Controlling Performance in Voice Conversion with Probabilistic Principal Component Analysis

A THESIS
SUBMITTED ON THE FOURTEENTH DAY OF MAY, 2004
TO THE DEPARTMENT OF
ELECTRICAL ENGINEERING AND COMPUTER SCIENCE
OF THE GRADUATE SCHOOL OF
TULANE UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE
BY

MARK MCMAHON WILDE

APPROVED:

ANDREW MARTINEZ Ph.D.
CHAIR

UVAIS QIDWAI Ph.D.

SERGEY DRAKUNOV Ph.D.

To my family and grandparents:

I dedicate my work to you, and it is in loving memory of my grandfather Joseph Raymond McMahon, Sr., that I dedicate it. Without your love and support throughout my life, I wouldn't be who I am today; nor would I have pursued my university studies this far.

Acknowledgement

I am grateful to Dr. Martinez for his brilliant advice and support during my work on this thesis. His insight into the domain of speech and signal processing and the problem of voice conversion has been a great asset in finishing this work. His deep knowledge of each model that I have used in the voice conversion system and his ability to quickly provide both philosophical and theoretical answers to my questions has proved invaluable. I thank the other two members of my committee, Dr. Qidwai and Dr. Drakunov for answering questions and taking the time to assist me with my work. In addition, the EECS department at Tulane has supported me financially for the past two years of my Master's work so I extend my warm appreciation. To my colleagues, Arman, Carr, Monika, Nick, Rafi, and Vipin, it's been wonderful being your friend for this time, and I hope that we see each other at future academic events. I thank all of those who participated in the subjective listening tests, especially Claire for transcribing the results.

I am indebted to Ben Gillett and Simon King from the University of Edinburgh for providing me with a base system for voice conversion. This system helped tremendously in developing and benchmarking my methods against previous work. I appreciate the following people for freeing their software and speech data:

- Ian Nabney from Aston University for the NETLAB toolbox [47] which provided algorithms for the Guassian Mixture Model and Probabilistic Principal Component Analysis.
- Mike Brookes from Imperial College for his Voicebox software for speech processing [11].
- Malcolm Slaney from IBM for his implementation of Dynamic Time Warping in the Auditory Toolbox [64].
- John Kominek and Alan Black from Carnegie Mellon for the Arctic speech corpus [34].

Lastly, I thank my family and friends for helping me with both moral support and love. You are the ones who mean the most to me, and I will always hold you dear to me.

MARK WILDE

Tulane University

Contents

List of Figures

Chapter 1

Introduction

SPEECH is arguably the most important mode of communication among humans. It is the way in which we express emotion, creative ideas, experiences, and knowledge. Our ability to form expeditious, concise structures with language and grammar separates us intellectually from all other species on our planet. This ability to convey pertinent information has historically ensured the continued advancement of the human race.

Of utmost importance in this communication is the speaker with whom we are communicating. Our ears' acuteness in perceiving the identity of the speaker has been the subject of research for some time now [37, 42]. When we hear a familiar speaker's voice, we make several psychological judgments about this person based on previous encounters.

- We grasp a feeling of this person's personality.
- We determine if we might be interested in what this person will talk about.
- We may prejudge the accuracy of their assertions.
- We determine if this person is trustworthy.

The ability to modify the characteristics of a speaker's voice to sound like another speaker's voice allows us to alter any of the above prejudgments of a prospective listener. This modification of a speaker's characteristics is called *voice conversion*.[1] Voice conversion is a powerful technology, and we now list some relevant applications.

[1] Voice conversion is also known in the literature as *voice morphing* [49, 79] and *voice transformation* [21], but we refer to it as voice conversion since original work on the topic refers to it as such [2].

1.1 Applications

Voice conversion is useful for many industries including the speech processing, medical, music, movie, military, intelligence, and security communities. For civilian applications of voice conversion technology, previous research has mentioned the concept of a *voice font* [75] that represents the salient features of a speaker's voice. We can imagine the associated digital rights for obtaining permission to use a certain speaker's voice font just as we license ideas with patents and marketing catch phrases with trademarks.

