Distribution Agreement

In presenting this thesis as a partial fulfillment of the requirements for a degree from Emory University, I hereby grant to Emory University and its agents the non-exclusive license to archive, make accessible, and display my thesis in whole or in part in all forms of media, now or hereafter now, including display on the World Wide Web. I understand that I may select some access restrictions as part of the online submission of this thesis. I retain all ownership rights to the copyright of the thesis. I also retain the right to use in future works (such as articles or books) all or part of this thesis.

Signature:

Jonathan Valyou April 7, 2022

Nonnegative Matrix Factorization for Music - Tuning the NMF Algorithm with Regularization

By

Jonathan Valyou

Elizabeth Newman Ph.D.
Advisor

Department of Mathematics

Elizabeth Newman Ph.D.
Advisor

Lars Ruthotto, Ph.D.
Committee Member

Simon Blakey, Ph.D.
Committee Member

2022

Nonnegative Matrix Factorization for Music - Tuning the NMF Algorithm with Regularization

By

Jonathan Valyou

Elizabeth Newman Ph.D.
Advisor

An abstract of
a thesis submitted to the Faculty of Emory College of Arts and Sciences of
Emory University in partial fulfillment
of the requirements of the degree of
Bachelor of Science with Honors

Department of Mathematics

2022

Abstract

Nonnegative Matrix Factorization for Music - Tuning the NMF Algorithm with Regularization
By Jonathan Valyou

The mathematics behind music is a work of art in itself. Mathematicians have been utilizing mathematical tools to analyze music for decades. One such tool is Nonnegative Matrix Factorization (NMF) which has been used to decompose an audio signal into fundamental components in a source separation application. The NMF algorithm in a musical interpretation takes a spectral object known as a spectrogram represented by a matrix and separates the spectrogram into a two nonnegative sparse matrix product where one matrix takes temporal information of the sources, and the other matrix gives the frequency information of the sources. While the basic NMF algorithm excels at handling small in complexity problems with little noise, it fails to successfully separate the sources for problems with many sources or bad-quality audio data. One solution to this limitation is the implementation of regularization into the NMF algorithm. Regularization aims to induce qualities into our matrix factorization such as promoting sparsity or smoothing temporal readings that will improve the source separation accuracy. In this paper, we hope to introduce the simple NMF algorithm, display the source separation application of NMF, and demonstrate the effects of a regularized NMF algorithm.

Nonnegative Matrix Factorization for Music - Tuning the NMF Algorithm with Regularization

By

Jonathan Valyou

Elizabeth Newman Ph.D.
Advisor

A thesis submitted to the Faculty of Emory College of Arts and Sciences
of Emory University in partial fulfillment
of the requirements of the degree of
Bachelor of Science with Honors

Department of Mathematics

2022

Acknowledgments

I would like to thank my research advisor, Dr. Elizabeth Newman, for her wealth of knowledge and unwavering support of my research development. I would also like to thank Emory College of Arts and Sciences, and in particular the Emory University Mathematics Department, for encouraging the creation of this thesis and the publication of the work into the Emory University Theses Archive. I would like to acknowledge my friends both at and outside of Emory who have always been willing to discuss ideas and keep me motivated. Finally, I would like to thank my parents, sister, and family for their unconditional support of all of my endeavors and their encouragement every step of the way.

I want to express that this project proved to be extremely interesting and enjoyable as both a mathematician and a musician. I have always been fascinated with the applicability of mathematics in an interdisciplinary fashion and I am fortunate to be able to show yet another tie between the disciplines of music and mathematics and how they can aid one another.

Contents

List of Figures

Chapter 1

Introduction

Nonnegative Matrix Factorization (NMF) is an algorithm that aims to achieve the notable data science task of feature extraction and separation for nonnegative information. As a diverse array of disciplines are constrained to nonnegative data, this factorization method has proven to be useful in its application across a spectrum of contexts including learned image representations and topic modeling via text mining [10], spatial resolution of astronomical spectra [6], and liquid chromatography/mass spectrometry [28].

In this thesis, we explore the application of NMF in source separation of acoustic data, specifically music. Source separation is the process of dividing up an object or data set into distinct groupings. In terms of music, we are trying to cluster audio data into groups based on their pitches and/or instrumentation. This is an important topic to study because it might be hard to hear everything in a piece simply through the human ear. While it is easy to pick out the notes and instrumental voices in simple examples, music can become very complicated when given a piece with several instruments and these instruments each are playing a wide range of notes throughout the arrangement making it more difficult for the listener to separate the instruments and pitches within the audio. Source separation allows us to break down the compli-

cated pieces into simpler lines for instrumentation separation or into simple notes for pitch separation. This is where NMF can play a critical role by making the source separation of a complex piece a much easier process.

The goal of NMF mathematically is to approximate $\mathbf{X} \approx \mathbf{WH}$ where $\mathbf{X}$ is a nonnegative matrix of size $m \times n$, $\mathbf{W}$ is a nonnegative matrix of size $m \times r$, and $\mathbf{H}$ is a nonnegative matrix of size $r \times n$. The given inputs for NMF are $\mathbf{X}$, a set number of basis vectors considered rank r, and some measurement of distance for minimizing the distance between our given matrix $\mathbf{X}$ and the product of our approximate factorization $\mathbf{WH}$. This method should yield an approximate factorization of $\mathbf{X}$ in the form of $\mathbf{WH}$. We now have the following optimization problem:

$$\min_{\mathbf{W},\mathbf{H}} D(\mathbf{X}, \mathbf{WH}) \quad \text{s.t.} \quad \mathbf{W} \in \mathbb{R}_+^{m \times r}, \mathbf{H} \in \mathbb{R}_+^{r \times n} \tag{1.1}$$

Due to the method's representation of all of the data points through a set of linear combinations of basis vectors that is determined by the rank r, NMF is considered a linear dimensionality reduction method. Additionally, the reason for restricting the matrix input and outputs to be nonnegative is for both the mathematical formulation of the methodology as well as the units of measure in applications driven by this restriction. For example, in this project, we are focusing on frequencies and time which are nonnegative measurement tools.

Academics unfamiliar with NMF may question why the matrix approximation is $\mathbf{WH}$ instead of the typically seen $\mathbf{WH}^\top$ in factorization methods. $\mathbf{WH}$ is perfectly fine for a factorization as long as the columns of $\mathbf{W}$ and the rows of $\mathbf{H}$ are equivalent and in this case equal to the matrix rank, or number of sources. By our definition in Equation (1.1), this factorization should work just fine while maintaining the rules of matrix multiplication such that $\mathbf{X} \approx \mathbf{WH}$.

Additionally, it is worth noting why NMF is preferred for this music analysis

problem as opposed to Singular Value Decomposition (SVD) which is likewise able to extract meaningful insights through dimensionality reduction. While the SVD is almost always a useful factorization, it is not guaranteed to yield positive entries thus making it ineffective in extracting data points corresponding to nonnegative measurements involved in music. Additionally, the SVD promotes orthogonality in order to extract eigenvalues/eigenvectors. This promotion often causes a lack of sparsity making visual interpretations of the matrices much more difficult. Meanwhile, NMF as discussed previously does have these positive entry restrictions which means its matrix decomposition provides much more useful information in the context of source separation. In contrast to SVD, NMF encourages sparsity through a parts-of-a-whole feature separation. This allows the identification and interpretation of the features of the data through visual matrix representations much easier.

In this paper, we show the effectiveness of utilizing NMF for audio source separation in music. Additionally, we demonstrate that source separation with NMF can be challenging when "noise" is present in the audio data and present a regularized NMF algorithm that is more robust to noisy data. The thesis is organized as follows, we give a brief overview of the NMF literature, including common algorithms and applications in Section 2. Then, we set up a crucial NMF input through the process of converting an audio file into a visual representation of matrix $\mathbf{X}$ known as a spectrogram in Section 3. Next, we derive the NMF algorithm in Section 4 followed by some illustrative examples in Section 5 of NMF for source separation of both pitches and instruments. Then, in Section 6, We introduce the challenges of source separation through NMF such as audio remnants (background and artificial noise) and how regularization can circumvent these challenges. Then we conduct experiments and explore the effectiveness of both NMF and Regularized NMF in source separation in Section 7. In Section 8, we discuss some key takeways and consider some intriguing future directions.

Chapter 2

Related Works

NMF was first used in analytical chemistry during the 1960s. The early years of NMF involved scientists using a linear mixing mathematical model for analyzing sample chemical spectra and determining the elemental composition of solutions [35]. After approximately three decades, the first modernly utilized NMF model was created in 1994 under the name of Positive Matrix Factorization (PMF) [26]. Prior to the late 1990s, researchers had not ventured to utilize NMF for purposes outside of physics, chemistry, and similar fields. However, the release of a paper by Lee and Seung in 1999 coined the modern name Nonnegative Matrix Factorization and expanded the applications into a variety of new fields such as feature extraction and data mining for image and text data [11, 17]. Chemists expanded the utilization of NMF for spectroscopy in determining and analyzing absorption spectra over the time of a reaction [19]. NMF has even aided cancer studies [36, 9].

Alongside these application expansions, NMF was beginning to be employed for audio data. NMF has been used to analyze music through applications such as source separation, music structure identification, and audio mosaicing using spatial-temporal data [21, 20, 29, 27]. In particular, using NMF for source separation would allow people to separate voices, sounds, and/or instruments nearly blind to the audio

source content (besides the number of sources present). Thus, source separation can be used for nearly automatic music composition transcription and could aid in the restoration of written musical scores that may have been lost over time [4].

In the NMF application of source separation, audio data is pre-processed into a nonnegative matrix which is visually represented by what is called a spectrogram [22]. With pre-processing completed, an NMF algorithm is equipped to be applied [12]. Other methods, such as deep neural networks [33], have been successful at audio data analysis, but our focus is on NMF.

To account for the variety of NMF applications, researchers created NMF variant algorithms [7] of the multiplicative update algorithm proposed by Lee and Seung [17]. Each proposed algorithm has an accompanied convergence analysis [16]. Some notable NMF variant algorithms for audio source separation include Sparse NMF [34], Convolutive NMF [30], and NMF Deconvolution [15]. These algorithms are well-suited for audio data as they promote a greater degree of source separation in often fewer iterations as opposed to the standard multiplicative update NMF. These variations were utilized for polyphonic, multiple part, music decomposition for which the basic NMF algorithm has challenges in source separating such as having voice parts of similar frequencies or the presence of background noise.

Regularization is one such modification to the NMF algorithm. We utilize regularization to encourage desirable properties of the factor matrices, $\mathbf{W}$ and $\mathbf{H}$, such as smoothness or sparsity and make sure that the NMF algorithm does not overfit the data to undesirable features present in an audio recording such as feedback noise or whispers that are picked up by the recording device. There have been several papers introducing various regularized NMF algorithms in a variety of fields. These include a sparse graph regularized method [1], a Huber Loss regularized method [36], and a method combining L1 Regularization and Tikhonov Regularization [32]. These proposed regularized NMF algorithms are specifically designed to make the NMF

application more effective at its given application but a common trend among regularized NMF regularization expressions is the promotion of sparsity and separation between points in space.

Chapter 3

From Recording to Spectrogram

Before NMF can aid in distinguishing the musical voices, we must first construct our matrix $\mathbf{X}$ from the audio data. Here, $\mathbf{X}$ is a spectrogram, a visualization of the embodied matrix where the time is represented along the columns of $\mathbf{X}$ and the frequency is represented across the rows of $\mathbf{X}$. The entries of $\mathbf{X}$ denote the amplitude or volume of the pitch for a specified pitch i and time j. In the spectrogram, we denote each different voice or instrumental part (the sources for which we want to distinguish), with a source number and sometimes a different color for a low total number of sources. For the purposes of reading this thesis, we have created a diagram in Figure 3.1 that will aid readers in understanding how to read certain visuals as encountered throughout this paper when and where we visualize $\mathbf{X}$, $\mathbf{W}$, and $\mathbf{H}$.

In order to obtain the spectrogram, we must go through a process to mathematically represent music. We start by taking an audio file that was recorded using any standard recording device. These audio files contain sound waves that give information about frequency and time in a given musical arrangement.

