CHORD-SEQUENCE-FACTORY: A CHORD ARRANGEMENT SYSTEM MODIFYING FACTORIZED CHORD SEQUENCE PROBABILITIES

Satoru Fukayama Kazuyoshi Yoshii Masataka Goto
National Institute of Advanced Industrial Science and Technology (AIST), Japan
{s.fukayama, k.yoshii, m.goto}@aist.go.jp

ABSTRACT

This paper presents a system named *ChordSequenceFactory* for automatically generating chord arrangements. A key element of musical composition is the arrangement of chord sequences because good chord arrangements have the potential to enrich the listening experience and create a pleasant feeling of surprise by borrowing elements from different musical styles in unexpected ways. While chord sequences have conventionally been modeled by using N-grams, generative grammars, or music theoretic rules, our system decomposes a matrix consisting of chord transition probabilities by using nonnegative matrix factorization. This enables us to not only generate chord sequences from scratch but also transfer characteristic transition patterns from one chord sequence to another. *ChordSequenceFactory* can assist users to edit chord sequences by modifying factorized chord transition probabilities and then automatically re-arranging them. By leveraging knowledge from chord sequences of over 2000 songs, our system can help users generate a wide range of musically interesting and entertaining chord arrangements.

1. INTRODUCTION

Chord sequences are essential when composing and arranging music. Different songs, composers, arrangers, and musical genres have different tendencies to use chord sequences, which contributes to increased variety in music. Each song could have different natural chord sequences that give different impressions. Although an arranger can change (*i.e.*, arrange) chord sequences of a song to alter its mood, this is very difficult for people who lack knowledge of chord sequences to arrange the chords in an appropriate way. The goal of this research is to assist people to generate variations of chord sequences from an input original sequence by leveraging knowledge from a large number of other existing chord sequences called *references*.

Chord sequence arrangement is a promising approach to create derivative works from existing songs. Although amateur creators could create such derivative works themselves, known as user-generated content (UGC) on video sharing services, the majority of users are generally music listeners who do not create much themselves. We aim to encourage such people to create more derivative works by changing chord sequences. This enables music listeners to personalize songs by customizing chord sequences without any special training [4]. Such content personalization was achieved by *Drumix* [15], a system for arranging the drum track in music audio signals, but content personalization using chord arrangements has not yet been studied.

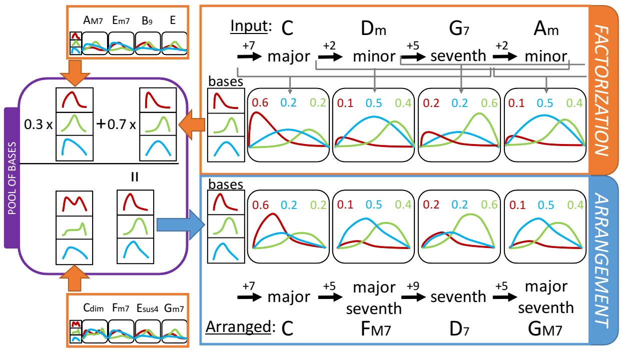

Figure 1. Overview of generating chord arrangements with *ChordSequenceFactory*.

To generate chord arrangements, it is necessary to model and manipulate chord sequences. Conventionally, models for generating chord sequences have been represented by probabilistic models or N-grams [1, 6, 8, 9, 11, 12, 14], genetic algorithms [2], generative grammars [10, 13], and exploiting examples or templates [3, 7]. Although these existing models can easily be used to generate new chord sequences from scratch [2], it is difficult to arrange chord sequences of an existing song while referring to other existing songs.

In this paper we propose a new mathematical formulation of chord sequences using nonnegative matrix factorization (NMF) and use it to build a system, *ChordSequenceFactory,* that enables users to arrange chord sequences of existing songs (Fig. 1). The chord sequence of each song (a sequence of symbols) is first converted into a *chord differential matrix* that represents chord transitions in short regions. The matrix is then decomposed into a set of bases (characteristic chord transition patterns) and a set of corresponding temporal activations using NMF. Chord arrangements can be achieved by interpolating the bases of a song with similar bases obtained from other songs while preserving the activations. An arranged chord sequence is

