Spring 5-2018

Vocal Processing with Spectral Analysis

Bradley Fitzgerald
bfitzgerald8812@gmail.com

VOCAL PROCESSING WITH SPECTRAL ANALYSIS

By

Bradley J. Fitzgerald

Honors Scholarship Project

Submitted to the Faculty of

Olivet Nazarene University

for partial fulfillment of the requirements for

GRADUATION WITH UNIVERSITY HONORS

March, 2018

BACHELOR OF SCIENCE

in

Engineering, Electrical Concentration

Scholarship Project Advisor (printed)	Signature	Date
Honors Council Chair (printed)	Signature	Date
Honors Council Member (printed)	Signature	Date

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

A well-known signal processing issue is that of the “cocktail party problem”, which refers to the need to be able to separate speakers from a mixture of voices. A solution to this problem could provide insight into signal separation in a variety of signal processing fields. In this study, a method of vocal signal processing was examined to determine if principal component analysis of spectral data may be used to characterize differences between speakers and if these differences may be used to separate mixtures of vocal signals. Processing was done on a set of voice recordings from 30 different speakers to create a projection matrix which could be used by an algorithm to identify the source of an unknown recording from one of the 30 speakers. Two different identification algorithms were tested. The first had an average correct prediction rate of 15.69%, while the second had an average correct prediction rate of 10.47%. Additionally, one principal component derived from the processing provided a notable distinction between principal values for male and female speakers. Males tended to produce positive principal values, while females tended to produce negative values. The success of the algorithm could be improved by implementing differentiation between time segments of speech and segments of silence. The incorporation of this distinction into the signal processing method was recommended as a topic for future study.

Keywords: vocal processing, spectral analysis, principal component analysis

INTRODUCTION

The digital age has produced a demand for signal processing techniques in various areas of study [1,2]. One such demand which has proved to be particularly difficult to address has been in the area of signal source separation; specifically, there is a call for a solution to the "cocktail party problem" [1, 3]. The cocktail party problem refers to the phenomenon experienced by humans in instances of a large gathering. When in a crowded room, one may be holding a conversation with another in the midst of various other voices speaking in the same vicinity. Little is known about the brain processes occurring during the processing of speech with background noise, yet unimpaired individuals are able to separate and group different sounds according to their origin while focusing on a single vocal signal [3]. This paper proposes a vocal analysis method which seeks to emulate this process through the production of a data projection matrix. This matrix is used to characterize unknown vocal signals with the goal of identifying their sources from within a set of recorded voices.

REVIEW OF LITERATURE

Linguistic Theory

Much of modern linguistic theory contains discussion on the components of speech, both from the perspective of production and of perception. Bowers, Kazanina, and Andermane explain that the traditional view is that a group of linguistic units known as phonemes can be used to represent the basic units of speech in a language [4]. The English language, for example, is commonly represented as having 44 attributed phonemes, including sounds created by each letter of the English alphabet in addition to some sounds created by the combination of letters,

such as /sh/, /th/, or /ch/ [4]. It is the arrangements of these phonemes which form the words available in a language.

While this is the traditional starting point from which speech is studied, it is not accepted without question as the ultimate representation of linguistic patterns. Phonemes can be broken down further into phones, which represent the unique ways in which a single phoneme can be pronounced, namely due to its orientation in a word [4]. For instance, the phoneme /t/ is articulated differently when oriented at the beginning of a word, as in “two”, and in the middle of a word, as in “steak”. Most arguments against phoneme theory advocate for a greater complexity and contextual nature of phoneme recognition [4, 5]. For instance, it has been shown that phoneme recognition does not occur as clearly outside of the context of speech as within speech [5]. Further, the perceptual learning of phonemes can be specific to the voice of the speaker and the ear of the listener, adding more to the complexity of phoneme recognition [4]. However, despite these challenges, this needn’t lead to the full dismissal of phoneme theory. Indeed, phoneme recognition is still used by researchers as a basis for measuring the quality of speech identification algorithms [6]. The fact that phoneme recognition can be altered depending on the context of the voice speaking implies a difference in phoneme production among speakers [5]. The goal of the method presented in this paper was to identify components of uniqueness among different speakers for the purpose of source separation; these components were identified through signal processing techniques including blind source separation and spectral analysis.

Blind Source Separation

The history of analysis techniques used to approach issues regarding speech processing has been relatively inconsistent and scattered. Researchers Hu and Loizou described how the

comparison of algorithms developed for speech enhancement is highly difficult due to the variety of ways in which these methods are presented [7]. Inconsistencies between methodologies and the absence of a common speech database reference are just a couple of the issues which stand in the way of clear comparison. In general, however, algorithms developed for the purpose of addressing the cocktail party problem do tend to fall under a common approach known as blind source separation [8]. Practical use of an algorithm which separates vocal signals, especially in real-time applications, requires the use of this approach. In blind source separation, a system receives a mixture of signals in a single input. The goal of the approach is to determine the original signals which have been mixed together. This approach has applications in many areas of signal processing, including those surrounding processing speech [8]. In the case of mixed voices, the mixed input signal's original source signals are the vocal signals from each individual. The practicality of blind source separation comes from its ability to perform source separation without prior knowledge of the signal sources.

Buchner and Aichner explained the standard approach to blind source separation [1]. Typically, blind source separation problems are approached with the assumption of a reverberant environment [1]. For example, this would hold true for many instances of the cocktail party problem, where sound waves from multiple speakers will bounce off of walls or objects and be projected around the room accordingly. Signals are often measured via multiple inputs, allowing the system to utilize reverberation as an aid in the process of separating signals. These are often referred to as multiple-input-multiple-output (MIMO) systems [1]. Such processes have practical application in certain areas of signal processing, including those related to imaging, due to the greater likelihood that certain systems may practically allow for multiple data input streams. However, in the area of speech processing, a multitude of data inputs is not

always practical. Applications such as voice recognition on mobile devices require the use of a single microphone, or at best, multiple microphones in very close proximity. Because of this, alternative source separation and spectral analysis methods must be examined in order to meet the needs of real-world applications.

One way in which this single input-stream requirement can be addressed is by measuring signals against a predetermined set of standard signal components. The work of Sirovich and Kirby, who worked in the field of facial recognition, is a classic example of this method [9]. They developed a methodology in which singular value decomposition was utilized to develop a set of images that formed the "building blocks" of constructing the picture of a face [9]. In terms of their usage of singular value decomposition, these images could be classified as eigenvectors. By this definition, the eigenvector images could be summed together, each scaled by a corresponding eigenvalue, to form an image of a face. Over time these eigenvector images came to be known as "eigenfaces". In a similar way, this research aims to develop a set of eigenvectors which may be used to separate and reconstruct human speech.

