MULTI-VOICE POLYPHONIC MUSIC TRANSCRIPTION USING EIGENINSTRUMENTS

Graham Grindlay and Daniel P. W. Ellis

LabROSA, Dept. of Electrical Engineering
Columbia University
{grindlay,dpwe}@ee.columbia.edu

ABSTRACT

We present a model-based approach to separating and transcribing single-channel, multi-instrument polyphonic music in a semi-blind fashion. Our system extends the non-negative matrix factorization (NMF) algorithm [1] to incorporate constraints on the basis vectors of the solution. In the context of music transcription, this allows us to encode prior knowledge about the space of possible instrument models as a parametric subspace we term "eigeninstruments". We evaluate our algorithm on several synthetic (MIDI) recordings containing different instrument mixtures. Averaged over both sources, we achieved a frame-level accuracy of over 68% on an excerpt of *Pachelbel's Canon* arranged for doublebass and piano and 72% on a mixture of overlapping melodies played by flute and violin.

Index Terms— Music transcription, NMF, eigenmodels, source separation

1. INTRODUCTION

Music transcription is the process of obtaining a symbolic description of the events in an audio recording. Of the possible note event properties, pitch, onset time, duration, and volume are most naturally associated with score transcription, but separating notes according to the different source instruments is also important. Accurately extracting this information from a mixture of polyphonic instruments is a challenging problem, particularly in the case where we have only a single observation channel.

In this paper we present a system for separating and transcribing multi-instrument, polyphonic music. Currently, we assume that the number of sources (instruments) in the target mixture is known *a priori*, although each of these sources may be fully polyphonic and the degree of polyphony of each instrument is not assumed to be known. Our approach is inspired by eigenvoice modeling [2, 3] in that we first learn a subspace of source models and then use this subspace as a constraint on the solution of the sources present in the target mixture. However, rather than using principal components analysis (PCA) for the initial decomposition and hidden Markov models for the source models, we use NMF for both. This leads to a two-tiered NMF variant which we call *Subspace NMF* (SsNMF). We refer to the models learned during the first stage of our decomposition as *eigeninstruments* to reinforce the idea that they are a basis for the model space, despite the fact that in this case they do not strictly conform to the definition of eigenvectors.

This work was supported by the NSF grant IIS-0713334. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the sponsors.

2. BACKGROUND

Polyphonic music transcription has proven to be one of the most challenging problems in music information retrieval. Although techniques such as independent components analysis (ICA) can work well when at least as many observation channels as sources are available, effective techniques for the single-channel case remain elusive.

Many different approaches to transcription have been proposed in the literature. Klapuri [4] estimates the notes present in each frame using an algorithm that iteratively estimates and removes the fundamental frequencies of notes present. Poliner and Ellis [5] treat transcription as a classification problem, using support vector machines to classify individual frames as to whether they contain particular notes or not. Recently, non-negative matrix factorization (NMF) [6, 7] has become an increasingly popular technique for polyphonic transcription. When used with magnitude spectra, NMF has an elegant interpretation as a basis decomposition $V = WH$ where the basis vectors, W, correspond to note spectra and the weight vectors, H, give activation levels of those notes across time. Thus, for recordings containing only one source, H gives most of the information needed for transcription. This basic formulation can be extended to handle n instruments by simply interpreting the basis and weight matrices as having block forms:

$$V = WH = \left[W^1\ W^2 \cdots W^n\right] \begin{bmatrix} H^1 \\ H^2 \\ \vdots \\ H^n \end{bmatrix} \tag{1}$$

The difficulty in using (1) for transcription comes from the indeterminacy of W: without some prior knowledge, it is unclear how to properly assign each basis column W_i to its source submatrix, W^s. [1] Clearly, if W were known *a priori*, the problem would be substantially easier as we would only need to solve for H. However, in most cases we cannot assume that we will have access to the exact W used to produce V. A more general approach is to impose constraints on the solution space of W. We describe a method for incorporating these constraints next.

[1] We use calligraphic letters to denote data tensors (stacks of matrices), uppercase letters to denote matrices, bold lowercase to denote vectors, and regular lowercase to denote scalars. Superscripts index submatrices (both within matrices and within tensors) while subscripts index vectors in matrices and scalars in vectors. Double-subscripts are understood to refer to individual elements of matrices.

