REPORT: SEPARATION OF DRUMS FROM MUSIC SIGNALS

Course: Introduction to Audio Processing
Contributors: 50357871: Anh Pham
50359358: Minh Tran

Table of Contents

Introduction

In the paper ***Separation of a monaural audio signal into harmonic/percussive components by complementary diffusion on spectrogram,*** the authors Nobutaka Ono, Kenichi Miyamoto, Jonathan Le Roux, Hirokazu Kameoka, and Shigeki Sagayama have proposed a simple algorithm to separate a monaural audio signal into harmonic and percussive components. We aim to recreate the method and test the effects of different variables using Python in this experiment. [1]

Background and Overview [1]

Nowadays, it is evident that music is becoming more and more critical to people's lives. Smartphones are much more accessible, which means music streaming services and music apps like Spotify, Youtube Music, Apple Music, and Shazam have much more users than before. The need to process a database of millions of songs and deliver to millions of users has urged musical signal processing development. Various subject tasks have been discussed like audio music retrieval, audio onset detection, multiple fundamental frequency estimation, etcetera. [1]

The music signal usually consists of two different components: harmonic and percussive. The harmonic component is the pitched sound, and the percussive component comes from unpitched instruments. [2] Because they have many different spectral structures, their simultaneous presence will make some tasks harder. Thus, separation is desirable. [1] If the task requires processing pitched sounds, for example, active listening, audio sources remixing, and pre-processing, we will be interested in having only the harmonic component. In contrast, if we are doing percussive sound analysis, such as drum beat detection, decreasing the harmonic component is probably more ideal. [3]

In the paper, the authors present the formulation of the separation as an optimization problem and derive the fast iterative solution to it by the auxiliary function approach. [1] We will program the computations and present the results using numpy, librosa, matplotlib, and soundfile. We will also test multiple gammas and k_maxs and examine their effects by studying the signal-to-noise ratio related to corresponding variables.

Implementation

The separation of harmonic and percussive is implemented in the following steps:

- First, read the audio sample (in .wav format) using the Librosa library to obtain an array and sampling frequency.
- A range-compressed version of the power spectrum (W) is calculated using STST of the input signal and range compression coefficient (γ), using equation (24) from [1]. The optimal value of γ will be discussed in section Result analysis.
- Power spectrograms of harmonic (H) and percussive (P) are created. It's initial value equals half of the original signal's power spectrogram.
- Constants $k_{\max}$ – number of iterations that will be used to search for the optimal power spectrograms and $\sigma_{\mathrm{H}}, \sigma_{\mathrm{P}}$ – weights of the horizontal and vertical smoothness are initialized. σ_{H} and σ_{P} is set to 1, while the optimal value of $k_{\max}$ will be discussed in section Result analysis.

- A while loop is created, where the values of harmonic (H) and percussive (P) are updated with delta (Δ), which is found using helper function *calculate_delta* that is used to implement
- Once the while loop is completed, H and P are binarized to arrays of zeros and W
- Before H and P are written into .wav files, Inverse Short Time Fourier Transform (ISTFT) is performed to those components using Librosa library.

Result analysis

Spectrogram

The difference between the components can be illustrated using power spectrograms. Below are three spectrograms of the provided audio file police03short.wav, with the original, harmonic, and percussive signals. The pre-set constants are $\gamma = 1, k_{\max} = 20$