The audio is spliced into short, segmented, slightly overlapping parts such that the amplitude and frequencies occurring at tiny intervals in time can be focused on. This process is called windowing. The smaller the window, the more time points and

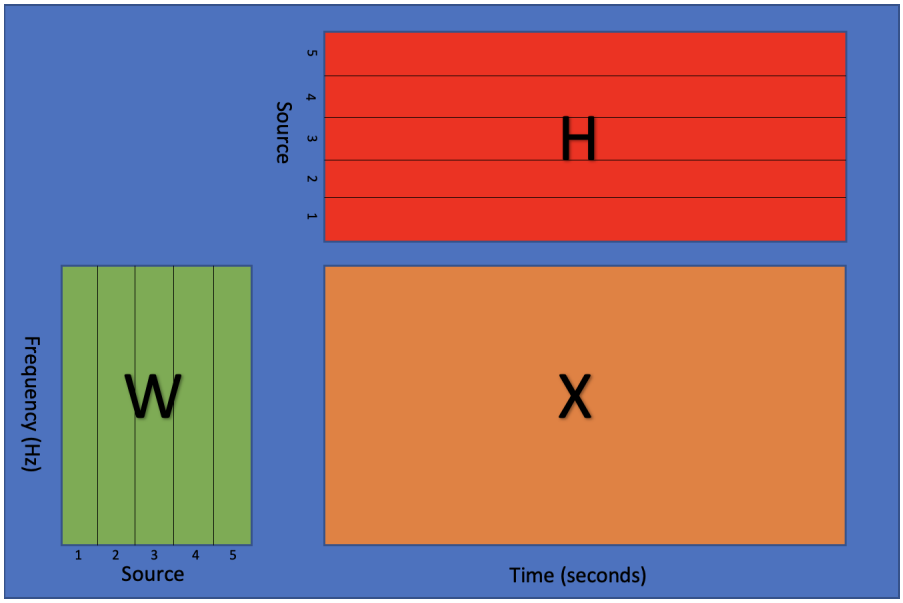

Figure 3.1: Visual Aid For Interpreting Diagrams – The spectrogram **X** is the bottom right corner with axes of time(seconds) and frequency(Hz). The visual representation of the frequency data for each source **W** is in the bottom left of the diagram. The visual representation of the temporal data for each source **H** is in the top right corner of the diagram.

therefore the better we can approximate the frequencies and the times at which they occur.

We proceed with a Fourier transformation to take these small audio intervals and translate them into mathematical points conveying frequency and time values. To do this, we model our segmented audio clips with sinusoidal functions. Therefore, we define a Discrete Fourier Transform (DFT):

$$D(k) = \sum_{n=0}^{N-1} x(n) \exp(-2\pi i k n/N) \tag{3.1}$$

that yields the Fourier coefficient $D(k)$ for a given window where k/N corresponds to one of a finite subset of frequencies and $x(n)$ is a finite sample from the window. This function is a one-to-one mapping from the signal space to the Fourier space. This is a discrete mapping as the sound signals are not continuous due to the windowing of the signal into a finite number of segments.

While the DFT reveals information about the frequencies in an audio recording, we also require the time values for when these frequencies occur. For this, we use the discrete Short-Time Fourier Transform (STFT). We piece together the DFT's into the STFT by defining a windowing function $w(n)$ to denote the segment of the audio that we are looking to transform, a hop size h to denote how much in time we move forward when we move to the next segment, and the time frame number m that we are currently looking at. Thus, we can write the STFT for a given time frame as:

$$\mathbf{X}(m,k) = \sum_{n=0}^{N-1} x(n+mh)w(n)\exp(-2\pi ikn/N) \tag{3.2}$$

Unlike the DFT, the STFT is now able to give us frequencies associated with the audio recording and when those frequencies are occurring which is all of the information needed to create the spectrogram.

One note to make is that the STFT and the DFT, like most Fourier transformations, have an output that typically exists in the complex vector spaces and therefore makes it difficult to visualize the result graphically. Thus, we take the log-spectral distance of the STFT to transform these complex values into real space to make them easier to visualize through the spectrogram. We illustrate the spectrogram in Figure 3.2.

As the amplitudes of the audio wave graph on top are much higher, the spectrogram below is shaded in with more color to indicate a louder volume of sound at the corresponding times. These sound intensities are entries in a matrix representation of the spectrogram. As these sound intensities are restricted to nonnegative values, we have the ability to factorize the spectrogram through NMF.

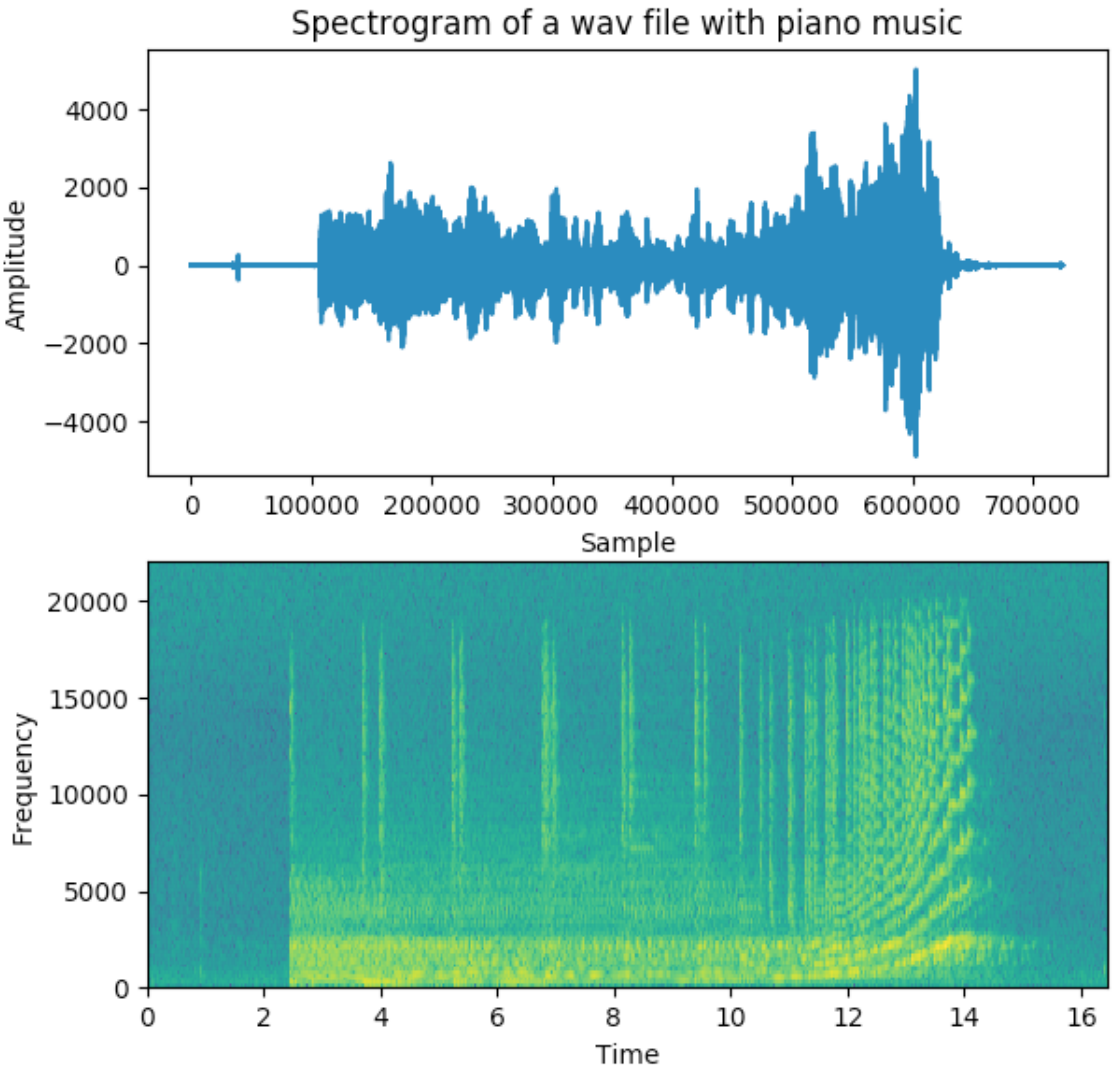

Figure 3.2: An Audio File And Its Respective Spectrogram - The top image is the visualization of a recorded audio signal with time (50000 units/1 second) and amplitude measuring the volume. The bottom image is the respective spectrogram **W** of the top image with time(seconds) and frequency(Hz). The yellow color in this diagram represents an existing frequency at a point in time of the recording whereas the blue represents the absence of a frequency at a particular point in time [8].

Chapter 4

Deriving the NMF Algorithm

With the spectrogram obtained, the input $\mathbf{X}$ for the NMF algorithm, it is necessary to derive the NMF algorithm. We further detail the preliminary optimization problem (1.1). First, we define $D(\mathbf{X}, \mathbf{WH})$ using the Frobenius norm: $\|\mathbf{X} - \mathbf{WH}\|_F^2$. Then we can rewrite our constraints where w_{ij} are the entries in $\mathbf{W}$ and h_{jk} are the entries in $\mathbf{H}$ such that $i = 1, 2, ..., m$, $j = 1, 2, ..., r$, $k = 1, 2, ..., n$. Therefore, the optimization problem becomes:

$$\min_{\mathbf{W},\mathbf{H}} \frac{1}{2}\|\mathbf{X} - \mathbf{WH}\|_F^2 \quad \text{s. t.} \quad \mathbf{W} \in \mathbb{R}_+^{m\times r}, \mathbf{H} \in \mathbb{R}_+^{r\times n} \tag{4.1}$$

This is a constrained optimization problem that is non-convex due to the optimization of both $\mathbf{W}$ and $\mathbf{H}$ simultaneously making it difficult to minimize the objective function. Typical algorithms use an alternating approach where we fix either $\mathbf{W}$ or $\mathbf{H}$ and take a step towards optimizing the matrix that is not fixed. Then we fix the other matrix and take a step to optimize the now not fixed matrix. Then, we keep switching which matrix is fixed and which is being updated until we reach some pre-defined tolerance level as described at the end of this section.

There are many ways to proceed with the updates of $\mathbf{W}$ and $\mathbf{H}$ such as the commonly used multiplicative and additive update rules. These update rules can be

derived from the Karush-Kuhn-Tucker (KKT) conditions [13, 14, 3] of (4.1).

The KKT conditions are the optimality conditions upon the constrained optimization problems that are necessary to indicate that a feasible point is optimal. In order to utilize the KKT conditions to determine optimality, the objective function must be convex and there must exist a feasible solution x^*. Then, we can determine an optimal solution when there exist $\lambda \in \mathbb{R}_+$ such that the four KKT conditions are met:

Theorem 4.0.1. *Necessary Conditions for KKT Conditions*

Consider the minimization problem:

$$\min_x f(x) \quad \text{s. t.} \quad a_i^\top x \geq b_i, \quad i = 1, 2, ..., m \tag{4.2}$$

where f is a continuously differentiable objective function, $a_i \in \mathbb{R}^n$, $b_i \in \mathbb{R}$, and x^* be a feasible solution and local minimum. Then there exist $\lambda_i \geq 0$ such that:

$$\nabla_x L(x^*, \lambda_i) = \nabla f(x^*) + \sum_{i=1}^{m} \lambda_i a_i = 0 \tag{4.3}$$

where L is the Lagrangian and

$$\lambda_i (a_i^\top x^* - b_i) = 0 \tag{4.4}$$

For our given optimization problem (4.1), there is at least one feasible solution where the KKT conditions in Theorem 4.0.1. hold true. So, we utilize the KKT conditions to derive the Euclidean Update rule for the NMF algorithm.