1.1.1 Intelligence and Military

Voice conversion technology is crucial for national security, and is a subset of technologies developed for information warfare. Voice conversion can be helpful in extracting information from an enemy such as a terrorist by mimicking the enemy's leader. In critical situations such as moments before an attack, any previously unknown information could be useful in thwarting the attack. In addition, it might be possible to persuade the enemy to complete some other task rather than the original intended task.

Similar applications are available to the military if they were able to intercept any of the enemy's communications. If they could identify the leader and communicate without being identified, they could likewise attempt to persuade the enemy. In both of these cases, most likely only limited training data for the voice conversion system is available. Thus, under these circumstances, it is crucial that the system still be able to synthesize high quality output.

1.1.2 Radio

Once a radio personality has established a listener base, usually these listeners can identify that radio personality's voice easily. If the radio personality is unavailable for one day, a substitute host is necessary. In some situations, listeners might not listen to the show that day if they didn't hear the familiar radio personality. Voice conversion solves this problem by transforming the substitute host's voice to sound like the radio personality's. Hopefully, this substition might increase the chances of maintaining the usual listener base. If the voice conversion system's output is convincing, the listeners might not be able to determine that it is really a different host (although perhaps they could decipher the difference from *what* the speaker says).

Another application is for the radio advertising industry. Usually companies hire a person with a famous voice to endorse a product over the radio. Often it may be expensive to fly this person to a recording studio to record the advertisement. Instead, this famous person could sell the rights to his or her voice font for use in the advertisement. This arrangement significantly decreases the overall cost of production if temporary rights to the voice font are reasonably priced.

1.1.3 Movies

Often, when popular movies are exported to foreign countries, a company in the foreign country creates a "foreign" version of the film. These "foreign" versions usually feature replacement speakers fluent in the foreign language and the original language of the movie. The replacements speak a translated version of the original performer's part with an effort to mimic the original performer so that the movie sounds more natural to the foreign listeners. This re-recording is called *dubbing*. Voice conversion allows the foreign speaker to sound like the original performer's voice. Thus, the original performance is available in any foreign country, and the intended audience receives a more natural version of the film without subtitles.

In animated films, it is necessary for a famous person to record all parts for the voice of an animated character. With voice conversion, this time-consuming task can be completed by someone who is not as expensive to hire, and that person's voice can be transformed to the famous person's after acquiring the digital rights to the famous person's voice font.

Television companies often want to broadcast popular movies after they have played for some time in theaters. Often, these popular movies contain several expletives which are not suitable for broadcast to the public. The television companies usually hire someone to substitute the expletives with more "TV-friendly" words. With voice conversion, it is possible to make the substituted words sound more like the original performer so that the transition is more natural.

With voice conversion, we can also use samples of a deceased famous person's voice to create new speech which that person might never have spoken. This technology can be integrated into future media to remind us of the deceased person and the older era in which that person lived. In *Forrest Gump* [80], if voice conversion technology had been readily available, producers could have been more creative with some of the historic scenes featuring Forrest dubbed in with famous figures of the 20^{th} century.

1.1.4 Music

Voice conversion technology is beneficial to the music industry because it allows someone to sound like a famous singer; this technology can be placed in a karaoke box. Companies such as Yamaha are already considering voice fonts as an option for music synthesizers and karaoke machines [1].

Also, to enhance the live performance of music, a singer can use voice conversion to imitate the sound of a trumpet or a guitar; or the trumpet player can make his or her instrument sound like the voice of the singer. One can imagine how interesting a concert would be if these complex interactions and mimickings were simultaneously occurring.

