Constrained Nonnegative Matrix Factorization with Applications to Music Transcription

University of Waterloo Technical Report CS-2014-27*

Daniel Recoskie and Richard Mann

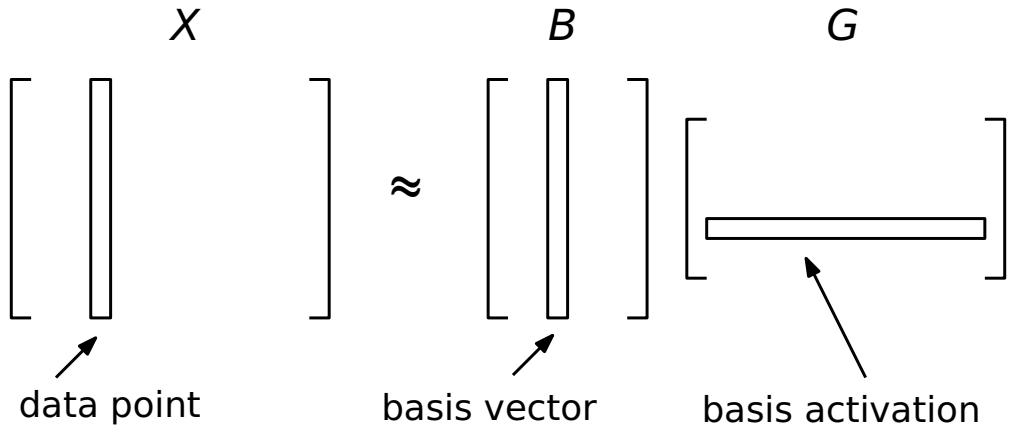
Cheriton School of Computer Science
University of Waterloo
Waterloo, Ontario, Canada
{dprecosk,mannr}@uwaterloo.ca

Abstract

We apply nonnegative matrix factorization to the task of music transcription. In music transcription we are given an audio recording of a musical piece and attempt to find the underlying sheet music which generated the music. We improve upon current transcription results by imposing novel temporal and sparsity constraints which exploit the structure of music. We demonstrate the effectiveness or our technique on the MAPS dataset.

1 Matrix Factorization

In this work we consider the problem of matrix factorization and its applications to music transcription. Given a data matrix X with columns as data points, we wish to approximate X as the product of two smaller matrices, B and G.



Furthermore, we restrict our matrices to be nonnegative. We make this restriction because many real world data is composed of nonnegative parts. We also consider the role constraints play in the model. More specifically, we add temporal and sparsity constraints in order to attain state of the art music transcription results.

*This work is a condensed version of the thesis presented in [16].

1.1 NMF in the literature

Nonnegative matrix factorization (NMF) has seen wide use in the literature. It was first used by Paatero and Tapper, but under the name positive matrix factorization [15]. Lee and Seung popularized the method as a tool for parts-based analysis [12, 13]. NMF has been used in a wide range of settings, from financial modelling to bioinformatics [9, 18]. For a collection of reviews see [2, 5, 6, 22]. In our work we concern ourselves with the task of music transcription.

Using NMF for music transcription was proposed by Smaragdis and Brown [17]. See Figure 1 for and outline of the process. Since then, there have been many improvements to the method. Some notable works include: a realtime sparse transcriber [8], a Bayesian approach [3], temporal and sparsity constraints [20, 21], and harmonic analysis [19]. For an exhaustive review of several music transcription methods, see [1].

