 144, a new organization of the material, allowing the piece to draw to a close) after a re-statement of the theme prior to 144.

The result shown in Figure 9(b) ($w_f = w_b = 12$) is close to that of Figure 9(c) ($w_f = w_b = 16$). Like the previous experiment with parameters $w_f = w_b = 8$, the algorithm selected time 76 rather than 80 as the left boundary for the middle section. The note material between 76 and 80 can belong to both A or E major, hence, according to the algorithm’s criteria of finding contextual boundaries, this answer is not incorrect. Time 112 marks the return fully to A major. As the window sizes grow beyond 16, time 148 re-emerges as a boundary in Figures 9(d) and (e). The E major section around 80-96 remains delineated in both charts.

The Argus algorithm computes the tonal discrepancy values in real time. The setting of the control limits requires an initialization period, after which the boundaries can also be selected in real time. The accuracy of the algorithm is somewhat sensitive to the window size. In McDowell’s piece, the best performance was achieved using window sizes 12 and 16 eighth notes, which correspond musically to 3 and 4 bars respectively, or approximately half a phrase of music, or 4 to 5 seconds.

5. Additional Examples

It is important to demonstrate that the proposed algorithm works for more than one example. This section presents two further examples using selections from Franz Schubert’s (1828) Impromptus and Moments Musicaux. The first example is the *Allegretto* from his Moment Musical D780 No. 6, composed in 1823-1828. The second example is the *Thema* from the Impromptu D935 No. 3, composed in 1827. The computational results for the two examples are presented and analyzed in Sections 5.1 and 5.2 respectively. In each case, the Argus algorithm located the tonal-segmentation boundaries with zero error.

5.1. Schubert: *Allegretto* from Moment Musical D780 No. 6

Figure 10 shows the *Allegretto* from Schubert’s Moment Musical D780 No. 6. In this piece, the composer marks the main key changes by key-signature changes written into the music, such as the one in the middle of the fourth line of the score at which the previous four flats (for A-flat major) are naturalized and the key signature switches to four sharps (for E major). There are four such key-change boundaries marked in this selection, between bars 28 and 29 (middle of the fourth line of music), between bars 39 and 40 (end of the fifth line),

in the middle of bar 65 (middle of the next-to-last line of music), and between bars 73 and 74 (four bars from the end).

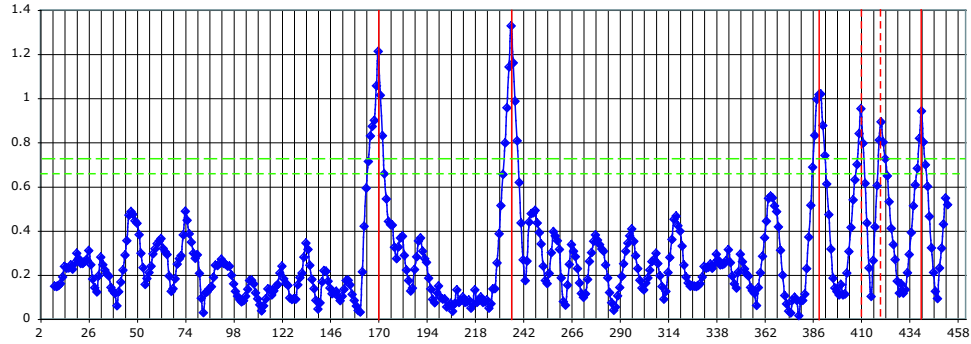


Figure 10: Schubert’s *Allegretto* from Moment Musical D780 No. 6

Figure 11 shows the computational results produced by the Argus algorithm. The horizontal axis marks time according to the number of eighth notes. After the first two lines of music, a repeat sign shows that the first two lines are to be repeated. A similar sign at the end of the excerpt shows that the third line through the end is to be repeated. The analysis in Figure 11 was performed on the excerpt without repeats. As in the previous section, the charts in Figure 11 show the time series for the distance between the forward and back window c.e.’s, this time for window sizes 9 and 18 eighth notes, i.e., 1.5 and 3 bars, respectively. Both charts show that the algorithm locates the key change boundaries exactly with zero error. In addition, the window-size-9 chart finds the local perturbation, a brief return to the flatted notes, embodied in the chords in bar 69 and the first two beats of bar 70. The algorithm’s boundaries corresponding to the main key changes are marked by solid vertical lines in Figures 10 and 11. Dotted lines delineate the boundaries of the local perturbation in Figures 10 and 11(a).

