HarmonyMixer: Mixing the Character of Chords among Polyphonic Audio

Satoru Fukayama   Masataka Goto
National Institute of Advanced Industrial Science and Technology (AIST), Japan
{s.fukayama, m.goto} [at] aist.go.jp

ABSTRACT

This paper describes *HarmonyMixer*, a method that enables a user without musical expertise to personalize the mood of existing polyphonic musical recordings by modifying their chord sequences. Our method lets the user choose a *reference song* with a character that the user wants reflected in chords of a *target song*. It is, however, difficult to modify chords in existing complex sound mixtures since technologies of sound source separation and multipitch analysis are not yet accurate enough for those mixtures. To overcome this difficulty, *HarmonyMixer* does not rely on those technologies and instead modifies chords by leveraging chromagrams. It first analyzes a chromagram feature matrix by using Bayesian non-parametric Non-negative Matrix Factorization, and then interpolates basis matrices obtained from *reference* and *target songs* to convert the chromagram of the *target song*. It finally modifies the spectrogram of the *target song* by reflecting the difference between the original and converted chromagrams while considering relations between frequency bins and chroma bins. Listening to the output from our method confirmed that modification of chords had been derived.

1. INTRODUCTION

While *Active Music Listening Interfaces* [1] allow content-based manipulations of audio signals, the personalization of chords in polyphonic audio has not yet been addressed. We introduce a method that enables users to direct chord sequence modifications in recordings of popular songs without musical expertise. The proposed method mixes up the character of chord sequences in two or more audio signals, which led us to name the method *HarmonyMixer*.

Editing the chords in a musical recording of popular songs is particularly a challenging task, especially for a user without musical expertise, since it requires significant musical knowledge to recognize the existing chords and how they may be altered without causing undesired dissonances. On the implementation side, it also requires techniques for extracting and altering the audio corresponding to the chords in polyphonic and multi-instrument audio. Having separate tracks of multi-track audio recordings is

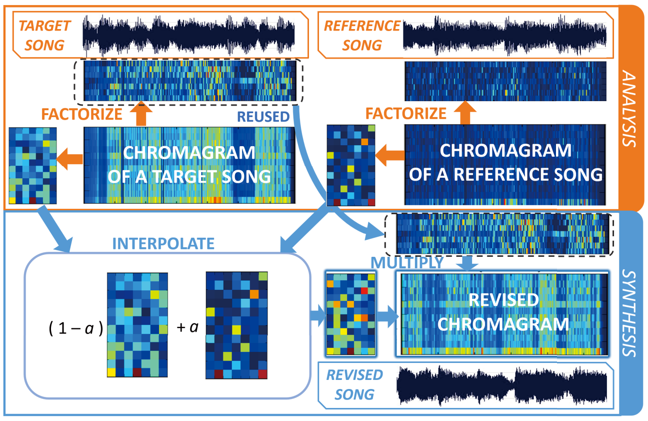

Figure 1. Process flow of *HarmonyMixer* mixing the character of chords among polyphonic audio. The method factorizes a matrix consisting of chroma vectors, interpolates the bases obtained through the factorization, and converts the spectrum to create an audible modification in the chords.

preferable when editing a chord sequence, but such tracks are often not available.

There are methods which aim to overcome difficulties of applying music-theoretical knowledge and enable users to modify music easily. *Drumix* [2], an active music listening interface for editing drum tracks, enables a user to modify drums in audio music signals. Concatenative synthesis approaches [3, 4, 5] were proposed to combine audio fragments in music database and enable the user to create or edit music easily by only choosing the audio fragments. *AutoMashUpper* [6], provided a powerful interactive tool to create mush-ups or arranging the mood of a song by converting and synchronizing songs. *ChordSequenceFactory* [7], which is our previous system sharing the motivation, can modify chord sequence described in chord symbols, but cannot be applied to acoustic music signals.

Developments in signal processing techniques seems to solve the problem on the implementation side of our task: modification of the polyphonic audio. In fact, as related works, multi-pitch analysis [8, 9, 10] and sound source separation techniques [11, 12, 13] have been proposed. Source separation techniques are also used for enhancement, suppression and re-panning of stereo mixtures [14]. A method for adaptive harmonization and pitch correction of polyphonic audio have been conducted by using multipitch analysis method [15]. However they only provide limited accuracies in estimating the pitches and separating sources where many tracks are mixed. Audio pitch-shifting using the constant-Q transform [16] has been proposed for key modulation of music audio. Phase vocoder based approach to pitch-shifting and harmonizing audio has also been conducted [17]. However they cannot be used to modify chord sequences in polyphonic and multi-instrument audio.

In this paper, we propose a method that enables a user to edit a *target song* by changing its chord sequence referring to a *reference song*. The *target song* is the song whose character of chords the user wants to modify. The *reference song* is the song which has the mood that the user wants to reflect in the *target song*. With our approach, users can edit the chord sequence in the *target song* by simply choosing a *reference song* and a mixture weight. The chord sequences of the *target song* and the *reference song* are analyzed automatically, and the chord sequence of the *target song* is edited reflecting the analyzed result of the *reference song*. Furthermore, the audio signal of the *target song* is converted in order to modify chords audibly, without using multi-pitch analysis methods. This method enables a user without musical expertise to modify a chord sequence since it is relatively easy for a user to choose a favorite song compared to analyzing and describing what kind of chord sequence that the user actually likes. In addition, the user can try various *reference songs* until he or she is satisfied with the result.

To achieve these functionality, we construct an analysis and synthesis framework of chords which can be applied to polyphonic music signals. The flow of our method is shown in Fig. 1.

