Source Separation of Instruments in Musical Audio Streams

Jason Cramer

***Abstract*—We investigate and implement a system that performs instrument source separation on a musical audio stream, in order to processes the instrument audio streams separately in some way. The problem is viewed as a matrix factorization problem on the spectrogram of each frame. We use Probabilistic Latent Component Analysis (PLCA) to build dictionarys for each instrument consisting of spectral basis vectors, and PLCA during run-time to reconstruct the sources in terms of the dictionarys. In addition, we adaptively update the dictionaries if the dictionaries do not sufficiently describe the data.**

***Index Terms*—music, audio, signal processing, source separation, probabilistic latent component analysis, PLCA, model, timbre, matrix factorization, expectation maximization, streams, representation, music information retrieval**

I. Introduction

RECOGNIZING instruments, and being able to mentally separate instruments when listening to a recording or attending a musical performance is something that is a relatively simple task for people, even without a lot of musical training. However, like many tasks that people excel at, it is a more challenging problem for a machine. If we could successfully train a computer for this task, it may be useful in audio production settings, where we might want to record or mix a performance without much equipment or setup, and have the computer recognize, separate, and process audio from different instruments using only a limited number of microphones. We consider the monoaural case, limiting ourselves to a single audio source. In order to be useful for tasks other than post-performance mastering, we would like to have the system work in real-time. Towards this goal, we also impose the structure of the audio signal to be a data stream, in the sense that we handle only a single segment of the entire signal at a time. There are numerous ways to formulate the problem of source separation and each of these perspectives has many ways of solving them. We will touch upon a few of these approaches.

II. Background

The problems of source separation is an active area of research, which falls under the broader field of music information retrieval (MIR). Popular applications of source separation include removing vocals from music, speaker identification and separation, music analysis (MIR), music transcription, and much more.

Perhaps the most popular approach to source separation is by non-negative matrix factorization (NMF), which is used to extract a set of spectral bases for sources. Wang and Mustafa [16] use a method that performs NMF on MFCC (Mel ceptrum coefficient matrices) matrices and uses K-nearest neighbors with a collection of similarly represented notes of various instruments. Cont, Dubnov, and Wessel [2] propose a somewhat similar approach, but use modulation spectra as their representation model, rather than STFT or MFCC matries, in addition to further constraining the NMF solutions to be sparse. However, their goal is only to recognize instruments and pitch, not to perform source separation. One of the other hugely popular algorithms for source separation is independent component analysis (ICA). ICA decomposes multivariate signals into seperate sources that are maximally statistically independent and non-Gaussian sources.

Coupling matrix factorization with probabilistic graphical models has been an area of interest recently. Mysore, Smaragdis, and Raj use a novel and successful approach that combines multiple dictionary learned using NMF with hidden Markov models for every speaker with an approach they call a non-negative factorial hidden Markov model (N-FHMM) [8]. Their approach works well in better separating the sources, but results in more artifacts. Smaragdis and Kim have recently explored a method that uses Markov random fields (MRF) to model probabilistic relationships between neighboring STFT bins in order to smooth out the spectrum. This is in order to avoid having disjoint and scattered spectral regions of a source where it does not seem to belong, which may add artifacts as well as suboptimally separate the sources. [7] This method generally outperforms NMF, aside from a few times.

One of the concerns with NMF and non-negative factorization methods is that magnitude of the spectrum is utilized, but not the phase. While the phase contains important aspects of the frequency content, most find it acceptable to discard, on the grounds that our ears are not sensitive to changes in phase. However, there have been attempts to rectify this. Parry and Essa describe an approach that does not directly estimate the phase, but probabilistically accounts for it when estimating the magnitude spectrum of the sources by putting a uniform distribution on the phase, and uses the geometric relationship with the magnitude to propose a different objective function and a different update rule. [10] They are able to achieve significant improvement over NMF. Yoshii, Tomioka, Mochihashi, and Goto take a different route, instead choosing to avoid the problem altogether. They use a tensor factorization method called positive-semidefinite tensor factorization (PSDTF) to estimate the time-domain signal directly. [17] They are able to get even more speed up, although at great computational costs.

Rather than exploring new methods, Ozerov, Vincent, and Bimbot chose to generalize a lot of the work that has been done with source separation [9] into a toolkit called the Flexible Audio Source Separation Toolkit (FASST) [11], that outlines the general methodology behind designing and experimenting with source separation algorithm, details a general mathematical framework, and implements it with a basic software framework.