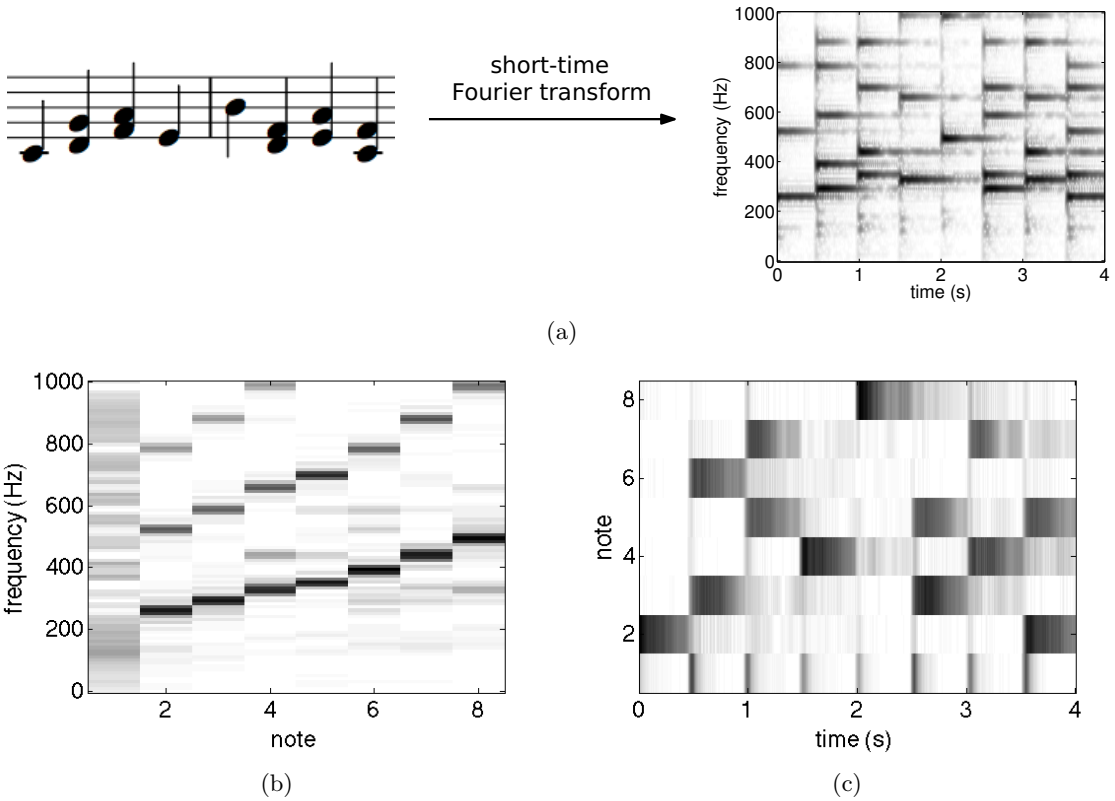


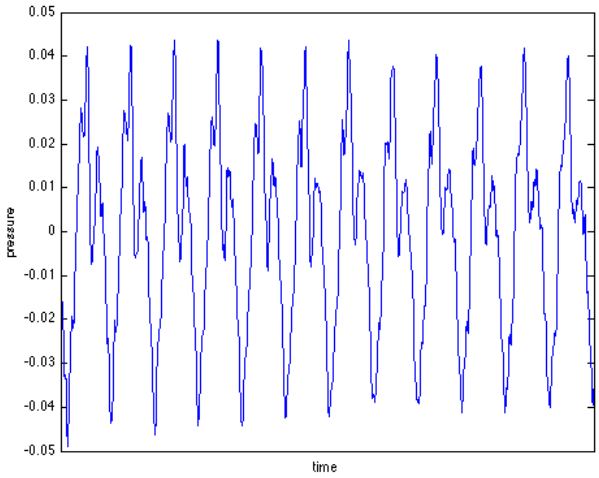
Figure 1: The short-time Fourier transform of an audio signal is taken to obtain the matrix X in (a). NMF produces (b) note matrix B and (c) transcription matrix G.

1.2 Outline

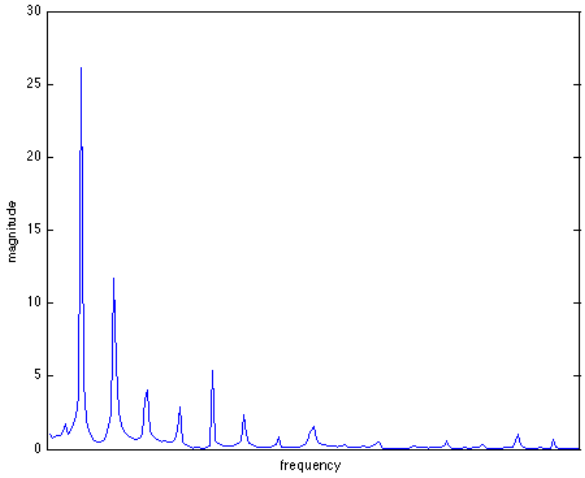
In this work we explore using NMF for music transcription. Section 2 briefly reviews time-frequency analysis. Section 3 describes our proposed changes to the traditional NMF model. Section 4 includes the results of our experiments. Finally, Section 5 contains our conclusion.

2 Time-frequency Analysis

In order to make use of NMF for music transcription we must transform our audio signals into a compatible form. Figure 2a shows an example of an audio signal represented as a pressure-time wave. We cannot discern much information from the audio when it is in this representation. However, if we consider the frequency representation of the signal as in Figure 2b, we are able to pick out the typical signature of a musical note.



(a)



(b)

Figure 2: (a) Example of an audio signal. (b) Magnitude of the Fourier transform of the audio signal in (a).

2.1 Fourier transform

We can make use of the Fourier transform in order to view audio signals in the frequency domain. The Fourier transform of a continuous signal, s, is defined as

$$S(\xi) = \int_{-\infty}^{\infty} s(t) e^{-2\pi i t \xi} dt \tag{1}$$

The inverse transform is defined as

$$s(t) = \int_{-\infty}^{\infty} S(\xi) e^{2\pi i t \xi} d\xi \tag{2}$$

For practical purposes, our signals will be discrete. Hence we need to modify the above transform to work with discrete signals. The discrete Fourier transform of a discrete signal $s_0, s_1, \ldots, s_{N-1}$ is

$$S_k = \sum_{n=0}^{N-1} s_n e^{-2\pi i k n / N} \tag{3}$$

Similarly, the inverse discrete Fourier transform is

$$s_n = \frac{1}{N}\sum_{k=0}^{N-1} S_k e^{2\pi i k n/N} \tag{4}$$

The Fourier transform is defined in terms of complex signals. For our purposes, we will only be concerned with the magnitude of the Fourier transform as seen in Figure 2b. By only considering the magnitude of the transform we lose the temporal information of the signal. We can remedy this shortcoming by modifying the Fourier transform.

2.2 Short-time Fourier transform

We can avoid losing temporal information by taking many transforms over small, time localized windows of the signal. This approach is called the short-time Fourier transform. The continuous version is defined as

$$S(\tau, \xi) = \int_{-\infty}^{\infty} s(t) w(t-\tau) e^{-2\pi i t \xi} dt \tag{5}$$

where w is the window function. Here S is a function of frequency and time. Similarly, the discrete short-time Fourier transform is defined as

$$S_{k,\xi} = \sum_{n=-\infty}^{\infty} s_n w_{n-k} e^{-2\pi i \xi n} \tag{6}$$

Consider the two signals in Figure 3. We see that we cannot distinguish between the two signals when we look at the magnitude of their Fourier transforms. However, the short-time Fourier transform shows clear differences.