3. SUBSPACE NMF

3.1. Eigeninstruments

Assume that we have a set of m instrument models $\mathcal{M}$ for use in training. In our case, each model $\mathcal{M}^j \in \mathcal{M}$ consists of p pitch-specific magnitude spectra, each with f frequency bins. Our first goal is to decompose $\mathcal{M}$ into a set of r instrument model bases $\mathcal{W}$ (an f-by-p-by-r tensor) and an r-by-m matrix of weights C, such that the j^{th} model $\mathcal{M}^j$ can be approximated as:

$$\mathcal{M}^j \approx \sum_a^r \mathcal{W}^a C_{aj} \tag{2}$$

When the rank of the decomposition r is equal to m, the reconstruction is perfect. In practice this decomposition is accomplished by concatenating each model's p vectors together into a "supervector" (each of which can be thought of as a point in (pf)-dimensional space) and then stacking these supervectors together to form a (pf)-by-m parameter matrix Θ. This parameter matrix can then be decomposed using regular NMF:

$$\Theta \approx \Omega C \tag{3}$$

where Ω is the (pf)-by-r matrix of basis vectors, which is just $\mathcal{W}$ in stacked (supervector) form. Although in theory one could use other decomposition techniques such as PCA for this step, our model parameters correspond to magnitude spectra and therefore the non-negativity constraint of NMF seems natural. It is also useful to have nonnegative bases foor the NMF in the next stage.

A new instrument model W^s can now be represented in terms of this basis using coefficient vector $\mathbf{b}$:

$$W^s = \sum_a^r \mathcal{W}^a \mathbf{b}_a \tag{4}$$

3.2. Update Equations

Once we have solved for the model basis $\mathcal{W}$, we can approximate the magnitude STFT V of a mixture of n unknown instruments as:

$$V \approx R = \sum_s^n W^s H^s = \sum_s^n \sum_a^r \mathcal{W}^a B_{as} H^s \tag{5}$$

This approximation is illustrated in Figure 1. We solve for B and H in a similar fashion to Lee and Seung [1]. We use the $\mathcal{I}$-divergence of the reconstruction R from the original data V as the basis of our objective function:

$$D(V||R) = \sum_{i,j} V_{ij} \log \frac{V_{ij}}{R_{ij}} - V_{ij} + R_{ij} \tag{6}$$

This quantity is difficult to optimize directly due to the log-of-sum terms. Following Lee and Seung, we use Jensen's inequality to upper-bound the divergence function with an auxillary cost function. This cost function can then be minimized by taking partial derivatives, setting to 0, and solving for B_{as}. This results in the following update equation for B_{as}:

$$B_{as} \leftarrow B'_{as} \frac{\sum_{ij} V_{ij} \frac{\mathcal{W}^a H}{\sum_{sa} \mathcal{W}^a B'_{as} H}}{\sum_{ij} (\mathcal{W}^a H)_{ij}} \tag{7}$$

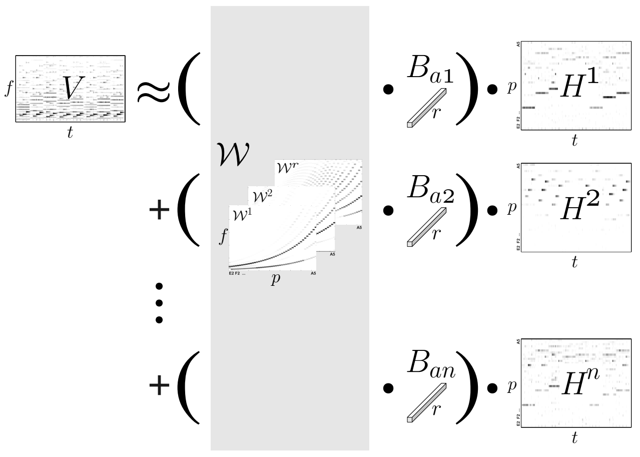

Figure 1: Illustration of the SsNMF decomposition of a spectrogram. The spectrogram is formed as the sum of several different instruments, each with its own set of per-note templates W^s and activations H^s. Each W^s, however, is formed as the weighted sum of a common eigeninstrument basis $\mathcal{W}$ collapsed via a vector of instrument-specific parameters B_s.

