Frequency Component Restoration for Music Sounds Using Local Probabilistic Models with Maximum Entropy Learning

Tomonori Izumitani, Kunio Kashino

NTT Communication Science Laboratories
3-1, Morinosato Wakamiya, Atsugi-shi, Kanagawa, Japan
izumi@eye.brl.ntt.co.jp, kunio@eye.brl.ntt.co.jp

Abstract

We propose a method that estimates frequency component structures from musical audio signals and restores missing components due to noise. Restoration has become important in various music information processing systems including music information retrieval. Our method comprises two steps: (1) pattern classification for the initial component-state estimation, and (2) state optimization by a generative model (Markov random fields; MRF). Throughout the method, we use a probabilistic model defined for each local region on a spectrogram. Unlike conventional MRF models, the model parameters are learned using a maximum entropy method. Experiments using artificial noisy sounds show that a combination of the above two steps improves the performance with respect to restoration accuracy and robustness, compared with the sole use of pattern classification or a generative model. The method achieves an F-measure greater than 0.6 even in periods where signals are replaced by noises. In addition, the method is shown to be effective even for audio signals of real instruments.

1. INTRODUCTION

Frequency components are time continuous peaks on a sound spectrogram such as a fundamental frequency (F0) component and its overtones. Estimating frequency component structure is an important task for various music information processing systems, because the structure is an important property of the instrumental and vocal sounds used in many music information processing systems, including a melody and bass-line extraction system[1] and a multiple F0 estimation system [2].

However, the estimation is not straightforward because the component structure in music signals can be easily broken by interfering factors generated by, for example, percussive sounds or noises. The restoration of such broken or missing components is therefore essential for many musical information processing tasks.

In this study, we focus on the problem of estimating frequency component structures from spectrograms of musical audio signals with noise. In particular, we consider a case where some frequency components are missing from musical audio signals due to noise.

Various methods that extract frequency components have been developed for partial tracking tasks, e.g. methods using the Kalman filter [3] and hidden Markov models [4], mainly in the context of sound analysis/synthesis. Here, we specifically address a case where the spectral peaks are intermittently hidden or erased by noise or other irregular sounds.

The method proposed in this paper is based on the Markov random field (MRF) [5] and the maximum entropy model (MEM) [6]. That is, the model represents local characteristics on a sound spectrogram by using a probabilistic model that is learned from data using MEM and estimates the optimal states by a generative approach [7].

However, a long time is needed for convergence and the estimation accuracy of the generative model is sensitive to initial states. To cope with this, here we introduce a pattern classification approach that reduces the number of iterations involved in the model. When we combine the pattern classification approach and the generative model, an initial state for the generative model is given by the pattern classification scheme. This improves the accuracy of the estimation and also reduces the calculation time.

Reyes-Gomez et al. proposed a method using probabilistic models that can restore spectral powers when some regions on a spectrogram are missing [8]. Our method is based on a learning scheme and so does not require explicit transition models given in advance. In addition, our method estimates frequency component structures without prior knowledge of noise locations.

This paper is organized as follows. Section 2 defines the problem. Section 3 describes the proposed methods. Then, we present our experimental results and discussion in section 4. Section 5 concludes the paper.

2. Estimating frequency component structure

Many musical sounds have frequency component structures, typically harmonic structures, that reflect the characteristics of instruments or vocal mechanisms. These structures are represented as peaks in the frequency direction extending in the time direction on a sound spectrogram.

In practice, a musical sound may contain various kinds of transient and irregular components, as found with percussive sounds and noises. In addition, the original instrumental or vocal sounds themselves have irregular components such as those appearing at the onset of a note. For simplicity, we treat these components as noises in this study.

Noises are represented as irregular patterns on a sound spectrogram [Fig. 1 (A)]. When peak extraction is applied to such signals, many transient power peaks are extracted simultaneously [Fig. 1 (B)].