Note that, in Figure 11(b), the lowest point is reached at the 218th eighth note, at the boundary between bars 36 and 37, which is the point at which the preceding three bars are repeated almost verbatim in the subsequent three bars. At this point, the note material in the windows before and after are almost identical, resulting in a discrepancy value near zero.

(a) $w_f = w_b = 9$, $\mu_d = 0.2863$, $\sigma_d = 0.2256$, $h_{0.95} = 0.6574$ and $h_{0.975} = 0.7285$.



(b) $w_f = w_b = 18$, $\mu_d = 0.3193$, $\sigma_d = 0.2441$, $h_{0.95} = 0.7209$ and $h_{0.975} = 0.7978$.

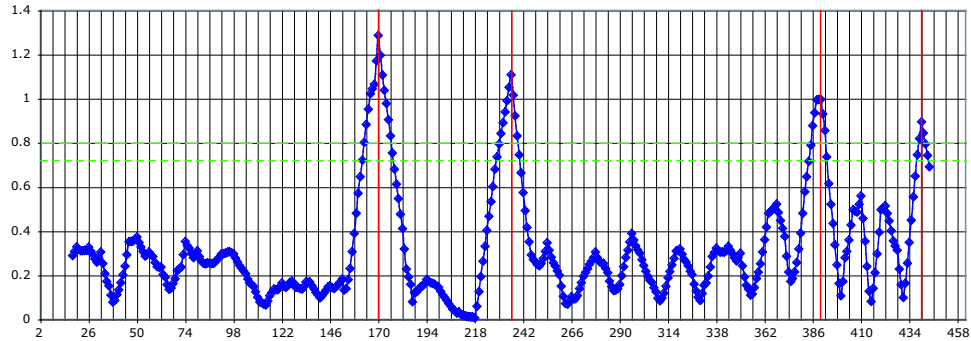


Figure 11: Tonal Discrepancies in Schubert’s *Allegretto* from Moment Musical D780 No. 6

5.2. Schubert: *Thema* from Impromptu D935 No. 3

This section examines another example by Schubert, the *Thema* from his “Impromptu” D935 No. 3, shown in Figure 12. The tonal boundaries in the previous example by Schubert had extreme changes that are readily apparent in its tonal discrepancy profile. The distance between a key with four flats and one with four sharps is a large one. As a result, the peaks in Figure 11 are well above the 97.5% threshold. The third impromptu from D935 exhibits a theme-and-variations structure, and its *Thema* modulates between neighboring keys related by a perfect fifth. In the *Thema*, the music modulates from B \flat major (having two flats in its key signature) to its neighboring key, F major (having one flat in its key signature). Figure 12 shows the *Thema* from this impromptu.

Now, for the manual analysis of this example. The selection is predominantly in the key of B \flat major. The gray (shaded) box marks the musical phrase that veers away from the B \flat major key context. The key changes are not marked by key-signature changes because of the slight difference between the key signatures (two flats versus one flat). Bars 1 through



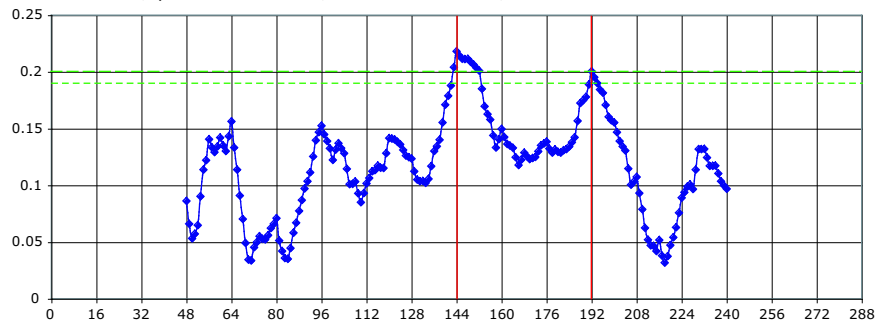
Figure 12: Schubert’s *Thema* from Impromptu D935 No. 3