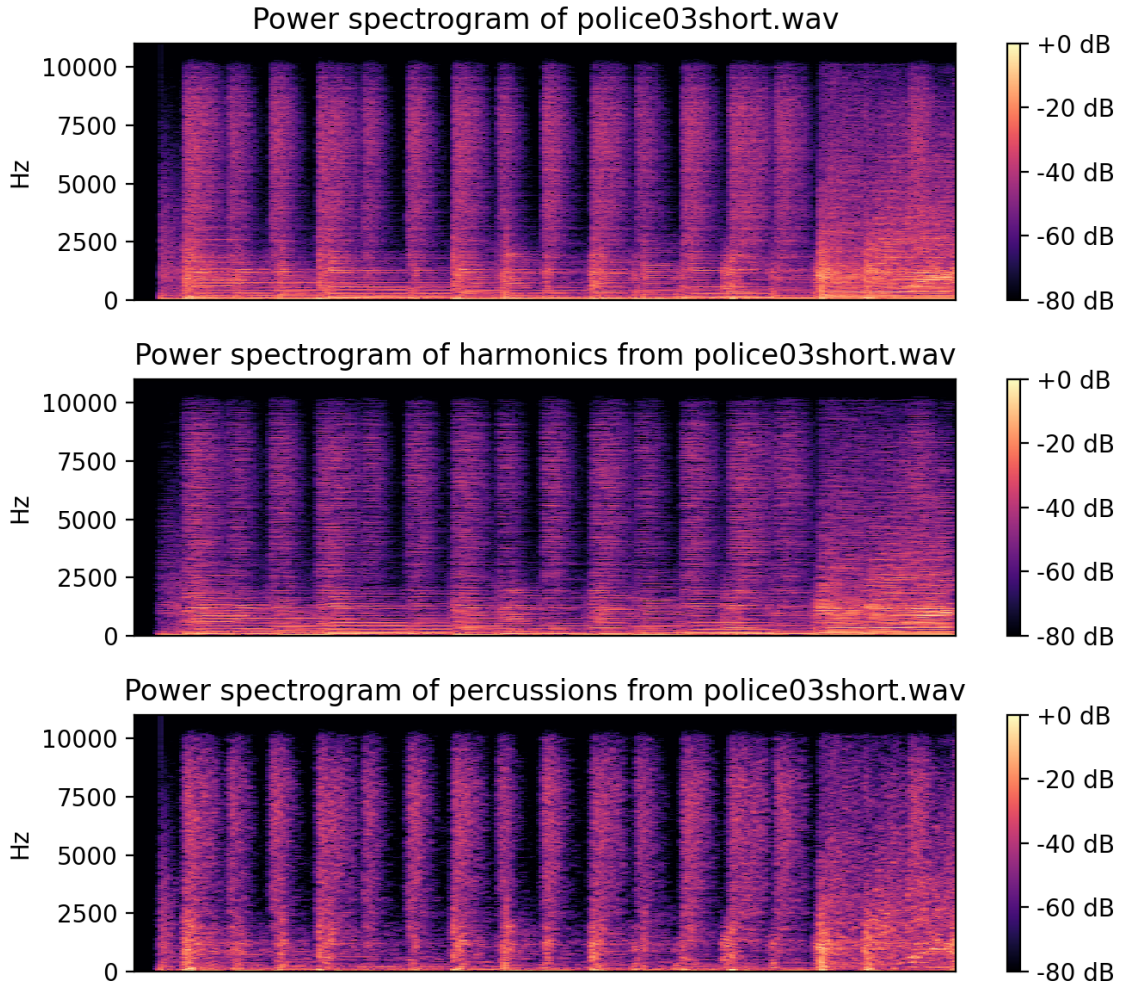

Figure 1: Spectrograms of the original and the components of police03short.wav

From the figure, we can observe that the harmonic component forms parallel ridges, while the percussive component is concentrated in a short time frame, i.e., having distinguishable vertical ridges. The spectrogram of the original audio signal is the sum (combination) of the two components.

Different values of γ are tested with the audio, but within a certain limit: $0 \leq \gamma \leq 1$. For each value of γ, two spectrograms are drawn, one for each component. Figure 2 illustrates the spectrograms with 5 different values of γ, while $k_{\max} = 20$.