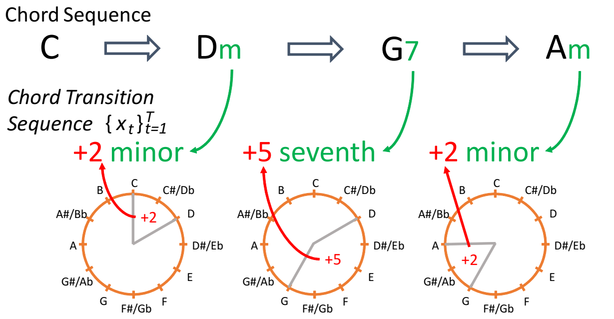

Figure 2. The representation of a sequence of a chord transition used in our method. Each transition is represented as a combination of the interval between the root notes of adjacent chords and the type of the latter chord.

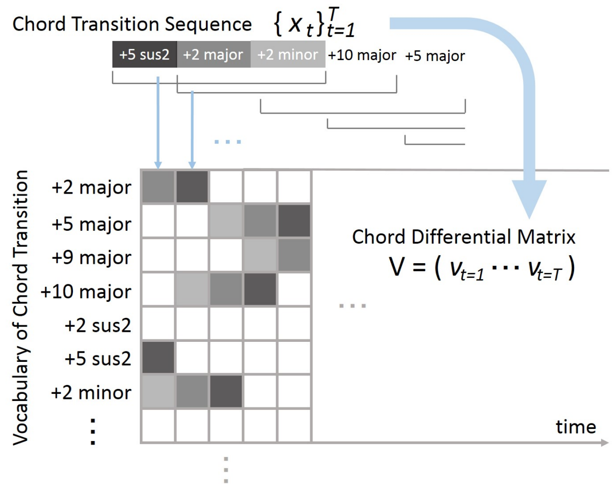

Figure 3. Representation of a chord transition sequence with the *chord differential matrix* representation.

finally generated from the reconstructed *chord differential matrix* as the product of the resulting bases and the original activations.

The remainder of this paper is structured as follows. In Section 2, we present the analysis and synthesis framework of chord symbol sequences based on the *chord differential matrix*. In Section 3, we present the formulation of chord arrangements. In Section 4, we present experimental results to illustrate the performance of the system *ChordSequenceFactory*. Finally in Sections 5 and 6 we include discussion and conclusions, which summarize the main contributions of the paper.

2. ANALYSIS AND SYNTHESIS OF SYMBOLIC CHORD SEQUENCES

2.1 Chord transition sequence

Chord names typically consist of up to three labels: root note, chord type, and bass note specification [5]. Since the bass notes are omitted in many cases, we treat chord information as a combination of the first two labels. In general, chord transitions are considered to be more important than the absolute pitch of the root note of each chord. This is supported by the fact that it is possible to transpose chord sequences into other tonalities without affecting the functions of chord idioms used in these sequences.

In this paper we do not define a vocabulary of chord names but rather, a vocabulary of chord "transitions" for modeling how adjacent chords are arranged in music. Let $\{c_n\}_{n=1}^{N}$ be the vocabulary of chord transitions, where N is the size of the vocabulary and each c_n is defined as follows:

$$c_n \equiv \mathrm{str}\,(\mathrm{INTERVAL}) + \mathrm{str}\,(\mathrm{TYPE})\,, \tag{1}$$

where INTERVAL indicates the difference in semitones between the root notes of a target chord and the previous chord and TYPE indicates the type of the target chord. An example chord transition sequence is shown in Fig. 2. Since we do not need to know the direction of root changes, the value of INTERVAL is restricted to a nonnegative integer. The operation str$(\cdot)$+str$(\cdot)$ means the string conjunction (*i.e.*, concatenation). Once a given sequence of chords is converted into a sequence of chord transitions according to the defined vocabulary, we can recover the original sequence if the first chord of the sequence is given. This representation has also been used in related work [6] for the same purpose of normalizing the tonality of chord sequences.

An advantage of this representation is that it can reduce the number of parameters to be dealt with. The typical approach to modeling chord sequences is to calculate a transition probability matrix over chord names defined in a vocabulary. This approach, however, requires a large number of parameters, *i.e.*, we need to deal with $M \times M$ parameters if M kinds of chord names are contained in the vocabulary. Since the chord type seems to be less dependent on the type of the previous chord name, we directly focus on the root-note interval and the current chord type.