First, we rewrite the objective function f of the optimization problem (4.1):

$$f(\mathbf{W}, \mathbf{H}) = \frac{1}{2} \|\mathbf{X} - \mathbf{WH}\|_F^2 = \frac{1}{2} \sum_{i=1}^{m} \sum_{k=1}^{n} (x_{ik} - \sum_{j=1}^{r} w_{ij} h_{jk})^2 \tag{4.5}$$

In order to write the KKT conditions, we must find the Lagrangian:

$$\begin{aligned} L(\mathbf{W},\mathbf{H},\mathbf{\Lambda},\mathbf{M}) &= \frac{1}{2}\sum_{i=1}^{m}\sum_{k=1}^{n}(x_{ik}-\sum_{j=1}^{r}w_{ij}h_{jk})^2+\sum_{i=1}^{m}\sum_{j=1}^{r}\lambda_{ij}(-w_{ij})+\sum_{j=1}^{r}\sum_{k=1}^{n}\mu_{jk}(-h_{jk}) \\ &= \frac{1}{2}\|\mathbf{X}-\mathbf{WH}\|_F^2-\langle\mathbf{\Lambda}\ ,\ \mathbf{W}\rangle-\langle\mathbf{M}\ ,\ \mathbf{H}\rangle \end{aligned} \tag{4.6}$$

where each λ_{ij},μ_{jk} $\in \mathbb{R}_+$ are the KKT multipliers that make up the entries of $\mathbf{\Lambda} \in \mathbb{R}^{m\times r}$ and $\mathbf{M} \in \mathbb{R}^{r\times n}$ respectively and $\langle\cdot\ ,\ \cdot\rangle$ is defined as an inner product. Additionally, we define $\odot$ to signify point-wise matrix multiplication.

The stationarity conditions for each entry are:

$$\nabla_{\mathbf{W}}L=(\mathbf{WH}-\mathbf{X})\mathbf{H}^\top-\mathbf{\Lambda} \qquad \text{where } \nabla_{\mathbf{W}}L\in\mathbb{R}^{m\times r} \tag{4.7}$$

$$\nabla_{\mathbf{H}}L=(\mathbf{WH}-\mathbf{X})\mathbf{W}^\top-\mathbf{M} \qquad \text{where } \nabla_{\mathbf{H}}L\in\mathbb{R}^{r\times n} \tag{4.8}$$

The complementary slackness conditions are:

$$\mathbf{\Lambda}\odot\mathbf{W}=\mathbf{0},\quad \mathbf{M}\odot\mathbf{H}=\mathbf{0} \tag{4.9}$$

The primal feasibility conditions are:

$$\mathbf{W}\in\mathbb{R}_+^{m\times r},\quad \mathbf{H}\in\mathbb{R}_+^{r\times n} \tag{4.10}$$

The dual feasibility conditions are:

$$\mathbf{\Lambda}\in\mathbb{R}_+^{\mathbf{m}\times\mathbf{r}},\quad \mathbf{M}\in\mathbb{R}_+^{\mathbf{r}\times\mathbf{n}} \tag{4.11}$$

Based on the stationarity conditions:

$$\nabla_{\mathbf{W}} L = (\mathbf{WH} - \mathbf{X})\mathbf{H}^\top - \mathbf{\Lambda} = \mathbf{0} \rightarrow (\mathbf{WHH}^\top - \mathbf{XH}^\top) = \mathbf{\Lambda} \tag{4.12}$$

We plug this into our complementary slackness conditions:

$$\mathbf{\Lambda} \odot \mathbf{W} = \mathbf{0} \rightarrow \mathbf{W} \odot (\mathbf{WHH}^\top - \mathbf{XH}^\top) = \mathbf{0} \tag{4.13}$$

$$\rightarrow \mathbf{W} \odot (\mathbf{WHH}^\top) = \mathbf{W} \odot (\mathbf{XH}^\top) \tag{4.14}$$

Through point-wise matrix division, we find:

$$\mathbf{W} = \mathbf{W} \odot \frac{\mathbf{XH}^\top}{\mathbf{WHH}^\top} \tag{4.15}$$

The above formulation is the multiplicative update. When $\mathbf{W}$ is not changing, the first order optimality conditions are met. Note that the computation utilizes point-wise division to produce a matrix of approximately ones theoretically from $\frac{\mathbf{XH}^\top}{\mathbf{WHH}^\top}$ if $\mathbf{WH} \approx \mathbf{X}$. This is a point-wise operation to maintain the shape of $\mathbf{W}$ over all iterations and as we update each entry in $\mathbf{W}$ individually by a scalar multiplicative update.

We update $\mathbf{H}$ using a similar process. Thus, the following overall multiplicative update rule is found where t is the current iteration number:

$$\mathbf{W}^{(t+1)} = \mathbf{W}^{(t)} \odot \frac{\mathbf{XH}^{(t)\top}}{\mathbf{W}^{(t)}\mathbf{H}^{(t)}\mathbf{H}^{(t)\top}} \tag{4.16}$$

$$\mathbf{H}^{(t+1)} = \mathbf{H}^{(t)} \odot \frac{\mathbf{W}^{(t+1)\top}\mathbf{X}}{\mathbf{W}^{(t+1)\top}\mathbf{W}^{(t+1)}\mathbf{H}^{(t)}} \tag{4.17}$$

After initializing $\mathbf{W}$ and $\mathbf{H}$ with nonnegative entries, our update rule preserves the nonnegativity throughout the iteration process. We iterate until we hit the stopping criteria. The stopping criteria is related to the optimality conditions in that we

want the algorithm to terminate when our stationarity conditions (4.7) (4.8) are satisfied. This would indicate that we have approximated a stationary point for our optimization problem.

This criteria could be one of several options or even a combination of the tolerance options. The stopping criteria could be set at a tolerance ϵ of convergence for both $\mathbf{W}$ and $\mathbf{H}$ to attain on a given iteration:

$$\|\mathbf{H}^{(t+1)} - \mathbf{H}^{(t)}\| < \epsilon, \quad \|\mathbf{W}^{(t+1)} - \mathbf{W}^{(t)}\| < \epsilon \tag{4.18}$$

Another option is to set the tolerance criteria at a minimum threshold ϵ for the objective function:

$$\|\mathbf{X} - \mathbf{W}^{(t)}\mathbf{H}^{(t)}\| < \epsilon \tag{4.19}$$

4.1 NMF Convergence

We discuss the convergence theory for the multiplicative update. First, let $F(h)$ be the objective function for a single column of $\mathbf{H}$:

$$F(h) = \frac{1}{2}||x - \mathbf{W}h||_2^2 \tag{4.20}$$

We define t to be the current iteration and h_t as the current column of $\mathbf{H}$. We can approximate h_t using a quadratic Taylor expansion:

$$F(h) \approx F(h_t) + \nabla F(h_t)^T(h - h_t) + \frac{1}{2}(h - h_t)^T\nabla^2 F(h_t)(h - h_t) \tag{4.21}$$

where $\nabla F(h_t) = -\mathbf{W}^T(x - \mathbf{W}h_t)$ and $\nabla^2 F(h_t) = \mathbf{W}^T\mathbf{W}$.

If we were to try to minimize this quadratic expansion with respect to h, we would

find that the nonnegativity of h is not guaranteed. Therefore, we instead create an auxilary function G such that G is equal to the objective function F at the current iterate and such that G majorizes F for all h:

$$G(h_t, h_t) = F(h_t), \quad G(h, h_t) \geq F(h) \tag{4.22}$$

We further define the auxiliary function such that minimizing G results in the derived multiplicative step. We repeatedly minimize to the local optimum of the G and then build a new G at each iteration. This means that the local minima of the updated auxilary function from iteration to iteration will be nonincreasing towards the lower bounded F. We cannot guarantee that this update rule will converge as we are taking infinite steps towards F and F does not guarentee nonnegative columns of $\mathbf{H}$. Pictorially, we can envision this in Figure 4.1.

For those interested in seeing how G is chosen and further details on NMF convergence, a full length proof of convergence can be found at [16].

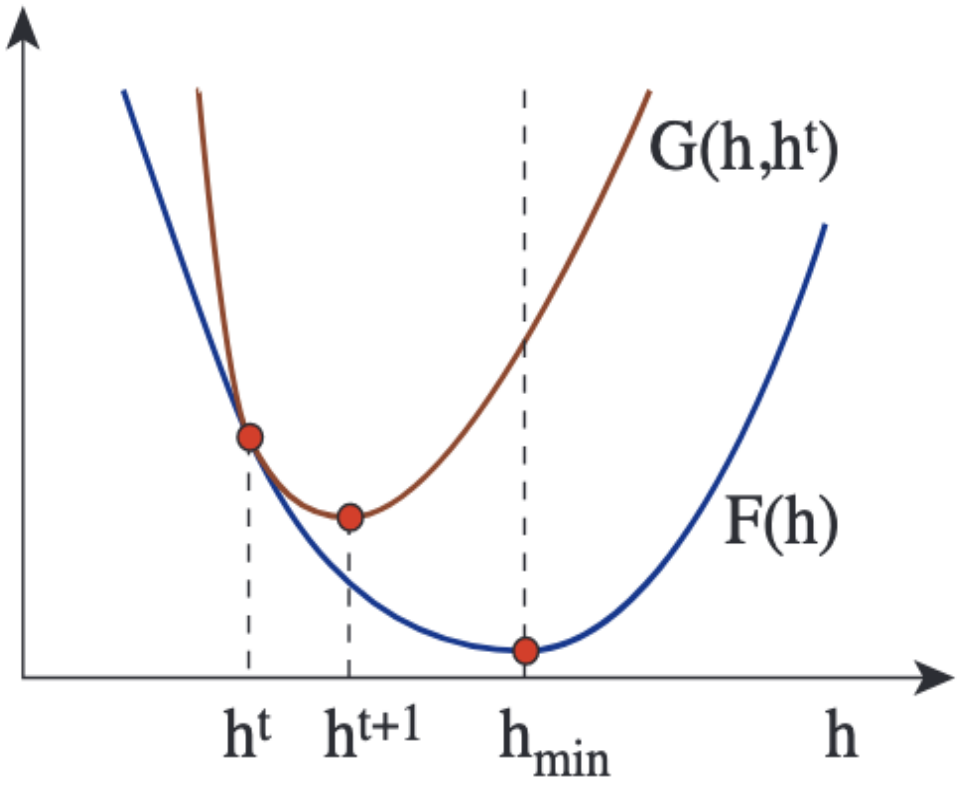

Figure 4.1: NMF Algorithm Convergence – An example of a defined auxiliary function $G(h, h^t)$ that is bounded below a given function $F(h)$ to demonstrate the convergence pattern of the NMF algorithm. With each iteration of the NMF algorithm, the current solution to the optimization problem moves from h^t to h^{t+1} and then the auxilary function changes such that we can continue to minimize toward h_{min}[16].

Chapter 5

Illustrative Examples

To visualize the NMF algorithm, we observe source separation through two examples: an audio of a C major scale on a clarinet (§5.1) and an audio of three drums playing simultaneously (§5.2). The NMF algorithm used comes from the NMF Toolbox [20] known as a Nonnegative Matrix Factor Deconvolution (NMFD) [31]. NMFD differs from NMF as it utilizes a convolution operator that shifts the columns of our spectrogram $\mathbf{X}$ to the right up to a set number of template frames. This allows NMFD to account for more temporal data from the spectrogram and improve the accuracy of the source separation by time within $\mathbf{H}$. This is favorable for these examples as we want to demonstrate the separation of sources application of NMF algorithms as clearly as possible.

5.1 C Scale Example

In this example, we took an audio sample of the C major scale [24].

Following the process in Section 3, we split the signal into slightly overlapping time frames and multiplied each frame by a window function to avoid numerical artifacts from splitting the signal. Finally, we obtain the columns of the spectrogram by taking a STFT at each time frame allowing us to create the spectrogram $\mathbf{X}$ of the C scale.

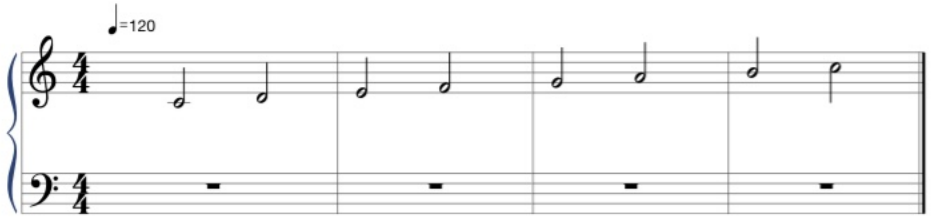

Figure 5.1: C Major Scale – The 8 notes of the C major scale arranged as half notes demonstrating how music is often visualized as a music score on sheet music. This image was arranged by Jonathan Valyou using the application Notation Pad.

