Python Implementation of Single Channel Mixed Speaker Separation using Nonnegative Matrix Factorization

Severin Ibarluzea and Joseph Lee

***Abstract*— A method for single channel mixed speaker separation using non-negative matrix factorization is analyzed and implemented. Short time fourier transforms are used to represent sounds as overlapping spectrograms. The method first initializes a set of basis matrices for each voice with training data using Nonnegative Matrix Factorization (NMF) with randomly initialized weight and basis matrices. A training conversation is then represented as a linear combination of the trained voices bases. This linear combination and trained bases can then be used to separate testing conversations. The optimal parameters for basis size and spectrogram length are experimentally determined.**

I. INTRODUCTION

An environment of mixed speakers can cause a single speaker to be difficult to hear for an observer. Isolating a single speaker in an active audio filtering system may then be desirable for human listeners.

Single channel conversation[1] separation is a difficult problem as only one input channel is provided. Similar problems, such as the cocktail party problem, rely on multiple observations from the same set of input sources, may be solved using techniques such as independent component analysis. However, single channel conversation separation must use a single observer. Therefore, a supervised voice detection algorithm using Non-negative Matrix Factorization (NMF) is used to train a model with the independent voices. The model is then used to extract the voices from a new mixed single channel conversation.

II. OBJECTIVE

A. Overview

The objective of the project is to train a Non-negative Matrix Factorization model with two independent voices and extract the two voices from a new input with the two voices added together. The results can be evaluated using Signal Distortion Ratio (SDR) [3].

$$SDR_g = 20log_{10}(\frac{||\lambda_g s_g||_2}{||\hat{s}_g - \lambda_g s_g||_2}) \tag{1}$$

The SDR measures the similarity of the shape of the extracted signal to the original signal if the extracted signal is allowed to be multiplied by a constant factor. Thus, the SDR will not change as a signal is scaled. This measurement matches more closely with qualitative interpretation of signals where the volume between the extracted signal and original signal does not change.

B. Related Work

In this application, NMF is used to decompose the training voices into a set of basis vectors which represent the learn voices. The basis vectors are used to extract the voices from mixed conversation signals. Other methods include an unsupervised NMF approach which extracts the input sources from a mixed signal without training data or prior information [4]. Another technique involves using NMF in a supervised setting and applying a wiener filter when reconstructing the signal which acts as a mask and scales the magnitudes of the mixed signal at its frequency components[1]. The technique used for Non-negative matrix factorization can be written as solving the following minimization problem:

$$min_{B,W} C(V, BW) \tag{2}$$

subject to the the elements of B and W being non-negative. Different NMF arise from differing cost functions. The first cost function is the L2 norm between V and BW, which produces the follow equation:

$$min_{B,W} D(||V - BW||_2^2) \tag{3}$$

where

$$D(||V - BW||_2^2) = \sum_{i,j}(V_{i,j} - (BW)_{i,j})^2 \tag{4}$$

The next cost function is the divergence of V from BW, which produces the follow equation:

$$min_{B,W} D(V||BW) \tag{5}$$

where

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

The second cost function was found to work well in audio source separation[1]. NMF using this cost function can be solved by alternating updates of **B** and **W** as follows:

$$B \leftarrow B \otimes \frac{\frac{V}{BW} W^T}{1 W^T}$$

$$W \leftarrow W \otimes \frac{B^T \frac{V}{BW}}{B^T 1}$$

*This work was done as the final project of ECSE 4530 at Rensselaer Polytechnic Institute

[1]A human conversation typically only has one speaker at any given time, when *conversation* is used in this paper it refers to humans simultaneous speaking