3 Nonnegative Matrix Factorization

3.1 Background

In the formulation of NMF we are given a data matrix $X \in \mathbb{R}^{d\times n}$ and are tasked with finding two matrices $B \in \mathbb{R}^{d\times r}$ and $G \in \mathbb{R}^{r\times n}$, such that BG approximates X. Here, n is the number of samples or data points, d is the dimension of each sample, and r is the number of basis vectors used to approximate the data. We evaluate the approximation using a cost function.

Lee and Seung [13] propose using divergence as a cost function[1] to measure the closeness of BG to X

$$D(X||BG) = \sum_{i=1}^{d}\sum_{j=1}^{n}\left(X_{ij}\log\frac{X_{ij}}{(BG)_{ij}} - X_{ij} + (BG)_{ij}\right) \tag{7}$$

where M_{ij} refers to the element of M in the i^{th} row and j^{th} column. We are free to choose other cost functions, but we focus on divergence in this work. Lee and Seung use the following iterative update rules for finding B and G.

$$B_{ij} = B_{ij}\frac{\sum_{a=1}^{n} G_{ja} X_{ia}/(BG)_{ia}}{\sum_{b=1}^{n} G_{jb}} \tag{8}$$

and

$$G_{ij} = G_{ij}\frac{\sum_{a=1}^{d} B_{ai} X_{aj}/(BG)_{aj}}{\sum_{b=1}^{d} B_{bi}} \tag{9}$$

Note that these rules are similar to gradient descent with the exception that there is no fixed step size. Indeed, let us consider the additive gradient descent rule:

$$G_{ij} = G_{ij} + \eta_{ij}\left[\sum_{a=1}^{d} B_{ai}\frac{X_{aj}}{(BG)_{aj}} - \sum_{a=1}^{d} B_{ai}\right] \tag{10}$$

[1]Note that equation 7 reduces to the Kullback-Leibler divergence when $\sum_{i=1}^{d}\sum_{j=1}^{n} X_{ij} = \sum_{i=1}^{d}\sum_{j=1}^{n}(BG)_{ij} = 1$. In this work, we scale X so that its maximum element is one.

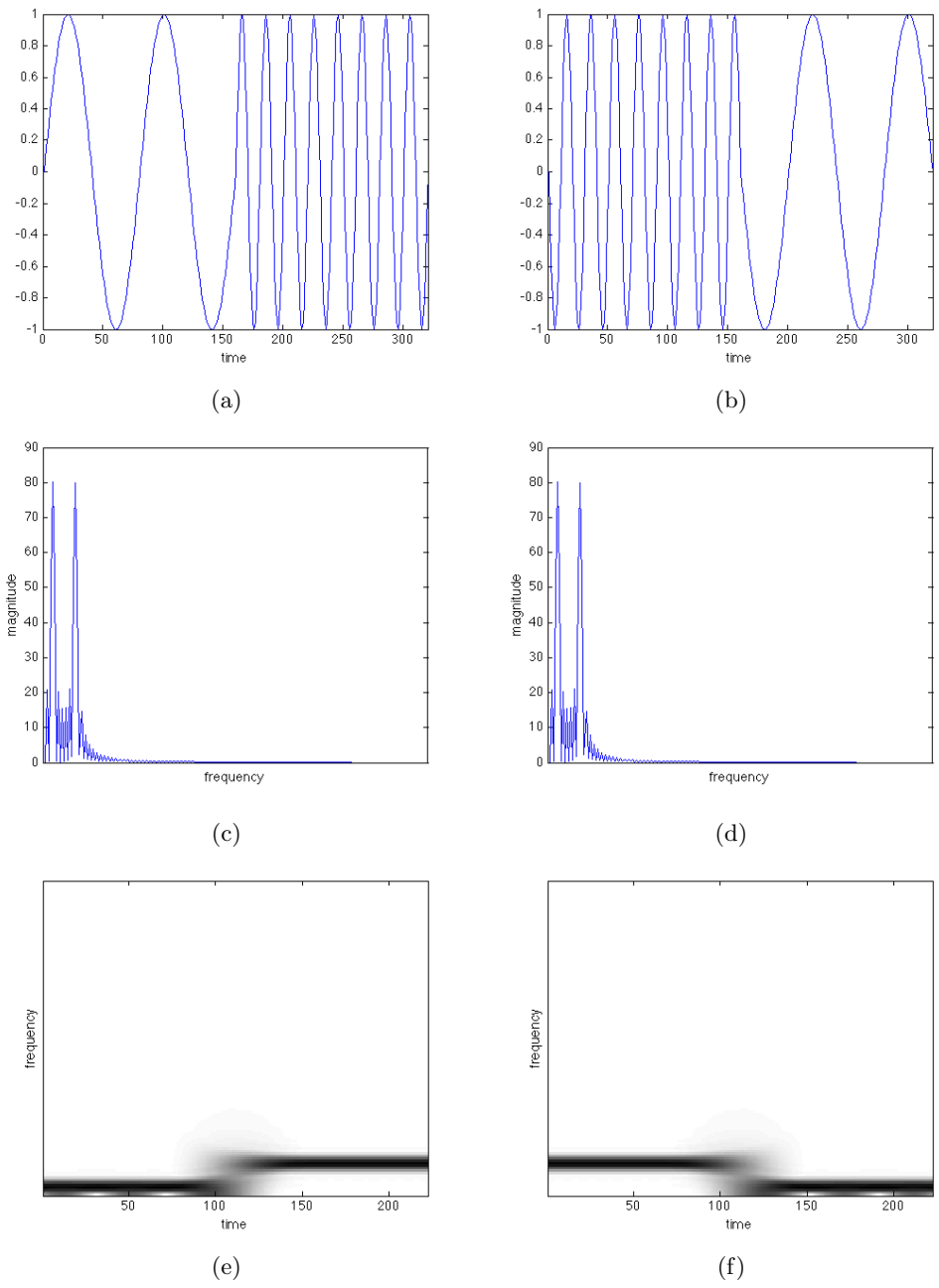
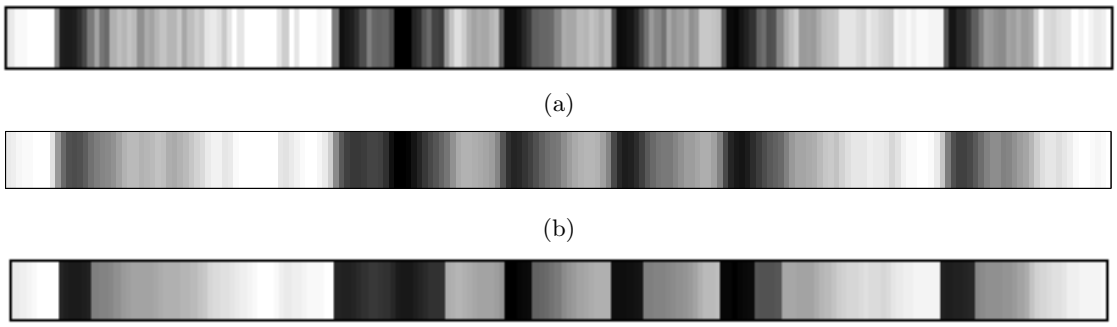