As an alternative, but numerically similar (if not identical [15]), approach to NMF, probabalistic latent component analysis (PLCA) is also quite popular. This is another non-negative factorization tool that models the spectrogram as a distribution over time and frequency, and the task becomes to decompose the spectrogram into probability matrix that acts as a set of spectral bases, a probability matrix of the magnitude of temporal activations at different times, and probabilistic weights for the latent components. [5] NMF and PLCA are similar, but PLCA has a more intrinsically probabilistic motivation behind it. PLCA also has a convolutive extension caled shift-invariant probabilistic latent component analysis (SI-PLCA) [14] that is popular and useful for performing audio segmentation and finding repeating patterns. The approach we use uses this PLCA, along with an adaptive unsupervised component to augment the spectral bases at run-time.

III. Method

In the first few sections, we detail our assumptions and constraints on the problem of source separation, and describe and motivate PLCA. We then describe how we estimate the source given the parameters from our model. We take a look at how training is carried out, giving an overview of the training data used, the data preprocessing performed, and the process of training itself. In the last section, we describe the run-time process.

A. Problem

Consider a sampled monoaural audio signal mix, that is streamed such that we receive a periodically recieve equally-sized segments of samples. Let $x_t[n] \in \mathbb{R}$ be the n-th sample during segment i, $\forall i \in \{0, \cdots, T-1\}$, where N is the length of each segment. The signal comprises of a mixture of $K > 1 \in \mathbb{Z}_+$ individual sources (instruments). Let $s_t^k[n] \in \mathbb{R}, \forall k \in \{1, \cdots, K\}$ be the n-th sample of segment t in source k. Our goal is to be able to recover the original K sources segments $s_t^k[n]$ for each segment.

We will do this be trying to separate the spectrogram of each segment in terms of latent components (spectral bases) that characterize each instrument, and reconstructing the source signals for those segments. We do this using an algorithm called Probabilistic Latent Component Analysis (PLCA). In order to simplify the discussion and the notation, when we discuss PLCA, we will drop the notation that indexes the segment. The procedure discussed in the next few sections applies to each segment individually. We will deal with segments again when we discuss the adaptive part of the procedure.

B. PLCA Model

Probabalistic Latent Component Analysis is a matrix factorization algorithm that probabilistically models the spectrogram of an audio signal as a histogram, that defines a joint spectral and temporal distribution over frequency and time. [13] In effect, we ultimately decompose the spectrogram into a a set of spectral bases and temporal activations weights.

Let X_{fm} represent the STFT of $x[n]$ at frequency f and STFT time-step m. Similarly, let S_{fm}^k be the STFT of source $s^k[n]$. Let $V_{fm} = |X_{fm}|$ be the magnitude of the STFT of an audio signal $x[n]$, also known as its spectrogram. We normalize $V_n[f]$ in order to obtain a distribution via

$$V_{fm} = \frac{P[f,m]}{A} = \frac{P[f,m]}{\sum_f \sum_m V_{fm}}$$

where we interpret $P[f|z]$ as a joint distribution of time and frequency.

We then decompose this as

$$P[f,m] = \sum_z P[f|z]P[m|z]P[z]$$

Here, $P[f|z]$ is a conditional multinomial distribution of frequency bins given a latent factor z. [5] The latent factor represents a feature associated with a spectral basis encoded by the distribution. $P[m|z]$ is also a conditional multinomial distribution of temporal activity given a latent factor. Similarly, this encodes the temporal basis associated with latent factor z. $P[z]$ is the multinomial prior on the latent variables, which serves as a weight for the latent function's contribution to the spectrogram.

This decomposition introduces an important interpretation of the spectrogram. The normalized spectrogram $P[f,m]$ can be thought of as a linear combination of the spectral (and temporal) contributions of each latent factors. In this form, our matrix factorization becomes [1]

$$P \approx WZH$$

where $W \in \mathbb{R}^{F \times J}$ is such that $W_{fz} = P[f|z]$, $Z \in \mathbb{R}^{J \times J}$ is such that Z is diagonal with entries $Z_{zz} = P[z]$, $J > 1 \in \mathbb{R}_+$ is the number of latent factors, and $H \in \mathbb{R}^{J \times N}$ is such that $H_{zm} = P[m|z]$. Note that since we are arbitrarily introducing the latent factors, we can choose J to be any (positive) number.

