Musical Metacreation: Papers from the 2012 AIIDE Workshop
AAAI Technical Report WS-12-16

FreshJam: Suggesting Continuations of Melodic Fragments in a Specific Style

Tom Collins
Center for Mind and Brain
University of California, Davis, CA 95618
tecollins@ucdavis.edu

Christian Coulon
IET Application Development
University of California, Davis, CA 95616
cdcoulon@ucdavis.edu

Abstract

Imagine that a budding composer suffers from writer's block partway through devising a melody. A system called FreshJam is demonstrated, which offers a solution to this problem in the form of an interactive composition assistant; an algorithm that analyzes the notes composed so far, makes a comparison with an indexed corpus of existing music, and suggests a possible next note by choosing randomly among continuations of matched melody fragments. We provide a demonstration of FreshJam as an aid in stylistic composition, and of its potential to be more iterative than existing composition assistants such as PG Music's Band in a Box or Microsoft's Songsmith.

There are many algorithms that, with one button click, generate entire melodies or passages of music in the style of another composer or period (Conklin and Witten 1995; Cope 2005; Pearce and Wiggins 2007). These one-click algorithms can be used to test theories of musical style, but are of little honest use to students of stylistic composition. This seems unfortunate, as some of the algorithms work by generating one note/chord at a time, and so are eminently suited to form the basis of a system for interactive music composition. Below we describe such a system, called *FreshJam*, in which users are able to request continuations to melodic fragments in a specific style. Compared with other models for compositional continuation (Cope 1997; Maxwell et al. 2012), the principles behind FreshJam are simple and few, and there is an argument for considering and evaluating simpler models before developing more complex alternatives.

The term *FreshJam* refers to the piano-roll, web-based interface in which users compose a melody, and the term *composition assistant* refers to the functionality by which the user can request a continuation for their melody fragment (see www.projectfreshjam.com for a video demonstration). Figure 1 shows a schematic for the composition assistant. At the start of the flow diagram, the user has composed a fragment of n melody notes. Unsure how to proceed, they request a continuation from the composition assistant. If the user has not already selected a composer/style for the source of continuations, they are prompted to do so at this point. Some composers/styles are indicated in Fig. 1 in the box labeled *state transition tables* (preanalyzed melodies; details in next section). Users can switch between styles mid-melody if desired. Solid arrows in Fig. 1 indicate four completely autonomous steps that occur immediately (analysis, lookup, suggestion, and conversion). A representation based on beat of the measure and transposition invariance is used to lookup the state transition table, and the suggested next note is a random choice among continuations of matching representations (details in subsequent section). Although the suggested continuation is returned immediately, we envisage the current version of the assistant being used for composition rather than improvisation (Pachet 2002; Keller et al. 2012). At the end of the flow diagram in Fig. 1, dashed arrows indicate that the user auditions the $(n + 1)$ melody notes and decides whether to reject the suggested continuation, revise it, or accept and continue composing.

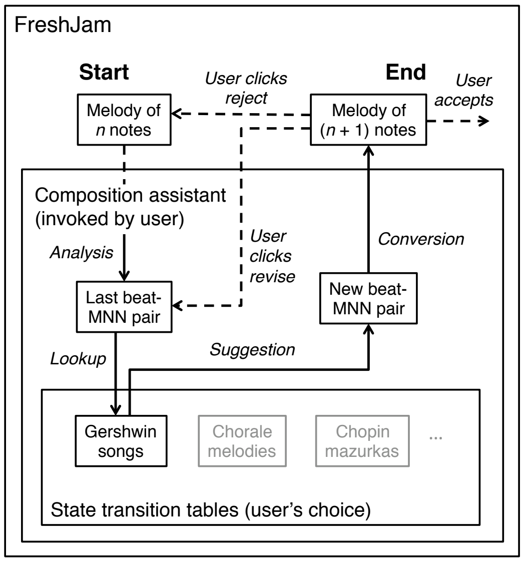

Figure 1: *Schematic for FreshJam and the embedded composition assistant. Dashed arrows indicate user decisions, and solid arrows for autonomous steps.*