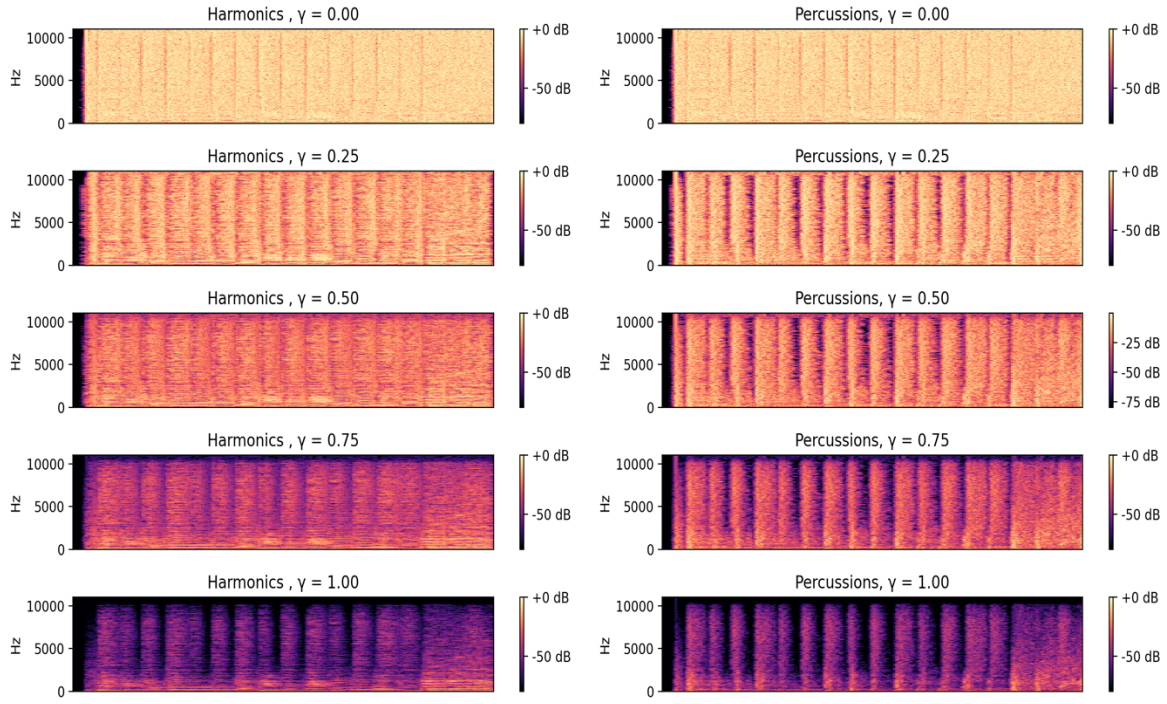

Figure 2: Harmonics and percussions with different range of compression values

It is clear to observe that the different between the components is clearer when γ increases. Hence, the optimal value for γ is $\gamma = 1$.

On the other hand, different values of number of iterations are tested, each produced one spectrogram for each component. Here we set the maximum number of iterations to be 50 and there is 5 different number of iterations. As proved above, the value of γ will always be set to 1.