Spectral Analysis

Often, the data available from the initial form of a signal may be insufficient for the purposes of source recognition or separation. In such cases, it becomes beneficial to perform certain operations on the signal which allows them to be observed from different perspectives. One widely used technique is known as the Fourier transform [10]. The namesake of this operation, Joseph Fourier, postulated that any signal may be observed as the summation of a set of sinusoidal waves [11]. The Fourier transform allows a signal to be decomposed into a set of cosine or sine waves that vary in amplitude and frequency, according to the original signal [11]. This operation transforms the original signal from the time domain to the frequency

domain. The resulting frequency components are known as spectra. These spectra may be observed and analyzed to determine significant pieces of information about the composition of the original signal. It is the observation of these frequency spectra that is known as spectral analysis. Nakatsuji and Omatu demonstrated the use of the Fourier transform as a preprocessing technique used in real-time spectral analysis [10]. In their research, spectral analysis was repeated as new samples were added to the initial set, which would consistently update the spectral data to reflect the current data set [10]. This approach provides an expansion on traditional spectral analysis, which typically utilizes a single initial instance of a Fourier or equivalent transform to produce frequency data which is then compared with all later data [10]. Despite its usefulness across a wide variety of fields, the Fourier transform in its classic form is a relatively lengthy computational process by modern standards. This has posed an issue with the use of the Fourier transform in fields where computational speed and efficiency are limiting factors [2]. Such fields include that of speech processing, where research on signal separation is done for the benefit of devices such as hearing aids or speech-to-text applications. In such devices, which often utilize embedded systems, the allowable complexity of an algorithm is limited by memory storage and computational speed.

In these applications it becomes useful to analyze data using a modified Fourier transform known as the fast Fourier transform [2]. This transform requires fewer computations than the traditional discrete Fourier transform, making it a commonly used technique. Paèz and Garzòn demonstrated that the fast Fourier transform could be used in a spectrographic analysis application which offered a reduced computation time when compared to equivalent applications used by traditional entities such as MATLAB [2]. In doing so, they showed that the

fast Fourier transform has practical use in increasing the efficiency of spectral speech analysis in embedded systems [2].

Independent and Principal Component Analysis

Among researchers addressing issues of signal processing and source separation there are two widely-used techniques known as Principal Component Analysis (PCA) and Independent Component Analysis (ICA). Both of these techniques have seen use in the field of data processing, namely for the purpose of data reduction and analysis [12]. It is important to distinguish between these two techniques in order to understand the specific benefits of each in different applications. Both techniques aim to decompose a set of data into generalized components. In PCA, correlation between portions of the original data set is used to determine the most common, and thus most significant, components of the set [12,13]. These components are compiled and listed with the goal of eliminating redundancy by containing as much significance as possible within each component [12]. The most significant difference between this approach and that of ICA is that PCA utilizes some sample data or prior knowledge to be used in the decomposition process. It requires previous information about the sources being separated, making it a difficult technique to be utilized effectively in a true problem of blind source separation [12]. ICA was developed in response to this shortcoming as a more adaptable form of PCA [12]. ICA performs a similar function, but begins only with the assumption that the data set to be examined is a linear mixture of independent source signals [12]. The lack of a necessity for prior information on the sources of the mixture has made ICA a widely utilized technique in the field of blind source separation [14]. However, ICA alone tends to be insufficient in particularly complex problems, as the number of mixture signals usually must be

greater than the number of source signals [14]. Most methodologies involving the use of this technique require pairing it with an additional signal-identification technique [8].

Mori et al. described a step in the right direction by utilizing independent component analysis with a binary masking technique in order to overcome reliance on a MIMO system [8]. They implemented ICA based on a single-input-multiple-output (SIMO) model and experimented with the algorithm's ability to separate mixtures of vocal signals [8]. This methodology was shown to significantly improve upon the performance of ICA alone [8]. Interestingly, the process still utilized multiple microphones to produce the greatest accuracy, but each input was analyzed individually using the SIMO model [8]. Mori et al. described the algorithm as being effective when the number of mixed vocal signals was less than or equal to the number of microphones utilized as inputs [8]. Thus, the ability to perform vocal source separation with fewer – and ideally, a single – microphone remained an unanswered problem.

Lu et al. made significant progress in terms of developing a blind source separation algorithm truly meant for a single stream of input data [14]. Their process combines ICA with higher-order statistics to extract significant component data from a single input stream [14]. The process was shown to be a successful starting point for algorithms seeking to answer the call for single-channel blind source separation [14]. It was noted that the algorithm was limited, like most, on the number of original source signals which could be separated [14]. Although this was a step forward for the problem of blind source separation, this particular process is not necessarily aimed towards the issues surrounding the cocktail party problem specifically. To create successful blind source separation algorithms for more specific applications, it is necessary that work be done to bridge the gap between the particular nuances of such applications and generalized source separation techniques such as this.

One of the key inspirations of this research was a study performed by Makarewicz and Makarewicz in the field of source separation [13]. In their study, they examined a possible method of addressing the problem of remotely determining the mineral content of pyroxene mixtures. The goal of such research was to develop methods by which the Compact Reconnaissance Imaging Spectrometer for Mars (CRISM) might analyze the mineral content of soils with which it comes into contact. They examined light spectra produced by pyroxene mixtures, using PCA to decompose this information into eigenvectors. Examination of these eigenvectors as principal components was used to correlate them with properties of the mixture, such as the percent clinopyroxene. This correlation was then used to create a projection matrix by which unknown samples could be characterized. This process was successful in characterizing composition and grain size of unknown mixtures. Here, we sought to enact a similar methodology directed towards the separation of vocal signals. In a manner similar to the studies put forward by Makarewicz and Makarewicz [13], as well as by Kirby and Sirovich[9], an algorithm was created to identify a set of speech eigenvectors which can be used to characterize unknown speech samples and mixtures via a projection matrix.

METHODS

Participants

Vocal samples gathered for this research were obtained voluntarily from 30 individuals, including 16 male and 14 female participants. 28 of the participants fell within the range of 18-23 years of age, while 2 were 47 years of age. 29 of the participants were Caucasian, and 1 female was of Hispanic heritage.

Materials

The script used in all voice recordings was obtained from a short story titled “Arthur the Rat” [15]. This story was used by the Dictionary of American Regional English (DARE) in the collection of voice samples from across the United States during fieldwork completed between 1965 and 1970 [15]. The passage is specifically designed to include phonetic representation of all phonemes present in American English [15].

All voice samples were recorded on a Samsung Galaxy S7 smartphone using the microphone of a standard Samsung headset. Windows Movie Maker was used to edit recordings, and VLC Media Player software was used for the conversion of audio files. All further data processing and analysis was completed using FreeMat, a free, open source coding environment similar to products such as MATLAB.