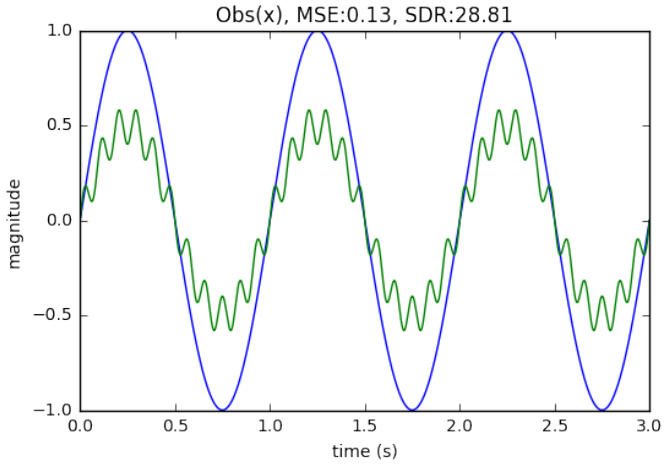
III. Data Collection

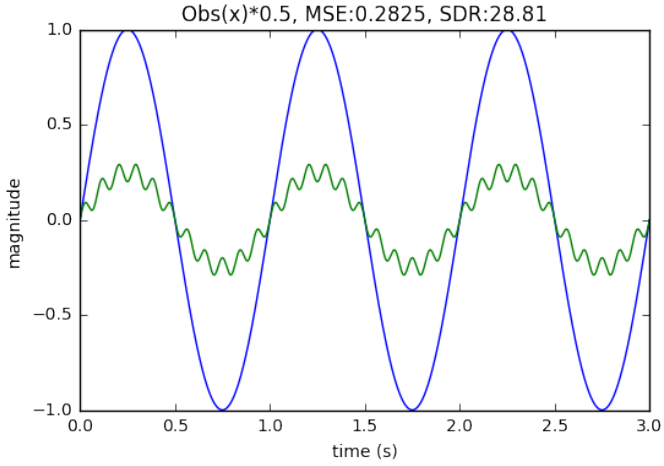
Data was collected from Carnegie Mellon University's *Robust Speech Recognition Group Census Database* [5], also know as AN4 and the Alphanumeric database. It includes various recordings of subjects who were asked to spell out personal information, and speak randomly generated sequences of words contained control words. All the recordings were sampled at 16 kHz.

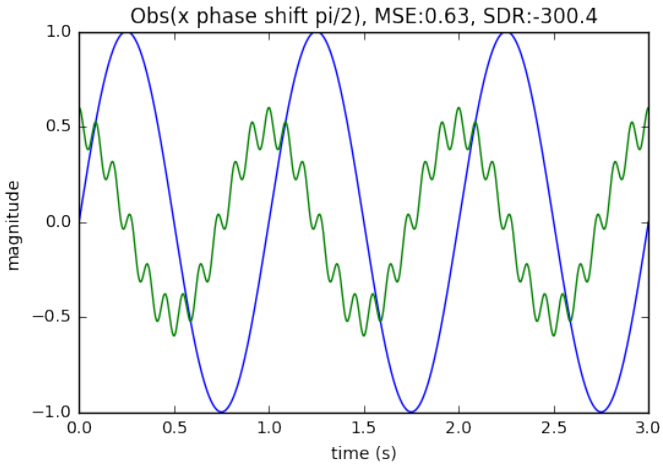
IV. Technical Approach

A. Signal Distortion Ratio vs Mean Squared Error

The difference between SDR and mean squared error (MSE) is shown in the figures below where `Obs(x)` is the noisy observation of a signal x. The differences in MSE implies a MSE approach fails to recognize the smaller original signal with noise as being similar to the original signal. The SDR for the signal with a smaller amplitude stays the same. When a dramatic phase shift occurs, both the MSE and SDR assign values implying a large error in the reconstructed signal.







The SDR better reflects a qualitative analysis where volume is ignored.

B. Supervised Learning Approach

A supervised learning approach was taken to learn the basis vectors of each voice in the time-frequency domain. The voices data was imported as a set of single dimensional arrays. A voice sample is converted into a magnitude spectorgram by taking discrete fourier transforms of the input signal. A discrete fourier transform (DFT) is performed on a framelength of points on the input signal and then shifted an offset of half the framelength. This step is repeat on the entire input signal and each DFT appended to the magnitude spectrogram matrix **V** as a column vector. The tuning parameters for this approach are the framelength and number of basis vectors; the optimal parameters are estimated experimentally.

The matrix **V** is broken down into the matrix **B** and **W** through non-negative matrix factorization. The columns of matrix **B** represent the basis vectors, and the weights of the associated basis vectors appear in the columns of **W**.

$$V \approx BW \tag{7}$$

Given the input spectrogram matrix V_{N_z,N_s} and the desired number of basis, N_b, the matrices **B** and **W** can be set to a random matrices of size N_z x N_b and N_b x N_s, respectively.

For each voice, the spectrogram matrix **V** can be used to determine the the basis and weight matrices using non-negative matrix factorization.