In the analysis phase, audio signals of the *target song* and the *reference song* are converted into chromagrams (matrices which consist of chroma vectors). Extraction of the frequently observed pattern of pitch set in songs is mathematically formulated as Non-negative Matrix Factorization [8] of a chromagram matrices. Since we do not know exactly how many patterns exist in songs, the number of the patterns and the patterns themselves are simultaneously estimated by applying Bayesian non-parametric Non-negative Matrix Factorization (BNMF) [18] to our task.

In the synthesis phase, characters of chords which are extracted from the *target song* and the *reference song* are mixed up by the linear interpolation of bases. A chromagram is re-generated through multiplication of the interpolated bases and the original activations of the *target song*. Finally, the audio signal is converted to create audible modifications. This audio conversion is not exploiting multi-pitch analysis or source-separation techniques, but adding and reducing the sound by searching the optimal note sequence by using dynamic programming. The note sequence achieves modification with similar harmonic structure observed in the *target song*.

The structure of this paper is as follows: the discussions on how the analysis phase and the synthesis phase can be formalized are described in Section 2 and 3 respectively. Experiments are described in Section 4 and 5 for validating the analysis phase and synthesis phase of our method, and we report the result of the generated audio and discuss the further perspectives. Section 6 summarizes our findings.

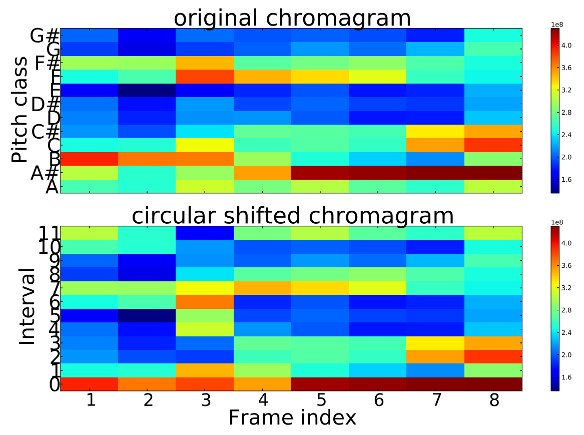

Figure 2. Example of chromagram and its circular shifted chromagram. It is created by a circular shift as whose maximum value within a chroma vector to be the first element (interval=0). With this feature, typical chord types are observed removing the transposition of chord driven by the change of the root note.

2. ANALYSIS OF CHORDS

2.1 Acoustic feature for chord characteristics

A chroma vector is an acoustic feature which has been shown to be effective in analyzing pitch content [6], and especially chords [19] in acoustic signals. We can obtain chroma vectors by summing up the frequency spectral amplitude in analysis frames, ignoring the octave differences but corresponding to the notes in chromatic scale (C, C#, ... , A, A#, B). Chroma vectors are effectively used in the audio chord recognition research [20, 21, 22].

When chroma vector analysis is conducted over a short time frame of the music, a set of pitches observed in the frame includes frequently observed pitches such as major third, minor third triad and dominant 7th. In addition, 9th and 11th notes of chord tones, and non-chord notes in the vocal melodies are also simultaneously observed.

The modification of relative intervals between the notes contained in a chord can achieve changes in character of it. We wish to represent these aspects of chord character with an acoustic feature vector, but a chroma vector with the original definition is insufficient, since the feature changes depending on the transposition of keys even the relative intervals between the notes are the same. We would like to find a feature that is robust to the difference derived from the transposition of chords.

Therefore, we apply a circular shift of the chroma vector so that the first element of the vector has the maximum value. The example of a chromagram and the circular shifted chormagram is shown in Fig. 2. We expect that ordinary chord types, such as major 3rd or diminished 7th, will appear in this shifted chroma vector as the peak values.

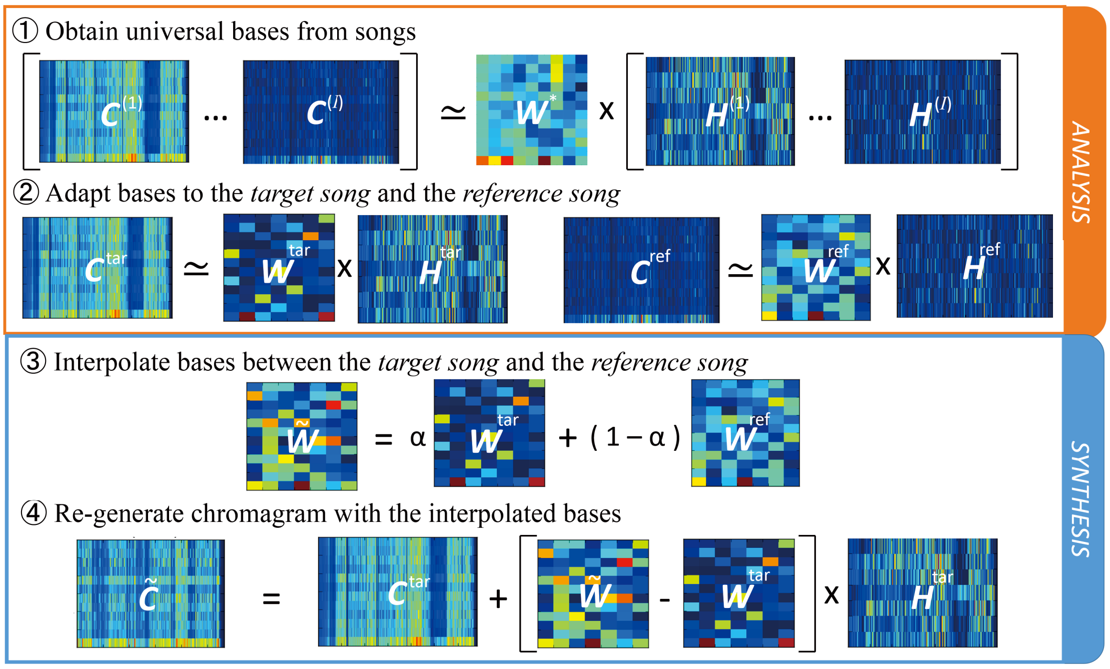
① Obtain universal bases from songs
C(1)
C(I)
W*
H(1)
H(I)
② Adapt bases to the target song and the reference song
Ctar
Wtar
Htar
Cref
Wref
Href
③ Interpolate bases between the target song and the reference song
W̃ = α Wtar + (1 − α) Wref
④ Re-generate chromagram with the interpolated bases
C̃ = Ctar + [W̃ - Wtar] x Htar
ANALYSIS
SYNTHESIS