Data Collection

Participants were recorded in a quiet room and asked to read the entirety of the “Arthur the Rat” passage with a natural tone, comfortable pace, and slightly raised volume for the sake of producing clear recordings. The microphone was held by the speaker at a distance of approximately 6 inches in front of the mouth. The speakers were told to not stop due to any mistakes in pronunciation or reading that may occur during the recording.

Any mistakes which caused the speaker to deviate from the given script as well as additional comments made by the speaker were later removed from the recording using Windows Movie Maker and saved as an mp4 file. These files were then imported into VLC Media Player to be converted to a standardized audio format. Each mp4 file was converted to a waveform audio file (WAV) format with a single channel, a bitrate of 88 kilobytes per second, and a sample rate of 11025 Hertz. The bitrate and sample rate were chosen to increase

efficiency by limiting the amount of data which would need to be processed for each vocal sample.

WAV files were imported into FreeMat for processing as follows. The data from the recording was divided into time segments with a 90% overlap between consecutive segments and 4096 data samples per segment. This segmenting of the recording provided small samples of data of identical sizes for all recordings; all remaining processing of the recordings was done according to these segments. The fast Fourier transform of each time segment was calculated to produce a frequency spectrum for the sample and was stored in an *n* by 2048 spectra matrix, where *n* represented the number of time segments created for the audio file. Each spectrum was normalized before being stored into the spectra matrix in order to account for differences in volume between speakers. Figure 1 illustrates sample spectra of one recorded speaker.

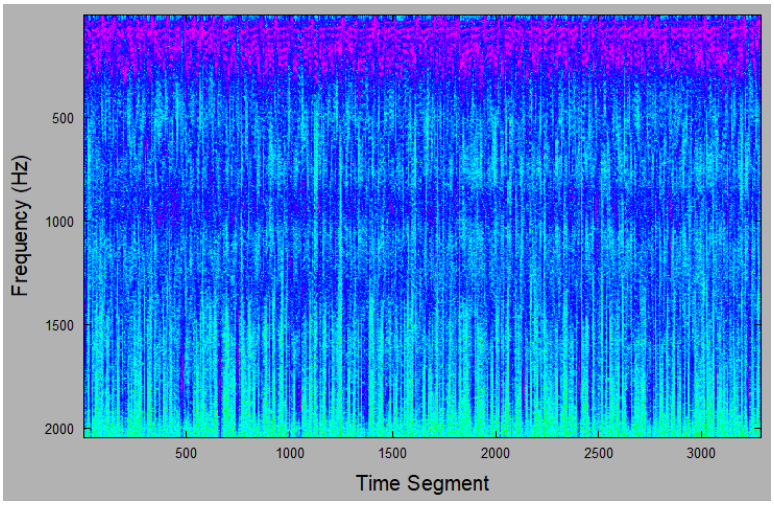

Figure 1. Normalized frequency spectra for each time segment from Speaker #1. Frequency spectra represent the results of the Fast Fourier Transform on the original recording's time segments. Color contrast illustrates intensity of frequencies in each time segment. Red indicates weak intensity, while blue indicates strong intensity.

Singular Value Decomposition

Following the production of the frequency spectra for each recording, the spectra were each individually processed. Singular Value Decomposition (SVD) was performed on each set of spectra representing the 30 speakers, producing 30 unique sets of principal values and principal vectors. The SVD function orders the produced principal vectors in order of significance, placing the most significant vectors at the front of the matrix. The first 50, and thus the 50 most significant vectors, were saved for each of the 30 spectra sets.

This process was completed for all 30 recordings, producing 30 sets of 50 principal vectors. These matrices were then concatenated into a single matrix on which SVD was completed a final time. This produced a set of 1500 principal vectors representing the full span of the 30 recorded vocal signals. This set was saved as the final principal vectors matrix, Matrix A; this matrix was not changed from this point forward in the process. Matrix A was then used to create a projection matrix of principal values which characterized each original speech signal in terms of these principal vectors. This was completed by reproducing the spectra for each speaker in the same manner described before. A principal value matrix was calculated by multiplying a speaker's spectra matrix by the Matrix A. Then, for every speaker, the principal values corresponding to a single principal vector were averaged and the standard deviation found. Matrix B stored the averaged principal values for each speaker, and Matrix C stored the standard deviations of the principal values for each speaker. Data from Matrix B and C for the 10 most significant principal vectors is presented in the Appendix. This completed the preprocessing of data into projection matrices which would then be used to predict speakers.

Speaker Prediction – Algorithm 1

The difference between the two algorithms came in the way that weights were applied to each principal value in the set. A principal value that has a larger general magnitude among all the speakers is one that is more significant and prevalent throughout all the samples. Because of this, the weight given to each principal value in Algorithm 1 was calculated by summing the magnitudes of a particular principal value for all speakers, shown as the vertical columns in Appendix A. Then, Z-scores were calculated by dividing the difference between the measured principal value and the average principal value of a known speaker from the Appendix A database by the standard deviation of that principal value for a known speaker from the Appendix B database. These Z-scores were calculated for each measured principal value and possible speaker combination. Finally, the measure used to compare speaker possibilities was calculated by summing the weight multiplied by the Z-score of every principal value for a known speaker. The calculation used to produce these values is shown in Equation 1,

$$M = \sum\left(\frac{\alpha-\mu}{\sigma} * W\right) \quad (1)$$

where:

M = Z-score sum,

α = measured time segment principal value,

μ = average speaker principal value,

σ = speaker's principal value standard deviation, and

W = principal vector weight.

This produced a single, weighted Z-score sum M for every possible speaker from the database. The speaker with the lowest value M was chosen as the predicted speaker for the given time

segment. This process was completed for each time segment in the spectra set, allowing the algorithm to predict which speaker produced the recording for every time segment.

Speaker Prediction – Algorithm 2

Algorithm 2 followed a similar methodology to the first. The spectra and principal values for the unknown speaker were produced in the same manner. However, the weights applied to each principal vector were removed. As stated, the SVD function used to calculate the final principal vectors matrix arranges the principal vectors in order of significance, placing the most significant vectors at the front of the matrix. Thus, the first columns in the principal value database represent the most significant data. For this algorithm, it was chosen to only calculate Z-scores for the first 10 sets of principal values from the database. Z-scores were calculated and summed in the same manner as in Algorithm 1. No weights were applied to the calculated Z-scores. The calculation used to produce the Z-score sum to be compared between speakers is shown in Equation 2,

$$M = \sum_{i=1}^{10} \left(\frac{a-\mu}{\sigma}\right) \quad (2)$$

The speaker with the lowest Z-score total was again chosen as the algorithm's predicted speaker for a given time segment. Algorithm 2 was completed for the full duration of each original voice recording to determine its effectiveness at determining source speakers.