A solution is found by via an iterative updates using the expectation-maximization algorithm, which are derived from minimizing Kullback Leibler (KL) divergence: [6]

$$d_{\text{KL}}(P[f,m], \hat{P}[f,m]) = \sum_f \sum_k P[f,m] \log \frac{P[f,m]}{\hat{P}[f,m]}$$

where, $P[f,m]$ is the actual normalized spectrogram, and $\hat{P}[f,m]$ is the factorization approximation. For the EM algorithm, first, all of the unknown parameters are initialized at random. The iterative updates are as such: [12], [13]

a) E Step: Estimate the posterior distribution, the conditional of the latent factor at a certain time, given a frequency bin

$$P[z|f,m] \leftarrow \frac{P[z]P[f|z]P[m|z]}{\sum_{\tilde{z}} P[\tilde{z}]P[f|\tilde{z}]P[m|\tilde{z}]}$$

b) M Step: Estimate the spectral bases and the latent factor priors:

$$P[z] \leftarrow \frac{\sum_m \sum_f V_{fm} P[z|f,m]}{\sum_{\widetilde{z}} \sum_m \sum_f V_{fm} P[\widetilde{z}|f,m]}$$

$$P[m|z] \leftarrow \frac{\sum_f V_{fm} P_m[z|f]}{\sum_{\widetilde{m}} \sum_f V_{f\widetilde{m}} P[z|f,\widetilde{m}]}$$

$$P[f|z] \leftarrow \frac{\sum_m V_{fm} P_m[z|f]}{\sum_m \sum_{\widetilde{f}} V_{\widetilde{f}m} P[z|\widetilde{f},m]}$$

It is worth recalling that with this factorization, we are claiming that $P_m[z|f]$ consists of a sum of contributions from each spectral basis. Remember that we can choose J as we wish. Additionally, in matrix form, J has no effects on the dimensions of the resulting product. Therefore, we can reasonably extract out the parts of the signal containing the latent factors encoded in the spectral bases. By only using specific latent factors z, we can extract contributions to the signal containing features of interest. If we can associate the latent factors with an instrument, we can reasonably extract the contributions of these latent factors. In matrix form, this amounts to selecting the relevant rows, diagonals, and columns from the matrices that contain quantities associated with these latent factors, we can create a spectrogram containing only their contributions, or more importantly, the instrument's contributions. In effect, we can separate the instrument from the mix by recovering a time-domain signal from this spectrogram. This can be done by constraining the bases $P[f|z]$ to be spectral bases describing an instrument. After we learning representative spectral bases, our goal then becomes estimating $P^k[f,m]$ for each source instrument.

C. Source Separation Procedure

We obtain dictionaries $\mathcal{S}_k$ for each source $k \in \{1, \cdots, K\}$, each of which contains J_k spectral bases z, represented by column vectors that encode $P[f|z]$. Let $\mathbf{S} = \cup_k \mathcal{S}_k$. If we can reasonably assume that the k sources are all that contribute to the spectral content of the audio, then we can fix $\{P[f|z]\}_{z\in\mathbf{S}}$, and estimate the other parameters in the same way using the J latent factors. That is, we fix $W = W_{\mathbf{S}}$, where each column corresponds to $\{P[f|z]\}_{z\in\mathbf{S}}$ and estimate Z and H. This constrains the spectral bases present in the signal to be the bases of the sources when performing the factorization.

After we obtain the matrices $Z_{\mathbf{S}}$ and $H_{\mathbf{S}}$, we can separate the sources by reconstructing a normalized spectrogram using only the latent factors in a single dictionary at a time. Let $W_{\mathcal{S}_k}$ be a matrix containing the columns of $W_{\mathbf{S}}$, $Z_{\mathcal{S}_k}$ be a diagonal matrix containing the diagonal entries of $Z_{\mathbf{S}}$, and $H_{\mathcal{S}_k}$ be a matrix containing the rows of $H_{\mathbf{S}}$, all corresponding to the latent factors $z \in \mathcal{S}_k$ in the same respective order, such that we obtain

$$\hat{P}_{\mathcal{S}_k} = W_{\mathcal{S}_k} S_{\mathcal{S}_k} G_{\mathcal{S}_k}$$

corresponding to

$$\hat{P}^k[f,m] = \sum_{z\in\mathcal{S}_k} P[f|z]P[z]P[m|z]$$

We estimate the spectrogram of source k via:

$$\hat{V}^k_{fm} = P[m]\hat{P}[f,m]$$

In order to recover the STFT of source k we re-use the phase from the original signal. This results in a bit of reconstruction error, but a common justification is the human auditory system is largely insensitive to phase differences. Therefore, reusing the original phase results in relatively small perceptual discrepancies. As mentioned, there have been multiple methods of estimating the phase, or estimating the magnitude in a way that is aware of phase. Regardless, we estimate the STFT as

$$\hat{S}^k_{fm} = \hat{V}^k_{fm} e^{j\angle X_{fm}}$$

where $j = \sqrt{-1}$, and we recover the time-domain signal for source k, $\hat{s}^k[n]$, by taking the inverse STFT.