Creating the state transition table from stylistically homogeneous melodies

To create the state transition table for a specific style, a corpus of existing melodies is converted from point set representations, $D = \{\mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_n\}$, to state-context pairs,

$$L_n = \big((\mathbf{s}_1, c_1), (\mathbf{s}_2, c_2), \ldots, (\mathbf{s}_n, c_n)\big). \qquad (1)$$

The dimensions of the point set are onset measured in quarter-note beats, MIDI note number (MNN), morphetic pitch number (MPN, necessary for correct pitch spelling, see Meredith 2006), duration in quarter-note beats, staff number, and loudness (or velocity, 0-127). Each point in the set D represents a note (or several notes in the case of ties). The point $\mathbf{d}_i$ is converted to a so-called *beat-MNN state* $\mathbf{s}_i$, which is an ordered pair of numbers. The first number is the beat of the measure on which $\mathbf{d}_i$ begins, calculated from the point onset and overall time signature. For instance, counting from zero and with four beats in each measure, the onset 25 maps to beat 2 ($= 25 \bmod 4 + 1$). The second number is MNN relative to tonal center, which can be determined from the point MNN, the mean MNN of the point set, and the overall key of the melody. For example, in a G-major melody with mean MNN 72, MNN 63 maps to -4 ($= 63 - 67$), as 67 is the MNN of the tonic note closest to the mean MNN. The rationale behind this choice of state representation is that if two notes/points $\mathbf{d}_i, \mathbf{d}_j$, possibly from different melodies, map to the same beat-MNN state $\mathbf{s}$, then they are more likely to be perceived as metrically and tonally equivalent than a randomly chosen pair of notes (Huron 2006).

As well as being converted to a state $\mathbf{s}_i$, each point $\mathbf{d}_i$ is also converted to a context c_i (eq. 1). The context c_i is itself a list, containing contextual information for converting the state $\mathbf{s}_i$ into a new point $\mathbf{e}$ and hence a new melody note. Our use of a *context* is similar to Cope's (1997; 2005) use of a *lexicon*, but we give more details on the information stored and its use. The context consists of a string identifier for the melody, the point $\mathbf{d}_i$, the transformation applied to the MNN, and the key of the melody. Details of how this information is used to construct a new point $\mathbf{e}$ are given in the next section. Letting L_n be the list of state-context pairs for a corpus of stylistically homogeneous melodies, construction of the state transition table begins by defining a column of unique states $\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_m$ observed in L_n. A unique state $\mathbf{u}_i$ is used to index all occurrences of the beat-MNN state across the corpus, recording *what happened next on each occasion* as a list of state-context pairs denoted $L(\mathbf{u}_i)$. Elements of $L(\mathbf{u}_i)$ form the basis of candidate continuations for the beat-MNN state $\mathbf{u}_i$. A random, equiprobable selection from $L(\mathbf{u}_i)$ is equivalent to sampling from distributions in the transition matrix for a first-order Markov model over the space of beat-MNN states. The difference between our approach and a standard Markov model is that retaining a context c_i alongside the state $\mathbf{s}_i$ makes this information available when converting states back into new melody notes.

Using the composition assistant in FreshJam

When a FreshJam user invokes the composition assistant, their melody is converted to a point set D and the last point $\mathbf{d}$ is converted to a state $\mathbf{s}$. This state is searched amongst the unique states $\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_m$ occurring in the corpus (see above), and if $\mathbf{s} = \mathbf{u}_i$ for some $i \in \{1, 2, \ldots, m\}$, then an element of the list of candidate continuations $L(\mathbf{u}_i)$ will be sampled without replacement. The sampled candidate continuation, denoted $(\mathbf{s}_j, c_j)$, must be converted to a new point $\mathbf{e}$ and hence a new melody note for the user to audition. (If no such state $\mathbf{u}_i$ exists then $\mathbf{s}$ is not observed in the corpus, and a message to this effect is returned.)