8 are in the key of B \flat major. The first bar of music after the first repeat sign (bar 9) shifts towards the G minor chord, which belongs in both B \flat major and F major, beginning the transition away from the old and into the new key. The C major chord in the middle of the following bar is the first strong indicator that the new key could be F major, which is confirmed at the beginning of bar 11. The music from the middle of bar 10 through the end of bar 12 can most likely be heard in the key of F major. Technically, bars 9 and 10 constitute a “gray zone” in between the two main keys, and any notes from the second half of bar 8 through bar 12 can belong to the key of F major. B \flat major returns in bars 13 through 18. Counting in sixteenth notes, borders 128 and 152 mark the maximum span of the ambiguous zone between the B \flat major and F major key regions, and borders 136 and 192 define the maximum span of the F major section.

The segmentation boundaries are computed for input data without the musical repeats, as was the case with the moment musicaux example in Section 5.1. Figure 13 charts the computed tonal discrepancies over time (counting the number of sixteenth notes) for window sizes 48 and 64, which correspond to three and four bars respectively. For window size 48, strong boundaries emerge at 144 and 192. For window size 64, strong boundaries emerge at 152 and 192, with a third highest peak (below the thresholds) at 128. Solid vertical lines mark the algorithm’s choice of tonal boundaries for the 97.5% threshold in Figure 13, when

$w_f = w_b = 48$ and $w_f = w_b = 64$. The corresponding boundaries for window size 64 are marked in the score in Figure 12.

(a) $w_f = w_b = 48$, $\mu_d = 0.1188$, $\sigma_d = 0.0435$, $h_{0.95} = 0.1903$ and $h_{0.975} = 0.2040$.



(b) $w_f = w_b = 64$, $\mu_d = 0.1010$, $\sigma_d = 0.0273$, $h_{0.95} = 0.1458$ and $h_{0.975} = 0.1544$.

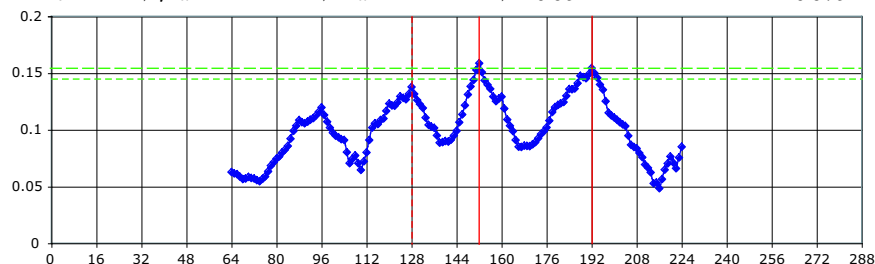


Figure 13: Tonal Discrepancies in Schubert’s *Thema* from Impromptu D935 No. 3

Using a window size of 64 sixteenth notes, i.e. four bars, the boundaries emerge at the middle of bar 10 (sixteenth note 152) and at the end of the gray box (sixteenth note 192.) A local and third highest peak at the beginning of bar 9 (sixteenth note 128) signifies the departure from the B \flat major context, even though the peak does not rise above the two thresholds.

Using a window size of 48 sixteenth notes, i.e. three bars, the algorithm selects boundaries at the barline between bars 9 and 10 (sixteenth note 152) and at the end of the gray box. One may debate where exactly in bars 9-10 the F major region begins; but, all the note material between the algorithm’s boundaries for window sizes 48 and 64 are irrefutably in the key of F major. Although the boundary choices at window size 48 (three bars) are also reasonable, a human listener is more likely to agree with the algorithm’s boundaries using window size 64 (four bars).

The fact that the algorithm achieved the best results for both the McDowell’s tonally ambiguous “To a Wild Rose” and Schubert’s *Thema* from Impromptu D935 No. 3 using

windows of size four bars may be due in part to the four-bar phrase lengths or to the four-bar length of the section in the foreign key.