D. Adaptive Dictionary Updates

In order to better adapt to the specific mixture signal we are processing, in addition to utilizing the temporal succession of audio segments, we add the ability to adaptively update the dictionary in an unsupervised way which considers the previous audio segment. If our dictionary insufficiently describes the current segment, then we update the spectral bases to better describe the current segment, as well as the previous. The reasoning behind including the previous frame is to prevent the update from overfitting on the current frame, while still trying to adapt to the current frame. The approach in [3] is used for semi-supervised learning, where we have no spectral bases for a source. However, we already have dictionaries trained for the instruments, so this allows us to start off with some general knowledge about the sources beforehand and adapt slightly to better describe what is currently being processed.

For our metric of how well our spectral bases describe the current segment, we use KL divergence between the normalized spectrogram "distribution" estimated by running PLCA constrained by the spectral bases from our dictionaries, and the true "distribution" obtained from the spectrogram. We want a series of updates that will decrease divergence from the distribution of the current segment, as well as decreasing divergence from the distribution of the previous segment, to a lesser degree. Let $P_t[f,m]$ be our estimate of the current normalized spectrogram at segment t. Let $Q_t[f,m] = \frac{V_{t,fm}}{\sum_m \sum_f V_{t,fm}}$ be the actual normalized spectrogram at segment t. The proposed objective to minimize is:

$$\underset{P[f|z],P[z,m],\forall z}{\arg\min} \; d_{\mathrm{KL}}(Q_t[f,m]||P_t[f,m]) + \frac{\alpha}{N} d_{\mathrm{KL}}(Q_{t-1}[f,m]||P_t[f,m])$$

$\forall t > 1$, where $\alpha \in \mathbb{R}_+$ is a hyperparameter that controls how much the the divergence of the previous segment contributes. We use an algorithm adapted from [3] to minimize this object.

a) Adaptive Dictionary Update Algorithm: For each segment $t > 1$, compare the divergence $d_{\text{KL}}(Q_t[f,m]||P_m[f,m])$ against a threshold hyperparameter θ_{KL}. If the threshold is not exceeded, we do not update the dictionary, and process the frames in the usual way. If the threshold is exceeded, we perform the following EM algorithm, which essentially performs the updates in [3] across all of the time frames in the segment. Let the subscripts t in the following refer to the segment that the quantities belong to, and let $P_t[z,m] = P_t[m|z]P[z]$. We have the updates:

b) E Step:

$$P_t[z|f,m] \leftarrow \frac{P_{t-1}[f,m]P[z|f]}{\sum_{\tilde{z}} P_{t-1}[\tilde{z},m]P_{t-1}[f|\tilde{z}]}$$

c) M Step:

$$\phi_t[f|z] \leftarrow V_{t,fm}P_t[f|z] + \frac{\alpha}{N}V_{t-1,fm}P_{t-1}[f|z]$$

$$\phi_t[z,m] \leftarrow \sum_f V_{t,fm}P_t[z|f,m]$$

Normalize $\phi_t[f|z]$ and $\phi_t[z,m]$ to get the updated distributions $P_t[f|z]$ and $P_t[z,m]$.

Using our updated parameters, we estimate the spectrogram for source k, $\hat{V}^k_{t,fm}$, as:

$$\hat{V}^k_{t,fm} = \frac{\sum_{z \in \mathcal{S}_k} P[f|z]P_t[z,m]}{\sum_z P[f|z]P_t[z,m]}$$

We then reconstruct the STFT and signal segment for the sources in the same way done in the non-adaptive approach.