For this example, the spectrogram is present in the bottom right corner of Figure 5.2. After running the NMFD algorithm, we output **W** in the bottom left corner of Figure 5.2 and **H** in the top right corner of Figure 5.2.

Each column of **W** represents a source. The entries within each column correspond to the different frequencies detected for a source. The matrix size is $m \times r$ with m frequencies r sources. In this case, we have 8 distinct notes in the C scale so we chose the rank of the matrix to be $r = 8$.

Similarly, each row of **H** represents a source. The entries within each row correspond to the time points where a source is active. The size of **H** is $r \times n$ where r is the number of sources and n is the number of time points dependent on the STFT set-up. In this visualization of matrix **H**, we can see the times where a specific note is played as they are each represented by a large area of gray coloring. In fact, since the time range runs in chronological order identically to the audio sample and the knowledge that the audio sample is an ascending C scale, we can actually identify which source maps to which note in this NMF output. For example, the left-most gray section starting from time at 0 seconds occurs in the source row labeled 4 and since the first note in the C scale is a low C, source 4 must correspond to the low C in our NMF output. That means we can also look back at our visual representation

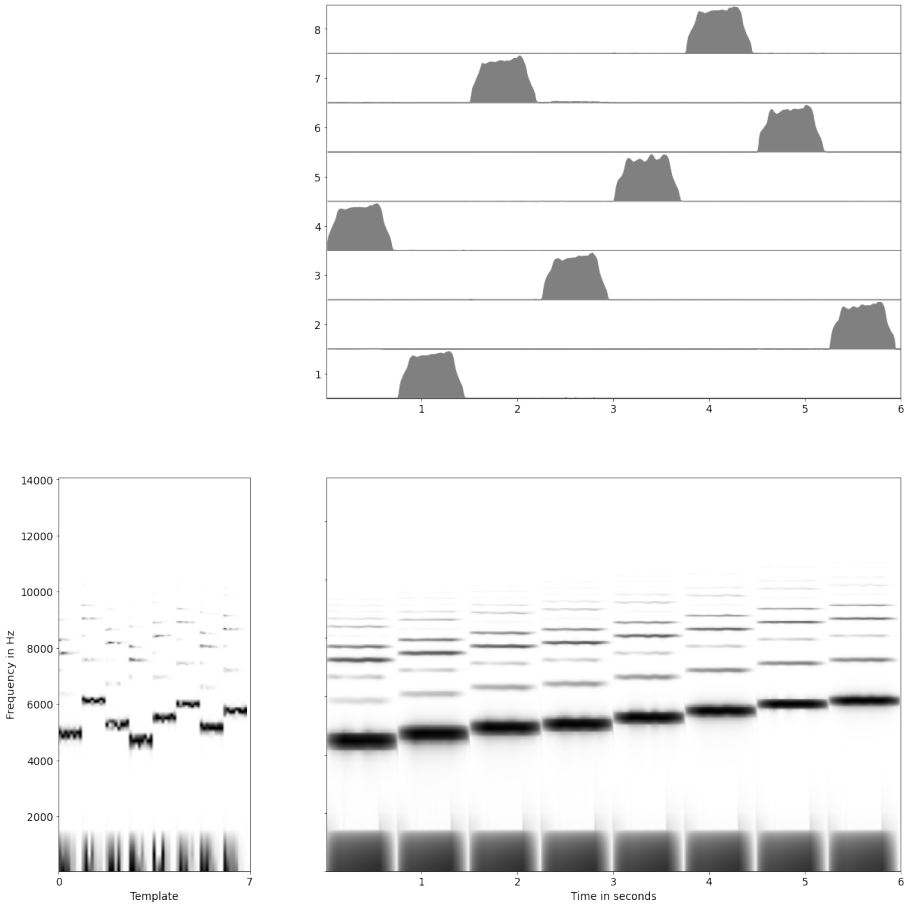

Figure 5.2: Visualizing NMF Through C Major Scale- The NMFD algorithm was utilized with input parameters of 8 sources for 8 distinct pitches, 300 iterations, 8 Template Frames to specify 8 Convolutions, randomly initialized **W**, and uniformly initialized **H**. For information on how to interpret this diagram, see Figure 3.1.

of **W** at the column labeled source 4 and see the frequencies that will be associated with the low C pitch of our scale. This process can be repeated for any of the other notes in the C scale.

It should be noted that our **W** and **H** are both sparse as represented with white space in our visualization in 5.2. In fact, **W** and **H** appear to have a greater degree of sparsity than the spectrogram **X**.

Convergence in C Scale Example

We can validate the NMF convergence claim from the end of (§4) by graphing the value of the objective function, $\|\mathbf{X} - \mathbf{WH}\|_F$, over several iterations of the basic NMF algorithm. In Figure 5.3, we see that as the iteration number increases for the Scale Example, the value of the objective function decreases monotonically. Since this graph is on a logarithmic scale, we can see that the change in the objective function quickly decreases within the first few iterations and has relatively smaller changes but can always be minimized further.

Figure 5.4 shows the change in $\mathbf{W}$ and $\mathbf{H}$ that occurs at each iteration. These graphs computed the relative residuals, $\|\mathbf{H}^{(t+1)} - \mathbf{H}^{(t)}\|_F$ and $\|\mathbf{W}^{(t+1)} - \mathbf{W}^{(t)}\|_F$, of each matrix using the Frobenius Norm. Scaled logarithmically, we see the differences in both $\mathbf{W}$ and $\mathbf{H}$ between iterations tend to decrease, indicating that we may be converging upon a solution to the objective function. It is worth noting that unlike the previous figure measuring the change in the objective function, we do not see that the change in $\mathbf{W}$ and $\mathbf{H}$ is always monotonically decreasing. This is due to the fact that we are trying to minimize both $\mathbf{W}$ and $\mathbf{H}$ at each iteration using the alternating approach derived from the KKT conditions. As outlined in (§4), if the changes in both matrices reach a small enough point, we could utilize this as a stopping criteria for convergence. However, in this case for testing convergence, our stopping criteria was when the total number of iterations reached 50.

Overall, this demonstrates some methods of implementing stopping criteria on the NMF algorithm as previously mentioned in (§4).

5.2 Drum Sound Separation

Not only can NMF separate tones on a single instrument, but the algorithm can also distinguish between instruments in an audio recording as many instruments play at

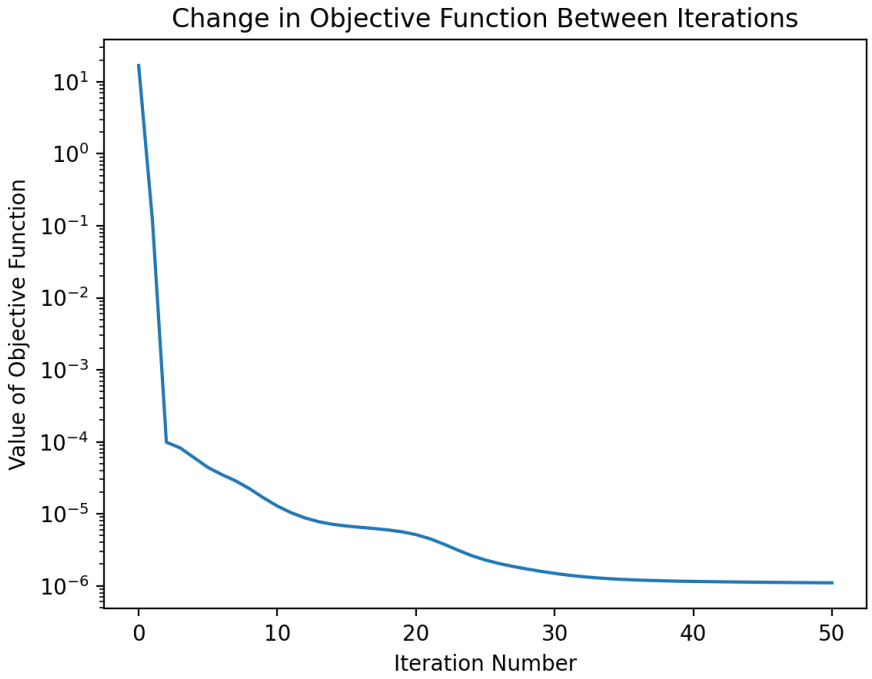

Figure 5.3: Convergence of the Objective Function Associated with NMF - The curve of this diagram measures how the objective function $f(\mathbf{W}, \mathbf{H})$ changes from iteration to iteration.

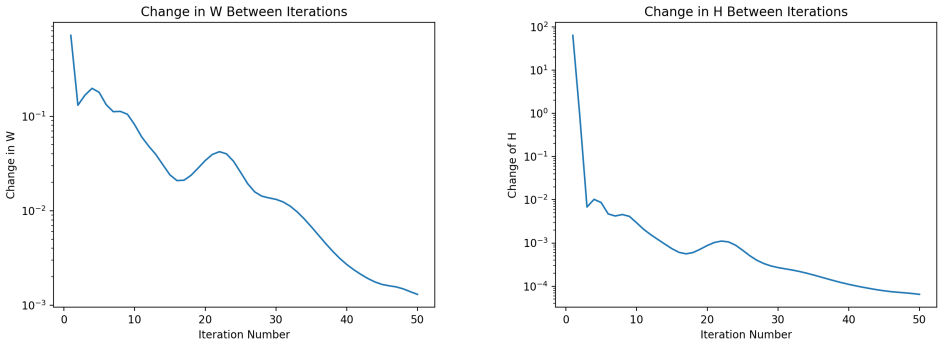

Figure 5.4: Measuring Change in H and W - The plot on the left depicts how much $\mathbf{W}$ changes between iterations while the plot on the right depicts how much $\mathbf{H}$ changes between iterations.

distinct frequency ranges. We want to show a more complicated example where the times at which sources occur overlap. For this example, an audio recording of 3 types of drums (a kick drum, a snare drum, and a ride cymbal) playing simultaneously was taken. Percussion instruments tend to emit an almost unique frequency combination allowing the NMF algorithm to perform source separation quite effectively. This is a slightly adapted version of a pre-existing example found in the Toolbox [20].

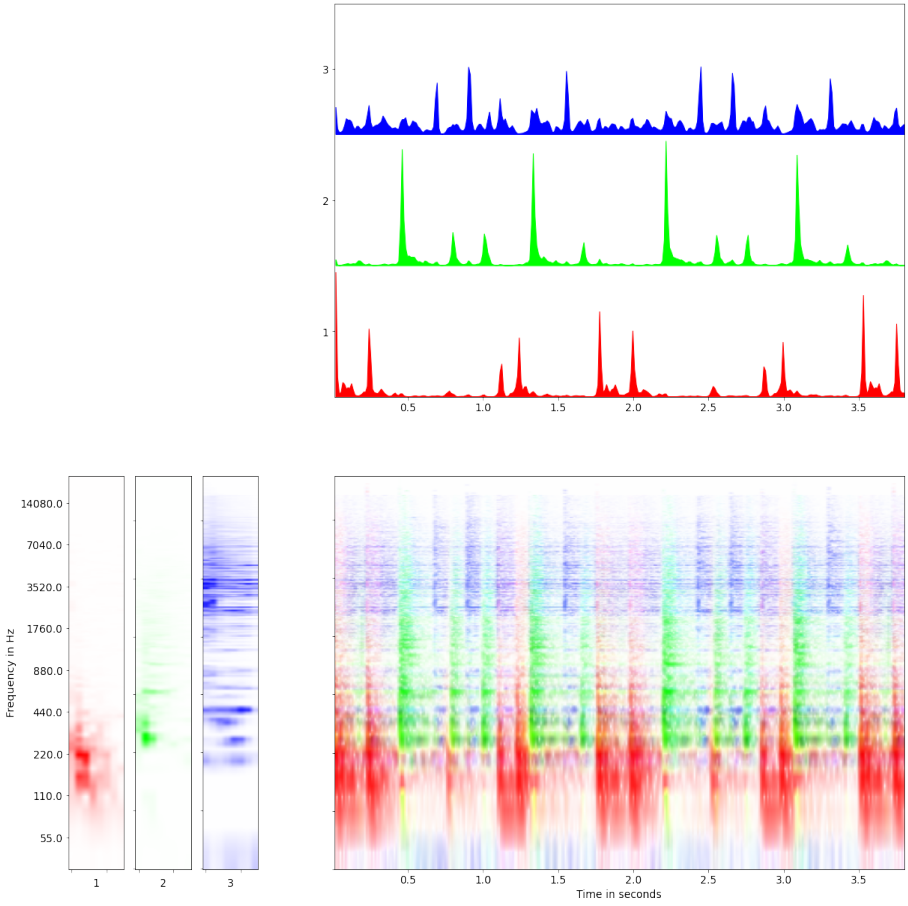

Figure 5.5: Visualizing NMF Through Drum Beats - The NMFD algorithm was utilized with input parameters of 3 sources for 3 distinct instruments, 30 iterations, 8 Template Frames to specify 8 Convolutions, randomly initialized **W**, and uniformly initialized **H**. The three colors represent each of the three percussion instruments: red represents the kick drum, green represents the snare drum, and blue represents the ride cymbal. For information on how to interpret this diagram, see Figure 3.1.