The onset of the new point/note $\mathbf{e}$ is calculated from the measure beat (first element of state $\mathbf{s}_j$) and the onset of the last melody note $\mathbf{d}$. The MNN of $\mathbf{e}$ is calculated from the MNN relative to tonal center (second element of $\mathbf{s}_j$), the specified (or estimated) key of the user's melody, and the transformation that was applied to the point from the corpus (retained in the context c_j). The remaining elements of $\mathbf{e}$ (MPN, duration, staff number, and loudness) are also calculated from the context c_j. The point $\mathbf{e}$ is appended to the set D and converted to a sound file for the user to audition. After audition the user may:

- reject the suggested continuation (*undo* functionality restores the previous point set D and melody);
- request a revised suggestion (the list $L(\mathbf{u}_i)$ of candidate continuations is resampled);
- continue composing with the suggested note as part of their melody.

Suggestions from the composition assistant can be requested occasionally or one after the other, and, in the latter case, the string identifier retained in the context safeguards against too many consecutive suggestions originating from the same corpus melody.

The result of a sample session in FreshJam is shown in Fig. 2, where the aim was to write a song melody in the style of George Gershwin (1898-1937). The ossia in Fig. 2 indicates points at which the composition assistant was invoked, with brackets for consecutive suggestions, crossed noteheads for rejected suggestions, and arrows for edits. It is evident from Fig. 2 and the video demonstration mentioned previously that using the state transition table reduces the huge number of possible continuations to what some composer(s) (e.g., Gershwin) would have done, whilst maintaining the potential for unusual or surprising suggestions. Figure 2 also illustrates how the composition assistant favors interaction and iteration. The user still has to devise the melody note by note, whereas with existing software such as PG Music's Band in a Box or Microsoft's Songsmith, whole accompaniments can be produced automatically.

Application and proposed evaluation

Precisely because FreshJam's composition assistant is geared towards interaction and iteration, it has potential as an aid in stylistic composition tasks. Based on statistics from the main examination boards, in the UK alone an estimated 50,000 students aged 16-18 undertake composition tasks each year, and stylistic composition is part of music syllabuses at higher education establishments across Europe and the USA. While we acknowledge that some tasks, such

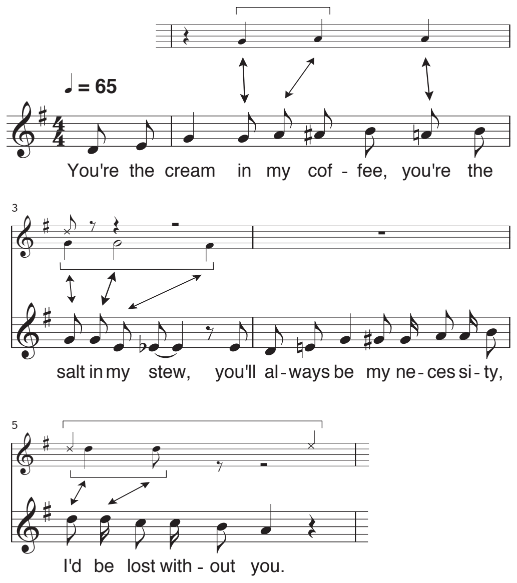
♩ = 65
You're the cream in my cof - fee, you're the
salt in my stew, you'll al - ways be my ne - ces si - ty,
I'd be lost with - out you.

Figure 2: *Sample session in FreshJam, with the aim of writing a song melody in the style of Gershwin. The ossia and annotations indicate suggestions from the composition assistant, as explained in the main text. Lyrics by Nat King Cole (1919-1965).*

as harmonization, are better catered for by systems other than FreshJam (Ebcioğlu 1994), we intend to evaluate the extent to which our composition assistant can have a positive effect on students' music education and work. It appears that no existing study on computer-based composition has considered *assistant software* as an independent variable (Hewitt 2009; Seddon and O'Neill 2001). As part of our evaluation, students will be assigned to different experimental conditions (use of composition assistant and control), and asked to complete compositional tasks. Separately, judges (music teachers and examiners) will rate the stylistic and creative success of these compositions (Pearce and Wiggins 2007; Hickey 2001). Analysis of variance can be employed to determine if use of FreshJam's music composition assistant leads to statistically significant improvements in the stylistic success of student compositions.