Figure 3. Overview of our analysis and synthesis framework of chords based on Non-negative Matrix Factorization of circular shifted chromagram: character of chords used in the *target song* and the *reference song* are extracted as frequently observed patterns of pitch set with two step of matrix factorization. The first step extracts the universal bases which are common in songs in the database, and the second step extracts bases adapted to each *target song* and *reference song*. To reflect the character of chords in the *reference song* to the *target song*, the extracted bases are linearly interpolated, and the chromagram is re-generated by adding the modification derived from the interpolation of the bases.

2.2 Convex representation of the chroma vector

The circular shifted chromagram can be represented as a convex combination of frequently observed patterns of notes, with time varying non-negative weights. This can be understood by considering that there are common notes within chords with different chord types. For instance, triad notes are common in major chord and 7th chord. Let the chroma vector for each analysis frame $\boldsymbol{c}_n = (c_1 \cdots c_{12})^{\mathrm{T}} \in \mathbb{R}^{12}$, $n = 1, \cdots, N$, with analysis frame $(n-1)\lambda \leq t < n\lambda$ whose length is λ. In addition, let the frequently observed patterns of pitch set $\boldsymbol{w}_k = (w_1 \cdots w_{12})^{\mathrm{T}} \in \mathbb{R}^{12}$, where $k = 1, \cdots, K$ are the indices of patterns. $\boldsymbol{x}^{\mathrm{T}}$ denotes the transposition of a vector $\boldsymbol{x}$. These patterns can be represented with the same dimension and property as the chroma vector. The chroma vector is represented with the convex combination as follows:

$$\boldsymbol{c}_n = \sum_{k=1}^{K} h_{kn} \boldsymbol{w}_k, \tag{1}$$

where h_{kn} is the non-negative weight for adding the pattern $\boldsymbol{w}_k$ at analysis frame n.

2.3 Obtaining patterns from a chromagram

To discover the frequently observed patterns of the pitch set from a chromagram, we can apply matrix factorization techniques. Let the matrix collecting the circular shifted chroma vectors of the ith song in the database be $\boldsymbol{C}^{(i)} = [\boldsymbol{c}_1^{(i)} \cdots \boldsymbol{c}_{N^{(i)}}^{(i)}]$, where the length of the sequence is denoted as $N^{(i)}$. The factorization is:

$$\boldsymbol{C}^{(i)} \simeq \boldsymbol{W}^{(i)} \boldsymbol{H}^{(i)} \tag{2}$$

$$= \left[\boldsymbol{w}_1^{(i)} \cdots \boldsymbol{w}_K^{(i)}\right] \begin{bmatrix} h_{11}^{(i)} & \cdots & h_{1N^{(i)}}^{(i)} \\ \vdots & \ddots & \vdots \\ h_{K1}^{(i)} & \cdots & h_{KN^{(i)}}^{(i)} \end{bmatrix} \tag{3}$$

where $\boldsymbol{W}^{(i)}$ is the matrix collecting the patterns obtain from the ith song $(i = 1, \cdots, I)$, and $\boldsymbol{H}^{(i)}$ is the matrix collecting the weights $(h_{1n}^{(i)} \cdots h_{Kn}^{(i)})$ for adding the patterns at each analyzing frame as

$$\boldsymbol{H}^{(i)} = [(h_{11}^{(i)} \cdots h_{K1}^{(i)})^{\mathrm{T}} \cdots (h_{1N^{(i)}}^{(i)} \cdots h_{KN^{(i)}}^{(i)})^{\mathrm{T}}]. \tag{4}$$

The factorization of Eq. 2 is possible with Non-negative Matrix Factorization (NMF) [8, 12], since the components in the matrices are all non-negative. NMF techniques tend to factorize into bases containing the patterns that are observed frequently in the data. Through this property, we expect typical combination of notes in chords to be obtained by applying NMF to the shifted chromagram. In the NMF context, the patterns and the weights given above are called *bases* and *activations*, respectively.

2.4 Comparing the character of chords

We can compare the character of chords among songs in the database by looking at the difference in bases that represent the character of chords. The difference in tension notes, for example can be observed as the different location of peaks in the corresponding chroma vector in a pair of bases. However, if NMF is applied to a song individually, the chord type represented by $\boldsymbol{w}_k^{(i)}$ is not always the same as the type represented by $\boldsymbol{w}_k^{(j)}$. We need to align the order of bases with a common order among all the songs to compare them to each other.