2.2 Analysis of a chord transition sequence based on a chord differential matrix

We now explain the probabilistic representation of a chord-transition sequence $\{x_t\}_{t=1}^{T}$, where $x_t \in \{c_n\}_{n=1}^{N}$ and T is the length of the sequence. We analyze the sequence on frame-by-frame basis using an exponentially-decaying window as shown in Fig. 3. This window is designed based on our assumption that the characteristics of chord transitions remain the same for some period of time because musical pieces are usually composed so that each section gives a coherent impression. The window is moved one by one from the first chord of the given sequence. In each frame t, we calculate a probability vector $\boldsymbol{v}_t \in \mathbb{R}^N$ such that the elements of the vector sum to unity. More specifically, v_{tn} is a ratio of chord transition c_n to all possible transitions in frame t, which is given by

$$v_{tn} = \frac{\sum_{0\leq\tau\leq\lambda} \delta_{c_n x_{t+\tau}} e^{-\tau}}{\sum_{0\leq\tau\leq\lambda} e^{-\tau}}, \tag{2}$$

where δ_{ij} is the Kronecker delta, λ is the window length, and $e^{-\tau}$ is a temporally-decaying weight. Note that $\boldsymbol{v}_t$ indicates a co-occurrence relationship between chord transi-

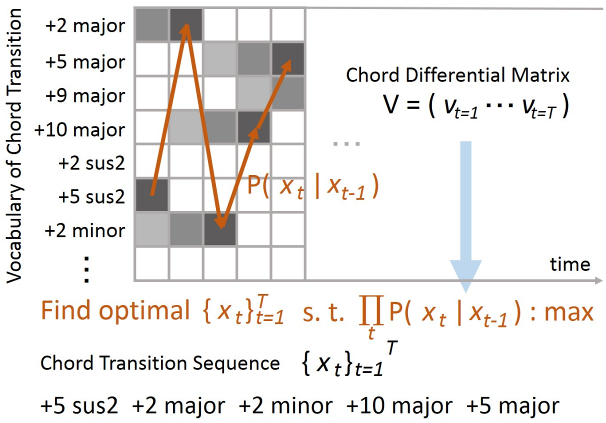

Figure 4. Regeneration of a chord transition sequence from a chord differential matrix representation by means of V and the bi-gram probability obtained *a priori* from the data.

tions in the vicinity of frame t. The process of representing the chord transition sequence as a *chord differential matrix* is shown in Fig. 3.

This frame-based vectorial representation of chord transition probabilities has useful properties for chord arrangement as follows:

- **Mood coherence in short durations**
  Although chords often change at bar boundaries, the mood does not change in such a short time span because we use an exponentially-decaying window of length λ.

- **Reproducibility of chord sequences**
  We can approximately reconstruct the original sequence $\{x_t\}_{t=1}^T$ from a sequence of probability vectors $\{\boldsymbol{v}_t\}_{t=1}^T$ in a principled manner.

2.3 Synthesis of a chord transition sequence

To regenerate a chord transition sequence from a chord differential matrix, we need to consider the transition between successive chord transitions x_t and x_{t-1}. Using the transition probability $P(x_t|x_{t-1})$ trained from the data, we can calculate the probability of observing $\{x_t\}_{t=1}^T$ as follows:

$$P\left(\{x_t\}_{t=1}^T\right) = \prod_{t=1}^{T} \left(\xi v_t\left(x_t\right) + \left(1-\xi\right) P\left(x_t|x_{t-1}\right)\right), \tag{3}$$

where ξ is an interpolation coefficient such that $0 \leq \xi \leq 1$.

The chord transition sequence $\{x_t^*\}_{t=1}^T$ is reconstructed by maximizing $P\left(\{x_t\}_{t=1}^T\right)$ as follows:

$$\{x_t^*\}_{t=1}^T = \underset{\{x_t\}_{t=1}^T}{\operatorname{argmax}}\, P\left(\{x_t\}_{t=1}^T\right). \tag{4}$$

Since there are N possibilities for each x_t, it is computationally infeasible to test all N^T possible sequences with the naive method of exhaustive search. Fortunately, we can obtain the solution $\{x_t^*\}_{t=1}^T$ with $O(N)$ using dynamic programming. The process for generating a sequence is shown in Fig. 4.