As one can see in Figure 5.5, our visualization is color-coded such that red represents the kick drum, green represents the snare drum, and blue represents the ride

cymbal. The same processes of setting up the spectrogram and running the NMFD algorithm as in the C scale example was utilized. However, one will note that the spectrogram is much more convoluted as we cannot discern an obvious pattern from the spectrogram like previously seen in the step-wise pattern of the C scale in Figure 5.2. This is most prominently due to the overlapping beats of the drums in the audio files. This makes it difficult to visualize the times and frequencies established and without the color scheme, it would be near impossible given simply the spectrogram.

However, The NMFD algorithm factors out **W** and **H** to reveal the separated sources as visualized in the bottom left corner and upper right corner of Figure 5.5 respectively allowing us to see the musical patterns of the audio more clearly. The spikes in the visualization in **H** represent the moments where one of our three percussion instruments is struck. Specifically, Source 1 corresponds with the kick drum, Source 2 corresponds with the snare drum, and Source 3 corresponds with the ride cymbal. The colored regions in **W** represent the range of frequencies that are attributed to each source throughout the audio file. An interesting aspect to note with this example is that Source 3, the ride cymbal, appears to have a much greater range of frequencies. This is due to the physical construction of the cymbal where the place where the cymbal is hit will vary with the frequency much for than the hit locations of the kick drum and the snare drum. Additionally, we can attribute this extended range to the ringing nature of a cymbal.

It should be noted that in **H**, we see that there is additional small levels of sound that are being picked up for each of the drums when they are not being struck. The majority of this additional sound recognition is due to the vibrations of the drums after the striking of one of the drums. These vibrations reverberate and so the audio file is able to capture this additional sound artifact. This is yet another complication that demonstrates why we need NMF to reveal details that we cannot see in a convoluted spectrogram of a given audio file.

Chapter 6

Regularized NMF

While NMF can separate sources well for audio files of higher quality (lack of feedback or noise, advanced recording captures, etc.), the NMF algorithm has a much harder time separating sources from a poor audio recording or a recording with background noise. A popular technique to perform source separation given noisy data is to implement regularization. This technique is often applied to circumvent overfitting to a data set and in terms of our noisy data problem, regularization aims to avoid overfitting the objective function to noise that is within our audio data.

In general, regularization is the act of adding a penalty expression to the cost function to promote desirable properties in the matrices over which we are optimizing, $\mathbf{W}$ and $\mathbf{H}$. We write our general regularized objective function as the following:

$$\min_{\mathbf{W},\mathbf{H}} \frac{1}{2}\|\mathbf{X} - \mathbf{WH}\|_F^2 + \beta R(\mathbf{W}) + \gamma S(\mathbf{H}) \tag{6.1}$$

where $\beta, \gamma \in \mathbb{R}$ are our regularization parameters corresponding to the respective regularization functions $R(\mathbf{W})$ and $S(\mathbf{H})$. In Equation (6.1), we focus on the standard practice of adding the regularization expression(s) to our given cost function in Equation (4.5). Some of the most popular types of regularization are Tikhonov

Regularization and L1 Regularization. We also want to determine our regularization parameter(s) β and/or γ which are constants that impact how much effect the regularization expression will have on our NMF model.

For example, if we want to add the following regularization expression to the cost function in Equation (4.5): $\frac{\beta}{2}\|\mathbf{W}\|_F^2$. Then, our objective function would take the form:

$$\min_{\mathbf{W},\mathbf{H}} \frac{1}{2}\|\mathbf{X}-\mathbf{W}\mathbf{H}\|_F^2 + \frac{\beta}{2}\|\mathbf{W}\|_F^2 \quad \text{s.t.} \quad \mathbf{W}\in\mathbb{R}_+^{m\times r}, \mathbf{H}\in\mathbb{R}_+^{r\times n}, \beta>0 \tag{6.2}$$

This regularization term aims to make $\mathbf{W}$ more favorable for source separation by punishing large entries in $\mathbf{W}$. This means that after applying the regularization term, $\mathbf{W}$ will have more smaller values.

The regularized objective function corresponds to different optimality conditions. We derive the optimality conditions and corresponding update rules for Equation (6.2) in Appendix A. The multiplicative update rules corresponding to this regularized objective function are:

$$\mathbf{W}^{(t+1)} = \mathbf{W}^{(t)} \odot \frac{\mathbf{X}\mathbf{H}^{(t)\top}}{\mathbf{W}^{(t)}\mathbf{H}^{(t)}\mathbf{H}^{(t)\top} + \beta\mathbf{W}^{(t)}} \tag{6.3}$$

$$\mathbf{H}^{(t+1)} = \mathbf{H}^{(t)} \odot \frac{\mathbf{W}^{(t+1)\top}\mathbf{X}}{\mathbf{W}^{(t+1)\top}\mathbf{W}^{(t+1)}\mathbf{H}^{(t)}} \tag{6.4}$$

And the derivation process as presented in (§8) will hold true for similar regularization expressions. Below, we have listed some additional types of regularized NMF problems that were tested on the experiment audio data with their respectively derived update rules:

$$\min_{\mathbf{W},\mathbf{H}} \|\mathbf{X} - \mathbf{WH}\|_F^2 + \frac{\gamma}{2}\|\mathbf{H}\|_F^2 \quad \text{s. t.} \quad \mathbf{W} \in \mathbb{R}_+^{m\times r}, \mathbf{H} \in \mathbb{R}_+^{r\times n}, \gamma > 0 \tag{6.5}$$

with update rules

$$\mathbf{W}^{(t+1)} = \mathbf{W}^{(t)} \odot \frac{\mathbf{X}\mathbf{H}^{(t)\top}}{\mathbf{W}^{(t)}\mathbf{H}^{(t)}\mathbf{H}^{(t)\top}}$$
$$\mathbf{H}^{(t+1)} = \mathbf{H}^{(t)} \odot \frac{\mathbf{W}^{(t+1)\top}\mathbf{X}}{\mathbf{W}^{(t+1)\top}\mathbf{W}^{(t+1)}\mathbf{H}^{(t)} + \gamma\mathbf{H}^{(t)}}$$

$$\min_{\mathbf{W},\mathbf{H}} \|\mathbf{X} - \mathbf{WH}\|_F^2 + \beta\|\mathbf{W}\|_{\text{sum}} \quad \text{s. t.} \quad \mathbf{W} \in \mathbb{R}_+^{m\times r}, \mathbf{H} \in \mathbb{R}_+^{r\times n}, \beta > 0 \tag{6.6}$$

with update rules

$$\mathbf{W}^{(t+1)} = \mathbf{W}^{(t)} \odot \frac{\mathbf{X}\mathbf{H}^{(t)\top}}{\mathbf{W}^{(t)}\mathbf{H}^{(t)}\mathbf{H}^{(t)\top} + \beta\mathbf{1}_W}$$
$$\mathbf{H}^{(t+1)} = \mathbf{H}^{(t)} \odot \frac{\mathbf{W}^{(t+1)\top}\mathbf{X}}{\mathbf{W}^{(t+1)\top}\mathbf{W}^{(t+1)}\mathbf{H}^{(t)}}$$

where $\|\mathbf{W}\|_{\text{sum}} = \sum_{i,j} |w_{ij}|$ and $\mathbf{1}_W \in \mathbb{R}^{m\times r}$ is a matrix of ones. This regularization expression encourages sparsity in $\mathbf{W}$ [23].

$$\min_{\mathbf{W},\mathbf{H}} \|\mathbf{X} - \mathbf{WH}\|_F^2 + \gamma\|\mathbf{H}\|_{\text{sum}} \quad \text{s. t.} \quad \mathbf{W} \in \mathbb{R}_+^{m\times r}, \mathbf{H} \in \mathbb{R}_+^{r\times n}, \gamma > 0 \tag{6.7}$$

with update rules

$$\mathbf{W}^{(t+1)} = \mathbf{W}^{(t)} \odot \frac{\mathbf{X}\mathbf{H}^{(t)\top}}{\mathbf{W}^{(t)}\mathbf{H}^{(t)}\mathbf{H}^{(t)\top}}$$
$$\mathbf{H}^{(t+1)} = \mathbf{H}^{(t)} \odot \frac{\mathbf{W}^{(t+1)\top}\mathbf{X}}{\mathbf{W}^{(t+1)\top}\mathbf{W}^{(t+1)}\mathbf{H}^{(t)} + \gamma\mathbf{1}_H}$$

where $\|\mathbf{H}\|_{\text{sum}} = \sum_{j,k} |h_{jk}|$ and $\mathbf{1}_H \in \mathbb{R}^{r \times n}$ is a matrix of ones. Finally, this regularization expression is similar to the previous expression as it encourages sparsity, but this time in $\mathbf{H}$.

There are several other regularization expressions that we could use. For example, a cancer data analysis paper [36] utilizes a manifold/graph regularization that aims to promote distance separation between more distinct data points. For more examples of regularization expressions, see the papers referenced at the end of (§2).

It is worth noting how we choose our constant regularization parameters β, γ. If β is too small, then there will be barely any difference between our regularized NMF and the non-regularized NMF algorithms making regularized NMF ineffective at source separating noisy data. If β is too large, then we are no longer trying to fit the data to our initial objective function but rather the regularization expression leading us to fit to the noise. Thus, when utilizing a regularization expression, we want to find a value for β that demonstrates effects from regularization but that does not over-regularize our problem. The same is true for γ.

Chapter 7

Experiments

All numerical experiments were run on a MacBook Pro 2018 model. Experiments were written in Python and run using PyCharm and Jupyter Notebooks. The NMF algorithms used originated from The NMF Toolbox written by Patricio López-Serrano, Christian Dittmar, Yiğitcan Özer, and Meinard Müller [20].

7.1 NMFD: Higher Complexity Audio Sample

This experiment aims to demonstrate how well NMF algorithms can handle source separation for higher complexity audio samples. The audio source used for this experiment came from [24], and it is a recording of Johann Sebastian Bach's Choral in BWV80 known as Ein feste Burg ist unser Gott in Equal Temperament. The score provided in Figure 7.1 that was taken from [25] shows the musical complexity of the piece.

The piece is much more complex than the examples from (§5) as it involves multiple melodies being played simultaneously through a Baroque music technique known as counterpoint. This means that we have multiple voices playing multiple notes (some different and some identical) in each of their melodic lines.