RESULTS

30 speakers were recorded while reading the given script. These recordings were processed by performing singular value decomposition on the spectral data from each recording to produce a projection matrix, which was used in the development of two speaker identification algorithms. The algorithms were developed in FreeMat to predict a speaker for

every time segment of a given recording. Both algorithms were run for the full duration of all 30 recorded speakers. A plot of the predictions made by the algorithm for a given time segment of a recording is shown in Figure 2. The accuracy of Algorithm 1 was computed for each speaker and displayed in Table I. The correct prediction rate represents the percentage of time segments of a given speaker for which the algorithm correctly predicted the identity of the speaker. Overall, the algorithm had an average correct prediction rate of 15.69%, with a standard error of 3.93%. These rates ranged from 0.22% to 85.69%. Algorithm 1 performed notably well with Speaker 8, which produced the highest correct prediction rate of 85.69%. The accuracy of Algorithm 2 was computed for each repetition and displayed in Table II. Overall, the algorithm had an average correct prediction rate of 10.47%, with a standard error of 2.82%. These rates ranged from 0.00% to 65.83%. As with Algorithm 1, this algorithm performed the best with Speaker 8, producing the highest Algorithm 2 correct prediction rate of 65.83%.

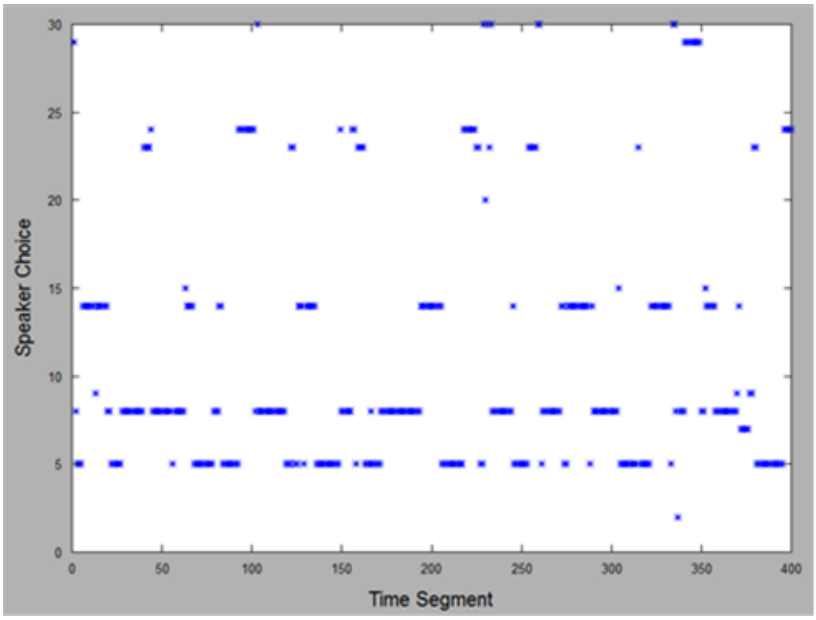

Figure 2. Algorithm 1 speaker predictions for first 500 time segments of Speaker 5. The vertical axis identifies Speaker choices 1-30, while the horizontal axis identifies the segment of time analyzed. Each "*" symbol shown on the plot represents a speaker prediction made by the algorithm during an iteration for each time segment.

Recording	Correct Prediction Rate	Recording	Correct Prediction Rate
1	0.64%	**16**	1.04%
2	11.18%	**17**	3.34%
3	1.67%	**18**	15.91%
4	1.13%	**19**	2.04%
5	24.87%	**20**	1.23%
6	0.36%	**21**	6.61%
7	1.65%	**22**	23.20%
8	85.69%	**23**	43.59%
9	8.49%	**24**	73.88%
10	0.22%	**25**	9.69%
11	1.14%	**26**	2.04%
12	0.00%	**27**	24.38%
13	1.99%	**28**	1.68%
14	36.97%	**29**	36.82%
15	16.11%	**30**	18.14%

Table 1. Algorithm 1 accuracy. Table 1 notates the percentage of time segments Algorithm 1 correctly predicted the recording's speaker for each of the 30 recordings. Algorithm 1 had a correct prediction rate greater than 10% for only 12 recordings.

Recording	Correct Prediction Rate	Recording	Correct Prediction Rate
1	0.00%	**16**	1.87%
2	3.41%	**17**	5.35%
3	4.51%	**18**	10.67%
4	0.97%	**19**	4.72%
5	30.31%	**20**	3.88%
6	0.58%	**21**	3.18%
7	1.46%	**22**	14.59%
8	65.83%	**23**	40.79%
9	0.74%	**24**	38.45%
10	0.58%	**25**	17.25%
11	0.39%	**26**	1.37%
12	0.08%	**27**	14.56%
13	0.97%	**28**	0.12%
14	24.60%	**29**	8.43%
15	3.43%	30	18.14%

Table 2. Algorithm 2 accuracy. Table 2 notates the percentage of time segments Algorithm 2 correctly predicted the recording's speaker for each of the 30 recordings. Algorithm 2 had a correct prediction rate greater than 10% for only 9 recordings.

DISCUSSION

This research had two objectives, the second of which depended upon the success of the first. The first objective was to determine whether principal component analysis of spectral voice data could be used to identify differences between speakers. Success in this objective would be characterized by an algorithm predicting the correct speaker for at least 70% of the recording for each of the 30 speakers, which was not met with the utilized methods. Algorithm 1 was unable to correctly guess the speaker of a recording for a majority of the total time segments analyzed. For 18 out of the 30 recordings, the algorithm correctly guessed the speaker less than 10% of the recording. Algorithm 2 resulted in a slight decrease in performance when compared to Algorithm 1, still yielding an insufficient success rate. For 19 out of the 30 recordings, Algorithm 2 correctly guessed the speaker less than 10% of the recording. Thus, both algorithms yielded similar results, with neither being able to consistently identify an unknown speaker for more than 70% of the speaker's recording.

It is worth noting that in both algorithms, a few individual speakers were guessed overwhelmingly often no matter which speaker was actually present in the recording. For instance, Algorithm 1 had a strong tendency to guess Speaker 8 and 24 for all of the recordings. This is likely the reason these few speakers had higher percentages of accuracy. This would imply that the few higher-performing recordings likely resulted from the tendency of the algorithm to become too focused on some characteristic of a certain speaker rather than a legitimate recognition of the speaker.