Figure 3: (a) A low frequency wave followed by a high frequency wave. (b) A signal opposite to (a). (c,d) Magnitudes of the Fourier transform of (a,b) respectively. (e,f) Magnitudes of the short-time Fourier transform of (a,b) respectively.

For small η_{ij} this update rule will decrease $D(X||BG)$. If we set

$$\eta_{ij} = \frac{G_{ij}}{\sum_{a=1}^{d} B_{ai}}$$

and substitute it into equation 10 we obtain equation 9. We can obtain equation 8 similarly. What remains is to show that equations 8 and 9 do not increase the cost functions. See [13] for a proof.



(a)

(b)

(c)

Figure 4: A row of G found by (a) standard NMF, (b) smoothed NMF, and (c) piecewise smooth NMF.

3.2 Constrained NMF

Let $F(X, BG) = D(X||BG)$, and define

$$\nabla_M F \triangleq \begin{pmatrix} \frac{\partial F}{\partial M_{11}} & \frac{\partial F}{\partial M_{12}} & \cdots & \frac{\partial F}{\partial M_{1j}} \\ \frac{\partial F}{\partial M_{21}} & \frac{\partial F}{\partial M_{22}} & \cdots & \frac{\partial F}{\partial M_{2j}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial F}{\partial M_{i1}} & \frac{\partial F}{\partial M_{i2}} & \cdots & \frac{\partial F}{\partial M_{ij}} \end{pmatrix} \tag{11}$$

for some $i \times j$ matrix M. Let $\nabla_M^+ F(X, BG)$ and $\nabla_M^- F(X, BG)$ be the absolute values of the positive and negative parts of $\nabla_M F(X, BG)$ respectively. We can rewrite update rules as [21]

$$B_{ij} = B_{ij} \frac{[\nabla_B^- F(X, BG)]_{ij}}{[\nabla_B^+ F(X, BG)]_{ij}} \tag{12}$$

and

$$G_{ij} = G_{ij} \frac{[\nabla_G^- F(X, BG)]_{ij}}{[\nabla_G^+ F(X, BG)]_{ij}} \tag{13}$$

Update rules of this form are often seen empirically in be nonincreasing [3]. Writing the update rules in this form allows us to easily add constraints to the cost function. We only consider constraints on G. Since B will represent the basis vectors of our data, we do not want to restrict its form.

3.2.1 Temporal coherence

In this work we are considering audio signals. Hence we can attempt to exploit the temporal coherence of the data. In the task of music transcription, the matrices B and G will represent the notes and transcription respectively. In order to exploit temporal coherence in our data, we may be tempted to smooth the rows of G by modifying the cost function to

$$D(X||BG) + \lambda \sum_{i=1}^{r} \sum_{j=2}^{n} (G_{ij} - G_{ij-1})^2 \tag{14}$$

where λ controls the weight of the constraint. This constraint attempts to smooth over the entire row. This leads to a problem because in music there are often abrupt breakpoints (note onsets and offsets). We can remedy this problem by allowing breakpoints in the rows of G.

Figure 4 shows the same row of G found by three different methods. Figure 4a is the results of standard NMF. Figure 4b shows the result of smoothing the entire row. Note how the breakpoints are removed. Finally, Figure 4c shows the result of our method that smooths the row but allows arbitrary breakpoints. With this method we are able to exploit temporal coherence effectively.

We can impose piecewise smoothness by modifying our cost function to be

$$D(X||BG) + \lambda \sum_{i=1}^{r} \sum_{j=2}^{n} \left(1 - e^{-(G_{ij} - G_{ij-1})^2 / 2\sigma^2}\right) \tag{15}$$