For the purposes of this project, scikit-learn's non-negative matrix factorization function was used. The objective function in this implementation is:

$$\begin{aligned} &0.5 * ||X - WH||_F ro^2 \\ &\quad + \alpha * l1_{ratio} * ||vec(W)||_1 \\ &\quad + \alpha * l1_{ratio} * ||vec(H)||_1 \\ &\quad + 0.5 * \alpha * (1 - l1_{ratio}) * ||W||_F ro^2 \\ &\quad + 0.5 * \alpha * (1 - l1_{ratio}) * ||H||_F ro^2 \end{aligned}$$

where:

$$||A||_Fro^2 = \sum_{i,j} A_{ij}^2 (Frobeniusnorm)$$
$$||vec(A)||_1 = \sum_{i,j} abs(A_{ij}) (ElementwiseL1norm)$$

The two voice spectrogram matrices can be decomposed using this function to approximate the basis matrices and weight matrices.

$$V_{voice1} \approx B_{voice1} W_{voice1}$$
$$V_{voice2} \approx B_{voice2} W_{voice2}$$

Once the basis of the two voices have been learned, the matrices can be used to extract the individual voices from conversations between the two voices.

Given a mixed signal of the two previous voices, new weights are calculated for the previous basis matrices. Let $V_{voice1,2}$ be a new sample created as a sum of different samples of the first two voices. Solving the following equation for $W_{voice1,2}$ will produces the weights corresponding to the basis vectors of each voice.

$$V_{voice1,2} \approx [B_{voice1} B_{voice2}] W_{voice1,2}$$

The mixed weights matrix can be broken down into the weights for the basis vectors of the two voices.

$$\begin{bmatrix} \hat{W}_{voice1} \\ \hat{W}_{voice2} \end{bmatrix} = W_{voice1,2}$$

The weights matrix are needed to create the voice from the calculated basis matrices. The matrices for the reconstructed voice spectogram matrix were calculated using a least squares solution to the linear matrix equation.

$$\hat{V}_{voice1} \approx B_{voice1} \hat{W}_{voice1}$$
$$\hat{V}_{voice2} \approx B_{voice2} \hat{W}_{voice2}$$

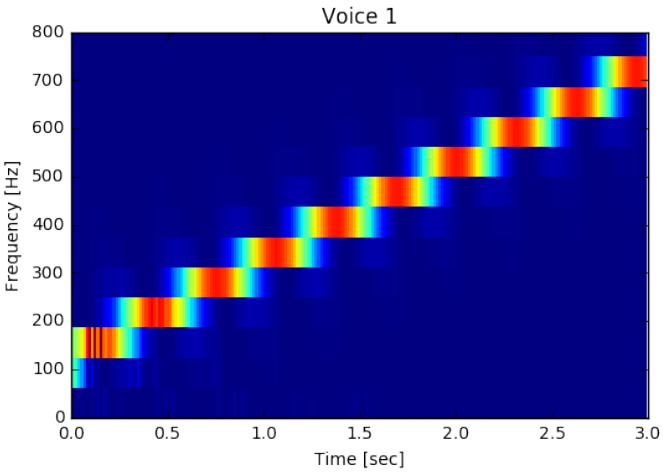
The extracted voices were then reconstructed from the spectogram using the inverse short-time fourier transform. The inverse short-time fourier transform performs an inverse discrete fourier transform on the column vectors in $\hat{V}$ to produce the extracted voice in the time-domain.

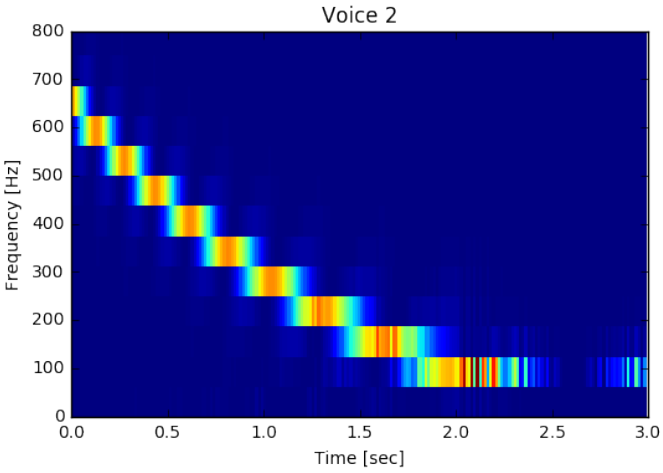
The extracted voices could then be analyzed for error using SDR and MSE.