In addition, voice conversion techniques have significant impact on the area of multichannel audio synthesis [43, 44]. This research focuses on transmitting audio acquired from several microphones at a concert. Using voice conversion techniques, these researchers formulate a model for synthesizing the extra channels of a recording from two base microphones. Then, they transmit the two base signals and the model for the other channels over the network. At the receiving end, the multichannel recording is reconstructed using the two base recordings and the model for the other channels.

1.1.5 Speech Processing

Automatic speaker identification, text-to-speech synthesis, and transmission of speech benefit from advancements in voice conversion. A colossal research effort has been funded for automatic speaker identification systems [13, 56] with the intent of integrating them as a biometric feature into authentication systems. The objective of an automatic speaker identification system is to determine the true identity of a speaker perhaps for clearance to a classified system. Voice conversion and speaker identification have an interesting symbiotic relationship that is analogous to the relationship between a computer cracker and a computer security expert. As crackers force security experts to be aware of the latest techniques that they are using and vice versa, voice conversion systems truly test the performance of automatic speaker identification systems. If someone is able to mimic a speaker effectively, he or she gain unauthorized access thus rendering the speaker identification system useless. Pellom investigated these applications in [53]. For the other direction of the relationship, speaker identification systems can test the performance of voice conversion systems as done previously in [3, 61].

A common, tedious problem in speech processing is the creation of speech databases for text to speech (TTS) synthesis. Each speaker in the database must speak a common set of utterances for training the speech synthesizer. With voice conversion, we can simply record one base speaker's utterances; then, we can record a much smaller subset of the utterances for every other speaker. We can train the voice conversion system with this smaller subset and synthesize speech with the voice conversion system. This new system alleviates much of the cost associated with producing synthetic speech. In earlier work, Kain [29] integrated his system of voice conversion into the open source Festival Speech Synthesis System [10].

Voice conversion can save some cost for the transmission of speech. Before transmitting speech, we can extract the salient features that identify a speaker and create a model of this speaker. We can then transmit the speaker model followed by the *normalized* speech over a channel. The normalized speech theoretically has less information so we can encode it in a more efficient manner. With a voice conversion system, the receiving end then reconstructs the speech using the speaker information and the normalized speech. Normalization of speech can also improve the robustness of speaker recognition systems.

1.1.6 Medical

Over 2.5 million Americans currently suffer from some sort of neurological impairment, and a huge portion of these suffer from *dysarthria* — a group of speech impediments which affect the production of speech [19]. Due to a malfunction in the mechanisms that contribute to the production of speech, dysarthric individuals' speech is unintelligible. This disability can affect these individuals' socioeconomic status in life. Currently, research is underway to see how voice conversion technology could be used in a wearable device to make dysarthric speech more intelligible [27, 77].

1.1.7 Internet Gaming or Chat

Individuals who are participating in on line gaming or chat activities may want to disguise their voice to protect their privacy; or, perhaps they may want to provide some humor for others by using voice conversion technology to sound like a famous person.

1.2 General Voice Conversion System

In voice conversion, we map the acoustic features of a *source* speaker to those of a *target* speaker. Figure 1.1 illustrates the typical model for a voice conversion system. We collect speech in a parallel training corpus[2] from both the source and target speaker for use in training the model. After training is complete, we predict what a target speaker sounds like using the information from the new speech of the source speaker.