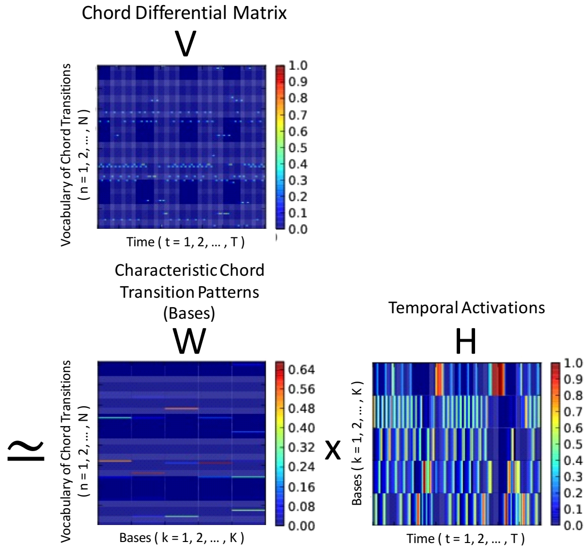

Figure 5. Factorizing the chord deferential matrix: Analysis of the frequent combinations and the temporal occurrences of chord transitions.

3. FORMULATION OF CHORD-SEQUENCE-FACTORY

3.1 Factorization of a chord differential matrix

An overview of *ChordSequenceFactory* is illustrated in Fig. 6. The mood of music varies according to sections of a song. For instance, some sections often use dominant intervals and other sections tend to use chords with more tension notes. Therefore we aim to identify the characteristic patterns of chord transitions that affect the mood of each section. We represent a probability vector $\boldsymbol{v}_t$ at each time t as a convex combination of multiple bases as follows:

$$\boldsymbol{v}_t = \sum_{k=1}^{K} h_{kt} \boldsymbol{w}_k, \tag{5}$$

where $\boldsymbol{w}_k, (k = 1, \cdots, K)$ denotes a characteristic chord transition pattern and h_{kt} is its weight. In order to decompose $\{\boldsymbol{v}_t\}_{t=1}^T$ using shared bases $\{\boldsymbol{w}_k\}_{k=1}^K$, we exploit nonnegative matrix factorization (NMF) (Fig. 5), *i.e.*,

$$V = (\boldsymbol{v}_1 \cdots \boldsymbol{v}_T) \tag{6}$$

is decomposed into matrices $W\,(N \times K)$ and $H\,(K \times T)$ as:

$$V \simeq WH, \tag{7}$$

where W is the matrix of bases:

$$W = (\boldsymbol{w}_1 \cdots \boldsymbol{w}_K) \tag{8}$$

and H is the matrix of activations:

$$H = \begin{pmatrix} h_{11} & \cdots & h_{1T} \\ \vdots & \ddots & \vdots \\ h_{K1} & \cdots & h_{KT} \end{pmatrix}. \tag{9}$$