Once we have solved for B_{as}, we can use the result with the previously obtained model basis $\mathcal{W}$ to obtain W^s as in (4). This is done for each source s and the results are combined into $W = \left[W^1\ W^2 \cdots W^n\right]$. The update for H is basically the same as Lee and Seung's algorithm except that, as with W, we interpret it in block form such that the first p rows correspond to H^1, the next p to H^2, and so on:

$$H_{ij} \leftarrow H'_{ij} \frac{\sum_k W_{ki} \frac{V_{kj}}{(WH)_{kj}}}{\sum_l W_{li}} \tag{8}$$

The complete SsNMF algorithm is therefore:

1. Calculate $\mathcal{W}$ from the set of training models, $\mathcal{M}$
2. Initialize parameters matrices B and H randomly
3. Update B using (7)
4. Solve for W^s for $s = 1..n$ as in (4)
5. Combine models: $W = \left[W^1\ W^2 \cdots W^n\right]$
6. Update H using (8)
7. If the algorithm has not converged, goto 3

4. EVALUATION

We conducted several two-instrument transcription experiments in order to test the efficacy of our technique. Due to the scarcity of accurately transcribed and aligned audio and score data, and also to provide easier, more consistent data, we used recordings synthesized from MIDI. We do, however, select different instruments to train our eigenistrument models than the ones used in testing.

A set of 30 instruments, selected to be representative of a wide range of likely instrument types, was used in our experiments. In each experiment, the two instrument types used in the test mixture were excluded from the training set, leaving 28 models from which to learn $\mathcal{W}$. Each of the instrument models consisted of 42 pitches, ranging from E2 to A5. Models were created by synthesizing each note at a sampling rate of 8kHz using *timidity*. For the training data, the *EAW* instrument patch set was used for synthesis, while for test data the *Fluidsynth* R3 soundfont set was used. This provided additional assurance that training and testing datasets were well-separated. Next, the average spectrum of each note in each instrument in the training set was determined. This was done using a 512-point STFT with 64ms Hamming window and 16ms hop. These spectra were then stacked together to form the training models, $\mathcal{M}$. For both experiments, we used a rank of 25 when learning the eigeninstrument models. This value was chosen empirically.

4.1. Metrics

In order to quantify our results, we use a set of frame-based metrics proposed by [5]. The metrics consist of a total error measure which is comprised of three more specific types of error:

- E_{tot}: total error summed over time (i.e. $E_{sub}+E_{miss}+E_{fa}$)
- E_{sub}: the substitution error rate
- E_{miss}: the error rate due to missed notes
- E_{fa}: the error rate due to false alarms
- *Acc*: the percentage of correctly classified time-pitch cells

These metrics require a binary decision for each element H^s_{ij} as to whether the note represented by row i is on or off at the time represented by column j. We use an edge-detection and thresholding method to convert each H^s into a binary pianoroll representation that can be compared with the original score.

4.2. Experiments

The first experiment consisted of a four-bar excerpt from *Pachelbel's Canon* synthesized using doublebass for the lower voice and piano for the upper voice. We used the same parameters as in model construction for the STFT, and the SsNMF algorithm was then run until convergence using randomly initialized parameters.

Figure 2 shows the pianoroll representation of the score (blue denotes times when doublebass notes are sounding while green denotes piano notes) and the H^1 and H^2 matrices learned by the algorithm. We can see that the algorithm has done a fairly good job of separating and transcribing the two sources, despite having no explicit information (other than the number of sources) about the instruments contained in the mixture. Table 1 gives the values of the metrics after post-processing H^1 and H^2. Most of the error seems to be due to misclassfications, although the piano part has a relatively high false-alarm error rate which we can see from Figure 2 is likely due to cross-source contamination. It is also interesting to look at the inferred eigeninstrument weights B as this gives some indication of how much of the separation performance is due to the algorithm finding distinct regions of model space for each of the sources. These are shown in Figure 3.