The objective of this study is to obtain frequency components in a sound spectrogram of musical audio signals containing noise [Fig. 1 (C-1), (C-2)]. Solely suppressing noise components is not necessarily sufficient because information embedded in frequency

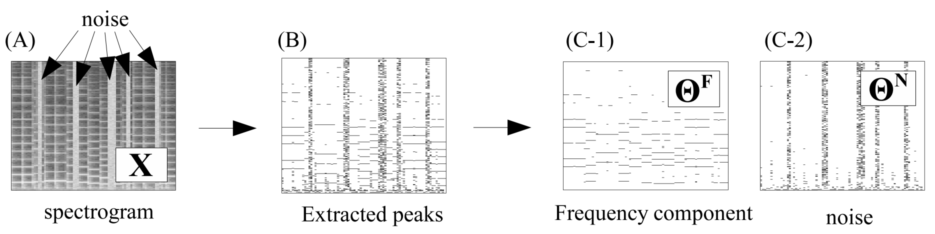

Figure 1: Estimation of frequency component structure. Spectrogram of musical audio signal (A), power peaks extracted from spectrogram (B), a frequency component structure extracted from power peaks, where hidden frequency components are interpolated (C-1), and noise is segregated from frequency component structure (C-2).

components is often hidden or erased by noises. Thus, we address the restoration of these hidden or erased components. To deal with both processes simultaneously, we consider the task as the estimation of the optimal frequency component structure when a spectrogram is given. It is formulated as the following *a posteriori* probability maximization model: in

$$\hat{\mathbf{\Theta}} = \arg\max_{\mathbf{\Theta}^{\mathrm{F}},\mathbf{\Theta}^{\mathrm{N}}} P(\mathbf{\Theta}^{\mathrm{F}}, \mathbf{\Theta}^{\mathrm{N}}|\mathrm{X}). \tag{1}$$

In this formulation, we have introduced an auxiliary matrix $\mathbf{\Theta}^{\mathrm{N}}$, which is also a binary state matrix that represents components originating in noise, to reflect the difference between the generation mechanism of observation X with and without noise.

In this paper, we represent the elements of matrices X, $\mathbf{\Theta}^{\mathrm{F}}$, and $\mathbf{\Theta}^{\mathrm{N}}$ as x_i, θ_i^{F}, and θ_i^{N}, respectively. Borrowing terminology from image processing, we call each pair comprising a frequency and time component, namely, a point i on the spectrogram, a "pixel". The two kinds of states that are represented by $\mathbf{\Theta}^{\mathrm{F}}$ and $\mathbf{\Theta}^{\mathrm{N}}$ are called the "frequency component state" and "noise state," respectively.

In the rest of this paper, we use notations $\mathbf{\Theta}$ and θ_i to simultaneously represent two state matrices, $\mathbf{\Theta}^{\mathrm{F}}$ and $\mathbf{\Theta}^{\mathrm{N}}$, and their elements, θ_i^{F} and θ_i^{N}.

The problem to be solved involves constructing an appropriate probabilistic model represented by eq. (1) and assigning an optimal state that satisfies eq. (1) from four possible states, namely, θ_i^{F} = 0/1 and θ_i^{N} = 0/1.

3. Method

3.1. Probabilistic models for microscopic characteristics

In order to build a probabilistic model that represents *a posteriori* probability in eq. (1), we assume that the state θ_i can be determined only by its neighborhood states and/or neighborhood observations.

This assumption is based on the following strong characteristics commonly possessed by a spectrogram of a musical audio signal: if a pixel is involved in a frequency component, neighboring pixels in the frequency direction tend to be excluded from any frequency components while those in the time direction tend to be included.

Therefore, the conditional probability of each pixel's state can be represented as follows:

$$P(\theta_i|\mathbf{\Theta}_{\setminus i}, \mathbf{X}) = P(\theta_i|\theta_{j\setminus i}, x_j, j \in G_i), \tag{2}$$

where G_i denotes a set of neighboring pixels around i including

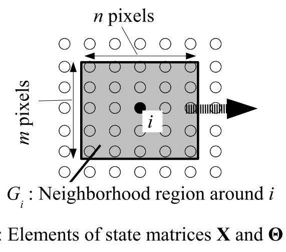
n pixels
m pixels
i
G_i : Neighborhood region around i
○ : Elements of state matrices $\mathbf{X}$ and $\mathbf{\Theta}$

Figure 2: A neighborhood region around a pixel at i.

i itself, $\mathbf{\Theta}_{\setminus i}$ denotes all elements of $\mathbf{\Theta}$ except for θ_i, and $\theta_{j\setminus i}$ denotes $\theta_j, (j \in G_i)$ except for θ_i.

In this study, we define the neighborhood as a rectangular region centered on the focusing pixel i, for simplicity (Fig. 2). The widths of a neighborhood region in the frequency and time direction are represented by m and n, respectively.