E. Training

For each instrument, we wish to learn, offline, a dictionary of spectral basis templates, in order to produce a factorization that constrained to fit the spectral characteristics of the instruments.

a) Training Data and Preprocessing: Our training data consists of samples of individual notes played on each instrument. As preprocessing, we trim off silence from the ends of the training data so that the more interesting spectral content is accentuated and so that training time is not wasted on processing silence. For each of these notes, we learn spectral basess, and combine them into a single column matrix of spectral basess. [15]

b) Harmonic Percussive Source Separation: Since the transients and harmonic content of a note are considered to be some of the best perceptual identifiers for instruments, we add an additional preprocessing step to split the training example into its harmonic and percussive (transient) components. To do this, we use HPSS (Harmonic Percussive Source Separation) [4]. The intuition behind HPSS is that harmonic components tend to be horizontal in the time-frequency representation, and percussive (noisy) components tend to be vertical in the time-frequency representation. With this in mind, we define two median filters on an TF power matrix V of a signal x:

$$g^h_{\text{med}}(V)[f,m] = \text{median}(\{V_{f(m-\ell_h)}, \cdots, V_{f(m+\ell_h)}\})$$

$$g^p_{\text{med}}(V)[f,m] = \text{median}(\{V_{(f-\ell_p)m}, \cdots, V_{(f+\ell_p),m}\})$$

where $g^h_{\text{med}}[f,m]$ is the harmonic median filter, $g^p_{\text{med}}[f,m]$ is the percussive median filter, and ℓ_h and ℓ_p are the lengths of the respective filters. The motivation for this is that we want each filter enhance the horizontal (harmonic) and vertical (percussive) features. After applying the median filters and obtaining the harmonically and percussively enhanced power matrices $V^h = g^h_{\text{med}}(V)$ and $V^p = g^p_{\text{med}}(V)$, we perform a binary mask using the following threshold functions:

$$M^h[n,k] = \begin{cases} 1: & V^h_{fm} \geq V^p_{fm} \\ 0: & \text{otherwise} \end{cases}$$

$$M^p[n,k] = \begin{cases} 1: & V^p_{fm} > V^h_{fm} \\ 0: & \text{otherwise} \end{cases}$$

This mask selects a time-frequency bin to belong to the harmonic component of the signal if it has more harmonic power than percussive power, and vice versa. We use this mask on the TF power matrix X to obtain the TF matrix of the separated harmonic and percussive components:

$$X_h = X \otimes M_h$$

$$X_p = X \otimes M_p$$

We then take the inverse transform to retrieve the time-domain signals for the harmonic and percussive components of the signal, $x^h[n]$ and $x^p[n]$. Towards emphasizing the transients and the harmonics individually as characteristic spectral components, we learn a spectral bases for each.

c) Learning Templates: To learn each instrument dictionary, we use PLCA to decompose $P[f,m]$ for the harmonic and percussive part of each training example and extract the matrix W and stack the columns horizontally. After we process all of the training examples, we store them for use during runtime.

F. Run-time

At the start of run-time, the user specifies which instruments are to be played, and the corresponding spectral basis dictionaries for each instrument are retrieved, and the spectral basis matrices for each instrument $W_{\mathcal{S}_k}$ are stacked horizontally into a large matrix of $P[f|z]$ vectors from each instrument. During each step of run-time, we process an individual audio segment, and separate the audio segment into into the specified instruments, performing the procedure in III-C and III-D. With the obtained signal, we can choose to process it in whatever way is desired, or simply output the separated sources into different audio channels. We repeat again for every subsequent segment m until the stream ceases.

IV. Experimental Results

We implemented the framework in Python using numpy, scipy, audiolab, and librosa. We also used an implementation of PLCA written for [1]. For our training examples we used instrument note samples made available by University of Iowa's Electronic Music Studios (http://theremin.music.uiowa.edu/MIS.html) and the Philharmonia Orchestra (http://www.

philharmonia.co.uk/explore/make_music). We made manual mixes out of loops found on http://www.free-loops.com. In addition, we extracted audio from some online videos to test on some real audio mixtures. To truncate the silence from the beginning and end of the training examples, we used Audacity, making use of its ”chain” feature. We ran on a number of non-adaptive trials, and only a few adaptive trials due to the increased processing time.

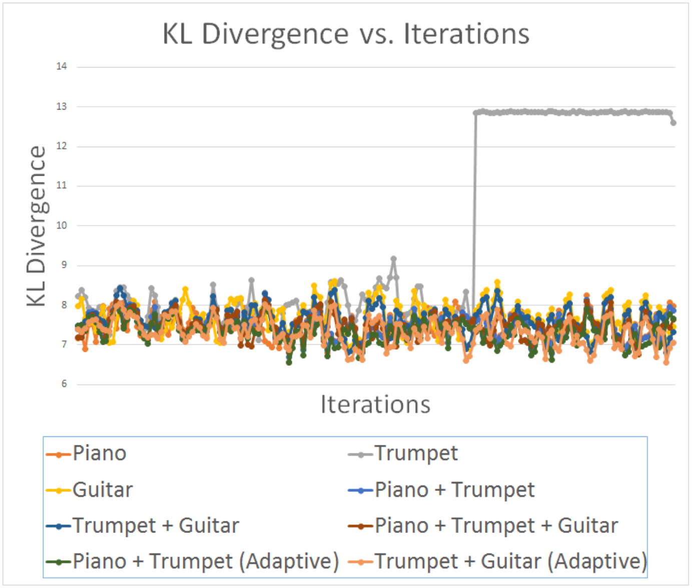
To evaluate performance for the manual mixes, we process sequential audio segments in the manner described in the previous section and compute the SNR (signal-to-noise ratio) for each source with respect to the original source audio:

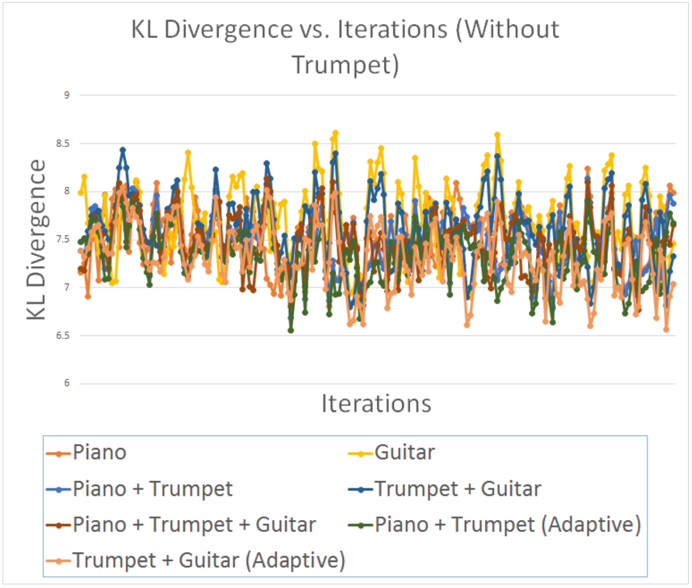
$$SNR^k = 10\log_{10}\frac{\|s^k\|_2^2}{\|\hat{s}^k - s^k\|_2^2}$$

where $s^k = [s_0^k[0], \cdots, s_0^k[N-1], s_1^k[0], \cdots, s_1^k[N-1], \cdots]^\intercal$ is a column vector containing the consecutive values of samples in consecutive segments. Similarly, $\hat{s}^k = [\hat{s}_0^k[0], \cdots, \hat{s}_0^k[N-1], \hat{s}_1^k[0], \cdots, \hat{s}_1^k[N-1], \cdots]^\intercal$. As is common with audio processing, better SNR does not always yield better sound results, but this still gives us a ballpark idea of how well we do.

Instrument Mix	Piano SNR (dB)	Trumpet SNR (dB)	Guitar SNR (dB)
Solo Piano (Non-Adaptive)	5.029141	N/A	N/A
Solo Trumpet (Non-Adaptive)	N/A	0.624250	N/A
Solo Guitar (Non-Adaptive)	N/A	N/A	1.631432
Piano + Trumpet (Non-Adaptive)	3.476929	2.310791	N/A
Trumpet + Guitar (Non-Adaptive)	N/A	1.411124	2.198955
Piano + Trumpet + Guitar (Non-Adaptive)	0.143035	0.566972	0.305995
Piano + Trumpet (Non-Adaptive)	-4.218275	-2.668547	N/A
Trumpet + Guitar (Adaptive)	N/A	-3.251632	-4.475748

We can also use the KL divergence as a measure of well our instrument dictionaries describe the spectrogram distribution over time, as well as how our adaptive updates improve the dictionary.





Overall, the results are mediocore in terms of SNR. [3] This is likely due to a lack of discriminative spectral bases from the training data, as well as a lack of sufficient training data to begin with. Disappointingly, the adaptive method seemed to make the separation much worse in terms of SNR. However, since we've mentioned that SNR is not a clear indicator of perceptual quality, we must listen to the results.