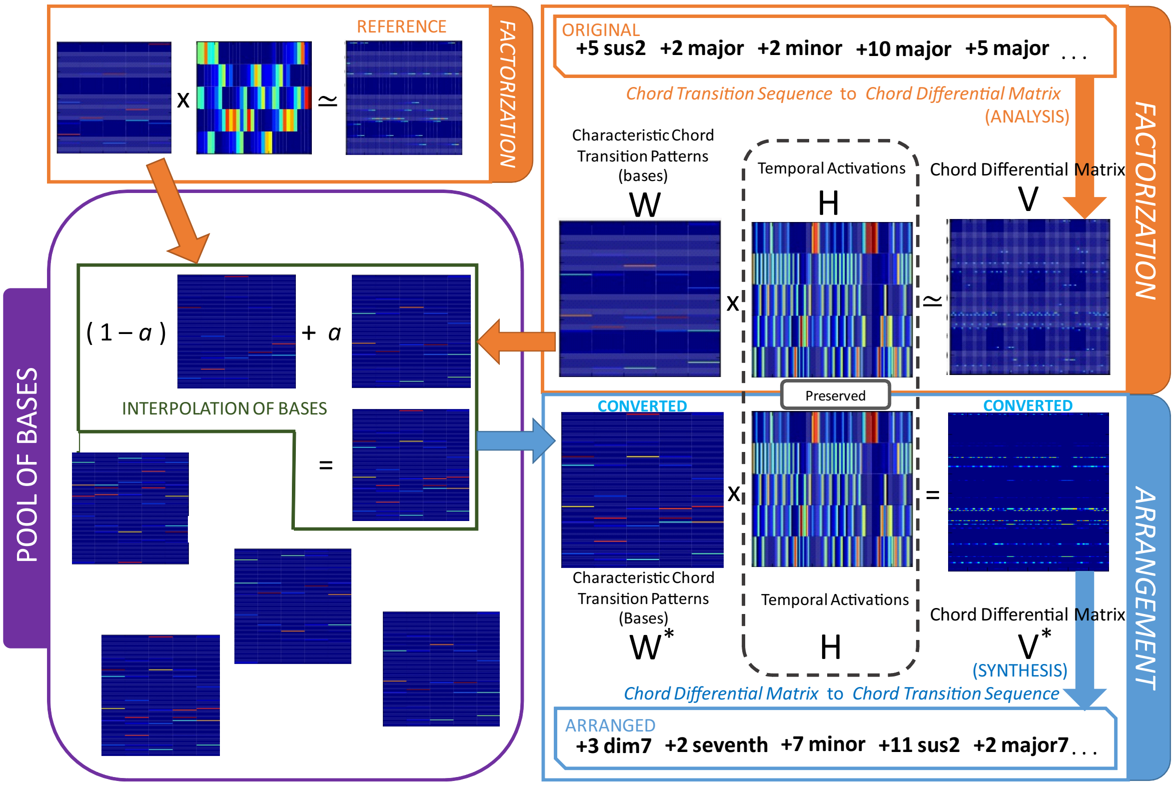

Figure 6. Overview of the process executed by *ChordSequenceFactory* for generating chord arrangements.

3.2 Interpolation

We want to modify the V of a target song by referring to chord transition patterns used in another song. More specifically, to obtain a reconstructed chord differential matrix V^*, we reuse the H of the target song and use a set of modified vectors $W^* = (\boldsymbol{w}_1^* \cdots \boldsymbol{w}_K^*)$ as follows:

$$V^* = (\boldsymbol{w}_1^* \cdots \boldsymbol{w}_K^*)\, H \qquad (10)$$
$$= W^* H. \qquad (11)$$

To obtain each modified vector $\boldsymbol{w}_i^*$, we interpolate a reference vector $\boldsymbol{w}_i^{\mathrm{ref}}$ of another song with the original vector $\boldsymbol{w}_i$. Since the original bases $\{\boldsymbol{w}_k\}_{k=1}^K$ and reference bases $\{\boldsymbol{w}_k^{\mathrm{ref}}\}_{k=1}^K$ are not guaranteed to have their index aligned, we associate each $\boldsymbol{w}_i$ with a reference $\boldsymbol{w}_{j(i)}^{\mathrm{ref}}$ such that $\boldsymbol{w}_i$ is closest to $\boldsymbol{w}_{j(i)}^{\mathrm{ref}}$, *i.e.*,

$$j(1), \cdots, j(K) = \underset{j(1),\cdots,j(K)}{\operatorname{argmax}} \sum_{k=1}^{K} D\left(\boldsymbol{w}_i \| \boldsymbol{w}_{j(i)}^{\mathrm{ref}}\right), \qquad (12)$$

where D is a distance measure based on the symmetric Kullback Leibler divergence:

$$D\left(\boldsymbol{w}_i \| \boldsymbol{w}_{j(i)}^{\mathrm{ref}}\right) = \sum_k w_{ik} \log \frac{w_{ik}}{w_{j(i)k}^{\mathrm{ref}}} + \sum_k w_{j(i)k}^{\mathrm{ref}} \log \frac{w_{j(i)k}^{\mathrm{ref}}}{w_{ik}}. \qquad (13)$$

Using the aligned indicies $j(1), \cdots, j(K)$, we can calculate the modified vector $\boldsymbol{w}_i^*$ as follows:

$$\boldsymbol{w}_i^* = a\boldsymbol{w}_i + (1-a)\, \boldsymbol{w}_{j(i)}^{\mathrm{ref}}, \qquad (14)$$

where a is the interpolation parameter such that $0 \le a \le 1$. The value of a represents how the mood of the original song is preserved through the chord arrangement.