Figure 7.2 is a visualization of the starting spectrogram and the visualized output

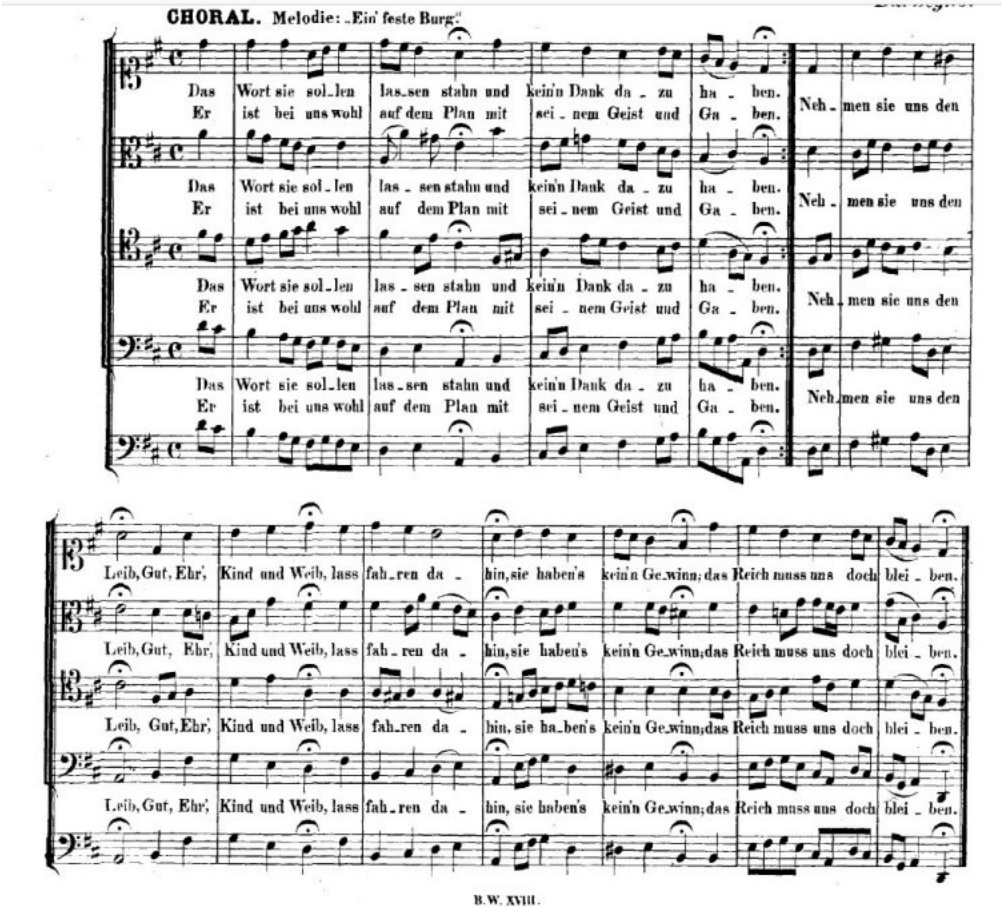

Figure 7.1: BWV80 Ein feste Burg ist unser Gott Score by Johannes Sebastian Bach - This is a polyphonic piece with 5 organ parts where three voices are in the alto clef range and two voices are in the bass clef range.

of the NMF model. Once again, the NMFD model from [20] was utilized. The algorithm was set to run for 200 iterations and the rank $r = 8$ was set. This rank was chosen to track the roughly 8 notes, ranging over various octaves, that seem to appear in the piece. As the five melodic lines of the choral are all played on the same instrument, the organ, we cannot simply distinguish the melodic lines based on the instrument as the melodic lines overlap in frequencies and thus making it much harder for NMF to distinguish the five melodic lines. It is also worth noting that the run time of the algorithm was approximately 10 minutes due to a high number of iterations but also due to the piece being 49 seconds in length as opposed to the previous examples being under 15 seconds.

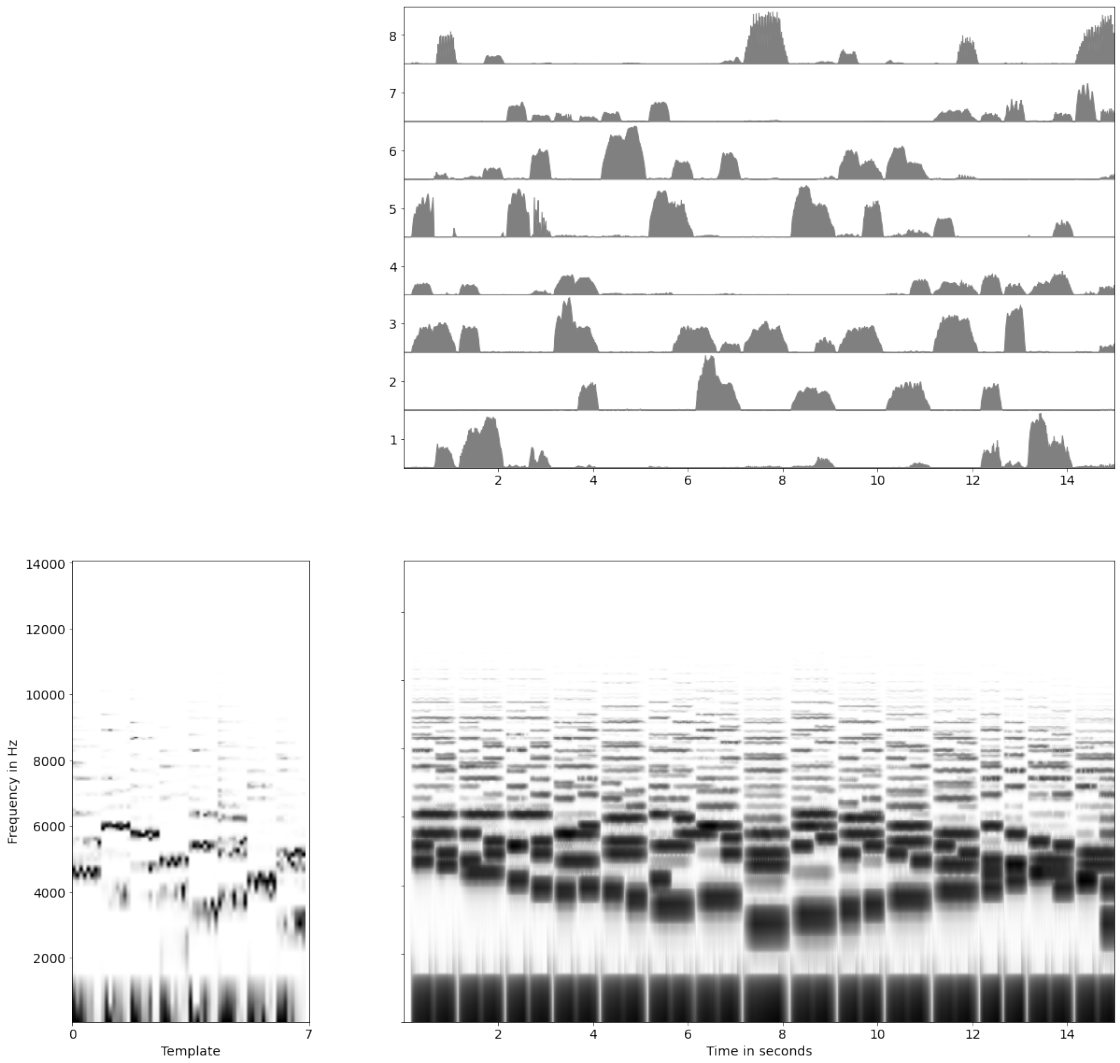

Figure 7.2: NMF Source Separation of Bach Choral BWV80 in Equal Temperament - The NMFD algorithm was utilized with input parameters of 8 sources for 8 distinct pitches, 200 iterations, 8 Template Frames to specify 8 Convolutions, randomly initialized **W**, and uniformly initialized **H**. For information on how to interpret this diagram, see Figure 3.1.

As one can see in Figure 7.2, the spectrogram appears much more convoluted and unable to pick out an easy pattern as we could in the scale example. However, similar to the drum example in (§5.2), after running the NMFD algorithm, we have some apparent patterns. Listening to the deconstructed recording that NMF produced, one can hear that each audio source contains primarily a single note that is played throughout the piece.

In, the visualization of **W**. We can see that compared with the scale example in (§5.1), each source appears to have a much greater range of frequencies within the frequency combinations. This is due to the audio recording not being bound within an octave as it previously was in the scale example. But as we see the ladder-like structure of frequencies in each source, it seems to confirm that often the two notes that are the same note in distinct octave ranges were categorized under the same source. That is, for example, a C note and an octave higher C note are both being categorized under the same source for all C notes and the same appears primarily true for most of the notes of the scale. Musically, this makes sense as notes of the same letter all share at least some of the volume at a certain frequency that is distinct from notes of a different letter.

7.2 NMFD: Symphony with Real Noise

This experiment aims to demonstrate how NMFD can separate out real captured "noise" from an audio sample. The audio sample once again comes from [24]. The sample is a four second snippet from a symphonic arrangement involving a full orchestra with an individual coughing twice over the instruments.

As seen in Figure 7.3, NMFD was able to separate the coughing noise from the orchestra by classifying the two distinct sources into different bins. Thus, NMF on its own is able to separate real-life noises that may impede audio recordings allowing

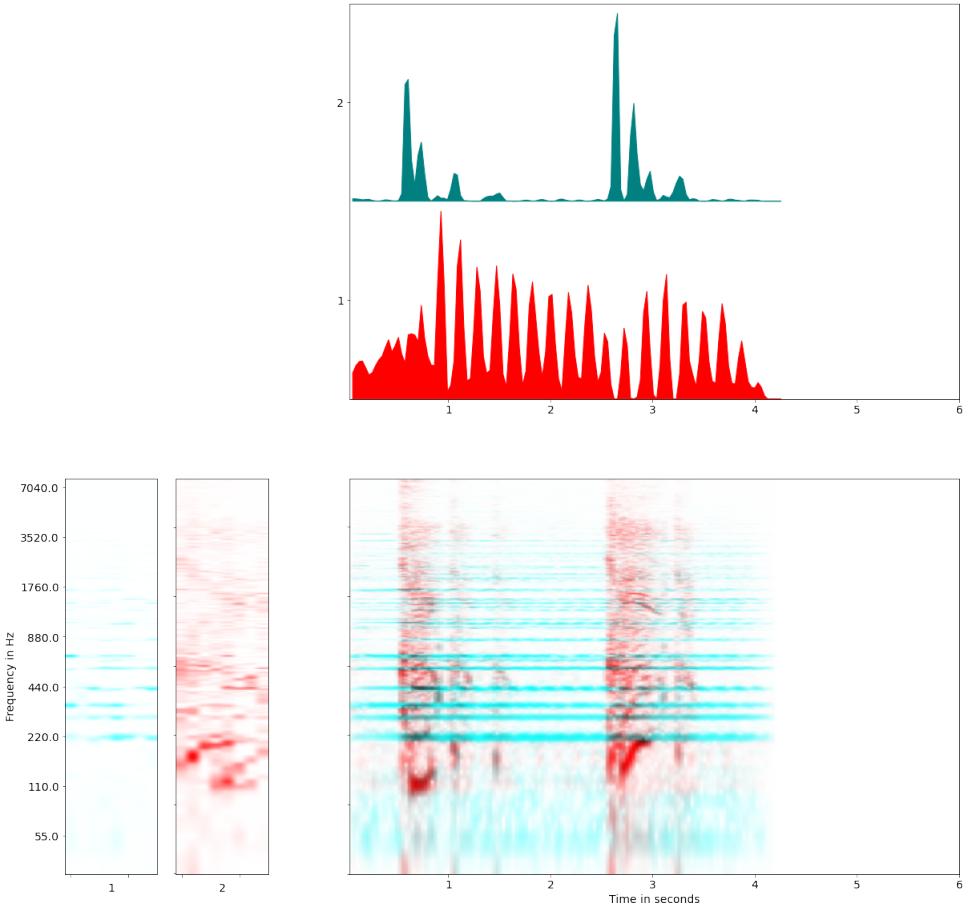

Figure 7.3: Visualizing NMF Separating Out Noise - The NMF algorithm with no regularization parameter was utilized with input parameters of 2 sources for the music and the noise, 200 iterations, randomly initialized **W**, and uniformly initialized **H**. Blue corresponds with the coughing noise and red corresponds with the orchestra.

individuals to obtain a separate recording nearly or fully free of noise as long as the noise does not match the frequency of the music. NMF was able to separate the cough without regularization as the "noise" is of a very distinct frequency compared to the orchestra and thus can be treated as if it were a separate source.