V. Results

A. Intermediate Results

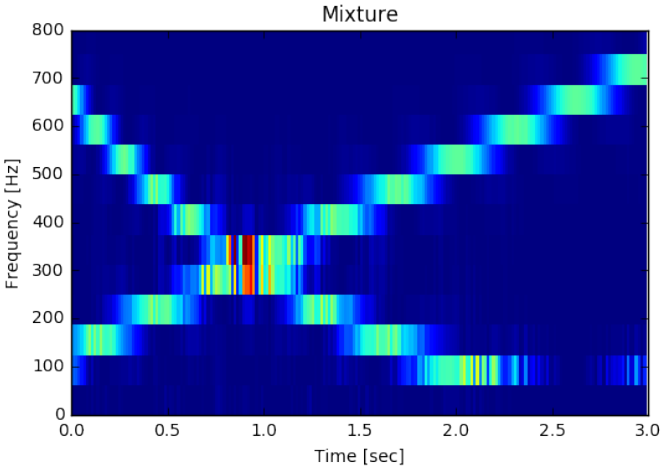
To test the system, a sine wave with an increasing frequency was used as voice 1, and a sine wave with a decreasing frequency was set to voice 2. Short-time fourier transforms were performed to create the spectograms.





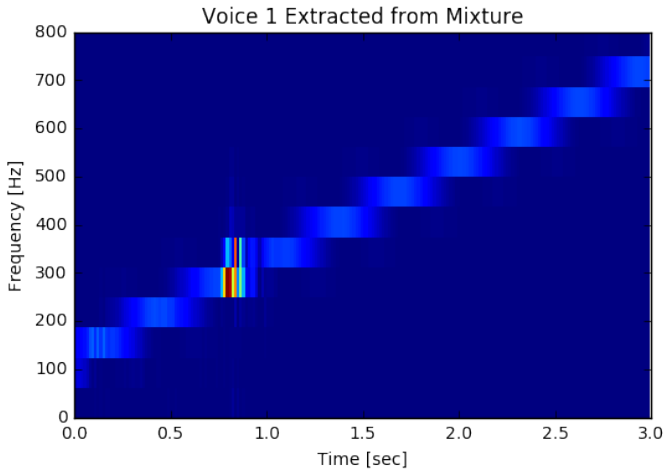
To train the model using the two voices, the basis matrices are calculated using non-negative matrix factorization.

A mixed signal was created by summing the two signals. A spectogram of the mixture was used as a test input to verify that voice 1 could be extracted.

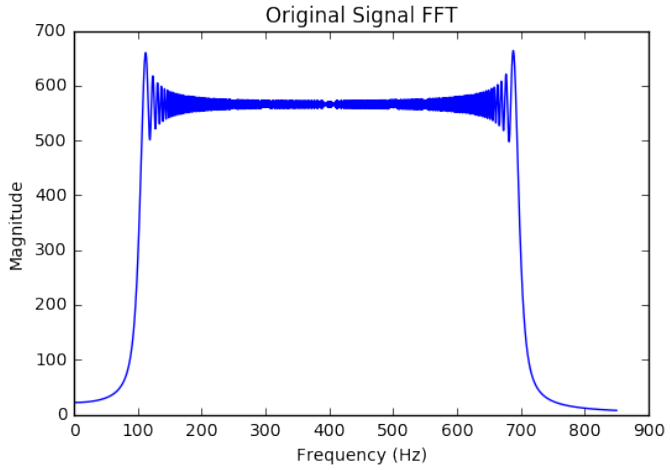


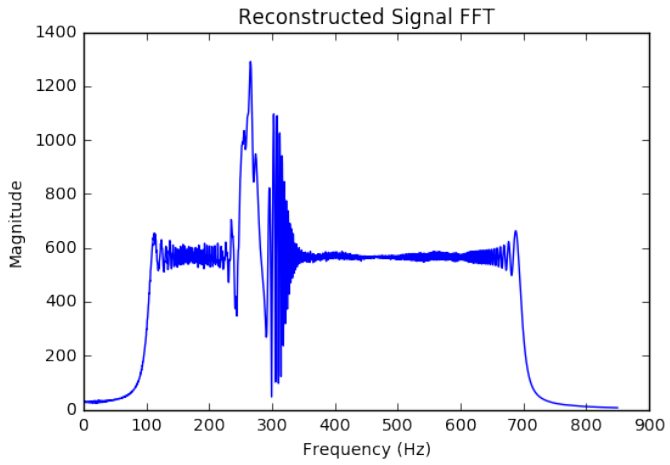
The weights for the basis matrices were calculated using

a least squares solution. The spectogram of the voice was then reconstructed using the basis and weight matrix.