The second experiment was designed to probe the abilities of the algorithm to handle mixtures containing sources not well-separated in frequency. We created a four-bar segment (see Figure 4 for score) containing two voices that occupied a narrow, common range. The melodies were designed to contain the same notes at a few points in time to examine the system's performance in these cases. The mixture was synthesized in the same way as

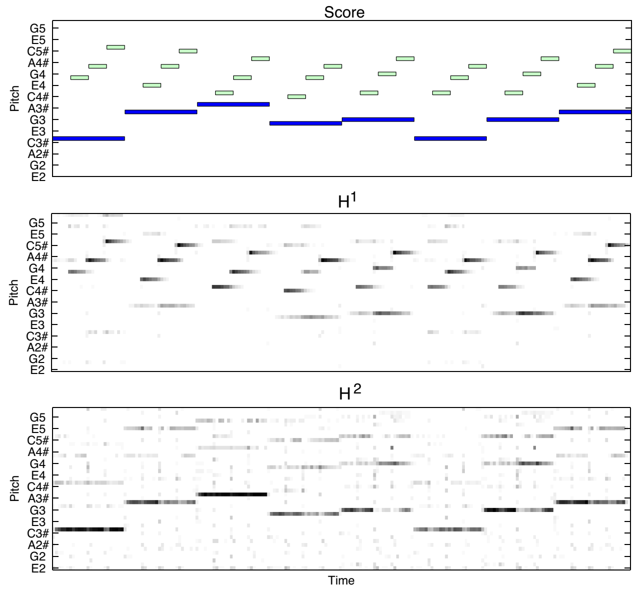

Figure 2: Pianoroll and raw transcription results of *Pachelbel's Canon*.

	Doublebass	Piano
E_{tot}	0.24	0.39
E_{sub}	0.06	0.08
E_{miss}	0.12	0.10
E_{fa}	0.06	0.20
Acc	0.73	0.64

Table 1: Evaluation metrics for *Pachelbel's Canon*.

Pachelbel's Canon, but violin and flute were used for the instruments as these seemed more appropriate for the range, as well as having quite distinct timbres. The raw transcription results are shown in Figure 5 while the binary metrics are given in Table 2.

While we might have expected the performance to be worse for this experiment, this is not case. As we can see from the metrics and by visual inspection of H^1 and H^2, the algorithm has performed remarkably well on this example. The reason for this surprise could be partly due to random initialization (although in our experience the algorithm is not overly sensitive to initial parameter settings) or the register of the voices. A more significant factor, however, is likely to be the match between the simple single-spectrum-per-note instrument model we use, and the actual instrument timbres. Additional experiments seem to confirm this theory: running the same experiment, but using doublebass and pi-

	Violin	Flute
E_{tot}	0.52	0.10
E_{sub}	0.04	0.01
E_{miss}	0.28	0.08
E_{fa}	0.21	0.01
Acc	0.55	0.89

Table 2: Evaluation metrics for the mixture of melodies performed by violin and flute.

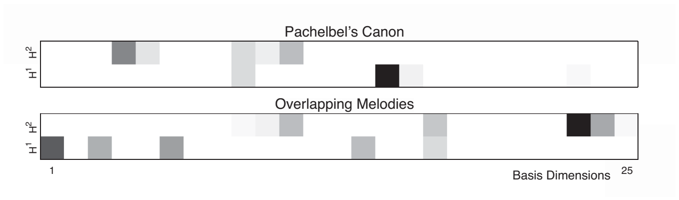

Figure 3: Learned weights (B) for the two sources in *Pachelbel's Canon* and the two sources in the overlapping melody mixture.

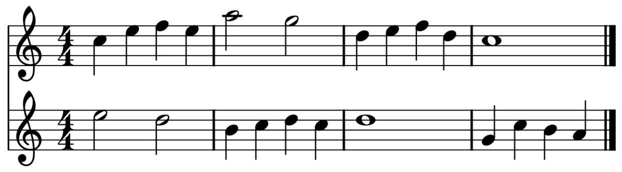

Figure 4: Score designed to contain two voices with substantial overlap.