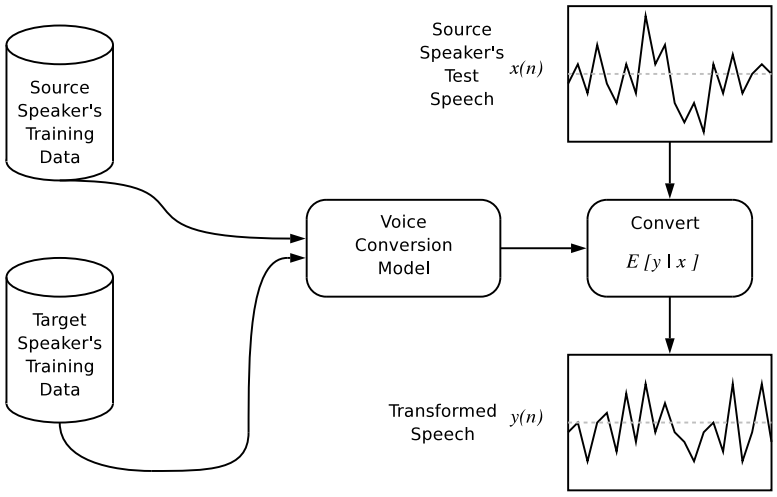

Figure 1.1: Voice Conversion Model

A typical voice conversion system consists of seven components: the speech corpus, time alignment of the speech, analysis, training, voice mapping, synthesis, and post-processing.

Speech corpus — We must collect the same speech from a set of speakers. This speech should contain an even distribution of the phonemes of the speakers' language so that we can model a variety of sounds; this variety improves the quality of the voice conversion system. In this work, we use the Arctic Corpus for training and testing [34].

Alignment of phonemes — We should have a robust method for time aligning the source and target speaker's speech on the order of millisecond by millisecond. Exact time alignment before training is necessary for optimal performance of this particular voice conversion system.

Analysis — We must determine the relevant acoustic features to use in training the system. These features should be able to represent a large portion of speech with only a few

[2] We use a *parallel* speech corpus in which the speakers say the same sentences. Recent studies of voice conversion [35, 45] have focused on converting using a non-parallel training corpus in which the speakers' utterances are different.

parameters. We discuss the features we use for representing speech in §2.2 and §2.4. Examples of typical features are the pitch or energy of a short segment of speech.

Training — We must figure out an appropriate model for training the voice conversion system.

Voice Mapping — We perform the mapping of the source speaker's features to the target speaker's. Usually, this mapping is a statistical expectation. Training and mapping are the crucial items that we focus on in this thesis while also being the subject of most research. We mention a few of the previously employed techniques in §2.7

Synthesis — We must synthesize the transformed features into high quality speech.

Post-processing — After synthesis, we might perform some final processing of the signal to improve its quality.

1.3 Thesis Outline

The following chapters are summarized below:

Chapter 2 introduces the mathematical model for human speech production and presents the features with which we represent speech.

Chapter 3 is an introduction to a previous model for voice conversion. We then discuss the present research, a method of modeling the probabilistic acoustic space of both speakers with a mixture of Probabilistic Principal Component Analyzers and performing conversion with this model.

Chapter 4 presents the objective and subjective results of our voice conversion system. It highlights the tradeoff of performance versus complexity that our system offers.

Chapter 5 concludes and also discusses future opportunities for research in voice conversion.

Chapter 2

Speech Model

IN ORDER to map the vocal tract characteristics from a source speaker to a target, we must first understand how humans produce speech. We present this description in the first section followed by the *source-filter* mathematical model of human speech production. We then discuss two features — *linear predictive coefficients* and *line spectrum frequencies* — which model a short segment of speech. Lastly, we discuss how to apply this mathematical model to voice conversion and finish by listing the previous methods researchers have used for voice conversion.

2.1 Physiology

The main organs of the human speech production system are in Figure 2.1. Listed in the order by which speech is produced, these organs are the lungs, the trachea, the vocal cords or *glottis*, the pharyngeal cavity, the oral cavity, and the nasal cavity. Usually, we consider both the pharyngeal and oral cavity as the *vocal tract*.

We produce speech by first compressing air in our lungs. This air is then pushed out of the lungs and into the trachea. If the sound to be produced is resonant (commonly called *voiced*), the vocal cords vibrate in an almost periodic sense thus "exciting" the air. After this excitation travels through the vocal cords, the vocal and nasal tracts' shape then determines the resultant sound output. The shape of the jaws, tongue, teeth, and lips (commonly called the *articulators*) affects the shape of both the vocal tract and nasal tract. At the lips, a final impedance can affect the sound. Examples of voiced sounds in American English are the sounds produced when pronouncing the vowels *a*, *e*, *i*, *o*, and *u*.