We propose a method comprising two steps for estimating $\mathbf{\Theta}$. The first step is based on pattern classification; this means that the probability of θ_i is represented as a function of only $x_j, (j \in G_i)$. We call this step "pixel classification."

In the second step, the probability is represented as a probability function using MRF [7], to find the optimum set of states by a generative approach. Although each of the above two steps can be used alone to estimate $\mathbf{\Theta}$, we find a combination of the two very promising as discussed below.

3.2. Pixel classification

The first step, the pixel classification, assumes that the state for every pixel is independent of each other. Therefore, probability of each θ_i in (2) can be written as

$$P(\theta_i|\theta_{j\setminus i}, x_j, j \in G_i) = P(\theta_i|x_j, j \in G_i), \tag{3}$$

and joint probability $P(\mathbf{\Theta}|\mathbf{X})$ is represented as a product of $P(\theta_i|x_j, j \in G_i)$ for all i.

The most probable state matrix $\mathbf{\Theta}$ can be estimated only by independently assigning a state that maximizes eq.(3) to each pixel i. This process classifies all the pixels into one for four states.

Because of the difficulties involved in manually constructing an appropriate probabilistic model, we utilize a supervised learning method to obtain $P(\theta_i|x_j, j \in G_i)$ from training data.

Various supervised learning methods can be employed for classification, including methods that have discriminant functions in non-probabilistic forms. Of these, we adopt the MEM, which can estimate a probability model. It is used for generative methods as shown in the following subsection. The MEM is described in detail in subsection 3.5.

Note that the less calculation is needed in this step than in the following step, because this step needs only one calculation of a conditional probability for each pixel.

3.3. Generative method using MRF

The pixel classification approach ignores relationships among states of multiple pixels. This may generate less probable combinations of states, especially in circumstances where the spectral peaks of frequency components disappear behind noises.

The MRF is a framework for representing the microscopic characteristics of graphs [5] in the same form as the conditional probability, $P(\theta_i|\theta_{j\setminus i}, x_j, j \in G_i)$, used in this study.

In this formulation, the probability cannot be calculated without determining neighboring states. Generative approaches are commonly used for estimating the most probable set of states from such a model. We use Gibbs sampling with simulated annealing as is often used with MRF. It starts with an initialized set of states $\mathbf{\Theta}^{\mathrm{I}}$ and iteratively generates θ_i according to $P(\theta_i|\theta_{j\setminus i}, x_j, j \in G_i)$, moving i and its neighborhood region. To find the states that yield the maximum value of the multi-peaked function, the method introduces the concept of "temperature" $T(t)$ that decreases along with iteration step t. We use $T(t) = C/\log(1+t)$ in accordance with the original MRF study [5].

As the iterations proceed, the probability peaks gradually sharpen if we regard the probability as a function proportional to $P^{1/T(t)}$.

Most MRF studies construct the probabilistic model by means of "potential functions," which adopt a small value for a state configuration that occurs easily. The conditional probability $P(\theta_i|\theta_{j\setminus i}, x_j, j \in G_i)$ is represented as an exponential of the potential function.

Usually, potential functions are manually defined based on simple rules that reflect *a priori* knowledge about the objects. However, it is difficult to design an appropriate function when we use various features that pixels within G_i have.

To overcome this problem, we use a supervised learning method, MEM, that directly estimates $P(\theta_i|\theta_{j\setminus i}, x_j, j \in G_i)$ from training data [7]. Using the method, we can easily extend the size of neighborhood region G_i, represented by m and n, without manually redesigning the potential function.

Hereafter, we refer to this generative process simply as "MRF."

3.4. Combination of pixel classification and MRF

A disadvantage of the generative process in MRF is that it often needs a long time for convergence. Moreover, an estimation is influenced by the initial state and it can be trapped in locally optimal states.

This is largely due to incorrectly estimated neighboring pixels. Thus, we want to avoid such a problem by finding an appropriate initial state matrix $\mathbf{\Theta}^{\mathrm{I}}$ that is close to the optimal one.

For this purpose, we propose a combination method that consists of the following steps:

1. Estimate a state matrix $\mathbf{\Theta}^{\mathrm{I}}$ by $P(\theta_i|x_j, j \in G_i)$ trained by the MEM (the pixel classification).
2. Set $\mathbf{\Theta}^{\mathrm{I}}$ as an initial state for Gibbs sampling in MRF.
3. Find local optimum states by applying Gibbs sampling with $C = 0$.