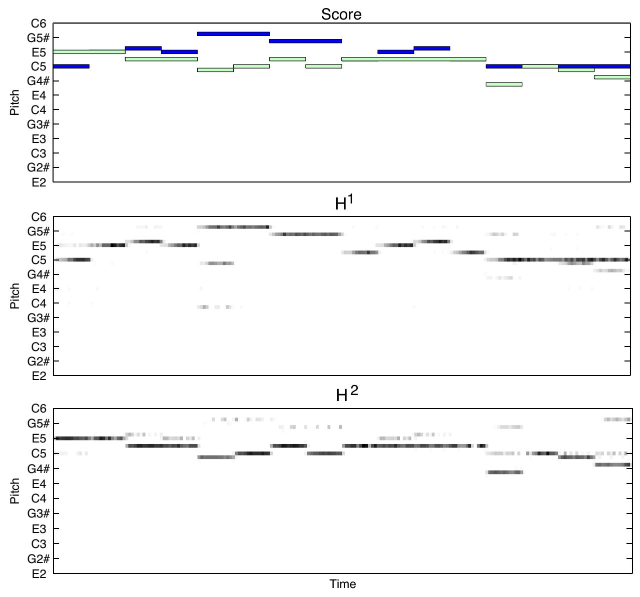

Figure 5: SsNMF transcription of the overlapping melody mixture.

ano as the instruments reveals a drop in performance, particularly for piano. Piano notes exhibit considerable variation in spectrum throughout their decay, and thus are difficult to fit with our current simple eigeninstrument model space.

5. DISCUSSION AND CONCLUSIONS

We have presented a novel data-driven approach to the challenging problem of polyphonic music transcription of multiple simultaneous instruments. The algorithm was tested on two synthetic polyphonic mixtures: an excerpt of *Pachelbel's Canon* and a mixture of two closely overlapping melodies, with average accuracy rates of 68% and 72%, respectively. Unlike most prior work in polyphonic transcription, these accuracies include the requirement that notes are correctly assigned to their respective instruments.

Although we believe that our results support the utility of the proposed approach, there are several areas in which improvements could be made. First, we intend to broaden our experiments to include more than two instruments as well as testing our algorithm on actual recorded audio. Although we have tried to avoid making our experimental setup too simplistic though the use of different synthesizers for training and test data and through the removal from the training set the instruments used in the test data, there is clearly no substitute for real recordings. Second, little attention was paid to initialization in our experiments, although NMF, and by extension our algorithm, are suceptible to local minima. There are numerous ways to combat this issue, including multiple restarts, annealing techniques, and instrument classifier methods. A third area for extension and improvement is the static-spectrum assumption that our model makes. Most instruments exhibit dynamic spectra over the course of a single note and we may be able to substantially improve performance by including this information in our model. Techniques such as convolutive NMF [8] which provide a means for incorporating temporal information into NMF may be useful in this context. We intend to explore these possibilities in the future.

6. REFERENCES

[1] D. Lee and H. Seung, "Algorithms for non-negative matrix factorization," in *Advances in Neural Information Processing Systems*, 2001, pp. 556–562.

[2] R. Kuhn, J. Junqua, P. Nguyen, and N. Niedzielski, "Rapid speaker identification in eigenvoice space," *IEEE Transactions on Speech and Audio Processing*, vol. 8, no. 6, pp. 695–707, November 2000.

[3] R. Weiss and D. Ellis, "Monaural speech separation using source-adapted models," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 2007, pp. 114–117.

[4] A. Klapuri, "Multiple fundamental frequency estimation based on harmonicity and spectral smoothness," *IEEE Transactions on Speech and Audio Processing*, vol. 11, no. 6, November 2003.

[5] G. Poliner and D. Ellis, "Improving generalization for classification-based polyphonic piano transcription," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 2007, pp. 86–89.

[6] P. Smaragdis and J. Brown, "Non-negative matrix factorization for polyphonic music transcription," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 2003, pp. 177–180.

[7] T. Virtanen and A. Klapuri, "Analysis of polyphonic audio using source-filter model and non-negative matrix factorization," in *Advances in Neural Information Processing Systems*, 2006.

[8] P. Smaragdis, "Non-negative matrix factor deconvolution; extraction of multiple sound sources from monophonic inputs," in *International Symposium on Independent Component Analysis and Blind Signal Separation*, 2004, pp. 494–499.