To do this, we can utilize the NMF property that the factorization result depends on the initial values set for the iterative solution. The overview of the process is shown in Fig. 3. We first obtain the *universal* bases $\boldsymbol{W}^* = [\boldsymbol{w}_1^* \cdots \boldsymbol{w}_K^*]$, which are the bases obtained from all songs in the database, with an appropriate number of bases to capture the characteristic patterns. We then decompose the chromagram of each song, with the initial value for the bases equal to the *universal* bases, with the fixed number of bases. Through the NMF property that the result largely depend on settings of initial value before the decomposition, we expect the bases that each base in $\boldsymbol{W}^{(i)}$ to be similar and aligned to the ones in $\boldsymbol{W}^*$, but adjusted for each song.

We can obtain *universal* bases by concatenating the shifted chromagrams for the whole songs in the database (index $i = 1, \cdots, I$, the corresponding shifted chromagram $\boldsymbol{C}^{(i)}$), and factorizing it as:

$$\left[\boldsymbol{C}^{(1)} \cdots \boldsymbol{C}^{(I)}\right] \simeq [\boldsymbol{w}_1^* \cdots \boldsymbol{w}_K^*] \left[\boldsymbol{H}^{(1)} \cdots \boldsymbol{H}^{(I)}\right]. \quad (5)$$

To adjust the number of bases with a probabilistic prior, we use Bayesian non-parametric Non-negative Matrix Factorization [18] to factorize the concatenated chromagram.

2.5 Use of bases as features for genre classification

With the method described above, we can extract the frequently observed pattern of pitch set of a song. This means that we can analyze the character of a song in respect to the chord types. If we assume that the chord types are important to distinguish the genre of a song, we can use the obtained bases as features for classifying songs into genres. We investigate this point in the evaluation section with genre classification task by using the obtained bases as features.

3. SYNTHESIS OF CHORDS

3.1 Mixing the character of chords

With the bases obtained through the analysis described in Section 2.4, we can modify the character of the circular shifted chromagram by switching or interpolating the bases. The overview of the process is shown in Fig. 3. Let the bases obtained from the *target song* be $\boldsymbol{W}^{\mathrm{tar}}$, and the activations be $\boldsymbol{H}^{\mathrm{tar}}$. We linearly interpolate between $\boldsymbol{W}^{\mathrm{tar}}$ and $\boldsymbol{W}^{\mathrm{ref}}$ via:

$$\widetilde{\boldsymbol{W}} = \alpha \boldsymbol{W}^{\mathrm{tar}} + (1-\alpha)\, \boldsymbol{W}^{\mathrm{ref}} \quad (6)$$

where $\alpha \in [0, 1]$ is the interpolation factor.

Furthermore, let the bases obtained from the *reference song* be $\boldsymbol{W}^{\mathrm{ref}}$. We can re-generate the new shifted chromagram $\widetilde{\boldsymbol{C}}$ that reflects the character of the *reference song*, by multiplying the interpolated bases $\widetilde{\boldsymbol{W}}$ with the original activation matrix. The reconstruction of the shifted chromagram is calculated as:

$$\widetilde{\boldsymbol{C}} = \boldsymbol{C}^{\mathrm{tar}} + \left(\widetilde{\boldsymbol{W}} - \boldsymbol{W}^{\mathrm{tar}}\right) \boldsymbol{H}^{\mathrm{tar}}. \quad (7)$$

By inversely shifting the chroma vectors in $\widetilde{\boldsymbol{C}}$ using the preserved index of the maximum value in each chroma vector, we obtain the converted chromagram that reflects the character of the *reference* song.

3.2 Generating audio from a chromagram

The inverse problem of generating an audio signal when given a converted chromagram cannot be solved uniquely. This is because the chromagram representation lacks the octave differences and it is difficult to determine the octaves to reflect the modification of the converted chromagram. For instance, adding energy equally to the every octave in the energy spectrum will not result in sounding like note being added to the music signal. On the other hand, concentrating energy in specific octave will result in generating sine wave, which is not adequate for converting the music signal.

Considering that there are certain degree of freedom in transferring the modification of chromagram to the spectrum, we need constraints on the property of the adding and reducing sounds. We put constraints on the property of the audio signal generation so that the generated signal has an audible modifications in chord sequence and is not unnatural as a music signal.

3.3 Constraints on adding and reducing sounds to achieve chord modifications

The adding or reducing sounds for achieving the modification of chromagram should satisfy the following four constraints:

3.3.1 Constraint 1: similarity in chromagram

When the added or reduced sound is converted into a chromagram, it should be similar to the difference between before and after of the conversion of the chromagram by the multiplication of interpolated bases and the activation.

We introduce the positive component $\Delta \boldsymbol{c}_n^+$ and negative component $\Delta \boldsymbol{c}_n^-$ of the difference between the converted chroma vector $\boldsymbol{c}_n^*$ and the original chroma vector $\boldsymbol{c}_n$ at analysis frame n:

$$\boldsymbol{c}_n^* - \boldsymbol{c}_n = \Delta \boldsymbol{c}_n^+ - \Delta \boldsymbol{c}_n^-, \quad (8)$$

where all elements in $\Delta \boldsymbol{c}_n^+$ and $\Delta \boldsymbol{c}_n^-$ are constrained to be positive. Each $\Delta \boldsymbol{c}_n^+$ and $\Delta \boldsymbol{c}_n^-$ correspond to the chroma vector for adding and reducing sounds, respectively.