3.3 Generation of chord arrangements

As discussed in Section 2.3, we can generate an arranged chord transition sequence from the modified chord differential matrix V^*. We interpolate the transition probability obtained from all songs in the database $P_{\mathrm{all}}(x_t|x_{t-1})$ and that obtained from the reference song $P_{\mathrm{ref}}(x_t|x_{t-1})$ with that obtained from the original song $P_{\mathrm{org}}(x_t|x_{t-1})$ as follows:

$$P^*(x_t|x_{t-1}) = \xi_1 P_{\mathrm{all}}(x_t|x_{t-1}) + \xi_2 P_{\mathrm{org}}(x_t|x_{t-1}) + \xi_3 P_{\mathrm{ref}}(x_t|x_{t-1}), \qquad (15)$$

where $\{\xi_i\}_{i=1}^3$ are the interpolation coefficients that sum to unity, *i.e.*, $\sum_i \xi_i = 1$. Using $P^*(x_t|x_{t-1})$, we can calculate the probability of observing $\{x_t\}_{t=1}^T$ as follows:

$$P\left(\{x_t\}_{t=1}^T\right) = \prod_{t=1}^{T} (\xi v_t^*(x_t) + (1-\xi)\, P^*(x_t|x_{t-1})). \qquad (16)$$

The arranged chord transition sequence is then obtained by maximizing $P\left(\{x_t\}_{t=1}^T\right)$ (see Section 2.3).

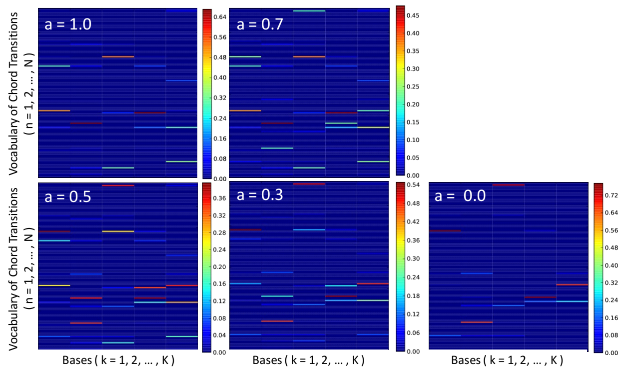

Figure 7. Examples of the bases interpolated with bases of a reference song: we can see that the chord usages originated in two different songs are combined, depending on the value of interpolation factor a.

4. EVALUATION

4.1 Experimental conditions

To evaluate our system, we used 2,123 lead sheets downloaded from the Wikifonia (www.wikifonia.org) website. All lead sheets we used were formatted in the MusicXml format including information of chord sequence. Each chord name in a file included the step (C, D, ...), alternation (#, ♭) of the root note, and the chord type.

To define a vocabulary $\{c_n\}_{n=1}^{N}$, we extracted chord transitions that appeared more than 10 times in the data. A special symbol "Unknown" $\in \{c_n\}_{n=1}^{N}$ was used for representing the rest. The vocabulary size was $N = 163$. Using the vocabulary, we represented each song as a chord transition sequence $\{x_t\}_{t=1}^{T}$ and calculated a chord differential matrix from that sequence by using an exponentially-decaying window of length $\lambda = 6$, corresponding to the number of chord changes. The transition probabilities between $\{c_n\}_{n=1}^{N}$ were calculated in advance for all 2,123 songs and for each song, respectively. The chord differential matrix was decomposed using NMF based on the Euclidean distance with $K = 5$.

We interpolated characteristic chord transition patterns (basis vectors) of a reference song with those of an original song according to an interpolation coefficient $a = 1.0$, 0.7, 0.5, 0.3, or 0.1. A chord transition sequence was synthesized from a modified chord differential matrix. Since the vocabulary of chord transitions only holds information of relative root positions, we set the root note of the initial chord to the original root note.

4.2 Experimental results and discussions

As shown in Fig. 7, we can combine two different characteristic chord transition patterns by controlling the interpolation factor a on the user interface of *ChordSequenceFactory* (Fig. 8). The generated examples of chord arrangements are shown in Table. 1. We confirmed that the mood of an original song can be modified while incorporating the mood of a reference song if these songs have some simillar characteristic chord transition patterns.