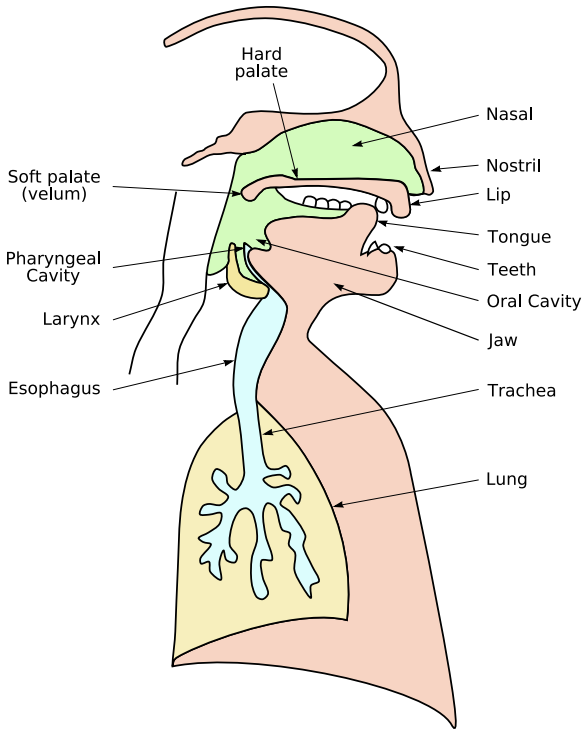

Figure 2.1: Human Speech Production

We produce *unvoiced* sounds by forcing air through the lungs and forming a constriction at some point in the vocal tract. Unvoiced utterances are noisy in nature; examples are the sounds produced when saying the American English consonants *s* and *f*.

This speech production process can produce three additional sounds which are a combination of voiced and unvoiced sounds.[1]

Mixed — These sounds are a mixture of both voiced and unvoiced sounds. Examples are the sounds created when pronouncing the letter *z* or *g*.

Plosive — We produce *plosive* sounds by forming a constriction at some point near the front of the vocal tract and then releasing the air pressure built up behind the constriction. Examples are the sounds of *p* or *t*.

Whisper — We produce a *whisper* by puffing air all the way from the lungs to the lips without any excitation at the vocal cords.

We give a simple, classic block-level description of the speech production process in Figure 2.2. The air travels from the lungs past the vocal cords and then splits after the

[1] We are restricting the possible sounds to those found in American English. Other interesting languages have more possible sounds.

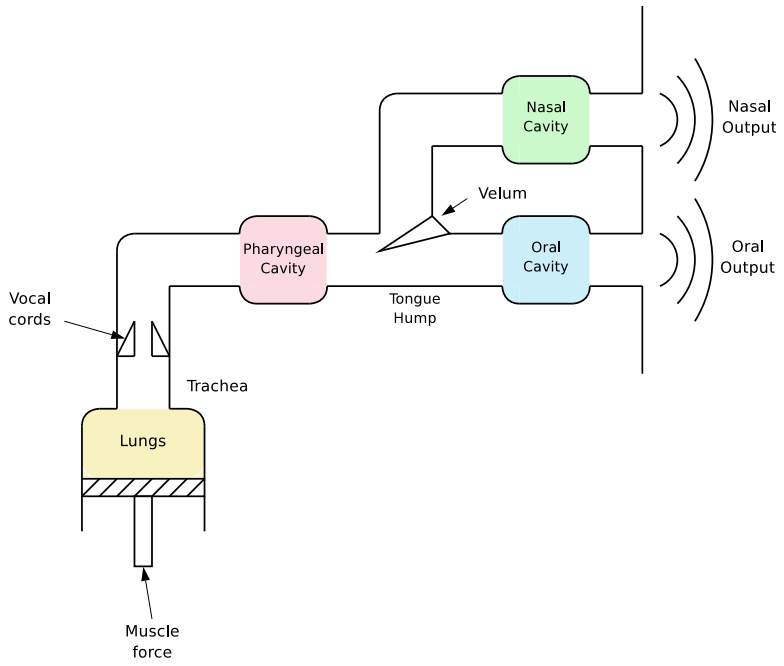

Figure 2.2: Human Speech Block–Level Model (adapted from [54])

pharyngeal cavity. It then travels through the oral and nasal cavities to reach its output at both the lips and nostrils.