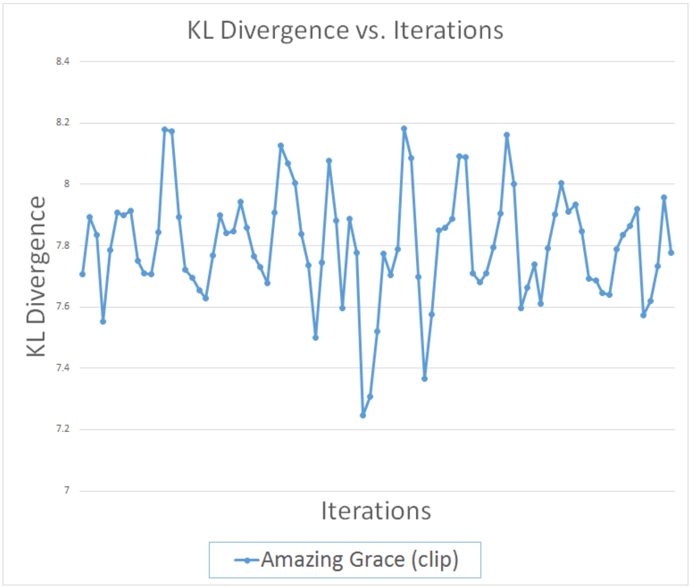
Interestingly the KL divergence does not vary much, aside from in the solo trumpet track where we see a sudden spike due to a large period of silence. This does suggest that the bases are more or less consistent for describing the instrument over time.

From subjectively listening, there were a lot of reconstruction issues and artifacts. Most notably, there was a noticeable crackling noise appearing at the ends of each segment, most likely due to windowing effects. This most likely contributed to a great deal of the SNR for each of the trials. In addition, clipping occasionally occurred, which drastically lowered the SNR in some cases. The separated sources did sound like the instrument they were supposed to. In fact, sometimes the SNR was lower for an instrument but sounded more discernible than the others, though this is not surprising as better SNR does not guarantee better perceptual approval. The fact that it, for the most part, accentuates or at least brings attention to the original source in the mix often is promising.

However, the quality of separation was generally not great, as a non-trivial or even fairly significant amount of the signal was part of another source. Again, we can attribute this to the spectral bases not being sufficiently discriminative to properly separate the sources. This was especially clear when trying to separate three instruments; there was so much overlap in spectral content that each of the separated sources contained quite a bit of the others. The majority of the time though, the instrument associated with the separated source was the most prominent.

It seemed that brighter timbres were emphasized much more, the descriptor ”brighter” generally being correlated with a higher spectral centroid, and often with more uniform power throughout its harmonics. The trumpet is the brightest of the instruments chosen, so it has the strongest spectral content in its harmonics. Since the overtones are so powerful and relatively uniform, any harmonic content is easily selected. This means for itself that it easily selects its own contributions to the mix, but also includes harmonic content from other signals. When listening to the examples, the trumpet tended to stick out in its separated audio more so than the other instruments in their separated audio. Piano is the next brightest, so it tends to stick out slightly more than guitar in its separated audio.

We ran our non-adaptive algorithm on a short clip from a video performance of Amazing Grace, played on guitar and piano. We get the following KL divergence trend:



This looks similar to what we had with our manually made mixes. Listening to the separated sources reveals that the trumpet was actually very well separated. However, the piano was not well separated, as the separated source audio contained a lot of trumpet. This may be due to the fact that in the song, the trumpet is the leading voice and is much louder than the piano. This would signify that the quality of the separation also relies on relative volume.

V. Future Work

The main issues faced were the sources were the quality and discrimination of the separation. This is often solved by adding regularization terms to the objective criterion, and has been approached by many with success. Using more training data would be good, but a dictionary learning algorithm that keeps the number of spectral spaces relatively small would be necessary, since the size of the basis matrix grows linearly with the size of the training data. A possibility could be to use an algorithm like SVD for dimensionality reduction. If we could reduce the basis space, we could try segmenting the training examples to try to learn more temporal variation in the spectrum. There are also formulations of PLCA that enforce sparsity, so exploring this could make the separation more discriminative, as well as help compensate for using less training data.

Since spectral bases can potentially occur in multiple instruments, it may be worth looking into further decomposing the PLCA representation, conditioning on the latent factors and adding a priors/weights for each instrument. Improving the adaptive algorithm is also worthwhile, as there are plenty of improvements and heuristics that could make for better adaptation.

Looking into ways to perform temporal smoothing between segments would be useful for avoiding the crackling artifacts. In addition, using an extra buffer that would be processed along with the current frame may make the effects of windowing less discontinuous.

Making training faster and more efficient would also make using more training data more feasible. Eventually, upon finding a set up that has sufficiently good results, building a real-time implementation and making it portable and distributable would be a good goal.