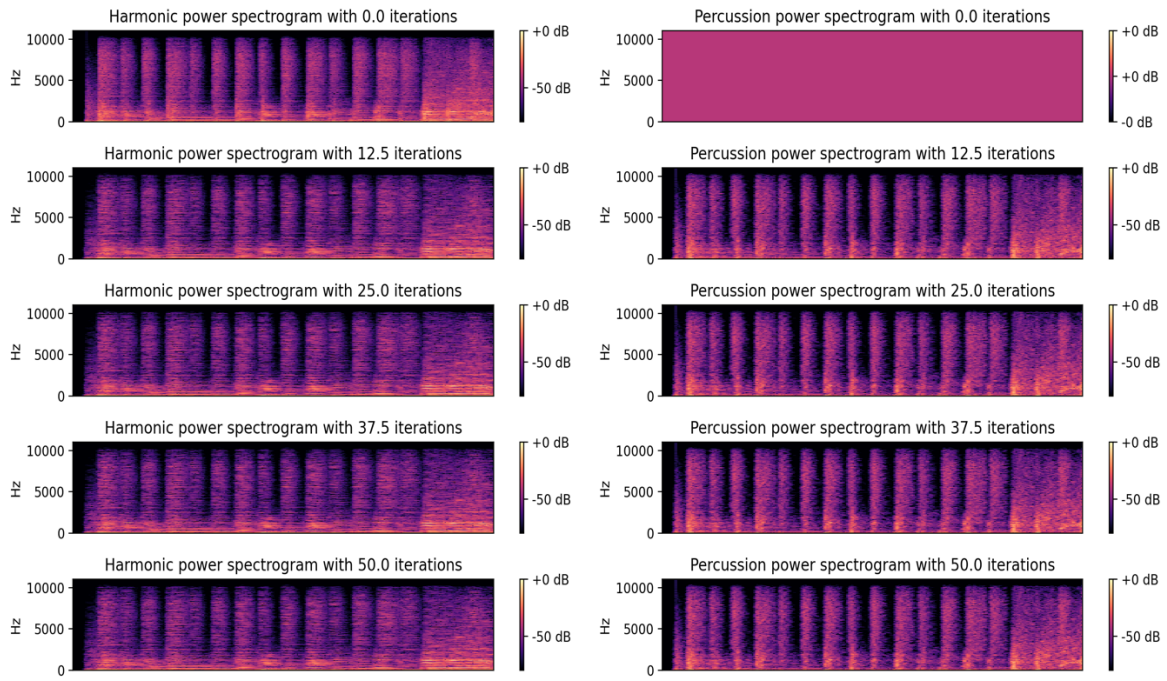

Figure 3: Harmonics and percussions with different number of iterations

When there is no iteration, there is no separation between the components, and all the values belongs to the harmonic spectrogram, which is not ideal. However, there is no noticeable difference between the harmonic spectrograms themselves (as well as percussion spectrograms) when a greater number of iterations are used. To enhance coding efficiency, the recommended value of number of iterations, $k_{\max}$, is set to 20.

Signal-to-noise ratio

Signal-to-noise ratio of each component can be calculated using the formula:

$$SNR = 10 \log_{10} \left(\frac{\sum_t s(t)^2}{\sum_t e(t)^2} \right)$$

where $s(t)$ is the original signal, and $e(t)$ is the original minus the separated signal.
The values of γ and $k_{\max}$ calculated above affects the signal-to-noise ratio, which is calculated using the above equation. The analysis of the same audio file is illustrated in the following plots.