Let $\Delta \hat{\boldsymbol{c}_n}$ be a chroma vector calculated from the generated audio of adding and reducing notes. We can define a measure for evaluating the difference between the newly generated $\Delta \hat{\boldsymbol{c}_n}$ and the modification components of the chroma vector $\Delta \boldsymbol{c}_n^+$ or $\Delta \boldsymbol{c}_n^-$ by calculating the L^2 norm:

$$D\left(\Delta \hat{\boldsymbol{c}_n} | \Delta \boldsymbol{c}_n^+\right) = \left|\Delta \hat{\boldsymbol{c}_n} - \Delta \boldsymbol{c}_n^+\right|^2, \quad (9)$$

$$D\left(\Delta \hat{\boldsymbol{c}_n} | \Delta \boldsymbol{c}_n^-\right) = \left|\Delta \hat{\boldsymbol{c}_n} - \Delta \boldsymbol{c}_n^-\right|^2 . \tag{10}$$

By minimizing the value of this measure, we can obtain audio that is similar to the difference of the chromagram.

3.3.2 Constraint 2: harmonic structure

The adding or reducing sound should hold a harmonic structure similar to the one in the *target song*. We exploit the pitch-shifted audio signals as sources to obtain harmonic structures for the adding or reducing sounds. We first apply wavelet transform to the pitch-shifted audio signals to analyze the power of each semitone in the chromatic scales. The harmonic structure can be extracted by using a harmonic comb filter.

3.3.3 Constraint 3: pitch range

The pitch range of the adding or reducing sound should be the medium voice range, in order not to result in dissonances with the melody lines (typically the vocal parts) and the bass lines. To impose a constraint, we use a function r which takes a larger value for the sound with the medium pitch range:

$$r\left((f,s)_n\right) = \log\left[\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}\left(f - f_c\right)^2\right)\right], \tag{11}$$

where $(f,s)_n$ represent the audio fragment with the length of the analyzing frame in pitch f in Hz obtained from the sth pitch transposed source at the nth analyzing frame. f_c and σ are the mean frequency and the frequency deviation of the adding or reducing sounds, respectively. We left the parameters f_c and σ as user parameters to change the generated result. In the experiment, these parameters were set heuristically as $f_c = 220$ and $\sigma = 0.1$.

3.3.4 Constraint 4: sustained notes

The adding or reducing sound should not change rapidly in time, since they should be the consisting notes of edited chords. We can put constraint on the length of the sounds by imposing costs on rapidly changing ones. We introduce the cost function q:

$$q\left((f,s)_{n-1} \to (f,s)_n\right) = \begin{cases} 0 & \left((f,s)_{n-1} = (f,s)_n\right) \\ a & \text{otherwise} \end{cases} \tag{12}$$

where $\to$ represents the transition between two sounds (i.e. notes). The parameter $a \in [0,1]$ controls the smoothness of note sequences, which in our experiments we varied between 0.0 (no smoothness) to 0.7 (more smoothness).

3.4 Searching the optimal note sequences for modifying chords

To obtain the series of sound, we can search for the sounds which minimize the cost function J^+, in which all three constraints are combined:

$$\begin{aligned} J^+\left((f,s)_{n-1} \to (f,s)_n\right) &= \lambda_\mathrm{d} D\left(\Delta \hat{\boldsymbol{c}_n} | \Delta \boldsymbol{c}^+\right) \\ &+ \lambda_\mathrm{r} r\left((f,s)_n\right) + \lambda_\mathrm{q} q\left((f,s)_{n-1} \to (f,s)_n\right), \end{aligned} \tag{13}$$

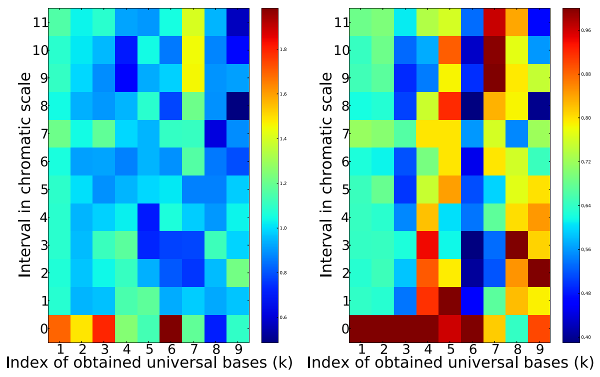

Figure 4. Bases obtained from a chromagram of all 100 songs by BNMF on the left. Normalized bases by dividing the values with the maximum value in each base are shown on the right. Typical chord types can be observed (minor in $k = 3$, major in $k = 6$).

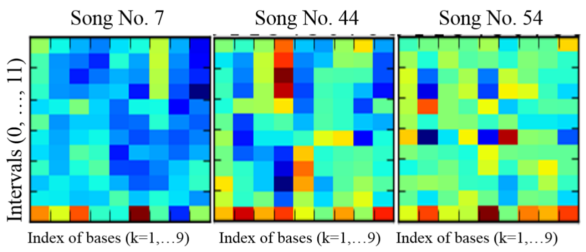

Figure 5. Bases obtained from three songs by NMF: the difference corresponding to the character of chords in each song can be observed.

where $\lambda_\mathrm{d}, \lambda_\mathrm{r}, \lambda_\mathrm{q}$ are parameters to control the weights of the constraints, which are heuristically set in the experiment. Searching for the adding or reducing sounds under these constraints can be formalized as:

$$\{(f,s)_n^*\}_{n=1}^N = \underset{\{(f,s)_n\}_{n=1}^N}{\operatorname{argmin}} \sum_{n=1}^{N} J^+\left((f,s)_{n-1} \to (f,s)_n\right), \tag{14}$$

where

$$\begin{aligned} & J^+\left((f,s)_0 \to (f,s)_1\right) \\ = & \frac{1}{\lambda_\mathrm{d} + \lambda_\mathrm{r}}\left(\lambda_\mathrm{d} D\left(\Delta \hat{c}_1 | \Delta \boldsymbol{c}_1^+\right) + \lambda_\mathrm{r} r\left((f,s)_1\right)\right) \end{aligned} \tag{15}$$

Since the sum of J^+ can be calculated recursively, we can use dynamic programming [23] to effectively search the optimal series of sound $\{(f,s)_n^*\}_{n=1}^N$.