The poor performance of the speaker identification algorithms was consistent with the data calculated and stored in the principal value and standard deviation Matrices B and C. By comparing the data in these two matrices, shown in Appendices A and B, it can be seen that the standard deviations for many principal value-speaker pairs were relatively high. In many cases,

the standard deviation and average principal value for a principal vector and speaker were on the same order of magnitude, indicating a high level of variance in most principal value representation throughout an individual recording. This increases the likelihood that principal values between speakers will overlap, which increases the difficulty of attempting to classify the speaker based on these values. Figure 3 illustrates a comparison of the representation of a particular principal value between two speakers. As is shown, a majority of the points from each speaker fall in the same general area on the plot. Ideally, the speakers would produce more separated clusters, which would indicate that the principal vector involved was a useful principal component to be used in recognizing differences between speakers.

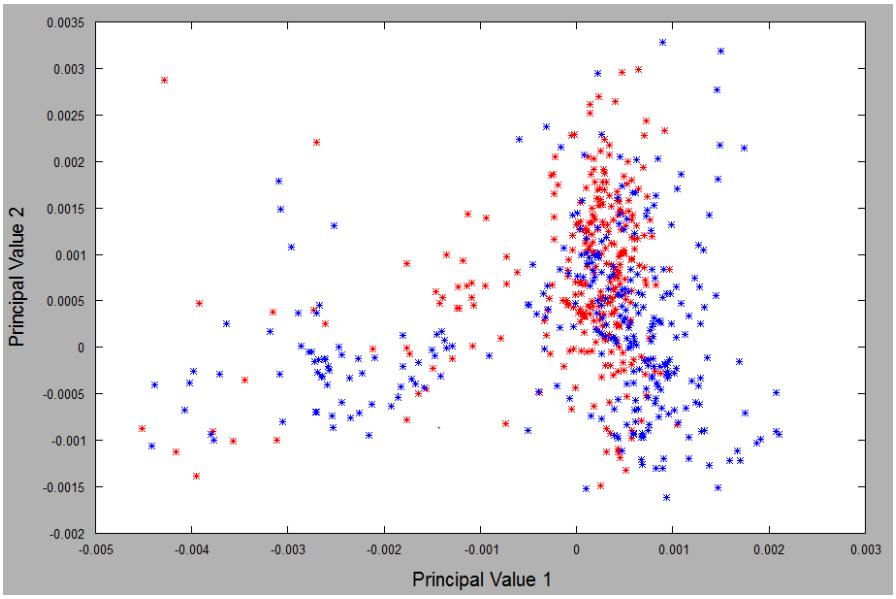

Figure 3. Interaction of two principal values for Speaker 1 (blue) and Speaker 2 (red). The principal values overlap between the two speakers for most of the region, making it difficult to use the interaction of the principal values to separate the speakers.

One piece of useful and interesting information was obtained from the results of the study. Participants were grouped by gender in the speaker order, such that Speakers 1-16 represented males and Speakers 17-30 represented females. Upon examination of the average

principal values for principal vector #4, shown in Appendix A, it was discovered that a distinction can be made between the average principal values for males and for females. This is illustrated in Figure 4. Male speakers tended to have negative values corresponding to principal vector #4, while females tended to have positive values. Table 3 compares the prediction of the gender of each speaker to the actual genders of the speakers after using the sign of the average of principal vector #4 values to predict the speaker. This method correctly identified the gender of 87.5% of male speakers and 85.7% of female speakers. An analysis of the implications of this principal vector would be an interesting subject for future research. It could prove beneficial to further examine the specific representation of the vector across the spectrum of speakers in order to determine if there is an identifiable characteristic which is described by the principal component. No clear correlation between these values and a specific speech characteristic was found in the brief analysis of this principal vector.

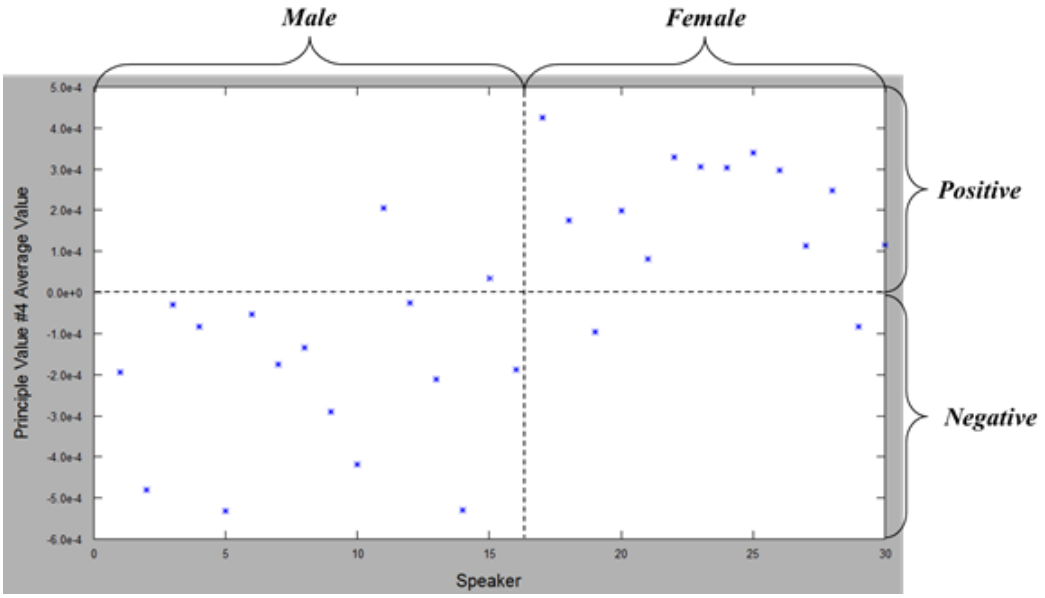

Figure 4. Comparison of principal values of principal vector #4 between males and females. Males tend to have negative principal values while females tend to have positive principal values, showing a potential for distinction between genders using principal vector #4.

Actual \ Predicted	Male	Female	Total
Male	14	2	16
Female	2	12	14
Total	16	14	30

Table 3. Gender prediction using principal values of principal vector #4. Table 3 shows the results of classifying the gender of a speaker according to the average principal value of principal vector #4 for the speaker. A positive average value predicted a female, while a negative average value predicted a male. This method correctly identified 14 out of 16 males and 12 out of 14 females.

Because the first objective of identifying spectral differences between speakers was unsuccessful, the second objective of separating mixed vocal signals was not attempted. This is because the developed algorithms were unable to correctly predict the source of a single-speaker recording, so they would not be effective in guessing the sources of mixed recordings. One potential source of error from the implemented methodology was the failure to distinguish between time segments where the speaker was talking as opposed to segments of no speech. The methods took no consideration of the difference between these two potential conditions, treating segments of silence identically as segments of speech. Addressing this issue would be a beneficial focus for continued study. Revising the computation method by eliminating the inclusion of data from segments of silence may prove better at highlighting the identifiable traits of different speakers.