VI. Conclusion

We have explored a method of source separation that involved using a PLCA approach combined with a semi-supervised adaptive approach. This first involved learning a set of spectral basis vectors for each instrument to form a dictionary. We added another step of separating the training data into the harmonics and transients in order to accentuate transients as a separate bases, since transients are extremely important for perceptual recognition of instruments. At run time forcing the spectrogram to factor with respect to the learned instrument bases of the instruments selected. We make it (optionally) adaptive by updating the dictionary with respect to the previous segment if the KL divergence exceeds a threshold. Due to limitations in the training process, the results were mediocre. Disappointingly, our adaptive approach made the results much worse. We also found that using such a relatively small data set in addition to common spectral features results in poor separation quality and discrimination. In addition, reconstruction and segmentation artifacts add a lot of noise. Nonetheless, the system does perform source separation to a degree and is a forward progress that can be improved upon for a more successful system.

The code can be found at https://github.com/jtcramer/streaminstrumentseparation.

REFERENCES

[1] Bello, R. W. J. (2010). Identifying repeated patterns in music using sparse convolutive non-negative matrix factorization.

[2] Cont, A., Dubnov, S., & Wessel, D. (2007). Realtime multiple-pitch and multiple-instrument recognition for music signals using sparse non-negative constraints. In Proceedings of Digital Audio Effects Conference (DAFx).

[3] Duan, Z., Mysore, G. J., & Smaragdis, P. (2012). Online PLCA for real-time semi-supervised source separation. In Latent Variable Analysis and Signal Separation (pp. 34-41). Springer Berlin Heidelberg.

[4] Fitzgerald, Derry. ”Harmonic/percussive separation using median filtering.” (2010).

[5] Guo, Y., & Zhu, M. (2011, May). Audio source separation by basis function adaptation. In Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on (pp. 2192-2195). IEEE.

[6] Lee, D. D., & Seung, H. S. (2001). Algorithms for non-negative matrix factorization. In Advances in neural information processing systems (pp. 556-562).

[7] Kim, M., & Smaragdis, P. (2013, September). Single channel source separation using smooth nonnegative matrix factorization with Markov random fields. In Machine Learning for Signal Processing (MLSP), 2013 IEEE International Workshop on (pp. 1-6). IEEE.

[8] Mysore, G. J., Smaragdis, P., & Raj, B. (2010). Non-negative hidden Markov modeling of audio with application to source separation. In Latent Variable Analysis and Signal Separation (pp. 140-148). Springer Berlin Heidelberg.

[9] Ozerov, A., Vincent, E., & Bimbot, F. (2012). A general flexible framework for the handling of prior information in audio source separation. Audio, Speech, and Language Processing, IEEE Transactions on, 20(4), 1118-1133.

[10] Parry, R. M., & Essa, I. (2007, April). Incorporating phase information for source separation via spectrogram factorization. In Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on (Vol. 2, pp. II-661). IEEE.

[11] Salan, Y., Vincent, E., Bertin, N., Souvira-Labastie, N., Jaureguiberry, X., Tran, D. T., & Bimbot, F. (2014, March). The Flexible Audio Source Separation Toolbox Version 2.0. In ICASSP.

[12] Shashanka, M., Raj, B., & Smaragdis, P. (2008). Probabilistic Latent Variable Models as Nonnegative Factorizations. Computational Intelligence and Neuroscience, 2008, 1-8.

[13] Smaragdis, P., & Mysore, G. J. (2009, October). Separation by humming: User-guided sound extraction from monophonic mixtures. In Applications of Signal Processing to Audio and Acoustics, 2009. WASPAA'09. IEEE Workshop on (pp. 69-72). IEEE.

[14] Smaragdis, P., & Raj, B. (2007). Shift-invariant probabilistic latent component analysis. Journal of Machine Learning Research.

[15] Smaragdis, P., Raj, B., & Shashanka, M. (2007). Supervised and semi-supervised separation of sounds from single-channel mixtures. In Independent Component Analysis and Signal Separation (pp. 414-421). Springer Berlin Heidelberg.

[16] Wang, W., & Mustafa, H. (2011). Single channel music sound separation based on spectrogram decomposition and note classification. In Exploring Music Contents (pp. 84-101). Springer Berlin Heidelberg.

[17] Yoshii, K., Tomioka, R., Mochihashi, D., & Goto, M. (2013). Beyond NMF: Time-Domain Audio Source Separation without Phase Reconstruction. In ISMIR (pp. 369-374).

Jason Cramer is an undergraduate at the University of California, Berkeley in the Electrical Engineering and Computer Science Department, and works with the Center for New Music and Audio Technologies