The magnitude of a fast fourier transform (FFT) of the signal was then plotted. Distortion can be seen where the two signals had intersected on the spectrogram at approximately 300Hz in the reconstructed signal (shown below).





The SDR of the reconstructed signal was 27.86, the MSE was 0.036. Qualitatively, the reconstructed signal sounds very similar to the original, with a noticeable change in volume around the 0.8 second point. This distortion implies that intersections in time-frequency domain (STFT domain) cause distortions in the produced signal.

B. Final Results

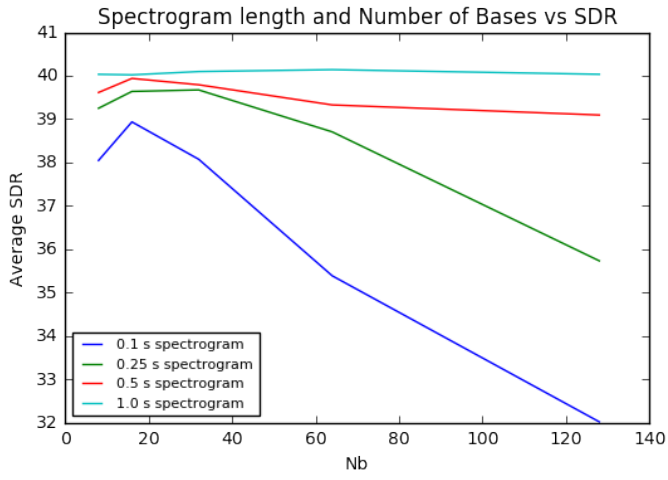
Parameter tuning was performed to determine the optimal parameters for N_b and N_s (where N_b is the number of bases, and N_s is the framelength or number of samples in each spectrogram). N_s represents a spectrogram of length $\frac{N_s}{f_s}$ seconds, where f_s is the sampling frequency. 20 combinations of the 16 available voices were used to find the average SNR and MSE for various combinations of N_b and N_s. The training and validation process took 2.167 hours on an Intel i5 Gen 2 processor in a Thinkpad laptop. The weights were generated for both voices, then extraction was attempted for both voices. The equations used to acquire the average SDR and average MSE are shown below.

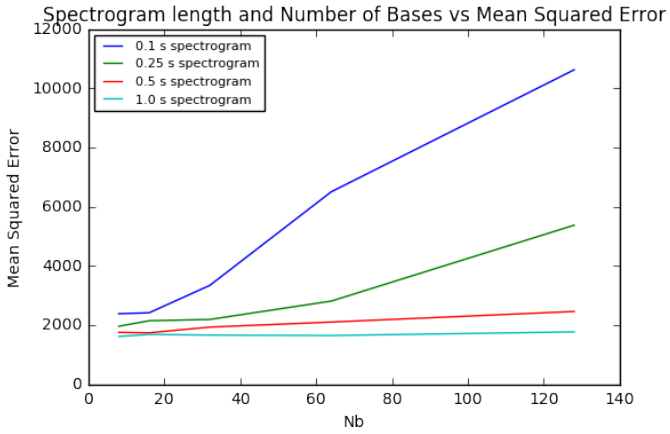
Given sufficient computation time, examples at 4 second

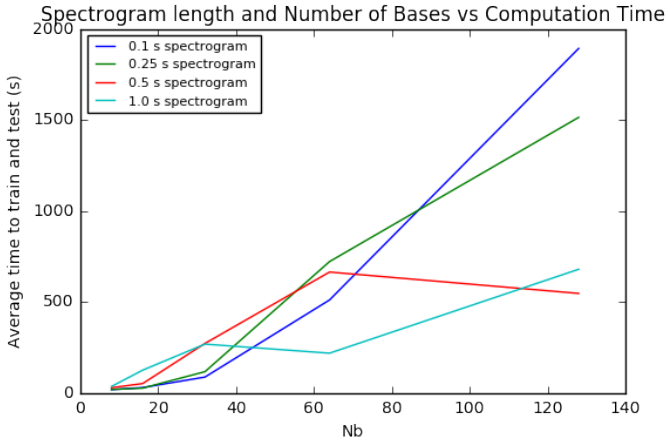
$$\text{SDR}_{\text{avg}} = \frac{1}{20}\sum_{0}^{20}\frac{\text{SDR}(v_1, \hat{v}_1) + \text{SDR}(v_2, \hat{v}_2))}{2} \tag{8}$$