C is a constant for the temperature scheduling function $T(t) = C/\log(1+t)$. C=0 means states θ_i that increase the probability are accepted in the iterative process.

3.5. MEM

The MEM can directly estimate the posterior probability $P(\omega|D)$ from training data. Here, D denotes a data sample and ω a class or category. In this study, ω corresponds to θ_i and each sample D corresponds to a pixel i on a spectrogram, which is characterized by $\theta_{j\setminus i}$ and x_j, ($j \in G_i$).

To define $P(\omega|D)$, multiple features are extracted for a given D using feature functions $f_l(D,\omega)$, $(l = 1, 2, ..., F)$. Each function is defined by a combination of data D and a class ω, and it has binary values (0/1).

In MEM, $P(\omega|D)$ is represented as

$$P_\Lambda(\omega|D) = \frac{1}{Z(D)} \exp\left(\sum_l \lambda_l f_l(D,\omega)\right), \tag{4}$$

where $Z(D)$ is a normalization term and $\Lambda = (\lambda_1, ...\lambda_F)$ are model parameters.

The model parameters Λ are estimated so as to maximize the entropy of joint probability $P(\omega, D)$ under the condition that the expectations of $f_l(D,\omega)$ from $P(\omega|D)$ are equal to $\tilde{E}(f_l)$, which is obtained simply by counting the number of cases where $f_l(D,\omega)$ = 1 in the training data, for all l.

Generalized iterative scaling (GIS) or the improved iterative scaling (IIS) method, which are kinds of hill-climbing methods, are commonly used to estimate Λ [6].

3.6. Defining feature function for MEM

Observations $x_j, (j \in G_i)$ are used to define feature functions f_l for pixel classification. For MRF, neighboring states around i, namely, $\theta^{\mathrm{F}}_{j\setminus i}$ and $\theta^{\mathrm{N}}_{j\setminus i}$ are used in addition to x_j.

In preparation for defining f_l, we introduce an auxiliary feature function $g_{l'}(D)$, which also takes a binary value and does not depend on ω. In the following two subsections, we propose auxiliary feature functions for the pixel classifier and MRF.

3.6.1. Auxiliary feature function for pixel classification

For pixel classification, auxiliary feature functions $g_{l'}$ are defined only by $x_j, (j \in G_i)$. Figure 3 (A) shows the procedure for obtaining $g_{l'}$.

First, we use spectral powers $x_j, (j \in G_i)$, which have real values. For each j, x_j is quantized into on/off states of B bins. Namely, one of B bins, $x_j^1, ..., x_j^B$, is set at 1 according to the value of x_j and the rest are set at 0. For example, when B=3 and 0.33< x_j <0.67, auxiliary feature functions (x_j^1, x_j^2, x_j^3) corresponding to the three bins become $(0, 1, 0)$, where every x_j is normalized to 0 - 1. Note that a feature function can only have a binary value in the MEM.

Second, we introduce binary value $\mathrm{x^P}_j$, which has a value of 1 if a pixel j forms a power peak in the frequency direction on the spectrogram. This is introduced because spectral power peaks provide rich information about frequency components, especially in a clear sound without noise.