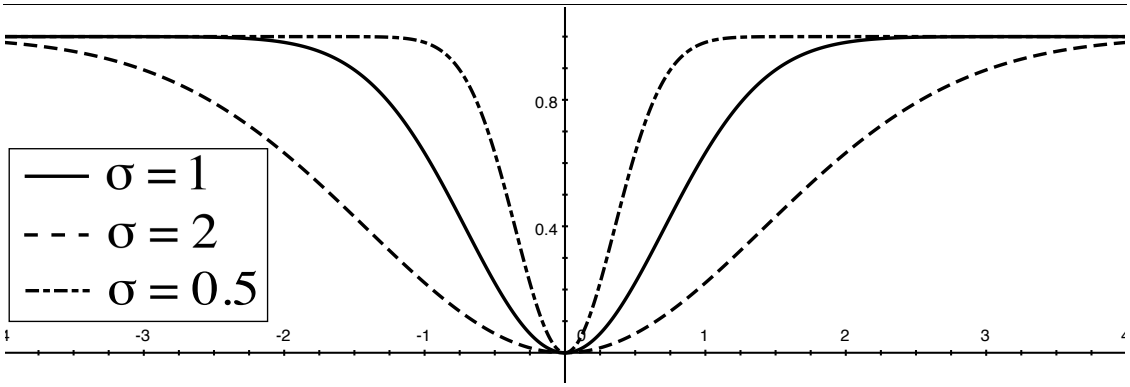


Figure 5: Plots of the function $1 - e^{-x^2/\sigma^2}$ for varying σ values.

See Figure 5 for plots of the function $1 - e^{-x^2/\sigma^2}$. The curve is similar to a quadratic for small values of x, and constant for large values of x. Similar functions are used for outlier detection in robust statistics [4].

Since the constraint only relies on G, the update rule for B remains unchanged. In order to modify the update rule for G, we must find the gradient of the cost functions with respect to G.

$$\nabla_G D(X||BG) = B^\intercal \left(\mathbf{1} - \frac{X}{BG}\right) \tag{16}$$

where $\mathbf{1}$ is the all ones matrix the same size as X.

$$\left[\nabla_G \lambda \sum_{i=1}^{r}\sum_{j=2}^{n}\left(1 - e^{-(G_{ij}-G_{ij-1})^2/2\sigma^2}\right)\right]_{ij} =$$

$$\frac{\lambda}{\sigma^2}\left(e^{-(G_{ij}-G_{ij-1})^2/2\sigma^2}(G_{ij} - G_{ij-1}) - e^{-(G_{ij+1}-G_{ij})^2/2\sigma^2}(G_{ij+1} - G_{ij})\right) \tag{17}$$

The update rule is then

$$G_{ij} = G_{ij}\,\frac{\left[B^\intercal \frac{X}{BG}\right]_{ij} + \frac{\lambda}{\sigma^2}\left(G_{ij-1}\cdot e^{-(G_{ij}-G_{ij-1})^2/2\sigma^2} + G_{ij+1}\cdot e^{-(G_{ij+1}-G_{ij})^2/2\sigma^2}\right)}{[B^\intercal \mathbf{1}]_{ij} + \frac{\lambda}{\sigma^2}G_{ij}\left(e^{-(G_{ij}-G_{ij-1})^2/2\sigma^2} + e^{-(G_{ij+1}-G_{ij})^2/2\sigma^2}\right)} \tag{18}$$

Jio and Qian propose a similar rule in their work on hyper spectral unmixing [11]. A proof of convergence can be found there.

We can further exploit temporal information by noting that breakpoints often coincide (i.e. several notes in a musical piece are often played simultaneously). We can make use of this correlation by adding a new constraint which favours aligned breakpoints. See Figure 6 for two rows of G before and after the alignment constraint is added.

We align breakpoints by modifying our cost function to be

$$D(X||BG) + \lambda \sum_{j=2}^{n}\left(1 - e^{-\sum_{i=1}^{r}(G_{ij}-G_{ij-1})^2/2\sigma^2}\right) \tag{19}$$

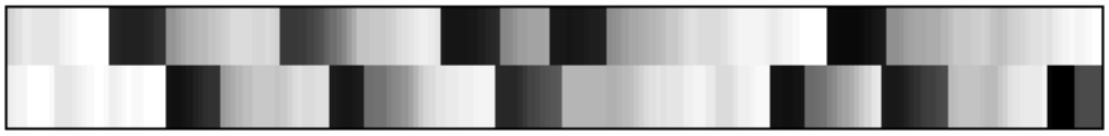
Similar to the last constraint, a single breakpoint receives a fixed cost of λ. The main difference with this new constraint is that multiple breakpoints that occur at the same time are assigned a total cost of λ instead of a cost of λ per breakpoint. We again determine the gradient with respect to G.

$$\left[\nabla_G \lambda \sum_{j=2}^{n}\left(1 - e^{-\sum_{i=1}^{r}(G_{ij}-G_{ij-1})^2/2\sigma^2}\right)\right]_{ij} =$$

$$\frac{\lambda}{\sigma^2}\left(e^{-\sum_{i=1}^{r}(G_{ij}-G_{ij-1})^2/2\sigma^2}(G_{ij} - G_{ij-1}) - e^{-\sum_{i=1}^{r}(G_{ij+1}-G_{ij})^2/2\sigma^2}(G_{ij+1} - G_{ij})\right) \tag{20}$$

The new update rule for G becomes

$$G_{ij} = G_{ij}\,\frac{\left[B^\intercal \frac{X}{BG}\right]_{ij} + \frac{\lambda}{\sigma^2}\left(G_{ij-1}\cdot e^{-\sum_{i=1}^{r}(G_{ij}-G_{ij-1})^2/2\sigma^2} + G_{ij+1}\cdot e^{-\sum_{i=1}^{r}(G_{ij+1}-G_{ij})^2/2\sigma^2}\right)}{[B^\intercal \mathbf{1}]_{ij} + \frac{\lambda}{\sigma^2}G_{ij}\left(e^{-\sum_{i=1}^{r}(G_{ij}-G_{ij-1})^2/2\sigma^2} + e^{-\sum_{i=1}^{r}(G_{ij+1}-G_{ij})^2/2\sigma^2}\right)} \tag{21}$$



(a)

(b)

Figure 6: Two rows of G found by imposing (a) piecewise smoothness and (b) piecewise smoothness with aligned breakpoints.