A method of identifying differences between vocal signals through principal component analysis of spectral data was studied. The method was not successful in identifying differences such that they could be used to identify different speakers. One of the principal vectors created showed a difference between the corresponding principal values for males and females, identifying the vector as a potentially useful tool in identifying the gender of a speaker. More

analysis of this vector is recommended to determine if it can be correlated with a specific characteristic of speech. In addition, future work is recommended in which the method used here be modified to better account for segments of silence from the speaker in a recording.

REFERENCES

[1] H. Buchner and R. Aichner, “A generalization of blind source separation algorithms for convolutive mixtures based on second-order statistics,” *IEEE Trans. Speech Audio Process.,* vol. 13, no. 1, pp. 120–134, Jan. 2005.

[2] N. F. G. Páez and J. S. E. Garzón, “Performance evaluation of software for the spectral analysis of speech signals in a MIPS based architecture,” *Eval. Desempeño Softw. Para El Análisis Espectral Señales Voz En Una Arquit. MIPS*, vol. 34, no. 2, pp. 309–332, Jul. 2016.

[3] S. Evans, C. McGettigan, Z. K. Agnew, S. Rosen, and S. K. Scott, “Getting the Cocktail Party Started: Masking Effects in Speech Perception,” *J. Cogn. Neurosci*., vol. 28, no. 3, pp. 483–500, Mar. 2016.

[4] J. S. Bowers, N. Kazanina, and N. Andermane, “Spoken word identification involves accessing position invariant phoneme representations,” *J. Mem. Lang.,* vol. 87, pp. 71–83, Apr. 2016.

[5] E. Reinisch, D. R. Wozny, H. Mitterer, and L. L. Holt, “Phonetic category recalibration: What are the categories?,” *J. Phon*., vol. 45, no. Supplement C, pp. 91–105, Jul. 2014.

[6] A. Jeanvoine, D. Gnansia, E. Truy, and C. Berger-Vachon, “Influence of noise reduction algorithms on phonemes recognition in the case of Binaural Cochlear Implant coding,” *Technol. Disabil*., vol. 27, no. 1/2, pp. 51–63, Jan. 2015.

[7] Y. Hu and P. C. Loizou, “Subjective comparison and evaluation of speech enhancement algorithms,” *Speech Commun*., vol. 49, no. 7/8, pp. 588–601, Jul. 2007.

[8] Y. Mori et al., “Blind Separation of Acoustic Signals Combining SIMO-Model-Based Independent Component Analysis and Binary Masking,” EURASIP *J. Adv. Signal Process*., vol. 2006, no. 1, p. 034970, Dec. 2006.

[9] M. Kirby and L. Sirovich, “Application of the Karhuen-Loeve Procedure for the Characterization of Human Faces,” *IEEE Trans. Pattern Anal. Mach. Intell*., vol. 12, no. 1, pp. 103–108, Jan. 1990.

[10] H. Nakatsuji and S. Omatu, “Real Time Spectral Analysis,” *IEEJ Trans. Electron. Inf. Syst*., vol. 126, no. 3, pp. 383–388, 2006.

[11] T. A. Gallagher, A. J. Nemeth, and L. Hacein-Bey, “An Introduction to the Fourier Transform: Relationship to MRI,” *Am. J. Roentgenol*., vol. 190, no. 5, pp. 1396–1405, May 2008.

[12] C. Bugli and P. Lambert, “Comparison between Principal Component Analysis and Independent Component Analysis in Electroencephalograms Modelling,” *Biom. J*., vol. 49, no. 2, pp. 312–327, Apr. 2007.

[13] J. Makarewicz and H. Makarewicz, “Spectral mixture decomposition using principal component analysis applied to pyroxene mixtures,” presented at IEEE Whispers, Gainesville, FL, 2013.

[14] G. Lu, M. Xiao, P. Wei, and H. Zhang, “A New Method of Blind Source Separation Using Single-Channel ICA Based on Higher-Order Statistics,” *Mathematical Problems in Engineering*, 2015. [Online]. Available:https://www.hindawi.com/journals/mpe/2015/439264/. [Accessed: 27-Sep-2017].

[15] Dictionary of American Regional English, (2013). “Arthur the Rat.” [Online]. Available: http://dare.wisc.edu/audio/arthur-the-rat. [Accessed: 6-Sept-2016].