Through informal evaluation and listening tests we found our system was, in many cases, able to provide musically coherent chord sequence arrangements (Table. 1). However, our evaluation also revealed some limitations that should be addressed in our future work. Since the vocabulary of chord transitions defined for the system does not include the absolute pitch for the root note, the generated results tended to exhibit transposition frequently. In addition, finding the optimal number of bases when decomposing the probability should be investigated for better performances. Finally, a slider for changing the activation can bring about more effects in the generated sequence.

Although simply adding a seventh note to chord sequences (rather than using our approach) has a similar effect on chord arrangements, dissonance may appear in the connection between the chords. In contrast, with our system, dissonant chords can be avoided by using the constraints given by the transition probabilities. Furthermore, there are no restrictions in terms of combining two songs that have different structures, since the interpolation is done between the decomposed basis.

Conventional methods based on N-grams cannot control the dynamic characteristics of chord transitions and need label sequences to reflect human intention. In our method, we can see that the probability is changed for each time t and that the activations at time t play the same role as the label sequences in the conventional methods.

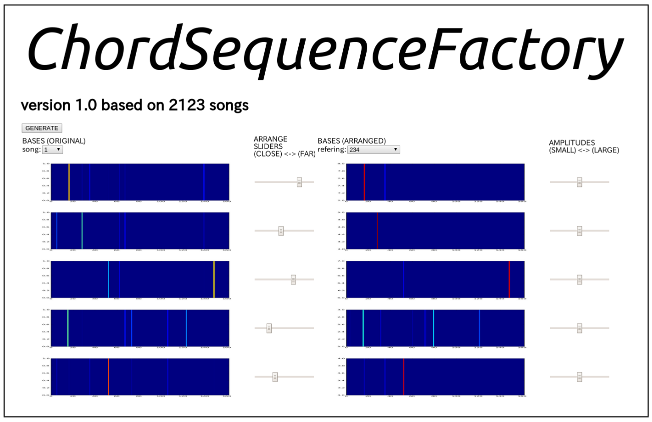

Figure 8. User interface of *ChordSequenceFactory*: users can change the interpolation factor with sliders corresponding to each base.

5. CONCLUSIONS

We have described a system *ChordSequenceFactory* that can assist a user to arrange chord symbol sequences of a song by finding latent frequent patterns of chord transitions and modifying them on the basis of other songs. We proposed a new analysis and synthesis framework for chord symbol sequences, where the temporal changes of chord transition sequences are represented as a chord differential matrix. NMF is used to decompose this matrix into bases corresponding to the frequent patterns of chord transitions and activations corresponding to their temporal occurrences. The matrix can then be updated for a new arrangement by modifying these bases by mixing them with

ORIGINAL			Cmin	B♭maj	E♭sus2	Fmaj	Gmin	Fmaj	B♭maj	Gmaj
	reference	a								
ARRANGED	A	0.7	Cmaj7	Fmaj7	B♭7	E♭7	A♭maj6+9	G♭maj	Bmaj	Gmin6
		0.6	Cmaj	B♭maj	E♭7	A♭7	D♭maj6+9	Bmaj	Emaj	Cmin6
		0.5	Cmaj	B♭maj	Cmaj	Dmaj	Cmaj	B♭maj	E♭maj	A♭min
	B	0.7	Cdim	E♭dim	E♭7	E♭7	F#dim	Adim	D7	Gmin6
		0.6	Cdim	E♭dim	E♭7	E♭7	F#dim	Adim	D7	Gmin6
		0.5	Cmaj	B♭maj	Cmaj	Dmaj	Cmaj	B♭maj	E♭maj	B♭min6
	C	0.7	Cmaj	Fmaj	F7	F7	B♭maj	E♭maj	A♭maj	Gsus4
		0.6	Cmaj	Fmaj	F7	F7	B♭maj	E♭maj	A♭maj	Gsus4
		0.5	Cmaj	Fmaj	F7	F7	B♭maj	E♭maj	A♭maj	Gsus4
	D	0.7	Cmin7	Dmin7	Dmaj7	Dmaj7	Emin7	Amaj7	Dmaj7	Cmaj7
		0.6	Cmin7	Dmin7	Dmaj7	Dmaj7	Emin7	Amaj7	Dmaj7	Cmaj7
		0.5	Cmaj	B♭maj	Cmaj	Dmaj	Cmaj	B♭maj	E♭maj7	A♭min
	E	0.7	Cmaj6	Fmaj6	B♭sus4	E♭sus4	A♭maj6	D♭maj6	F#7	Bmin
		0.6	Cmaj6	Fmaj6	B♭sus4	E♭sus4	A♭maj6	D♭maj6	F#7	Bmin
		0.5	Cmaj6	Fmaj6	B♭sus4	E♭sus4	A♭maj6	D♭maj6	F#major	Bmin