Conclusion

The details of FreshJam—a piano-roll, web-based interface with functionality for suggesting stylistic continuations to melodic fragments—have been presented and discussed. A sample FreshJam session (Fig. 2) demonstrated that use of the composition assistant is interactive and iterative, but not overly facile. As such, it could help transport music education 'towards a holistic model of artistic practice mediated through the effective use of ICT [information and communication technologies] . . .[away from] traditional or pre-existing musical practice merely done with ICT' (Savage 2005). Future work on FreshJam will include investigating improvisatory interaction, continuations for fuller musical textures, and composing backward from a target note.

References

Conklin, D., and Witten, I. H. 1995. Multiple viewpoint systems for music prediction. *Journal of New Music Research* 24(1):51–73.

Cope, D. 1997. The composer's underscoring environment: CUE. *Computer Music Journal* 21(3):20–37.

Cope, D. 2005. *Computer models of musical creativity*. Cambridge, MA: MIT Press.

Ebcioğlu, K. 1994. An expert system for harmonizing chorales in the style of J.S. Bach. In Balaban, M.; Ebcioğlu, K.; and Laske, O., eds., *Understanding music with AI: Perspectives on music cognition*. Menlo Park, CA: AAAI Press. 294–333.

Hewitt, A. 2009. Some features of children's composing in a computer-based environment: the influence of age, task familiarity and formal instrumental music tuition. *Journal of Music, Technology and Education* 2(1):5–24.

Hickey, M. 2001. An application of Amabile's consensual assessment technique for rating the creativity of children's musical compositions. *Journal of Research in Music Education* 49(3):234–244.

Huron, D. 2006. *Sweet anticipation: music and the psychology of expectation*. Cambridge, MA: MIT Press.

Keller, R. M.; Toman-Yih, A.; Schofield, A.; and Merritt, Z. 2012. A creative improvisational companion based on idiomatic harmonic bricks. In Maher, M. L. et al., ed., *Proceedings of the International Conference on Computational Creativity*, 155–159. Dublin, Ireland: University College Dublin.

Maxwell, J. B.; Eigenfeldt, A.; Pasquier, P.; and Gonzalez Thomas, N. 2012. MusiCOG: a cognitive architecture for music learning and generation. In *Proceedings of the Sound and Music Computing Conference*, 9 pages. Copenhagen: Aalborg University Copenhagen.

Meredith, D. 2006. The *ps13* pitch spelling algorithm. *Journal of New Music Research* 35(2):121–159.

Pachet, F. 2002. Interacting with a musical learning system: the continuator. In Anagnostopoulou, C.; Ferrand, M.; and Smaill, A., eds., *Music and Artificial Intelligence: Proceedings of the International Conference on Music and Artificial Intelligence*, Lecture Notes in Artificial Intelligence. Berlin: Springer-Verlag. 119–132.

Pearce, M. T., and Wiggins, G. A. 2007. Evaluating cognitive models of musical composition. In Cardoso, A., and Wiggins, G. A., eds., *Proceedings of the International Joint Workshop on Computational Creativity*, 73–80. London, UK: Goldsmiths, University of London.

Savage, J. 2005. Information communication technologies as a tool for re-imagining music education in the 21st century. *International Journal of Education and the Arts* 6(2). Retrieved February 25, 2011 from http://www.ijea.org/v6n2/.

Seddon, F. A., and O'Neill, S. A. 2001. An evaluation study of computer-based compositions by children with and without prior experience of formal instrumental music tuition. *Psychology of Music* 29(1):4–19.