Additionally, it is once again worth noting the frequency range that is captured by the matrix **W**. We see that the frequency range of the coughing noise has very distinct frequencies within a smaller range compared to the sound produced by the orchestra which is not as distinct and varies to a much greater magnitude. This difference is due to the sources themselves. The cough is by a singular person who's

cough is at some position in their vocal range whereas the orchestra is full of a variety of instruments each playing notes with different emitted frequencies.

7.3 Regularized NMF: C Scale with Induced Noise

This experiment aims to demonstrate the practical use adding regularization to the NMF algorithm. In this experiment, we manually induced noise by perturbing the initial matrix **X**. For the purpose of this example, we wrote the following code:

```
n_lev = 0.001
A = A + n_lev * np.linalg.norm(A) * abs(np.random.randn(*A.shape))
```

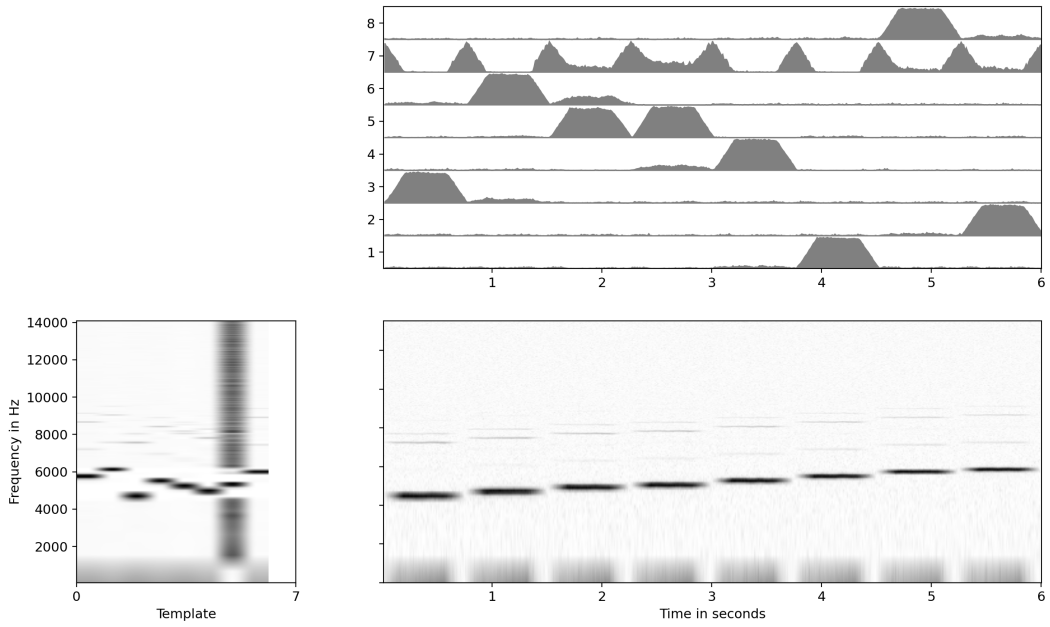

Figure 7.4: Visualizing NMF Through C Scale with Induced Noise - The NMF algorithm with no regularization parameter was utilized with input parameters of 8 sources for 8 distinct pitches, 50 iterations, randomly initialized **W**, and uniformly initialized **H**.

As we can see in Figure 7.4, the spectrogram in the bottom right corner now has some additional black dots throughout the visualization. This indicates the added

noise. We see that the non-regularized NMF algorithm struggles to separate all of the sources as we previously did in the illustrative examples (§5). While the algorithm was able to separate several of the sources, the algorithm was unable to pick out all 8 notes of the scale individually. We see in the visualization of $\mathbf{H}$ that the seventh source, the second from the top row, from the figure is registering several spikes in sound and that two of the notes are being combined to make up the fifth source. Based on the times of the spikes in the seventh source of the $\mathbf{H}$ visualization and our knowledge that there are breaks in sound between each note in the audio recording, we can infer that the seventh source is picking up induced noise that was added to any point where there is silence.

Additionally, it is worth noting that the two notes that are being combined on the scale are the E and the F of the scale. From a music perspective, the E and the F are the two closest notes of the scale in terms of frequency. This is due to the two notes being only a half step apart. Thus, it makes sense that when the silence induced with noise is distinct enough from the other tones to take up one of the source bins, NMF would try to combine the two most similar notes of the scale.

To overcome these challenges, we find a regularization expression that source separates the notes more effectively. The regularized NMF algorithm using the objective function and updates rules in Equation (6.7) was selected as the most favorable regularized NMF algorithm. The added regularization expression $\gamma\|\mathbf{H}\|_1$ was selected to promote sparsity in $\mathbf{H}$ so therefore, small amounts of noise added during very low volume levels should tend towards zero in the objective function. Meanwhile our regularized NMF should distinguish between the pitches that are more prominent in the audio by leaving their volume level further from zero.

In Figure 7.5, we have implemented and run the regularized NMF algorithm as presented in Equation (6.7) on the C Scale example with the same amount of added noise as in Figure 7.4. The regularization parameter was set as $\gamma = 5 \times 10^{-6}$. Now,

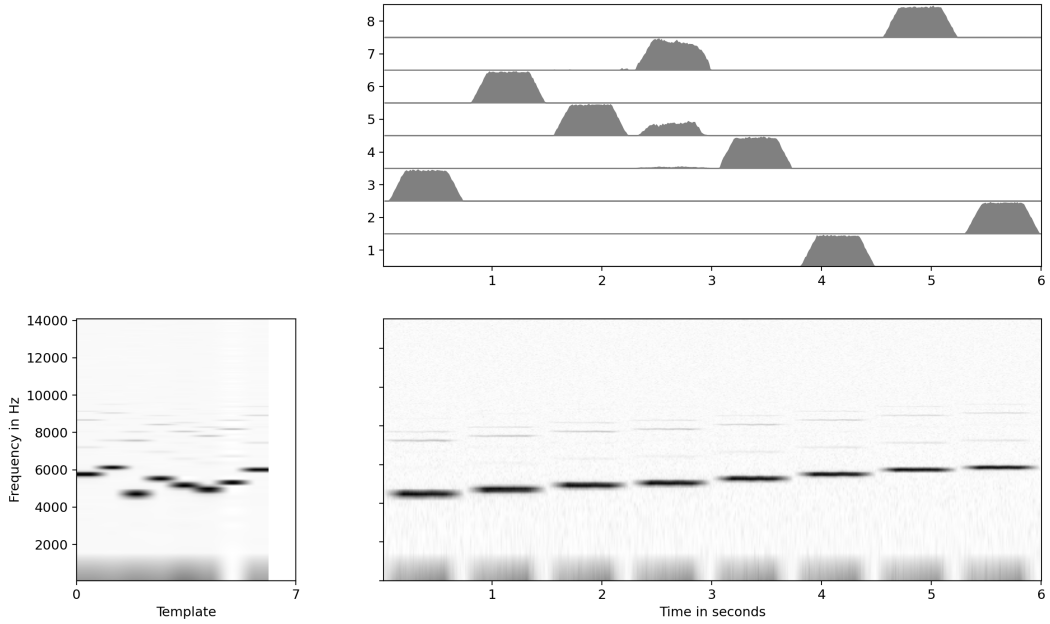

Figure 7.5: Visualizing Regularized NMF Through C Scale with Induced Noise - The Regularized NMF algorithm with regularization expression $\gamma\|\mathbf{H}\|_1$ was utilized with input parameters of a regularization parameter of $\gamma = 5\times10^{-6}$, 8 sources for 8 distinct pitches, 50 iterations, randomly initialized $\mathbf{W}$, and uniformly initialized $\mathbf{H}$.

the visualization of $\mathbf{H}$ reflects nearly all eight notes of the scale distinctly. We see that the seventh source is now one block in time rather than reflecting the added noise as it previously did. However, it is not perfect as the fifth source still has some of the sound of the F combined with the E (but most of the F is now classified in the seventh source). Yet, this source separation is much better and has essentially been able to avoid overfitting to the noise and still performing an excellent source separation.

It is worth noting why this noisy scale example required regularized NMF while the noisy symphony example (§7.2) did not. In the noisy symphony example, we are trying to separate a human cough, our noise, from the symphony. Thus, the noise is being treated as a separate source. Meanwhile, in this noisy scale example, we want to source separate out the 8 notes of the scale without treating the noise as an

extraneous source. We cannot simply treat the induced noise as another source since the noise has not only altered silence in the audio but also the notes themselves.

Chapter 8

Conclusions and Future Directions

In this paper, we have shown the applicability of Nonnegative Matrix Factorization to audio data, specifically music. We described NMF (§1) and derived the NMF algorithm utilizing KKT conditions (§4). We showed that the basic NMF algorithm works well on some simple audio recordings (§5) as well as a longer and more complex audio recordings (§7.1). We were able to utilize the NMF application of source separation to distinguish both notes (§5.1) and instruments (§5.2). NMF was able to reveal musical patterns that one could not simply make from looking at a cluttered spectrogram that occurs when music moves from monotonic to polyphonic.

After adding noise to simulate a poor-quality audio recording with background sound, we saw that the basic NMF struggles to separate all sources. We derived and implemented a regularized NMF algorithm once again using the KKT conditions (§6). After implementing the regularized NMF, we saw that the algorithm was able to avoid overfitting to added background noise in the perturbed audio data and produce a cleaner source separation(§7.3).

NMF can prove to be an extremely powerful tool for musicians with a lack of resources. If a musician wants to play a piece but only has an audio file without the sheet music, NMF can help the musician piece together the sheet music. As long as

they have knowledge of what notes correspond to certain frequency combinations and access to an NMF algorithm, one can run NMF or regularized NMF for noisy data and they would get **W** where they can identify which source corresponds to which note and then use **H** to piece together the rhythm of the notes leading them to have nearly all the information they need to transcribe the piece onto sheet music. This is valuable for if one loses their written score or if one does not have perfect pitch and cannot identify notes by ear [4].

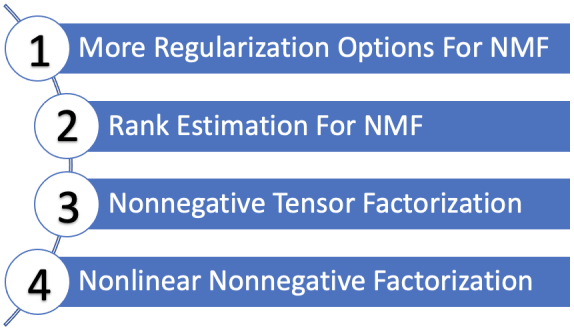

Figure 8.1: Future NMF Research Avenues

There are several future directions that this project could take on after this thesis. The most obvious option is to explore other types of regularized NMF algorithms. For example, one could look into a finite-difference based approach where we apply a matrix **B** formed from discretizing finite difference approximations to either **W** or **H** within a chosen norm creating a regularization expression such as $\beta\|\mathbf{HB}\|_1$. Another regularization approach that we are currently working towards is an NMF algorithm with sparse graph/manifold regularization as discussed in [36]. This regularization would take the discretized audio data and promote both sparsity and distance separation. We have begun the implementation of this regularization within our code and hope to continue working on this future direction in the coming months.

Another interesting direction would be to explore automatic hyperparameter selection such as a rank predictor for NMF. Currently, the NMF and regularized NMF algorithms require one to know the rank (or how many pitches for each distinct in-

strument) in order to produce a favorable source separation output. However, if NMF could predict the rank, one would not need to have any knowledge of the number of pitches and instruments present in the audio recording. One idea is to use cross-validation to identify a potential predicted rank through machine learning training and validate the predicted rank through machine learning validation [18].