3.2.2 Sparsity

We now consider element wise sparsity of our G matrix. We can impose this constraint with the following cost term:

$$\lambda \sum_{i=1}^{d} \sum_{j=1}^{n} (1 - e^{-G_{ij}^2/2\sigma^2}) \tag{22}$$

with the gradient with respect to G being

$$\left[\nabla_G \lambda \sum_{i=1}^{d} \sum_{j=1}^{n} (1 - e^{-G_{ij}^2/2\sigma^2}) \right]_{ij} = \frac{\lambda}{\sigma^2} G_{ij} \, e^{-G_{ij}^2/2\sigma^2} \tag{23}$$

The update rule for G is:

$$G_{ij} = G_{ij} \frac{\left[B^\intercal \frac{X}{BG} \right]_{ij}}{\left[B^\intercal \mathbf{1} \right]_{ij} + \frac{\lambda}{\sigma^2} G_{ij} \, e^{-G_{ij}^2/2\sigma^2}} \tag{24}$$

4 Music transcription

In the problem of music transcription, we are given an audio recording of music and are tasked with finding the underlying sheet music which generated the music. We now outline the process as seen in Figure 1. First, we take the magnitude of the short time Fourier transform of our audio file. This transform, also called a spectrogram, is our data matrix X. In our experiments we used 100ms triangular windows overlapping by 50%. After finding the spectrogram, we compute B and G using the NMF update rules from the previous section. We used the following for estimating the σ value for each constraint.

- piecewise smooth: $\lambda \sum_{i=1}^{r} \sum_{j=2}^{n} \left(1 - e^{-(G_{ij} - G_{ij-1})^2/2\sigma^2} \right)$

$$\hat{\sigma} = std(\{G_{ij} - G_{ij-1} \mid 1 \le i \le d, 1 \le j \le n\}) \tag{25}$$

- aligned breakpoints: $\lambda \sum_{j=2}^{n} \left(1 - e^{-\sum_{i=1}^{r} (G_{ij} - G_{ij-1})^2/2\sigma^2} \right)$

$$\hat{\sigma} = std\left(\left\{ \sum_{j=2}^{n} (G_{ij} - G_{ij-1}) \mid 1 \le i \le d \right\} \right) \tag{26}$$

- element-wise sparsity: $\lambda \sum_{i=1}^{d} \sum_{j=1}^{n} (1 - e^{-G_{i,j}^2/2\sigma^2})$

$$\hat{\sigma} = std(\{G_{ij} \mid 1 \le i \le d, 1 \le j \le n\}) \tag{27}$$

Algorithm 1 Constrained NMF algorithm

for $scaling\_factor \leftarrow 10..\frac{1}{10}$ **do**
  for $i \leftarrow 1..num\_constraints$ **do**
    $\sigma_i \leftarrow scaling\_factor \cdot estimate(\sigma_i)$
  end for
  while change in $F(B,G) > \epsilon$ **do**
    $B_{ij} \leftarrow B_{ij} \frac{[\nabla^- F(X,BG)]_{ij}}{[\nabla^+ F(X,BG)]_{ij}}$
    $B \leftarrow scale(B)$
    $G_{ij} \leftarrow G_{ij} \frac{[\nabla^- F(X,BG)]_{ij}}{[\nabla^+ F(X,BG)]_{ij}}$
  end while
end for

where $std(S)$ is the standard deviation of the elements of the set S.

Ideally, the columns of B will correspond exactly to the notes contained in our musical piece, and G will be our transcription. The final step is assigning specific notes to each columns of B. We can do so by looking at the lowest frequency peak of each column. Note that in addition to the notes, one column of B generally corresponds to a noise vector. This can be ignored in the final transcription.

See Algorithm 1 for an outline of the constrained NMF algorithm. The B and G matrices are randomly initialized prior to running the algorithm. We make use of a scaling factor that ranges from 10 to $\frac{1}{10}$. This is done so that our constraint curves start out wide and become narrower at each iteration. See [16] for more detail. The columns of the B matrix are scaled at each iteration so that they each sum to one. After we have found G, we threshold its values to obtain our final transcription as follows,

$$G_{ij} = \begin{cases} 1 & \text{if } G_{ij} \geq \sigma \\ 0 & \text{if } G_{ij} < \sigma \end{cases}$$

where σ is the standard deviation of the elements of G.

For a given instrument there is generally a fixed set of possible notes that can be produced. As such, it is often more efficient to compute our B matrix once prior to transcription. Once we have learned B we hold it fixed and only update G in our NMF algorithm. We used this approach in our experiments.

We evaluate our transcription results using precision, recall, F-score, and mean overlap ratio (MOR). The first three measure the accuracy of our transcribed note onsets and are described below,

$$\text{precision} = \frac{tp}{tp + fp}, \quad \text{recall} = \frac{tp}{tp + fn}, \quad \text{F-score} = 2\frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

where tp and fp are the number of true and false positives respectively, and fn is the number of false negatives. A note onset is considered correct if it occurs within 50ms of the ground truth.

MOR measures note duration accuracy. Let on_g and on_t be the ground truth onset time and transcribed onset time respectively. Define off_g and off_t for offset times similarly. The overlap ratio is then

$$\frac{\min\{off_g, off_t\} - \max\{on_g, on_t\}}{\max\{off_g, off_t\} - \min\{on_g, on_t\}}$$

MOR is the average overlap ratio taken over all correctly transcribed notes.