2.1.1 Speaker-specific Characteristics

The shape of the vocal tract not only determines the sound produced but also gives cues to the identity of the speaker because of its distinct shape. Three key features of speech — *segmental*, *suprasegmental*, and *linguistic* — aid in the process by which humans identify a speaker.

Segmental — Segmental features refer to the quality or tone of a person's voice. These features correlate with the length and shape of a person's vocal and nasal tract as well as the excitation produced at the vocal cords.

Suprasegmental — These features are the pitch and prosodic features of a person's voice. The way in which a person accents certain words can help distinguish that person. A native of New Orleans usually pronounces Tulane as (Too´ lān) as where most people from the North say (Teh lān´).

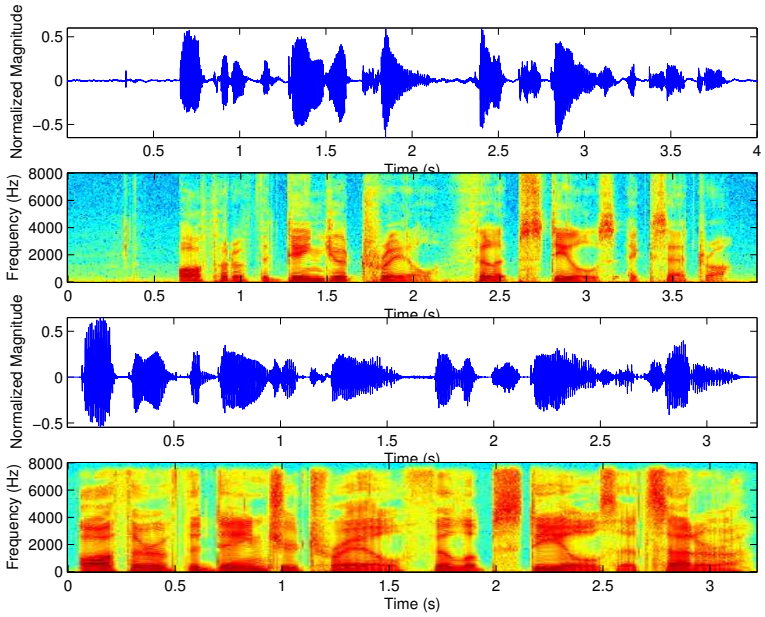

Figure 2.3: Speech signals and spectrograms for Speaker A (top) and Speaker B (bottom)

Linguistic — A person's choice of words can reveal that person's identity. For instance, if a sentence contains phrases such as "It's awesome, baby!" or "Serendipity, baby!", we identify the speaker as either the sports announcer Dick Vitale or some impersonator of him. We do not consider linguistic cues in this thesis as they are in the domain of natural language processing.

It is easy for a speaker to change suprasegmental cues by slowing or speeding up the rate of speech or by mimicking another speaker. Therefore, these cues assist to a certain extent in identifying a speaker, but the most important cues are segmental because they are related to the shape of his or her vocal and nasal tract. It is almost impossible for a speaker to change this shape unless they are talented or have surgical modifications.

When producing speech, our articulators change shape over time to produce the different sounds of a language. The acoustic filter produced by the articulators has peaks in the spectrum which are called the *formant* frequencies because they form the distinct sound produced. Thus, the formant frequencies and their bandwidths are highly correlated with the identity of a speaker. In Figure 2.3, we present the time-domain waveform of two different speakers saying "Author of the Danger Trail, Philip Steels, etc." with their spectrograms beneath. We see how the magnitude of each formant changes over time with respect to the different sounds uttered.