$$\text{MSE}_{\text{avg}} = \frac{1}{20}\sum_{0}^{20}\frac{\text{MSE}(v_1, \hat{v}_1) + \text{MSE}(v_2, \hat{v}_2))}{2} \tag{9}$$

The results of the parameter sweep can be seen below. Longer spectrogram lengths outperformed shorter spectrogram windows in both MSE and SDR. Generally speaking, longer spectrogram windows also took less time to compute.



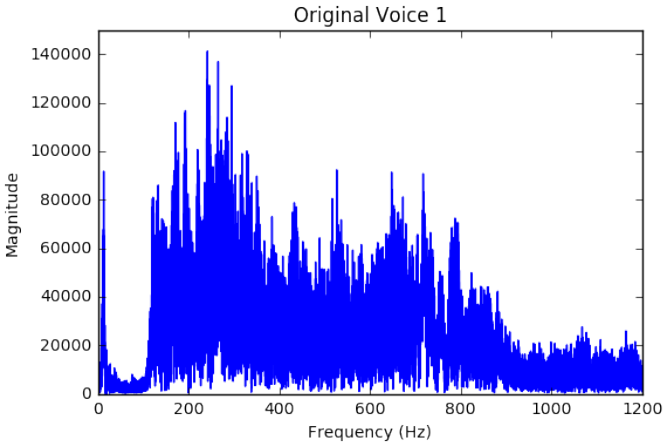


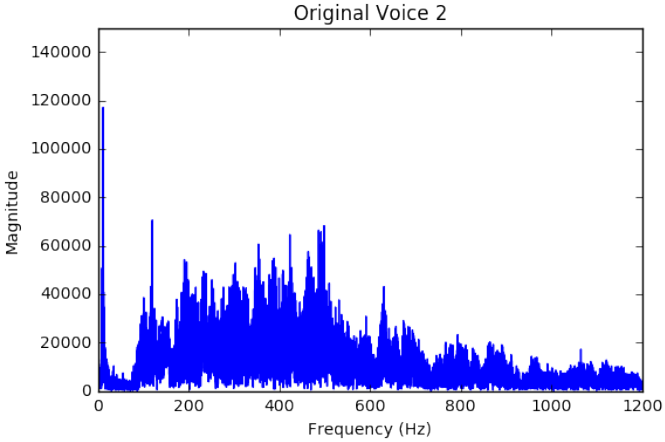


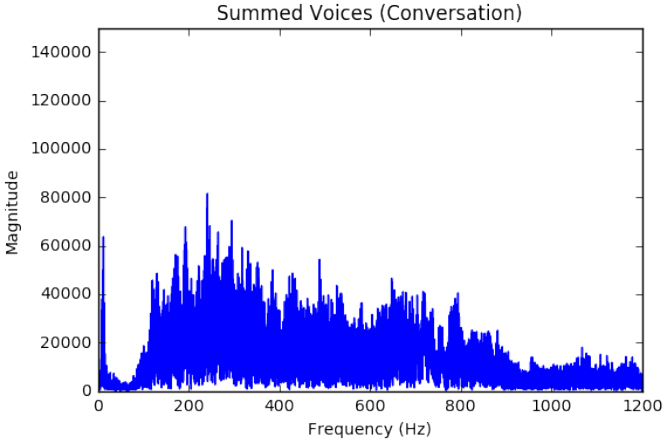
The relationship between spectrogram length and computation time implies that a longer STFT takes a longer time to compute. That is, there are more individual spectrograms with a small spectrogram window, and the number of individual spectrograms is a good indicator of computation time. It can also been seen that the number of bases increases the computation time. This is intuitive considering this increases the size of all matrix operations in the cost function.