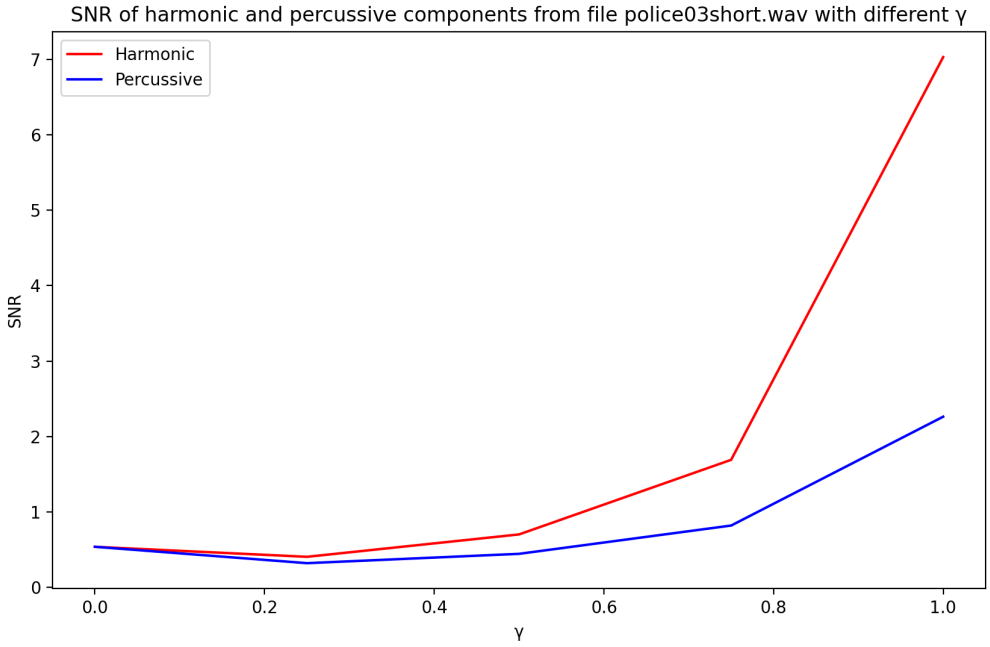

Figure 4: SNR analysis of different range of compression values

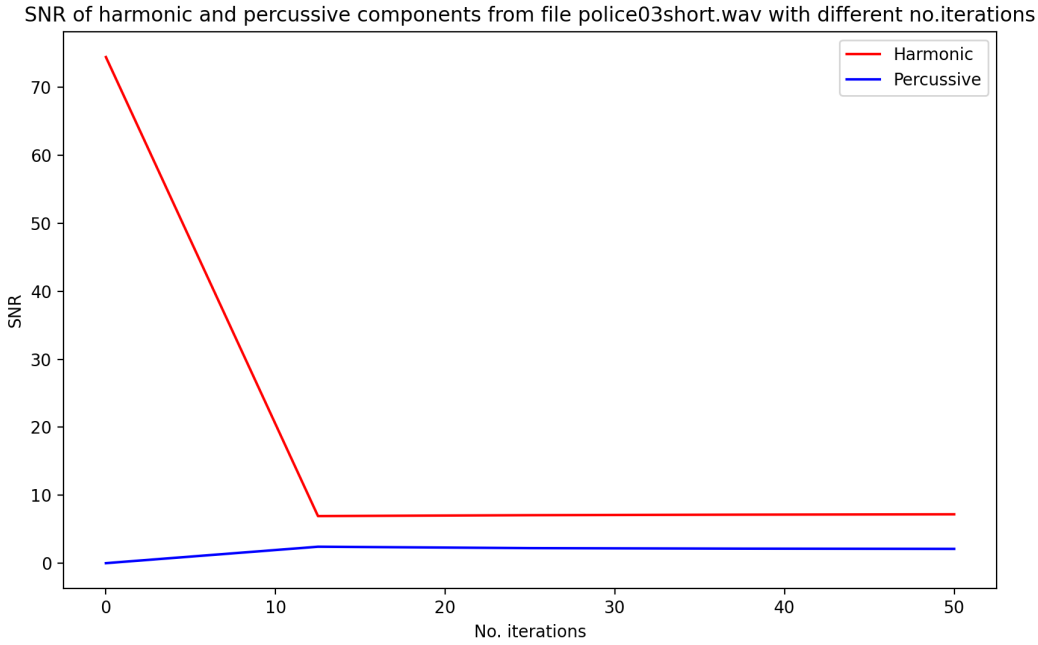

Figure 5: SNR analysis with different number of iterations

From the plots, setting the range compression value to maximum produce the highest SNR, which minimizes the effect of noise, but there is no such difference if we increase the number of iterations. As concluded above, it is recommended to set $\gamma = 1$ and $k_{\max} = 20$

Evaluation

Audio outputs evaluation

During the last step of components separation, the signal of each component is saved to two separate audio files (in .wav format). Listening to both files confirm the previous observations:

- Harmonic component's file contains mostly melody, while the percussive one contains bass and drum sound.
- Increasing range compression coefficient will produce less noise in both files
- Decreasing or increasing number of iterations does not have any big difference, but it should be set to at least 20.

Different audio types

The spectrograms of the audio file project_test1.wav are presented below.