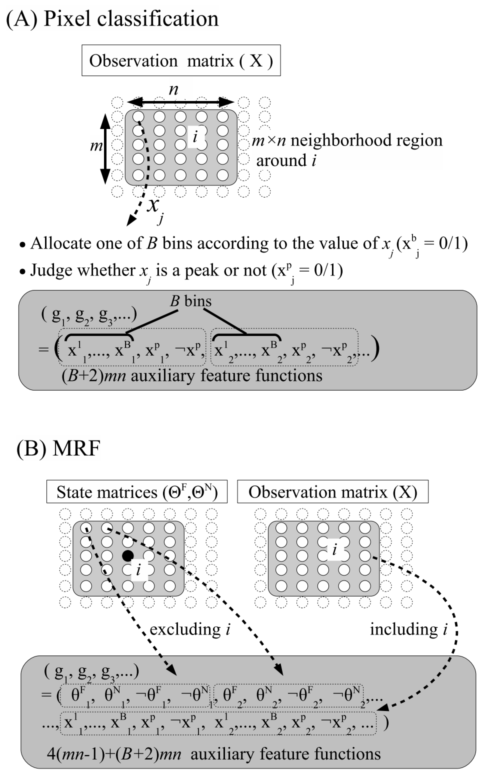
(A) Pixel classification
Observation matrix (X)
n
m
i
m×n neighborhood region around i
x_j
• Allocate one of B bins according to the value of x_j (x^b_j = 0/1)
• Judge whether x_j is a peak or not (x^p_j = 0/1)
B bins
(g_1, g_2, g_3,...)
= (x^1_1,..., x^B_1, x^p_1, ¬x^p_1, x^1_2,..., x^B_2, x^p_2, ¬x^p_2,...)
(B+2)mn auxiliary feature functions
(B) MRF
State matrices (Θ^F,Θ^N)
Observation matrix (X)
excluding i
including i
(g_1, g_2, g_3,...)
= (θ^F_1, θ^N_1, ¬θ^F_1, ¬θ^N_1, θ^F_2, θ^N_2, ¬θ^F_2, ¬θ^N_2,...
..., x^1_1,..., x^B_1, x^p_1, ¬x^p_1, x^1_2,..., x^B_2, x^p_2, ¬x^p_2, ...)
4(mn-1)+(B+2)mn auxiliary feature functions

Figure 3: Making auxiliary feature functions $(g_1, g_2, ...)$ from the neighborhood region around a pixel i. (A) Auxiliary feature functions for the pixel classification are extracted only by observation matrix $\mathbf{X}$. (B) For MRF, state matrices $\mathbf{\Theta}^{\rm F}$ and $\mathbf{\Theta}^{\rm N}$ are used in addition to $\mathbf{X}$.

We additionally introduce the negative $\neg {\rm x^p}_j$, which corresponds to each ${\rm x^p}_j$. This value is used to keep the sum of all $f_l(D, \omega)$ in the training samples constant and thereby simplifies the MEM learning using GIS.

This procedure yields $(B$+2$)mn$ auxiliary feature functions.

3.6.2. *Auxiliary feature function for MRF*

For MRF, auxiliary feature functions $g_{l'}$ include information about neighboring states excluding the state of pixel i, in addition to observations.

Figure 3 (B) shows auxiliary feature functions for MRF. A frequency component state and noise state, $\theta^{\rm F}_{j\setminus i}$ and $\theta^{\rm N}_{j\setminus i}$, $(j \in G_i)$ are used for $g_{l'}$ in addition to functions defined for pixel classification. The negative values, $\neg\theta^{\rm F}_{j\setminus i}$ and $\neg\theta^{\rm N}_{j\setminus i}$, which correspond to $\theta^{\rm F}_{j\setminus i}$ and $\theta^{\rm N}_{j\setminus i}$, are also introduced as auxiliary feature functions given for pixel classification. Consequently, 4(mn-1)+(2+B)mn auxiliary feature functions are obtained.

3.6.3. *Generate feature function*

By using the above definitions, a feature function $f_l = f_{\omega^{\rm F}\omega^{\rm N}l'}$ can be defined for all four combinations of frequency component and noise states, $\omega^{\rm F} = 0/1$ and $\omega^{\rm N} = 0/1$, using auxiliary feature functions $g_{l'}$ as follows:

$$f_{\omega^{\rm F}\omega^{\rm N}l'}(D, \theta^{\rm F}_i, \theta^{\rm N}_i) = \begin{cases} g_{l'}(D) & \text{if } \theta^{\rm F}_i = \omega^{\rm F} \text{ and } \theta^{\rm N}_i = \omega^{\rm N}, \\ 0 & \text{otherwise.} \end{cases} \quad (5)$$

4. Results and Discussion

We evaluated the proposed method described in the previous section in terms of frequency component restoration accuracy. We first checked each step of the method under controlled conditions using artificially synthesized audio signals. Then, we applied the method to musical phrases generated from real instruments.

4.1. Preparing training and test data from artificial sounds

Artificial sounds were synthesized for use as the ground truth for the frequency component and noise state matrix. The ground truth is needed for the MEM learning procedure and evaluation.

First, we prepared different tones for training and testing. Sounds close to a sawtooth wave with ten harmonics were used for training and those close to square wave with five harmonics were used for testing. We then generated two different 3.5-sec musical phrases using these sounds by connecting seven 500-ms single notes for training and test data.