APPENDIX A – PRINCIPAL COMPONENT VALUE AVERAGES DATABASE

Speaker	Principal Value Averages ($x10^{-3}$)									
	PV #1	PV #2	PV #3	PV #4	PV #5	PV #6	PV #7	PV #8	PV #9	PV #10
1	0.0452	0.5662	0.2814	-0.1934	-0.2749	-0.7227	-0.8242	-1.1918	-0.7681	0.3751
2	0.3190	0.1517	0.2028	-0.4807	-0.4787	-0.7702	-0.8529	-1.4408	-1.1884	-0.3927
3	0.3015	0.0707	0.3259	-0.0300	-0.1625	-0.5728	-1.1162	-0.9804	-0.6102	0.8653
4	0.0366	0.0209	0.5392	-0.0824	0.0017	-0.2281	-0.6325	-0.8196	-0.5445	0.5931
5	-0.0791	0.0824	-0.1112	-0.5312	-0.4871	-1.2939	-0.8713	-1.4575	-1.3701	-0.2886
6	0.1408	-0.5756	0.3126	-0.0529	-0.2599	-0.1516	-0.9759	-0.8654	-0.6473	0.4085
7	0.0804	0.2443	0.2125	-0.1756	-0.0777	-0.5396	-0.6809	-0.8179	-0.7653	0.8757
8	-0.4990	0.4669	0.5833	-0.1350	-0.2172	-0.7032	-0.9615	-1.2488	-0.8943	0.8484
9	-0.1322	0.4463	0.2742	-0.2904	0.0078	-0.8190	-0.7830	-1.0411	-0.8435	0.7497
10	0.4358	0.5125	0.0233	-0.4175	-0.2654	-0.6145	-0.6428	-0.9316	-0.6248	0.6519
11	-0.1462	-0.0188	0.6042	0.2503	-0.1013	-0.6089	-1.074	-1.0096	-0.6970	0.4292
12	-0.0103	-0.0125	0.3685	-0.0251	-0.2963	-0.5653	-0.9377	-1.3918	-0.8695	0.3473
13	-0.0483	0.3446	0.0204	-0.2110	0.0100	-0.4681	-0.6503	-1.0603	-0.4283	1.0583
14	-0.0408	0.4588	0.0779	-0.296	-0.1844	-0.8445	-0.6822	-1.3058	-0.9105	0.6539
15	-0.0610	0.6312	0.5220	0.0341	0.0128	-0.4754	-0.7025	-1.0465	-0.5791	1.0855
16	0.0576	0.0695	0.1062	-0.1882	0.0251	-0.4045	-0.6532	-0.9177	-0.3895	0.8304
17	-0.2538	-0.1426	0.8340	0.4246	0.2567	-0.3045	-1.1500	-0.7884	-0.7352	0.8307
18	0.0391	0.3136	0.7607	0.1750	0.2813	-0.5528	-0.9035	-0.9280	-0.4043	1.1424
19	-0.1309	0.4559	0.6917	-0.0952	-0.2631	-0.8418	-1.1642	-1.0935	-0.8034	0.0357
20	0.2611	0.3345	0.8030	0.1986	0.0233	-0.3189	-0.8086	-0.6824	-0.5099	0.5250
21	-0.1126	0.6196	0.5727	0.0817	-0.2174	-0.6005	-0.8510	-1.3072	-0.6940	0.5982
22	0.0428	-0.1171	0.5259	0.3287	0.1095	-0.2379	-0.9821	-0.8300	-0.4992	0.7498
23	-0.0729	-0.1303	0.9140	0.3052	0.0291	-0.5530	-1.2767	-1.1022	-0.5934	0.5312
24	0.0172	-0.1421	0.8481	0.3033	0.3141	-0.5593	-1.0014	-0.8723	-0.8292	0.5440
25	-0.2938	0.2973	0.6129	0.3403	0.0004	-0.7349	-0.8868	-0.9079	-0.8620	0.1419
26	0.0114	0.0532	0.7529	0.2974	0.0350	-0.6165	-0.9651	-1.0628	-0.6633	0.3192
27	-0.9182	0.6916	0.7484	0.1126	0.1586	-0.0925	-0.3902	-0.3826	-0.5065	0.2003
28	-0.1699	0.2432	0.6477	0.2471	-0.0772	-0.3912	-0.9046	-0.7085	-0.5124	0.5651
29	-0.0668	0.1830	0.5966	-0.0833	0.0670	-0.6030	-0.7233	-0.9447	-0.5394	0.7478
30	-0.2687	0.7080	1.0226	0.1161	0.1683	-0.4511	-0.7025	-0.7335	-0.6888	0.2551

APPENDIX B – PRINCIPAL VALUE STANDARD DEVIATION DATABASE

Speaker	Principal Value Standard Deviation ($x10^{-3}$)									
	PV#1	**PV #2**	**PV #3**	**PV #4**	**PV #5**	**PV #6**	**PV #7**	**PV #8**	**PV #9**	**PV #10**
1	1.1412	0.7337	0.7714	1.0605	0.8183	0.8736	1.1658	1.1392	1.1003	0.9014
2	0.9338	0.6400	0.6436	1.0013	0.7168	0.8664	1.0100	0.9494	0.9206	0.8596
3	1.3529	0.8303	1.0941	1.1936	0.9362	0.8367	1.3238	1.3067	1.3802	1.0125
4	0.9075	0.6293	0.6241	0.7811	0.8254	0.7272	0.9662	0.7528	0.9605	0.8383
5	1.2082	0.8143	0.7827	1.2978	0.8307	0.9203	1.3625	1.0379	1.4875	1.5200
6	0.9251	0.6792	0.5827	0.790	0.6861	0.6951	0.9524	0.8367	0.9797	0.8102
7	1.0907	0.8110	0.8357	1.3787	0.8713	0.9114	1.0779	0.9742	1.1179	0.7900
8	1.3874	1.0686	1.1201	1.4416	1.4362	1.2135	1.5373	1.5681	1.4813	1.3884
9	1.4497	0.8425	0.9118	1.3445	0.9625	1.0606	1.3920	1.2423	1.1751	1.0378
10	1.1252	0.6813	0.8175	1.1064	0.8218	1.0169	1.1782	1.1261	0.9978	0.8240
11	1.2407	0.916	0.8153	0.9668	0.9108	0.9010	1.4049	1.0943	1.3122	1.3668
12	0.9074	0.729	0.6289	0.9130	0.7141	0.6945	0.9137	0.9808	1.1010	1.1019
13	1.0916	0.8474	0.9174	1.0768	1.0417	0.9622	1.1328	1.0701	1.2463	0.8847
14	1.3222	0.8700	0.8489	1.5190	0.9987	1.1433	1.3402	1.1237	1.2353	1.1093
15	1.2529	0.8668	0.9741	1.2885	1.2176	1.0453	1.2751	1.1630	1.2363	0.9948
16	1.0230	0.7618	0.8534	1.2875	0.8644	0.9927	1.0653	1.0863	1.1876	0.8357
17	1.2422	0.9537	1.0037	1.0528	1.2919	1.0360	1.3629	1.2937	1.2299	1.1686
18	1.3224	0.8589	1.0925	1.5525	0.9669	1.3602	1.2417	1.5642	1.2074	1.0593
19	0.9760	0.7898	0.8882	0.9932	0.8899	1.2505	1.2028	1.0380	0.8923	1.0680
20	1.2615	0.9881	0.9593	1.2814	1.1996	1.3500	1.4718	1.4165	1.1257	0.9915
21	0.9673	0.8482	1.0705	1.4054	1.0254	1.3736	1.1511	1.4147	0.8843	1.0021
22	1.5144	0.9601	1.0178	1.5375	0.9787	1.3772	1.500	1.4284	1.3047	0.8619
23	1.4231	1.1126	1.0759	1.3719	1.3906	1.3676	1.6692	1.5724	1.4062	1.3103
24	1.4031	1.0358	1.1526	1.5463	1.2673	1.2452	1.592	1.5680	1.2194	1.0835
25	1.1523	1.0978	0.9455	1.0323	1.1598	1.0777	1.3552	1.2845	1.1695	1.1632
26	1.2364	0.8998	0.9090	1.2904	1.0969	1.2965	1.3391	1.3153	1.1160	1.0253
27	1.1827	0.8524	0.8138	0.9016	1.273	1.1157	1.2143	1.2279	0.9792	0.9708
28	0.9511	0.7838	0.6751	0.7797	0.9532	0.7541	0.9813	0.9664	0.8991	0.9255
29	0.9358	0.8013	0.8805	1.2616	0.9327	1.2047	1.0931	1.1953	0.8972	0.9335
30	2.5023	1.1500	1.2359	0.9633	0.8473	0.8513	0.7499	0.5440	0.5739	0.5422

APPENDIX C – "ARTHUR THE RAT" SCRIPT [15]

"Once there was a young rat named Arthur, who could never make up his mind. Whenever his friends asked him if he would like to go out with them, he would only answer, 'I don't know.' He wouldn't say 'yes' or 'no' either. He would always shirk making a choice. His aunt Helen said to him, 'Now look here. No one is going to care for you if you carry on like this. You have no more mind than a blade of grass.'