4. EXPERIMENTS WITH THE ANALYSIS FRAMEWORK

We evaluated whether our approach using Non-negative Matrix Factorization can extract character of chord sequences from the chromagrams. First, we verify that chord notes with typical intervals such as major 3rd, minor 3rd are observed in the obtained bases. Furthermore, we check the differences in patterns for each song which are obtained by factorizing the chromagram. Second, we investigate whether the bases obtained with our method hold information regarding the genre of a song. The investigation is done in the context of genre classification.

4.1 Evaluation of extracting character of chords

Experiments were conducted with 100 songs of the RWC Music Database (RWC-MDB-P-2001 No. 01 - No. 100) [24]. The songs in the database all included vocal components and most include drum tracks. The audio signals were monophonic, sampled at 44.1 kHz with 16-bit encoding. The chromagrams were calculated with short-time Fourier transform using a Hamming window 0.8 s in length with 0.4 s overlap.

First, the chromagrams of 100 songs were concatenated into a single chromagram which was then factorized using BNMF. The initial number of bases was 50, which converged into 9 after 100 algorithmic iterations. The obtained 9 bases are shown in Fig. 4. Second, chromagrams for each of the songs were factorized by NMF with the initial bases set as those obtained by BNMF. We observed changes in the values of the bases after the NMF iterations. Selected examples of the bases obtained with the method are shown in Fig. 5. Distance measures for both BNMF and NMF were KL-divergences.

In the bases obtained when BNMF was applied to 100 songs, the consisting notes of the major 3rd chord, minor 3rd chord and the diminished chord were observed. This indicates that typical chord types can be learned without prior knowledge by applying BNMF to chromagrams. Changes on 7th or 9th notes were observed in the bases obtained on individual songs, which indicates that our methods are able to capture the character of chords.

4.2 Evaluation with genre classification

As explained in Section 2.5, we can expect the result of genre classification by using chroma vector patterns obtained with our method as features to be reasonably accurate.

We used 100 songs from RWC Music Database (RWC-MDB-G-2001 No. 1 - No. 100) with a genre label as ground truth for each song. The database consists of 33 clusters of genres. Each cluster contains 3 songs.

After converting the audio signals into 16-bit encoded, 44.1-kHz sampling, monaural signals, we applied short-time Fourier transform (frame length: 0.8 s, frame shift length 0.4 s, Hamming window) to them to obtain a time-frequency representation. The chroma vector patterns obtained from each song were used as features for genre classification. Hierarchical clustering was conducted from the distance matrix [25]. The method used for the clustering was the Ward method, which minimizes the squared sum of distances within a cluster. We chose the five largest clusters in the result and calculated a histogram of the genre labels in each cluster.

The classification results are shown in Fig. 6. In clusters 1, 3 and 5, the major genre labels observed were ”dance”, ”vocal” and ”classical”, respectively. The labels ”vocal” and ”dance” were especially condensed in particular clusters and rarely appeared in the other clusters.

These results indicate that the features we used in mixing the character of the chord sequence hold information related to genre, and they potentially can be used to edit the character of a chord sequence. Chroma vectors are already often used in genre classification tasks, and it is not surprising that a feature derived from a chroma vector is effective as shown in our experiment. However it was not obvious that the bases extracted from the chromagram still hold information of genre and suitable for editing the audio. We confirmed that the extracted features from chroma vectors hold information about the genre of each *reference song*.

5. EXPERIMENTS WITH SYNTHESIS FRAMEWORK

5.1 Preliminary experiment for modifying chords

We first conducted a preliminary experiment to find whether we can edit chords in polyphonic audio with our method by manually converting the chromagram. We prepared audio of an organ playing a major triad chord. The chromagram of this audio was manually converted to achieve the chord modification from a major triad chord to a seventh chord. The added seventh note was clearly generated with our method. The audio samples are available at `http://staff.aist.go.jp/s.fukayama/ICMCSMC2014/`.

These results indicate, in very simple condition of audio source (only one instrument played, and no drums or vocal) and with the correct modification on chromagram, our proposed method can modify chords in polyphonic audio signal. We confirmed audible changes in the generated audio. The converted sound shows the proof of our concept, although the quality of the added sounds were not satisfactory compared to that of the original instrument sounds.

5.2 Experiments with a song database

We then conducted experiments in a more realistic situation using audio including drums and a vocal part. The following generated audio are available at `http://staff.aist.go.jp/s.fukayama/ICMCSMC2014/`. We used RWC-MDB-P-2001 No. 63 from the RWC Music Database as the *target song* [24]. We chose RWC-MDB-G-2001 No. 32 as the *reference song* whose genre label is ”jazz”. The chromagram was converted through our proposed method by executing interpolation of obtained NMF bases.

Tuning parameter a in Eq. (12) to control the smoothness of adding sounds was varied from 0.0 to 0.7. With the variation of a, we obtained rapidly changing adding notes when $a = 0.0$, and stable notes but not corresponding to the chord modifications when $a = 0.7$.