Next, for each musical sound, two parts of the original 3.5-sec musical sound were replaced by 200- and 300-ms-long Gaussian noises. The noise locations were different. With this procedure, traces of frequency components entirely disappeared within the noise regions.

All audio signals were sampled at Fs = 11,025 Hz. To create observation matrices X, sound spectrograms were generated using short-time Fourier transforms (STFT). The window size was set at 1024. We tried four kinds of shift for the window, L = 128, 256, 512, and 1024.

The spectral power values in decibels were normalized into a range of [0, 1] from the power of 100 neighboring components in the frequency direction. These were used in composing the observation matrix X.

The ground truth of the frequency components was created based on fundamental frequencies of the generated phrase and the harmonics that each tone should have. The ground truth of the noise components was created from spectral peaks within noise periods and peaks not included in frequency components.

Throughout the experiments, we used B = 3 for quantization of spectral powers. It was chosen by preliminary experiments using the same data set as [7]. For the MEM learning, 15,000 pixels in a spectrogram were randomly selected for each L.

4.2. Evaluation of accuracy

We examined the performance of the method for the following four experiment patterns:

1. Pixel classification (PC)
2. MRF with $C = 1$ for slow convergence (MRF1) [7]
3. MRF with $C = 0$ for fast convergence (MRF2)[7]
4. Combination of pixel classification and MRF (PC+MRF),

where each term in parentheses is an abbreviation of the corresponding experiment. For MRF, we tried two kinds of convergence

Table 1: The number of pixels for various STFT window shifts (L). Two columns labeled with "freq. c." show the number of pixels involved in the frequency components.

Window shift	Whole period (3.5s)		Noise period (0.5s)	
(L)	total	freq. c.	total	freq. c.
128	150,822	1,470	23,085	225
256	75,411	735	12,312	120
512	37,962	370	6,156	60
1024	18,981	185	3,591	35

Table 2: The best estimation of frequency components in the noise region for each experimental pattern. Values in parentheses denote the F-measure for whole 3.5-sec periods.

	L	m	n	F-measure	F-measure(PC+MRF)
PC	1024	3	7	0.66 (0.90)	0.64 (0.88)
MRF1(C=1)	1024	3	11	0.53 (0.84)	0.65 (0.88)
MRF2(C=0)	1024	3	11	0.51 (0.77)	0.65 (0.88)
PC+MRF	512	5	11	0.71 (0.88)	-

rates that were controlled by a constant C of temperature scheduling function $T(t)$. Initial states $\mathbf{\Theta}^{\mathrm{I}}$ for both experiments MRF1 and MRF2 were randomly generated.

We tried various sizes of neighborhood region for all the experiment patterns, namely, m=3, 5, 7, 9, 11 and n=3, 5, 7, 9, 11.

For generative processes in the experiments MRF1, MRF2, and PC+MRF, the convergence was judged by the number of generated pixel states that were different from the previous iteration step. We stopped the iteration if the number became less than 0.01 % of the total pixels.

Table 1 shows the total number of evaluated pixels and the number of pixels that composed the frequency components within the whole period of the test phrase and the periods where Gaussian noises were added. Noise periods are defined if at least half the samples within an STFT window originate in added Gaussian noise.

The number of pixels that compose frequency components, which are treated as positive patterns, constitute only about 1% of the total pixels. In such cases, performance is often measured by precision (P) and recall (R), especially in information retrieval tasks.

We adopt F-measure (F) as an evaluation measure. It is defined as the harmonic mean of precision and recall, and it takes a value in the [0, 1] range. These measures are defined as

$$P = \frac{TP}{TP+FP}, \quad R = \frac{TP}{TP+FN}, \quad F = \frac{2PR}{P+R},$$

where TP, FP and FN denote the number of true positive, false positive and false negative samples, respectively.

Table 2 shows the frequency component estimation performance within the 0.5-sec noise region, with parameter values yielding the best F-measure for each experimental pattern.

The best performance of all is obtained when the experiment PC+MRF is used with L=512, m=5, and n=11. It achieves an F-measure of 0.71, where precision is 0.90 and recall is 0.58. This means that more than half the frequency components, which had disappeared due to noise, were restored and 90% of the estimated frequency components were actually correct.