One rainy day, the rats heard a great noise in the loft. The pine rafters were all rotten, so that the barn was rather unsafe. At last the joists gave way and fell to the ground. The walls shook and all the rats' hair stood on end with fear and horror. 'This won't do,' said the captain. 'I'll send out scouts to search for a new home.'

Within five hours the ten scouts came back and said, 'We found a stone house where there is room and board for us all. There is a kindly horse named Nelly, a cow, a calf, and a garden with an elm tree.' The rats crawled out of their little houses and stood on the floor in a long line. Just then the old one saw Arthur. 'Stop,' he ordered coarsely. 'You are coming, of course?' 'I'm not certain,' said Arthur, undaunted. 'The roof may not come down yet.' 'Well,' said the angry old rat, 'we can't wait for you to join us. Right about face. March!'

Arthur stood and watched them hurry away. 'I think I'll go tomorrow,' he calmly said to himself, 'but then again, I don't know; it's so nice and snug here.'

That night there was a big crash. In the morning some men—with some boys and girls —rode up and looked at the barn. One of them moved a board and he saw a young rat, quite dead, half in and half out of his hole. Thus the shirker got his due."

APPENDIX D – FREEMAT CODE

```
%Initial_Processing.m
%Takes .wav audio files of all 30 speakers and performs FFT & SVD on
each

for i=1:30     %process recordings for Speakers 1 through 30
    %load individual speaker
    speaker =['Person_' num2str(i) '.dat'];
    load(speaker);

    %read audio file
    [z, SR, bits] = wavread(speaker);
    time_chunk_length = 4096;    % number of samples
    overlap = 0.9;     %90 percent overlap

    %calculate number of time segments and size of spectra matrix
    num_of_time_chunks = floor((length(z)-time_chunk_length*overlap)/
    (time_chunk_length*(1-overlap)));
    spectra = zeros(num_of_time_chunks, time_chunk_length/2);

    %process each time segment
    for i=1:num_of_time_chunks
      time_chunk = z(((i-1)*time_chunk_length*(1-overlap)+1):((i-1)*
       time_chunk_length*(1-overlap)+time_chunk_length));
      %fast fourier transform of  time segment
      spectrum = abs(fft(time_chunk));
      spectrum = spectrum/sum(spectrum);     %normalize spectrum
      %add spectrum to total spectra matrix
      spectra(i,:) = spectrum(1:time_chunk_length/2);
    end

    %preform singular value decomposition
    [U,S,V] = svd(spectra);
    values = U*S;
    vectors = V';

    %save data for each speaker
    save_name = ['Person_' num2str(i) '_decomp.dat'];
    save(save_name, 'values', 'vectors', 'spectra')

end
```

```
%Combine_Vectors.m
%combines first 50 principal vectors from each speaker and performs
SVD on total set


all_vectors = [];     %initialize blank matrix
for i=1:30     %load all speakers
    speaker =['Person_' num2str(i) '_decomp.dat']
    load(speaker);
         %add first 50 principal vectors from speaker to total set
    all_vectors = [all_vectors; vectors(1:50, 1:2048)];
end

     %perform SVD on total set
[U,S,V] = svd(all_vectors);
values = U*S;
vectors = V';

     %save vectors from SVD
save final_vectors.dat vectors
```

```
%Value_Compute.m
%Computes principal values for each speaker based on previously
defined principal vectors

load final_vectors.dat;     %load final vectors

     %compute for all 30 speakers
for i=1:30
    speaker =['Person_' num2str(n) '_decomp.dat']
    load(speaker);
         %calculate values
    values = spectra*inv(vectors);
         %save values for each speaker
    save_name = [speaker '_Values.dat'];
    save(save_name,  'values');

end
```

```
%Values_Database.m
%Computes averages and standard deviations of values for each speaker
%and stores in matrix databases

     %initialize database matrices
v_average_DB = [];
v_stdev_DB = [];
     %compute for all 30 speakers
for i=1:30
    speaker =['Person_' num2str(n) '.dat_Values.dat'];
    speaker=speaker
    load(speaker)
         %calculate averages and standard devations
    v_averages = sum(values)/size(values, 1);
    v_stdevs = std(values);
         %add to databases
    v_average_DB(n,:) = v_averages;
    v_stdev_DB(n,:) = v_stdevs;

end
     %save databases
save('value_DB.dat', 'v_average_DB', 'v_stdev_DB');
```

```
%Algorithm_1.m
%Computes sum of weighted z-scores comparing principal values to make
prediction of which speaker is talking

     %load values database and final principal vectors
load value_DB.dat;
load final_vectors.dat;

for i=1:30
    speaker =['Person_' num2str(n) '_Values.dat'];
    speaker=speaker
    load(speaker)
         %initialize blank people vector
    people = zeros(1,size(values,1));
         %predict speaker for each time segment
    for i=(1:size(values,1))
        measure = sum(repmat(sum(abs(v_average_DB)), [30,1])'.*
        abs((((repmat(values(i,:), [30,1]) - v_average_DB)./
        v_stdev_DB)')));
        [measure_min, person] = min(measure);

        people(1,i) = person;
    end
         %calculate percentage of correct predictions
    num_correct = 0;
    for i=1:size(values, 1)
        if (people(i) == n)
            num_correct = num_correct + 1;
        end
    end
    percent_correct = num_correct / size(values, 1)
         %save results
    save_name = ['Person_' num2str(n) '_Results_1'];
    save(save_name, 'people');
end
```

```
%Algorithm_2.m
%Computes sum of unweighted z-scores comparing principal values to
make prediction of which speaker is talking

     %load values database and final principal vectors
load value_DB.dat;
load final_vectors.dat;

for i=1:30
    speaker =['Person_' num2str(n) '_Values.dat'];
    speaker=speaker
    load(speaker)
         %initialize blank people vector
    people = zeros(1,size(values,1));
         %predict speaker for each time segment
    for i=(1:size(values,1))
        measure = sum(abs((((repmat(values(i,1:30), [30,1]) -
        v_average_DB(:, 1:30))./v_stdev_DB(:, 1:30))')));
        [measure_min, person] = min(measure);
        people(1,i) = person;
    end
         %calculate percentage of correct predictions
    num_correct = 0;
    for i=1:size(values, 1)
        if (people(i) == n)
            num_correct = num_correct + 1;
        end
    end
    percent_correct = num_correct / size(values, 1)
         %save results
    save_name = ['Person_' num2str(n) '_Result_2'];
    save(save_name, 'people');
end
```