As shown in Fig. 7, the differences of power in chromagram were approximately reproduced with the sound generated by our proposed method in both positive and negative components.

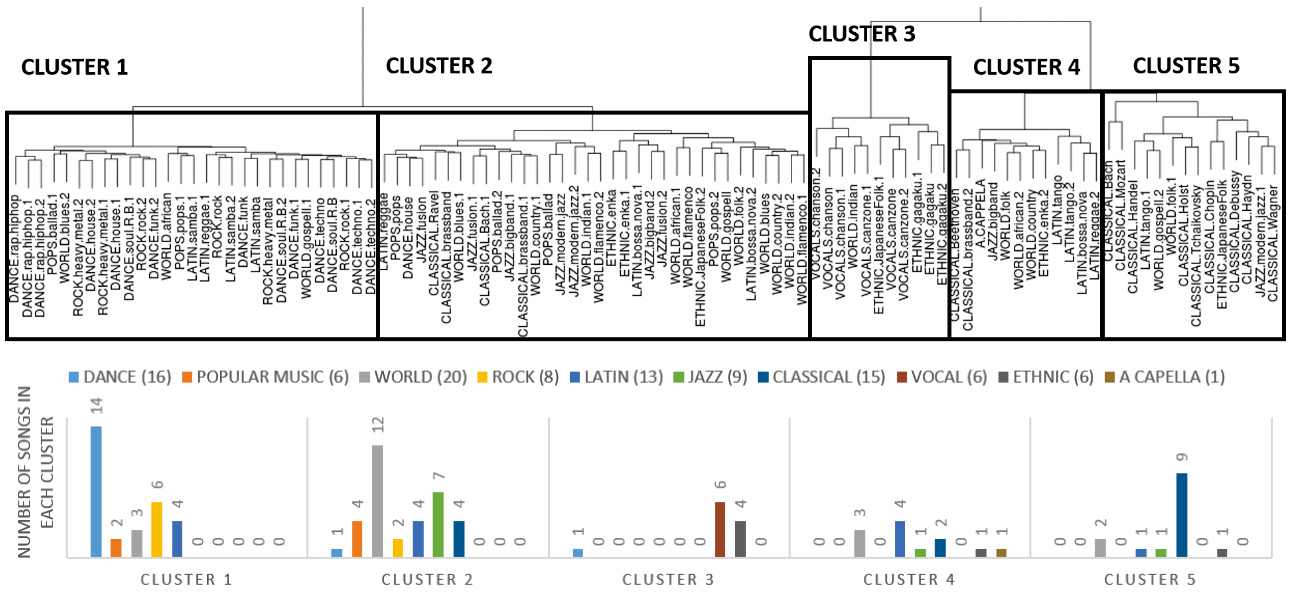

Figure 6. The result of hierarchical clustering (ward method, L^2norm) using bases obtained with factorization of chromagram as features. Dendrogram of the cluster (above) and histogram of number of songs for each genre in clusters (below). The database contained 3 songs per every 33 midium-large genre clusters and 1 a cappella song (100 songs as a whole). Number of songs for each genre is shown in the brackets of each legend. We found 5 relatively large clusters containing songs with similar genre labels.

6. CONCLUSION

We have described a method for modifying the chords of existing polyphonic music recordings. Our main contributions are:

- By simply choosing a *reference song*, user can reflect its character of chords in the *target song* by converting the chromagram by applying Non-negative Matrix Factorization,
- Polyphonic music signals of the *target song* can be converted to achieve the modification in the chromagram, by searching the sequence of pitches to add or reduce in the music signal of the *target song*.

We plan to put constraints on the harmonic structure of the adding sound to refine the sound quality. We aim to emphasize the more characteristic features derived from the *reference song*. The heuristic parameters for generating sounds from chromagrams are also planned to be investigated further via music theoretical discussions. We focused on mixing up the character of chords within two songs, but our theory can be applied to mixing among several songs or clusters of songs corresponding to the genre labels.

Acknowledgments

This work was supported in part by OngaCREST, CREST, JST. We thank Matt McVicar (AIST) and Matthew Davies (INESC TEC) for their helpful comments on this paper and Kazuyoshi Yoshii (Kyoto University) for providing support in the experiments.

7. REFERENCES

[1] M. Goto, “Active music listening interfaces based on signal processing,” in *Proceedings of ICASSP 2007*, 2007, pp. 1441–1444.

[2] K. Yoshii, M. Goto, K. Komatani, T. Ogata, and H. Okuno, “Drumix: An audio player with real-time drum-part rearrangement functions for active music listening,” *IPSJ Digital Courier*, vol. 3, pp. 134–144, 2007.

[3] R. B. Dannenberg, “Concatenative synthesis using score-aligned transcriptions,” in *Proceedings of ICMC 2006*, 2006.

[4] D. Schwarz, “Corpus-based concatenative synthesis: Assembling sounds by content-based selection of units from large sound databases,” *IEEE Signal Processing Magazine Special Section: Signal Processing for Sound Synthesis*, vol. 35, pp. 3–22, 2006.

[5] J. J. Aucouturier, F. Pachet, and P. Hanappe, “From sound sampling to song sampling,” in *Proceedings of ISMIR 2004*, 2004.

[6] M. E. P. Davies, P. Hamel, K. Yoshii, and M. Goto, “Automashupper: An automatic multi-song mashup system,” in *Proceedings of ISMIR 2013*, 2013, pp. 575–580.