The right column of Table 2 shows the performance of the experiment PC+MRF using the same parameters with the corresponding method. Also in these cases, the combination method outperforms MRF and it yields almost the same performance as pixel classification. This indicates that the combinatorial method is robust with respect to parameter variation.

Figure 4 shows the performance obtained with various parameter settings around L=512, m=5, and n=11 which yields the best performance. Figure 4 (A) shows the relationship between STFT window shift (L) and F-measures. When L=128 or 256, F-measure for every experimental pattern was suppressed at a low level. In each case, the pixel size in the time direction is very short, 12 ms and 23 ms, respectively. This indicates that the local probability models are unsuitable when a neighborhood region is small compared with the length of audio events such as a noise or musical notes.

In Figure 4 (B) and (C), various sizes of neighborhood region were employed. The tendency of the experiment PC/PC+MRF was quite different from that of the experiment MRF1/MRF2. In particular, PC and PC+MRF were found to be very robust with respect to variation of n.

We observed the following characteristics throughout all the experiments.

- The combination method (PC+MRF) worked much better than the MRF based method as long as L and n were not too small.
- The MRF generative process in the combination method improved the estimation only by the pixel classification in almost all cases.
- Estimation only by MRF (MRF1/2) was unstable for variations in the embedded parameters.

The only difference between the experiment MRF2 and PC+MRF is the initial states $\mathbf{\Theta}^{\mathrm{I}}$ in the generative process. This indicates that the appropriate setting of the initial states for the MRF generative process is practically essential and pixel classification can estimate the initial state.

Appropriate initial state setting also reduces the number of iteration steps needed for convergence. When the parameters were chosen as L=512, m=5, n=11, the experiment MRF1 took 279 steps, MRF2 18, and PC+MRF only 5 steps, while the computational cost for each step was identical for all three methods.

4.3. Application to real instrumental sounds

We also applied the proposed method to real instrumental sounds. We chose acoustic sounds of piano, violin, flute, trumpet, marimba, and alto (vocal) played with the normal playing style from the RWC Musical Instrument Sound Database (RWC-MDB-I-2001 No. 01-50) [9]. We then generated 7.5-sec musical phrases with five short Gaussian noises, 1.1-sec in total, for each instrumental sound.

We defined the ground truth data from power peaks close to the fundamental frequency and its harmonics that were expected from the notes of the musical phrase.

The same experimental conditions and the same training data as the previous subsection were used for the frequency component estimations.

Table 3 shows the performance of each method when applied to real instrumental sounds using two parameter sets: (A) L=1024, m=5, n=7 and (B) L=512, m=5, n=11. In the table, F-measure

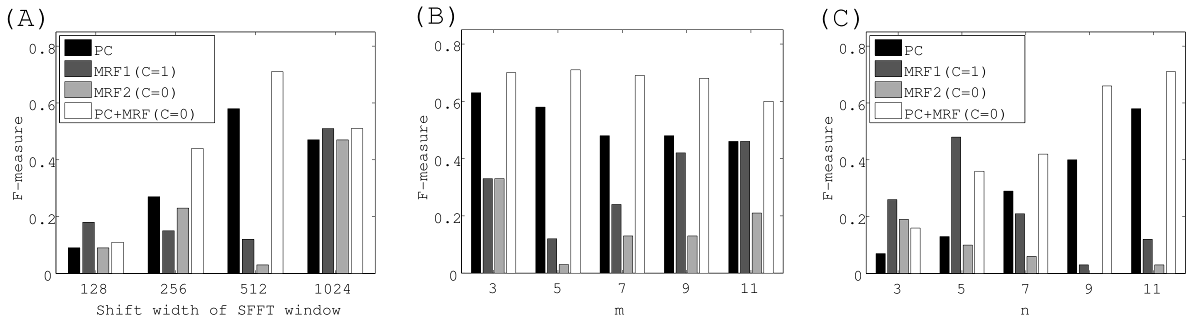

Figure 4: Comparison of F-measure of four experimental patterns. (A) Changing shift width of STFT L where m=5 and n=11. (B) Effect of neighborhood region size in the frequency direction where L=512, n=11. The notation is the same as (A) and (C). (C) Effect of neighborhood region size in the time direction where L=512, m=5.

Table 3: F-measure values for the restoration of the frequency components in noise region using real instrumental sounds.