6. Conclusions

The paper has presented a review of the spiral array model and its associated algorithms for tonal induction and approximation. An $O(n)$ algorithm for real-time segmentation was proposed, demonstrated using McDowell’s “To A Wild Rose” and tested using Schubert’s *Allegretto* from Moment Musical D780 No. 6 and *Thema* from Impromptu D935 No. 3. The proposed Argus algorithm has computational complexity $O(n)$, which lends itself well to testing on large databases.

6.1. Discussion on the Argus Algorithm

Preliminary tests have demonstrated that the Argus algorithm works best when the positions of the forward window pitch classes in the spiral array are distant from those of the back window, as was the case in Schubert’s *Allegretto* from Moment Musical D780 No.6. The algorithm also performs well when the boundary separates sections in distinct keys, as in Schubert’s *Thema* from Impromptu D935 No. 3. Even in the nebulous McDowell example, the algorithm produced reasonable answers; like Schubert’s *Thema*, the distinction in this case is also between neighboring keys related by a perfect fifth. The testing of the Argus algorithm on other works from a variety of different styles and against other algorithms (such as Temperley’s 1999) will be necessary to understand better the strengths and weaknesses of the approach.

Knowing the pitch names of the notes is an advantage in determining the tonal context and segmentation boundaries. This advantage is especially apparent in the Schubert D780 example, where the pitches are notated using names from distant keys. To make the experiments begin from the same input as humans, one can use numeric pitch information, such as in MIDI. In this case, pitch spelling would precede segmentation of the musical examples.

It is likely that the algorithm would not perform as well in distinguishing between relative major/minor modes, which share the same pitch set. And, as one reviewer points out, there is no distinction between genuinely harmonic notes and the passing notes; instead, the notes within the window are treated as a bag of undifferentiated unordered notes. In the McDowell example, the use of chromatic passing tones turned out to be a deviation from the diatonic

use of pitches in the rest of the piece that marked a point of structural significance. As this may not always be the case, future investigations can employ c.e. weights that quantify the “harmonicity” of the tones. Since harmonic tones tend to fall on strong beats, these weights can be correlated to the rhythmic profile of the examples.

The Argus approach calculates a tonal discrepancy between the forward and back windows at each instance in time. Apart from the application of finding tonal boundaries, another possible application is the quantification of tonal tension at each point in time. Every discrepancy value, not just the peaks, can be used to reveal information about the perceived distance between the pitch clusters forward and backward in time.

6.2. Other Applications

The potential applications of the tonal induction and segmentation algorithms include music visualization, automated accompaniment, music information retrieval, and expression synthesis. Accurate tonal analysis and segmentation is critical to content-based music information retrieval, i.e., retrieval based on the music itself rather than its title, composer, or performer(s). See the paper archives of the International Conference on Music Information Retrieval (<http://www.ismir.net>) for examples of recent work in this arena. When listening to music, we are cognizant of the pitch patterns relative one to another, and not the absolute pitches themselves. Tonal induction can be used to normalize a music database for comparison. Pitch spelling allows for more precise melody retrieval. Tonal segmentation can be used to index a database for faster retrieval.

Expert performance is the result of the manipulation of time structures (smooth displacements of beats and rhythms), dynamics (loudness), and articulation to present an expressive interpretation of the piece. The expressive gestures are most effective when they correspond to and exploit the expectations tied to the tonal structures of the piece. Hence, any system for expression synthesis will be greatly enhanced when coupled with accurate tonal-analysis tools. At a practical level, tonal analysis is essential in systems for musical improvisation. The ability to generate music in the same key and to know when the context changes is second nature to jazz musicians. Any successful computer system for improvisation requires a robust and real-time algorithms for tonal induction and segmentation.

Finally, any visualization of tonal music that is tied meaningfully to the content of the piece requires key analysis and segmentation. The geometry of the spiral array lends itself readily to visualization of tonal patterns. The integration of algorithms for tonal analysis

and segmentation into a real-time interactive music visualization system is part of a project called *MuSA.RT* – Music on the Spiral Array . Real-Time (Chew and François 2003).

Acknowledgments

The research has been funded by the Integrated Media Systems Center, a National Science Foundation Engineering Research Center, Cooperative Agreement EEC-9529152, and NSF grants ITR-0219912 and Career-0347988. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the author and do not necessarily reflect those of NSF.