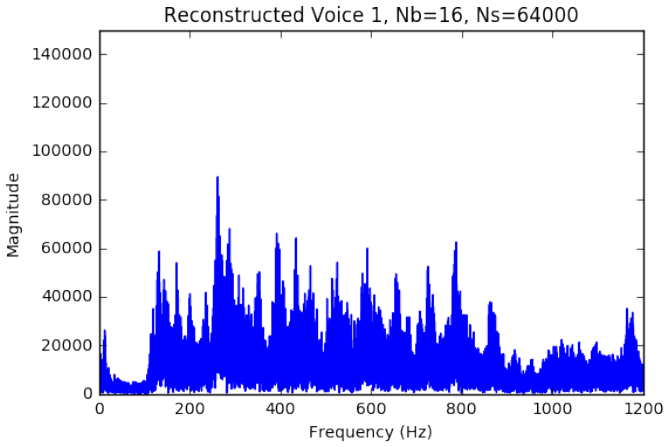
Increasing the number of bases increased the MSE but decreased the SDR (except for the optimal N_b value, 16). This implies that over-fitting is occurring for values greater than 16, and under-fitting is occurring for values under 16. Thus, $N_b = 16$ should be selected for testing. Longer time windows improve the results of the algorithm. Analyzing the results qualitatively, longer spectrograms seemed to create results that were more "echo-y".

An example was selected that reflects a good qualitative extraction. Using $N_s = f_s \times 4$ (four seconds) and $N_b = 16$, a voice was extracted from the dataset. Certain characteristics of Original Voice 1 can be seen in Reconstructed Voice 1, namely the amplitudes surrounding 800Hz. The reconstructed voice is decipherable, while the conversation may take several listens to decipher the content. The SDR of the example was 41.7, with a MSE of 484. The images below show the FFT of the original voices, the mixed voices and the reconstructed signal.









The reconstructed signal is available on the server at http://138.197.19.152:8888/ for listening. See notebook "Parameter Tuning, Testing and Error Measures"

VI. DISCUSSION AND FUTURE WORK

Given more time, a wiener filter can be applied as a soft mask to the mixed signal as noted by *Grais and Erdogan* [1]. It would scale the magnitudes of the spectogram of the mixed signal.

$$H_{Wiener} = \frac{\hat{S}^2}{\hat{S}^2 + \hat{M}^2} \tag{10}$$

$$\hat{S} = H_{Wiener} \otimes \hat{S} \tag{11}$$

The tests above were run on only a small number of possible N_b and N_s and with only a small number of possible combinations of audio samples (20 out of the 120 possible combinations with 16 voices). Further testing could expand on the parameter range and use more combinations for more robust error measures. For even more data, the *Amazon Polly*[6] text-to-speech framework could be used.

Further testing should also expand on the SDR and MSE to include an error measure that more closely reflects qualitative analysis. The effect on speech-to-text converters could be critically assessed.

ACKNOWLEDGMENT

Thanks to *Emad M. Grais* and *Hakan Erdogan*, whose paper on single channel speech music separation using nonnegative matrix factorization provided the basis for the technique that was used throughout this paper. Thanks to *Rich Radke*, who motivated us to pursue this problem via the final project of ECSE 4530, as well as teaching us everything we know about digital signal processing via his lectures, assessments and detailed notes.

REFERENCES

[1] Grais, Emad M., and Hakan Erdogan. "Single Channel Speech Music Separation Using Nonnegative Matrix Factorization and Spectral Masks." 2011 17th International Conference on Digital Signal Processing (DSP) (2011): n. pag. Web.

[2] Lefvre, Augustin, Franois Glineur, and P.-A. Absil. "A Convex Formulation for Informed Source Separation in the Single Channel Setting." Neurocomputing 141 (2014): 26-36. Web.

[3] Vincent, E., R. Gribonval, and C. Fevotte. "Performance Measurement in Blind Audio Source Separation." IEEE Transactions on Audio, Speech and Language Processing 14.4 (2006): 1462-469. Web.

[4] Virtanen, Tuomas. "Monaural Sound Source Separation by Nonnegative Matrix Factorization With Temporal Continuity and Sparseness Criteria." IEEE Transactions on Audio, Speech and Language Processing 15.3 (2007): 1066-074. Web.

[5] CMU Census Database (C1991). Raw audio. Alphanumeric database

[6] "Amazon Polly Lifelike Text-to-Speech." Amazon Web Services, Inc. N.p., n.d. Web. 07 Dec. 2016.