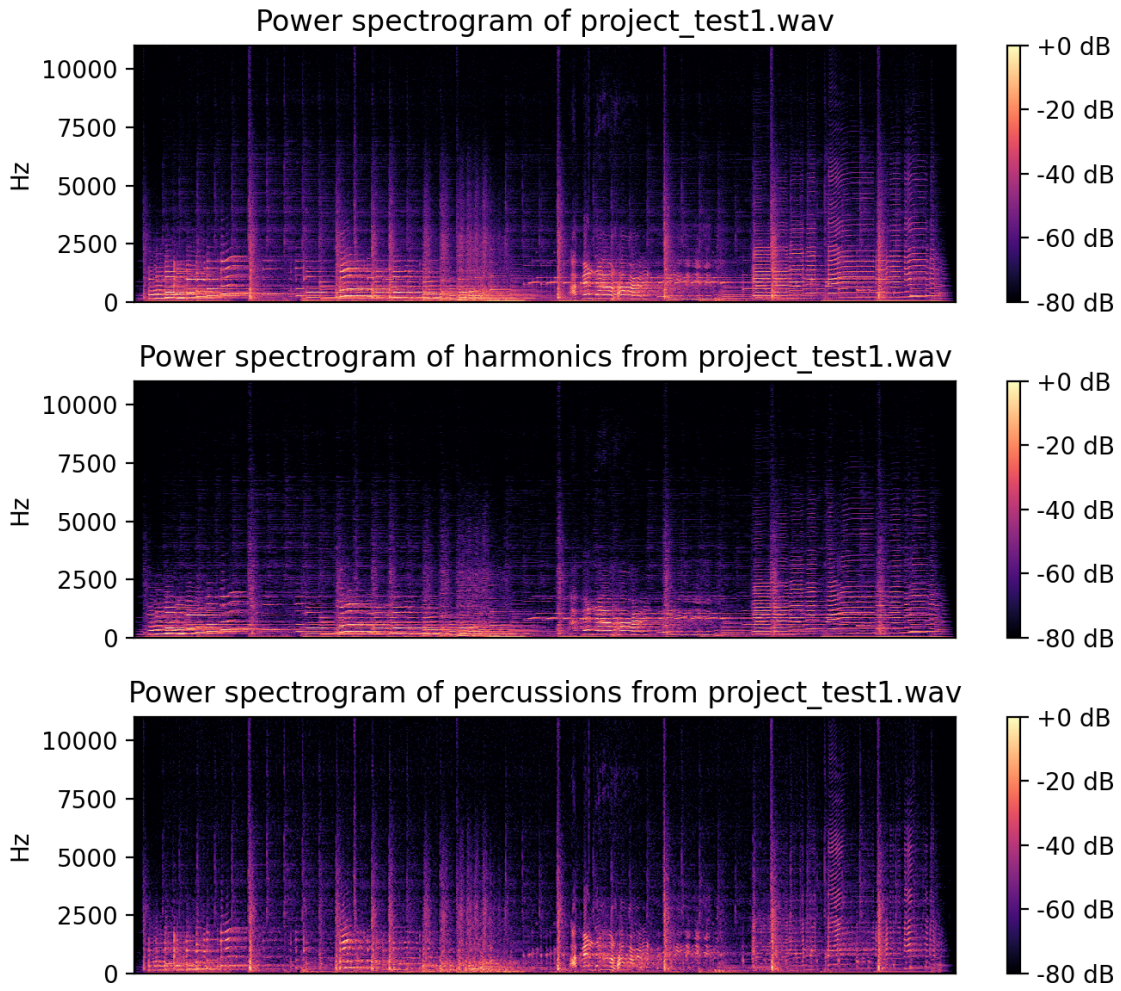

Figure 6: Power spectrograms of another audio file

By listening to this file and observe the original spectrogram, we can find out that there is no clear drums or bass sound. This makes the separation unclear. Hence, there is no noticeable different between the spectrograms of the components. Moreover, the last few second of the file contains singing, which makes it even harder for the operation to receive accuracies.

Reference

[1] Nobutaka Dno, Kenichi Miyamoto, Jonathan Le Roux, Hirokazu Kameoka, Shigeki Sagayama, ”Separation of a monaural audio signal into harmonic/percussive components by complementary diffusion on spectrogram”. Proc. EUSIPCO, Aug 2008.

[2] *Percussive & Harmonic Content Separation*. Percussive & harmonic content separation | Music Information Research | Institute for Language & Speech Processing. (n.d.). Retrieved December 16, 2021, from http://mir.ilsp.gr/harmonic_percussive_separation.html

[3] Wootaek Lim and Taejin Lee, ”Harmonic and Percussive Source Separation Using a Convolutional Auto Encoder”. Proc. EUSIPCO, 2017.