4.1 MAPS

Our experiments make use of the MAPS (MIDI Aligned Piano Sounds) dataset [10]. We compare our results to those found by Bertin et al. [3]. The MAPS dataset contains synthetic and real piano music. The synthetic music is generated from software, whereas the real piano music was recorded on a player piano. We transcribe a total of 30 pieces each of synthetic and real songs (truncated to 30s).

Tables 1 and 2 list the results reported by Bertin et al. (first six rows) and our results (last row). Note that in contrast to our method, the methods used by Bertin et al. do not learn B prior to transcription.

We report the values of λ which gave the best results in Tables 3, 4, and 5. The λ values corresponding to the piecewise smooth, piecewise smooth with aligned breakpoints, and sparsity constraints are denoted by λ_{sm1}, λ_{sm2}, and λ_{sp} respectively.

Algorithm	F-score	Precision	Recall	MOR
NMF/MU	0.549	0.634	0.561	0.512
Vincent'08	0.584	0.607	0.600	**0.548**
H-NMF/MU	0.524	0.587	0.591	0.460
S-NMF	0.495	0.624	0.433	0.507
Virtanen'07	0.536	0.559	0.564	0.521
HS-NMF	**0.607**	**0.658**	**0.645**	0.443
Constrained NMF	**0.706** ($\pm$0.055)	**0.727** ($\pm$0.051)	**0.701** ($\pm$0.066)	**0.529** ($\pm$0.035)

Table 1: The results of Bertin et al. and constrained NMF for synthetic music samples. The top two results in each column are in bold.

Algorithm	F-score	Precision	Recall	MOR
NMF/MU	0.408	0.433	0.434	0.477
Vincent'08	0.361	0.387	0.374	**0.500**
H-NMF/MU	0.413	0.430	0.427	0.446
S-NMF	0.366	0.462	0.320	0.456
Virtanen'07	0.336	0.342	0.348	0.471
HS-NMF	**0.450**	**0.466**	**0.453**	0.432
Constrained NMF	**0.539** ($\pm$0.063)	**0.563** ($\pm$0.068)	**0.544** ($\pm$0.071)	**0.565** ($\pm$0.040)

Table 2: The results of Bertin et al. and constrained NMF for real (ambient) music samples. The top two results in each column are in bold.

λ_{sm1}	λ_{sm2}	λ_{sp}	F-score	Precision	Recall	MOR
0.0	0.0	0.0	0.379 ($\pm$0.033)	0.257 ($\pm$0.027)	0.751 ($\pm$0.055)	0.574 ($\pm$0.033)
0.0	0.0	1.8	0.704 ($\pm$0.049)	0.665 ($\pm$0.046)	0.759 ($\pm$0.063)	0.521 ($\pm$0.035)
0.0	1.5	0.0	0.431 ($\pm$0.046)	0.376 ($\pm$0.037)	0.540 ($\pm$0.079)	0.541 ($\pm$0.051)
0.0	0.5	1.5	0.706 ($\pm$0.055)	0.727 ($\pm$0.051)	0.701 ($\pm$0.066)	0.529 ($\pm$0.035)
0.5	0.0	0.0	0.587 ($\pm$0.056)	0.608 ($\pm$0.045)	0.586 ($\pm$0.072)	0.549 ($\pm$0.049)
0.1	0.0	0.9	0.696 ($\pm$0.069)	0.730 ($\pm$0.072)	0.692 ($\pm$0.069)	0.544 ($\pm$0.042)

Table 3: Constrained NMF results for synthetic data with 95% confidence intervals.

λ_{sm1}	λ_{sm2}	λ_{sp}	F-score	Precision	Recall	MOR
0.0	0.0	0.0	0.251 ($\pm$0.030)	0.158 ($\pm$0.021)	0.644 ($\pm$0.066)	0.565 ($\pm$0.035)
0.0	0.0	2.4	0.502 ($\pm$0.055)	0.480 ($\pm$0.058)	0.542 ($\pm$0.058)	0.495 ($\pm$0.040)
0.0	1.0	0.0	0.367 ($\pm$0.040)	0.302 ($\pm$0.031)	0.511 ($\pm$0.078)	0.536 ($\pm$0.049)
0.0	0.5	1.5	0.493 ($\pm$0.060)	0.509 ($\pm$0.063)	0.503 ($\pm$0.067)	0.546 ($\pm$0.030)
0.5	0.0	0.0	0.456 ($\pm$0.052)	0.446 ($\pm$0.047)	0.509 ($\pm$0.078)	0.523 ($\pm$0.055)
0.1	0.0	0.9	0.539 ($\pm$0.063)	0.563 ($\pm$0.068)	0.544 ($\pm$0.071)	0.565 ($\pm$0.040)

Table 4: Constrained NMF results for real (ambient) data with 95% confidence intervals.

5 Conclusion

In this work we explored the effects of constraints on the NMF model. Specifically, we consider the task of music transcription and compare our results to the current state of the art. NMF is a suitable method for music transcription due to the additive nature of music. That is, music is composed of a sum of overlapping notes. NMF is able to exploit this additivity in order to give good transcription results. We show that by adding temporal and sparsity constraints we can improve the transcription performance over the state of the art.