[7] S. Fukayama, K. Yoshii, and M. Goto, “ChordSequenceFactory: A chord arrangement system modifying factorized chord sequence probabilities,” in *Proceedings of ISMIR 2013*, 2013, pp. 457–462.

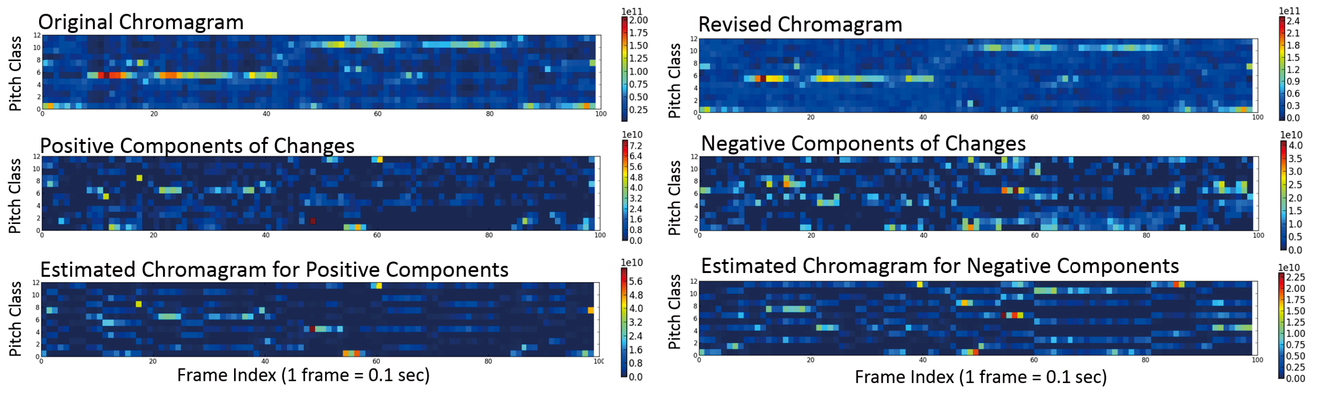

Figure 7. Original chromagram of the *target song* and the revised chromagram after the interpolation with the *reference song* are shown on the top. Two figures in the middle show the positive and the negative component of the chromagram difference from the original chromagram. The bottom two are the reproduced results with the proposed method, which show the both positive and negative components are approximately simulated.

[8] P. Smaragdis and J. Brown, "Non-negative matrix factorization for polyphonic music transcription," in *Proceedings of WASPAA 2003*, 2003, pp. 177–180.

[9] H. Kameoka, T. Nishimoto, and S. Sagayama, "A multipitch analyzer based on harmonic temporal structured clustering," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, pp. 982-994, no. 3, pp. 982–994, March 2007.

[10] A. Klapuri, "Multipitch analysis of polyphonic music and speech signals using an auditory model," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, pp. 255-266, no. 2, February 2008.

[11] E. Vincet, "Musical source separation using time-frequency priors," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 1, January 2006.

[12] T. Virtanen, "Monaural sound source separation by nonnegative matrix factorization with temporal continuity and sparseness criteria," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 3, pp. 1066–1074, March 2007.

[13] K. Itoyama, M. Goto, K. Komatani, T. Ogata, and H. G. Okuno, "Instrument equalizer for query-by-example retrieval: Improving sound source separation based on integrated harmonic and inharmonic models," in *Proceedings of ISMIR 2008*, 2008, pp. 133–138.

[14] C. Avendano, "Frequency-domain source identification and manipulation in stereo mixes for enhancemant, suppression and re-panning applications," in *Proceedings of WASPAA 2003*, 2003, pp. 55–58.

[15] M. Lagrange, G. Percival, and G. Tzanetakis, "Adaptive harmonization and pitch correction of polyphonic audio using spectral clustering," in *Proceedings of DAFx-07*, 2007, pp. 1–4.

[16] C. Schökhuber, A. Klapuri, and A. Sontacchi, "Audio pitch shifting using the constant-Q transform," *Journal of the Audio Engineering Society*, vol. 61, no. 7/8, pp. 562–572, 2013.

[17] J. Laroche and M. Dolson, "New phase-vocoder techniques for pitch-shifting, harmonizing and other exotic effects," in *Proceedings of WASPAA 1999*, 1999, pp. 91–94.

[18] M. Hoffman, D. Blei, and P. Cook, "Bayesian nonparametric matrix factorization for recorded music," in *Proceedings of ICML 2010*, 2010, pp. 439–446.

[19] T. Fujishima, "Realtime chord recognition of musical sound: a system using common lisp music," in *Proceedings of ICMC 1999*, 1999, pp. 464–467.

[20] A. Sheh and D. Ellis, "Chord segmentation and recognition using EM-trained hidden Markov models," in *Proceedings of ISMIR 2003*, 2003, pp. 183–189.

[21] M. Mauch and S. Dixon, "Simultaneous estimation of chords and musical context from audio," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 6, pp. 1280–1289, August 2010.

[22] M. McVicar, R. Santos-Rodríguez, Y. Ni, and T. De Bie, "Automatic chord estimation from audio: A review of the state of the art," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 2, pp. 556–575, February 2013.

[23] R. E. Bellman, *Dynamic Programming*. Dover Publications, 2003 (reprint).

[24] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, "RWC music database: Popular, classical, and jazz music databases," in *Proceedings of ISMIR 2002*, 2002, pp. 287–288.

[25] L. Kaufman and P. J. Rousseuw, *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley-Interscience, 2005.