(A) L=1024, m=5, n=7

instrument	PC	MRF1	MRF2	PC+MRF
piano	0.35 (0.73)	0.34 (0.71)	0.34 (0.70)	0.38 (0.73)
violin	0.21 (0.55)	0.19 (0.53)	0.19 (0.52)	0.22 (0.54)
flute	0.26 (0.50)	0.28 (0.49)	0.23 (0.45)	0.26 (0.49)
trumpet	0.28 (0.52)	0.29 (0.50)	0.28 (0.49)	0.30 (0.51)
marimba	0.10 (0.26)	0.08 (0.22)	0.11 (0.25)	0.08 (0.24)
alto	0.15 (0.33)	0.13 (0.30)	0.08 (0.23)	0.13 (0.31)

(B) L=512, m=5, n=11

instrument	PC	MRF1	MRF2	PC+MRF
piano	0.24 (0.74)	0.19 (0.60)	0.10 (0.50)	0.32 (0.74)
violin	0.21 (0.60)	0.10 (0.44)	0.01 (0.25)	0.25 (0.59)
flute	0.18 (0.54)	0.11 (0.37)	0.07 (0.26)	0.23 (0.54)
trumpet	0.26 (0.58)	0.13 (0.39)	0.01 (0.23)	0.32 (0.56)
marimba	0.13 (0.29)	0.00 (0.11)	0.00 (0.11)	0.16 (0.30)
alto	0.09 (0.37)	0.06 (0.26)	0.05 (0.13)	0.12 (0.36)

values for noise regions (1.1 sec) are shown and those for whole 7.5 sec sounds are shown in parentheses.

The total performance degraded compared with experiments using artificial sounds. This degradation occurred especially in terms of recall values, which decreased by around 0.1 - 0.2, under most conditions. This is because the method does not capture the complicated frequency component structures possessed by a real sound but not by the training data.

In Table 3 (A), similar results were obtained in each step of the method. However, especially with piano and trumpet data, the experiment PC+MRF achieved an F-measure greater than 0.3, although these sounds were not included in the training data. Adding real instruments' sounds to the training data may improve the performance.

Table 3(B) shows the results obtained under different conditions. In contrast to the experiments MRF1/MRF2, which captured very few frequency components, the experiment PC+MRF (and PC) performed as stably as under the conditions of Table 3(A).

5. Conclusion

We have proposed a method that restores missing frequency components from musical audio signals containing noise. The method exploits probabilistic models that represent local characteristics on a sound spectrogram. Specifically, the method features the combined use of pattern classification and a generative model. Test results showed that the combination method achieves better performance than the single uses of the pattern classification or the generative model from the viewpoints of restoration accuracy and robustness. Future work will include the application of the proposed method to real tasks such as music information retrieval and signal restoration.

6. References

[1] M. Goto, "A real-time music-scene-description system: predominant-F0 estimation for detecting melody and bass lines in real-world audio signals," *Speech Communication*, vol. 43, pp. 311–329, 2004.

[2] A. P. Klapuri, "Multiple fundamental frequency estimation based on harmonicity and spectral smoothness," *IEEE Trans. Speech and Audio Processing*, vol. 11, no. 6, pp. 804–816, 2003.

[3] A. Sterian and G. H. Wakefield, "A model-based approach to partial tracking for musical transcription," in *Proc. of SPIE98*, 1998, pp. 171–182.

[4] Ph. Depalle, G. García, and X. Rodet, "Tracking of partials for additive sound synthesis using hidden Markov models," in *Proc. of ICASSP93*, 1993, pp. I225–I228.

[5] S. Geman and D. Geman, "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. PAMI-6, no. 6, pp. 721–741, 1984.

[6] S. D. Pietra, V. D. Pietra, and J. Lafferty, "Inducing features of random fields," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 19, no. 4, pp. 380–393, 1997.

[7] T. Izumitani and K. Kashino, "Frequency component restoration for music sounds using a Markov random field and maximum entropy learning," in *Proc. of ICASSP2006*, 2006.

[8] M. Reyes-Gomez, N. Jojic, and D. Ellis, "Deformable spectrograms," in *Proc. of AI & Statistics 2005*, Robert G. Cowell and Zoubin Ghahramani, Eds., 2005, pp. 285–292.

[9] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, "RWC music database: Music genre database and musical instrument sound database," in *Proc. of ISMIR2003*, 2003, pp. 229–230.