There are several avenues of future research to consider. In this work we only use the divergence cost function. One may consider using other cost functions (e.g. squared Frodenius norm). Alternative

λ_{sm1}	λ_{sm2}	λ_{sp}	F-score	Precision	Recall	MOR
0.0	0.0	0.0	0.314 ($\pm$0.026)	0.198 ($\pm$0.019)	0.789 ($\pm$0.046)	0.631 ($\pm$0.026)
0.0	0.0	1.8	0.661 ($\pm$0.049)	0.597 ($\pm$0.058)	0.762 ($\pm$0.045)	0.558 ($\pm$0.031)
0.0	1.0	0.0	0.442 ($\pm$0.026)	0.341 ($\pm$0.021)	0.657 ($\pm$0.058)	0.628 ($\pm$0.031)
0.0	0.5	1.5	0.698 ($\pm$0.043)	0.685 ($\pm$0.046)	0.724 ($\pm$0.052)	0.584 ($\pm$0.031)
0.5	0.0	0.0	0.618 ($\pm$0.041)	0.591 ($\pm$0.037)	0.667 ($\pm$0.058)	0.641 ($\pm$0.036)
0.1	0.0	0.6	0.715 ($\pm$0.041)	0.699 ($\pm$0.045)	0.747 ($\pm$0.052)	0.637 ($\pm$0.025)

Table 5: Constrained NMF results for real (close) data with 95% confidence intervals.

optimization methods could also be explored such as regularized alternating least squares [7] or projected gradient descent [14]. Finally, it could be beneficial to add higher level concepts to the model. For example, we may wish to include tempo, time signature, and key.

References

[1] Emmanouil Benetos, Simon Dixon, Dimitrios Giannoulis, Holger Kirchhoff, and Anssi Klapuri. Automatic music transcription: challenges and future directions. *Journal of Intelligent Information Systems*, 41(3):407–434, 2013.

[2] Michael W. Berry, Murray Browne, Amy N. Langville, V. Paul Pauca, and Robert J. Plemmons. Algorithms and applications for approximate nonnegative matrix factorization. In *Computational Statistics and Data Analysis*, pages 155–173, 2006.

[3] N. Bertin, R. Badeau, and E. Vincent. Enforcing harmonicity and smoothness in bayesian non-negative matrix factorization applied to polyphonic music transcription. *Audio, Speech, and Language Processing, IEEE Transactions on*, 18(3):538–549, March 2010.

[4] G. E. P. Box and S. L. Andersen. Permutation theory in the derivation of robust criteria and the study of departures from assumption. *Journal of the Royal Statistical Society. Series B (Methodological)*, 17(1):pp. 1–34, 1955.

[5] Ioan Buciu. Non-negative matrix factorization, a new tool for feature extraction: Theory and applications. *Int. J. Computers, Communications and Control*, 3:67–74, 2006.

[6] Moody Chu and Robert Plemmons. Nonnegative matrix factorization and applications. *Bulletin of the International Linear Algebra Society*, 34:2–7, 2005.

[7] Andrzej Cichocki and Rafal Zdunek. Regularized alternating least squares algorithms for non-negative matrix/tensor factorization. In Derong Liu, Shumin Fei, Zengguang Hou, Huaguang Zhang, and Changyin Sun, editors, *Advances in Neural Networks ISNN 2007*, volume 4493 of *Lecture Notes in Computer Science*, pages 793–802. Springer Berlin Heidelberg, 2007.

[8] Arshia Cont. Realtime multiple pitch observation using sparse non-negative constraints. In *International Conference on Music Information Retrieval*, 2006.

[9] Konstantinos Drakakis, Scott Rickard, Ruair De Frin, and Andrzej Cichocki. Analysis of financial data using non-negative matrix factorization. *Int. Mathematical Forum*, 3(38):1853–1870, 2008.

[10] V. Emiya, R. Badeau, and B. David. Multipitch estimation of piano sounds using a new probabilistic spectral smoothness principle. *Trans. Audio, Speech and Lang. Proc.*, 18(6):1643–1654, August 2010.

[11] Sen Jia and Yuntao Qian. Constrained nonnegative matrix factorization for hyperspectral unmixing. *Geoscience and Remote Sensing, IEEE Transactions on*, 47(1):161–173, Jan 2009.

[12] Daniel D. Lee and H. Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

[13] Daniel D. Lee and H. Sebastian Seung. Algorithms for non-negative matrix factorization. In *NIPS*, pages 556–562. MIT Press, 2000.

[14] Chih-Jen Lin. Projected gradient methods for nonnegative matrix factorization. *Neural Comput.*, 19(10):2756–2779, October 2007.

[15] Pentti Paatero and Unto Tapper. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5(2):111–126, 1994.

[16] Daniel Recoskie. Constrained nonnegative matrix factorization with applications to music transcription. Master's thesis, University of Waterloo, http://hdl.handle.net/10012/8639, 2014.

[17] P. Smaragdis and J.C. Brown. Non-negative matrix factorization for polyphonic music transcription. In *Applications of Signal Processing to Audio and Acoustics, 2003 IEEE Workshop on.*, pages 177–180, Oct 2003.

[18] Suvrit Sra and Inderjit S. Dhillon. Nonnegative matrix approximation: Algorithms and applications. Technical Report TR-06-27, Department of Computer Science, the University of Texas at Austin, June 2006.

[19] E. Vincent, N. Bertin, and R. Badeau. Adaptive harmonic spectral decomposition for multiple pitch estimation. *Audio, Speech, and Language Processing, IEEE Transactions on*, 18(3):528–537, March 2010.

[20] T. Virtanen. Sound source separation using sparse coding with temporal continuity objective. In *Proc. Int. Comput. Music Conf*, pages 231–234, 2003.

[21] T. Virtanen. Monaural sound source separation by nonnegative matrix factorization with temporal continuity and sparseness criteria. *Audio, Speech, and Language Processing, IEEE Transactions on*, 15(3):1066–1074, March 2007.

[22] Yu-Xiong Wang and Yu-Jin Zhang. Nonnegative matrix factorization: A comprehensive review. *Knowledge and Data Engineering, IEEE Transactions on*, 25(6):1336–1353, June 2013.