2.2 Source-Filter Model

Speech is a highly *non-stationary* process; i.e., the statistics of the underlying signal vary with respect to time. But, for short segments of time, speech is either quasi-periodic or noisy so we can assume that speech is *wide-sense stationary* — the first and second order statistics remain constant — during these short segments. Thus, for these short segments of time, we can form a tractable model to represent the speech production process as described in [41].

A powerful method for modeling a discrete-time system for a short segment of time is the *auto-regressive moving average* (ARMA) model given by

$$s_n = -\sum_{k=1}^{p} a_k s_{n-k} + G \sum_{l=0}^{q} b_l u_{n-l} \text{ where } b_0 = 1 \tag{2.1}$$

where we model the discrete-time signal of interest s_n as a linear combination of its past values and also of present and past values of an external input u_n. The system transfer function $H(z)$ is thus

$$H(z) = \frac{S(z)}{U(z)} = G \frac{1 + \sum_{l=0}^{q} b_l z^{-l}}{1 + \sum_{k=1}^{p} a_k z^{-k}} \tag{2.2}$$

When processing speech, we usually restrict this powerful model simply to be an *auto-regressive* (AR) model.

$$s_n = -\sum_{k=1}^{p} a_k s_{n-k} + G u_n \text{ where } G u_n = e_n \tag{2.3}$$

For simplicity, we only allow one input Gu_n to the system and no other combinations of past input values. We can think of this input as the excitation e_n described in §2.1 that is produced by pushing air through the lungs and beyond the vocal cords. When the vocal cords vibrate quasi-periodically for voiced sounds, we model e_n as a periodic pulse train. When the vocal cords do not vibrate and some constriction is created in the vocal tract for unvoiced sounds, we model e_n as Gaussian white noise. This model is depicted in Figure 2.4 where the model switches randomly between the voiced and unvoiced source.

The transfer function of the AR model is then

$$H(z) = \frac{S(z)}{E(z)} = \frac{1}{1 + \sum_{k=1}^{p} a_k z^{-k}} \tag{2.4}$$

We notice several things about this simplified model. First, it is a digital filter representing the shape of the vocal tract. The articulators determine the shape of this acoustic filter for a short

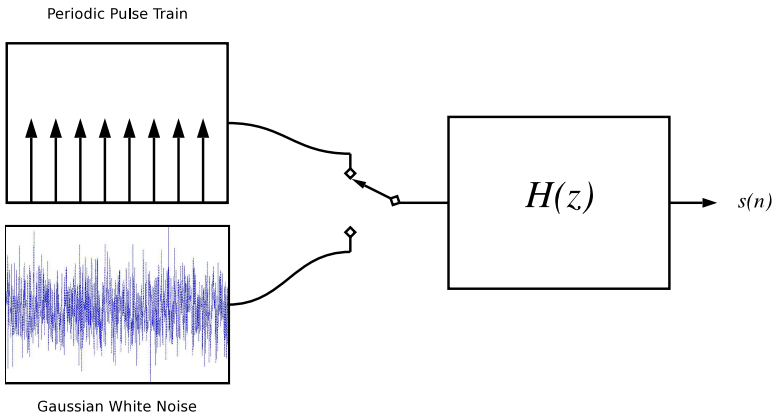

Figure 2.4: Mathematical Model of Speech Production (adapted from [41])

segment of time. Secondly, it is an *all-pole* digital filter because z exists only in the denominator; i.e., no value of z makes $H(z) = 0$, but several values make $H(z) = \infty$. Lastly, in order for the filter to be *stable*, all of these poles must be inside of the unit circle, i.e. $|z| < 1$. One last observation is that it is possible to find the spectrum of the excitation by "inverse filtering" the signal s_n as in the following equation.

$$S(z)\left(1 + \sum_{k=1}^{p} a_k z^{-k}\right) = S(z)A(z) = E(z) \tag{2.5}$$

It is useful in voice conversion to recover the excitation via this inverse filter.