References

- Cambouropoulos, E. 2003. Pitch spelling: a computational model. *Music Perception* **20** 411–430.
- Chew, E. 2000. Towards a mathematical model of tonality. Ph.D. thesis, Operations Research Center, Massachusetts Institute of Technology, Cambridge, MA.
- Chew, E. 2001. Modeling tonality: applications to music cognition. J. D. Moore, K. Steining, eds. 23rd Annual Meeting of the Cognitive Science Society, Edinburgh, Scotland, UK. Lawrence Erlbaum Associate Publishers, Mahwah, NJ/London. 206–211.
- Chew, E. 2002. The spiral array: an algorithm for determining key boundaries. C. Anagnostopoulou, M. Ferrand, A. Smail, eds. *Music and Artificial Intelligence*. Proc. 2nd Intl. Conf. LNCS/LNAI # 2445, Springer, Berlin, Germany. 18–31.
- Chew, E., Y.-C. Chen. 2003a. Mapping MIDI to the spiral array: disambiguating pitch spellings. H. K. Bhargava, N. Ye, eds. *Computational Modeling and Problem Solving in the Networked World*. Proc. 8th INFORMS Computer Society Conf. Kluwer, Dordrecht, the Netherlands. 259–275.
- Chew, E., Y.-C. Chen. 2003b. Determining context-defining windows: pitch spelling using the spiral array. Proc. 4th Intl. Conf. for Music Information Retrieval, Baltimore, MD.
- Chew, E., Y.-C. Chen. 2005. Real-time pitch spelling using the spiral array. *Computer Music Journal* **29** xx–xx.

- Chew, E., A. R. J. François. 2003. MuSA.RT – music on the spiral array . real-time. ACM Multimedia Conference 2003, Berkeley, CA. ACM Press, New York, NY. 448–449.
- Gomez, E. xxxx. Tonal description of polyphonic audio for music content processing. *INFORMS Journal on Computing* xx. xx–xx.
- Krumhansl, C. L. 1990. *Cognitive Foundations of Musical Pitch*. Oxford University Press, Oxford, UK.
- Longuet-Higgins, H. C. 1962. Two letters to a musical friend. *The Music Review* **23** 244–228, 271–280.
- Longuet-Higgins, H. C. 1976. Perception of melodies. *Nature* **263** 646–653.
- Longuet-Higgins, H. C., M. J. Steedman. 1971. On interpreting Bach. *Machine Intelligence* **6** 221–241.
- McDowell, E. 1896. To a wild rose. *Woodland Sketches* Op.51 No.1. EPS file typeset in SCORE by Craig Sapp. <http://ccrma-www.stanford.edu/~craig/score>.
- Meredith, D. 2003. Pitch spelling algorithms. Fifth Triennial ESCOM Conference, Hanover University of Music and Drama, Hanover, Germany. 204–207.
- Meredith, D. 2004. Comparing pitch spelling algorithms on a large corpus of tonal music. Proc. of Computer Music Modeling and Retrieval 2004, LNCS # 3310. Verlag, Berlin, Germany. 173–192.
- Pickens, J., J. P. Bello, G. Monti, M. Sandler, T. Crawford, M. Dovey, D. Byrd. 2003. Polyphonic score retrieval using polyphonic audio queries: a harmonic modeling approach. *Journal of New Music Research* **32** 223–236.
- Schubert, F. 1828. *Impromptus and moments musicaux*. G. Henle Verlag, Munich, Germany.
- Shmulevich, I., O. Yli-Harja. 2000. Localized key-finding: algorithms and applications. *Music Perception* **17** 531–544.
- Temperley, D. 1999. What’s key for key? The Krumhansl-Schmuckler key-finding algorithm reconsidered. *Music Perception* **17** 65–100.
- Temperley, D. 2001. *The Cognition of Basic Musical Structures*. MIT Press, Cambridge, MA.
- Toiviainen, P., C. L. Krumhansl. 2003. Measuring and modeling real-time responses to music: the dynamics of tonality induction. *Perception* **32** 741–766.