Table 1. Generated results of *ChordSequenceFactory* with five different reference songs (A,B,C,D,E). For each reference song, three values of interpolation factor $a = 0.7, 0.6, 0.5$ were used. The arranged chord sequences were decoded from the chord transition sequence $\{x_t\}_{t=1}^{T}$, by setting the root note of the first chord to C as in the original sequence.

similar bases in the pool of bases obtained from more than 2000 songs. The updated matrix is finally used to generate a new re-arranged chord sequence using dynamic programming. In our experience, *ChordSequenceFactory* generated musically interesting and entertaining chord arrangements. In the future, we plan to extend our framework to consider melody lines as constraints on arranged chord sequences. We will also include audio signal processing so that the chord differential matrix can be used directly on music audio signals.

Acknowledgement
This work was supported in part by OngaCREST, CREST, JST.

6. REFERENCES

[1] C. Ames: "The Markov Process as a Compositional Model: A Survey and Tutorial," *Leonardo Music Journal*, Vol. 22, No. 2, pp. 175–187, 1989.

[2] J. Biles: "GenJam: A Genetic Algorithm for Generating Jazz Solos," *Proceedings of ICMC*, pp. 131–137, 1994.

[3] D. Cope: *Experiments in Musical Intelligence*, A-R Editions, 1996.

[4] M. Goto: "Active Music Listening Interfaces based on Signal Processing," *Proceedings of ICASSP*, pp. 1441–1444, 2007.

[5] C. Harte, M. B. Sandler, S. A. Abdallah, E. Gómez: "Symbolic Representation of Musical Chords: A Proposed Syntax for Text Annotations," *Proceedings of the ISMIR conference*, pp. 66–71, 2005.

[6] M. Mauch, S. Dixon, C. Harte, M. Casey, B. Fields: "Discovering Chord Idioms through Beatles and Real Book Songs," *Proceedings of the ISMIR conference*, pp. 255–258, 2007.

[7] F. Pachet: "Surprising Harmonies," *International Journal of Computing Anticipatory Systems*, Vol. 4 1999.

[8] J. Paiement, D. Eck, S. Bengio: "A Probabilistic Model for Chord Progressions," *Proceedings of the ISMIR conference*, pp. 312–319, 2005.

[9] H. Papadopoulos, G. Peeters: "Large-scale study of chord estimation algorithms based on chroma representation and hmm," *Proceedings of the ISMIR conference*, pp 225-258, 2007.

[10] M. Rohermeier: "Towards a generative syntax of tonal harmony," *Journal of Mathematics and Music*, Vol. 5, No. 1, pp. 35–53, 2011.

[11] R. Scholz, E. Vincent, F. Bimbot: "Robust modeling of musical chord sequences using probabilistic N-grams," *Proceedings of the ICASSP*, pp. 53–56, 2009.

[12] A. Sheh, D. Ellis: "Chord Segmentation and Recognition using EM-Trained Hidden Markov Models," *Proceedings of the ISMIR conference*, pp. 185–191, 2003.

[13] M. J. Steedman: "A Generative Grammar for Jazz Chord Sequences," *Music Perception*, 2:1 pp. 52–77, 1984.

[14] D. Temperley: *Music and Probability*, The MIT Press, 2007.

[15] K. Yoshii, M. Goto, K. Komatani, T. Ogata and H. G. Okuno: "Drumix: An Audio Player with Real-time Drum-part Rearrangement Functions for Active Music Listening," *IPSJ Digital Courier*, Vol. 3, pp. 137–144, 2007.