Additionally, there are two direct extensions of this project to higher dimensionality. Since NMF works well for matrices, we could extend the project to create a Nonnegative Tensor Factorization (NTF) and look into regularizing the algorithm. NTF would allow us to extract music features from more than one audio file at a time and has previously been used in genre classification [5]. Also, our matrix systems are all linear. So it would be interesting to look into Nonlinear Nonnegative Factorization [2]. These two approaches could lead to even better results for complicated audio files that are on a much larger scale such as performing NMF on a large audio dataset, a symphony recording, or on a pop-song mashup.

For anyone interested in our edited NMF Toolbox code, please see the following GitHub repository: Regularized NMF Toolbox.

Appendix A

Example Derivation of Regularized NMF

The optimization problem (6.2) corresponds to a Euclidean Update rule as well. We rewrite the objective function $f2$ of (6.2):

$$f2(\mathbf{W},\mathbf{H},\beta) = |\mathbf{X}-\mathbf{WH}\|_F^2 + \frac{\beta}{2}\|\mathbf{W}\|_F^2 = \frac{1}{2}\sum_{i=1}^{m}\sum_{k=1}^{n}(x_{ik}-\sum_{j=1}^{r}w_{ij}h_{jk})^2 + \frac{\beta}{2}\sum_{i=1}^{m}\sum_{k=1}^{r}w_{ij}^2 \tag{A.1}$$

In order to write the KKT conditions, we form the Lagrangian:

$$L(\mathbf{W},\mathbf{H},\beta,\mathbf{\Lambda},\mathbf{M}) = |\mathbf{X}-\mathbf{WH}\|_F^2 + \frac{\beta}{2}\|\mathbf{W}\|_F^2 + \sum_{i=1}^{m}\sum_{j=1}^{r}\lambda_{ij}(-w_{ij}) + \sum_{j=1}^{r}\sum_{k=1}^{n}\mu_{jk}(-h_{jk}) \tag{A.2}$$

To establish the stationarity conditions, we need to calculate the partial derivatives of the Lagrangian in Equation (A.2) with respect to every possible entry point in $\mathbf{W}$ and $\mathbf{H}$. We can see that the stationarity conditions are where we see the biggest changes and this stems from the partial derivative calculations.

The stationarity conditions are:

$$\nabla_{\mathbf{W}} L = (\mathbf{WH} - \mathbf{X})\mathbf{H}^\top + \beta\mathbf{W} - \mathbf{\Lambda} \tag{A.3}$$

$$\nabla_{\mathbf{H}} L = (\mathbf{WH} - \mathbf{X})\mathbf{W}^\top - \mathbf{M} \tag{A.4}$$

The complementary slackness conditions are:

$$\mathbf{\Lambda} \odot \mathbf{W} = \mathbf{0}, \quad \mathbf{M} \odot \mathbf{H} = \mathbf{0} \tag{A.5}$$

The primal feasibility conditions are:

$$\mathbf{W} \in \mathbb{R}_+^{m \times r}, \quad \mathbf{H} \in \mathbb{R}_+^{r \times n} \tag{A.6}$$

The dual feasibility conditions are:

$$\mathbf{\Lambda} \in \mathbb{R}_+^{\mathbf{m} \times \mathbf{r}}, \quad \mathbf{M} \in \mathbb{R}_+^{\mathbf{r} \times \mathbf{n}} \tag{A.7}$$

Therefore, we can see that only the stationarity conditions were affected by adding our regularization term to the minimization problem and we are about to see this affect the multiplicative update rules as these are reliant upon the stationarity conditions. We can utilize the stationarity conditions in order to create the multiplicative update:

$$\nabla_{\mathbf{W}} L = (\mathbf{WH} - \mathbf{X})\mathbf{H}^\top + \beta\mathbf{W} - \mathbf{\Lambda} = \mathbf{0} \rightarrow (\mathbf{WHH}^\top - \mathbf{XH}^\top + \beta\mathbf{W}) = \mathbf{\Lambda}$$

We can now plug this into our complementary slackness condition:

$$\mathbf{\Lambda} \odot \mathbf{W} = \mathbf{0} \rightarrow \mathbf{W} \odot (\mathbf{WHH}^\top - \mathbf{XH}^\top + \beta\mathbf{W}) = \mathbf{0} \rightarrow \mathbf{W} \odot (\mathbf{WHH}^\top + \beta\mathbf{W}) = \mathbf{W} \odot (\mathbf{XH}^\top)$$

Using point-wise division:

$$\mathbf{W} = \mathbf{W} \odot \frac{\mathbf{X}\mathbf{H}^\top}{(\mathbf{W}\mathbf{H}\mathbf{H}^\top + \beta\mathbf{W})}$$

Note that this particular regularization expression simply changed the denominator of our update rule by taking into account the size of the w_{ij} found in $\mathbf{W}$. Like we previously did in the original derivation of the update rules, we can do a similar process to find the update rule for each iteration of $\mathbf{H}$. However, as our regularization expression does not involve $\mathbf{H}$, we saw that $\nabla_{\mathbf{H}} L$ remained unchanged and so the update rule for $\mathbf{H}$ is the same as the non-regularized NMF algorithm.

Thus, the following overall multiplicative update rule is found for a regularization expression involving taking the Frobenius norm of $\mathbf{W}$:

$$\mathbf{W}^{(t+1)} = \mathbf{W}^{(t)} \odot \frac{\mathbf{X}\mathbf{H}^{(t)\top}}{\mathbf{W}^{(t)}\mathbf{H}^{(t)}\mathbf{H}^{(t)\top} + \beta\mathbf{W}^{(t)}} \tag{A.8}$$

$$\mathbf{H}^{(t+1)} = \mathbf{H}^{(t)} \odot \frac{\mathbf{W}^{(t+1)\top}\mathbf{X}}{\mathbf{W}^{(t+1)\top}\mathbf{W}^{(t+1)}\mathbf{H}^{(t)}} \tag{A.9}$$

Bibliography

[1] S. Abdali and B. NaserSharif. Non-negative matrix factorization for speech/music separation using source dependent decomposition rank, temporal continuity term and filtering. *Biomedical Signal Processing and Control*, 36:168–175, 2017.

[2] Babajide O. Ayinde and Jacek M. Zurada. Deep learning of constrained autoencoders for enhanced understanding of data. *IEEE Transactions on Neural Networks and Learning Systems*, 29(9):3969–3979, 2018.

[3] Amir Beck. *Introduction to Nonlinear Optimization - Theory, Algorithms, and Applications with MATLAB*, volume 19 of *MOS-SIAM Series on Optimization*. SIAM, 2014.

[4] Emmanouil Benetos, Simon Dixon, Zhiyao Duan, and Sebastian Ewert. Automatic music transcription: An overview. *IEEE Signal Processing Magazine*, 36:20–30, 01 2019.

[5] Emmanouil Benetos and Constantine Kotropoulos. Non-negative tensor factorization applied to music genre classification. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(8):1955–1967, 2010.

[6] Olivier Berné, A Helens, Paolo Pilleri, and Christine Joblin. Non-negative matrix factorization pansharpening of hyperspectral data: An application to mid-infrared astronomy. pages 1–4, 06 2010.

[7] Andrzej Cichocki, Rafal Zdunek, Anh Huy Phan, and Shun-Ichi Amari. *Nonnegative Matrix and Tensor Factorizations: Applications to Explotatory Multi-Way Data Analysis and Blind Source Separation*. Wiley, 2009.

[8] Ketan Doshi. Audio deep learning made simple(part 1): State-of-the-art techniques.

[9] Flavia Esposito, Angelina Boccarelli, and Del Nicoletta. An nmf-based methodology for selecting biomarkers in the landscape of genes of heterogeneous cancer-associated fibroblast populations. *Bioinformatics and Biology Insights*, 14:117793222090682, 05 2020.

[10] Xiao Fu, Kejun Huang, Nicholas D. Sidiropoulos, and Wing-Kin Ma. Nonnegative matrix factorization for signal and data analytics: Identifiability, algorithms, and applications. *IEEE Signal Processing Magazine*, 36(2):59–80, Mar 2019.

[11] Nicolas Gillis. The why and how of nonnegative matrix factorization, 2014.

[12] Nicolas Gillis. *Nonnegative Matrix Factorization*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2020.

[13] William Karush. Minima of functions of several variables with inequalities as side constraints. 1939.

[14] Harold W. Kuhn and Albert W. Tucker. Nonlinear programming proceedings of the 2nd berkeley symposium on mathematical statistics and probability. *University of California Press*, pages 481–492, 1951.

[15] Clement Laroche, † Papadopoulos, Matthieu Kowalski, and Gaël Richard. Drum extraction in single channel audio signals using multi-layer non negative matrix factor deconvolution. 03 2017.

[16] Daniel Lee and Hyunjune Seung. Algorithms for non-negative matrix factorization. *Adv. Neural Inform. Process. Syst.*, 13, 02 2001.

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

[18] Seokjin Lee. Estimating the rank of a nonnegative matrix factorization model for automatic music transcription based on stein's unbiased risk estimator. *Applied Sciences*, 10:2911, 04 2020.

[19] Ping Liu, Xia Zhou, Yanling Li, Minqiang Li, Daoyang Yu, and Jinhuai Liu. The application of principal component analysis and non-negative matrix factorization to analyze time-resolved optical waveguide absorption spectroscopy data. *Anal. Methods*, 5:4454–4459, 2013.

[20] Patricio López-Serrano, Christian Dittmar, Yiğitcan Özer, and Meinard Müller. Nmf toolbox, 2019.

[21] Patricio López-Serrano, Christian Dittmar, Yiğitcan Özer, and Meinard Müller. Nmf toolbox: Music processing applications of nonnegative matrix factorization. In *Proceedings of the International Conference on Digital Audio Effects (DAFx)*, Birmingham, UK, September 2019.

[22] Meinard Müller. *Fundamentals of Music Processing: Audio, Analysis, Algorithms, Applications*. Springer Publishing Company, Incorporated, 1st edition, 2015.

[23] Andrew Y. Ng. Feature selection, l1 vs. l2 regularization, and rotational invariance. In *Proceedings of the Twenty-First International Conference on Machine Learning*, ICML '04, page 78, New York, NY, USA, 2004. Association for Computing Machinery.

[24] Stanford University Department of Music. Sound examples.

[25] Aryeh Oron. Chorale melodies used in bach's vocal works ein feste burg ist unser gott, 2018.

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

[27] Jeongsoo Park, Jaeyoung Shin, and Kyogu Lee. Separation of instrument sounds using non-negative matrix factorization with spectral envelope constraints, 2018.

[28] Jérémy Rapin, Antoine Souloumiac, Jérôme Bobin, Anthony Larue, Chistophe Junot, Minale Ouethrani, and Jean-Luc Starck. Application of Non-negative Matrix Factorization to LC/MS data. *Signal Processing: Image Communication*, 123:75–83, June 2016.

[29] P. Smaragdis and J.C. Brown. Non-negative matrix factorization for polyphonic music transcription. In *2003 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (IEEE Cat. No.03TH8684)*, pages 177–180, 2003.

[30] Paris Smaragdis. Convolutive speech bases and their application to supervised speech separation. *IEEE Trans. Speech Audio Process.*, 15(1):1–12, 2007.

[31] Paris Smaragdis and J. Brown. Non-negative matrix factor deconvolution; extraction of multiple sound sources from monophonic inputs. volume 3195, 09 2004.

[32] Leo Taslaman and Björn Nilsson. A framework for regularized non-negative matrix factorization, with application to the analysis of gene expression data. *PloS one*, 7:e46331, 11 2012.

[33] Emmanuel Vincent, Tuomas Virtanen, and Sharon Gannot. *Audio source separation and speech enhancement*. John Wiley & Sons, 1st edition, 2018.

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

[35] Richard M. Wallace. Analysis of absorption spectra of multicomponent systems1. *The Journal of Physical Chemistry*, 64(7):899–901, 1960.

[36] Chuan-Yuan Wang, Jin-Xing Liu, Na Yu, and Chun-Hou Zheng. Sparse graph regularization non-negative matrix factorization based on huber loss model for cancer data analysis. *Frontiers in Genetics*, 10, 2019.