Thus, with the model from Equation 2.4, we have a concise, parametric form for representing short segments of speech with p coefficients a_k. These a_k coefficients are the *linear predictive coefficients* (LPCs) since they predict the next value of the signal from past values of the signal and the present value of the excitation. The motivation for using the AR model in speech is the tractability of computing the a_k coefficients with Levinson's algorithm compared to the difficulties inherent in estimating an ARMA model's parameters.

2.3 Linear Prediction and the Levinson Algorithm

In this section, we provide a brief description of Levinson's algorithm which he formulated in 1947 [40] with performance improvements made by Durbin [18] and Robinson [58] for the specific problem of a time-series as we have with our speech model in Equation 2.3. Levinson's algorithm runs in $O(p^2)$ time compared with older methods which run in $O(p^3)$ time. For this reason, the linear predictive model's popularity soared.

In order to present Levinson's algorithm, we consider the time-domain speech model from Equation 2.3 under a different light. Suppose that we now try to *approximate* the signal s_n as a linear combination of its past values.

$$s_n \approx -\sum_{k=1}^{p} a_k s_{n-k} = \hat{s}(n) \tag{2.6}$$

We can describe the *residual* term as the difference between the approximation on the right hand side and the signal s_n as follows

$$e_n = s_n - \hat{s}_n = s_n + \boldsymbol{a}^T \boldsymbol{s} \tag{2.7}$$

where, for simplicity of notation, we now consider $\boldsymbol{a}$ to be a vector of the p LPCs and $\boldsymbol{s}$ to be the past p values of s_n. In order for this approximation to be reasonable, we should minimize the mean energy of the residual term e_n for a short segment of time. The energy of the residual e_n is given by

$$\begin{aligned} e_n^2 &= s_n^2 + 2s_n \boldsymbol{a}^T \boldsymbol{s} + (\boldsymbol{a}^T \boldsymbol{s})^2 && (2.8) \\ &= s_n^2 + 2s_n \boldsymbol{s}^T \boldsymbol{a} + \boldsymbol{a}^T \boldsymbol{s}\boldsymbol{s}^T \boldsymbol{a} && (2.9) \end{aligned}$$

and the mean energy is

$$\begin{aligned} E[e_n^2] &= E[s_n^2] + 2E[s_n \boldsymbol{s}^T]\boldsymbol{a} + \boldsymbol{a}^T E[\boldsymbol{s}\boldsymbol{s}^T]\boldsymbol{a} && (2.10) \\ &= E[s_n^2] + 2\boldsymbol{\rho}\boldsymbol{a} + \boldsymbol{a}^T \boldsymbol{R}\boldsymbol{a} && (2.11) \end{aligned}$$

where $\boldsymbol{\rho}$ is the *autocorrelation* between the signal s_n and its past values $\boldsymbol{s}$, and $\boldsymbol{R}$ is the $p \times p$ autocorrelation matrix of the signal s_n. To find the coefficients $\boldsymbol{a}$ which minimize the mean energy of the residual, we set the derivative to 0.

$$\frac{\partial E[e_n^2]}{\partial \boldsymbol{a}} = 2\boldsymbol{\rho} + 2\boldsymbol{R}\boldsymbol{a} \tag{2.12}$$

We find that the set of optimal coefficients $\boldsymbol{a}^*$ are given by

$$\boldsymbol{a}^* = -\boldsymbol{R}^{-1}\boldsymbol{\rho} \tag{2.13}$$

This optimal solution for $\boldsymbol{a}$ is well known as the Wiener-Hopf solution, and we can use Levinson's algorithm to solve for $\boldsymbol{a}$ recursively. The algorithm finds the p^{th} order solution of $\boldsymbol{a}$ by using the $(p-1)^{th}$ order solution. We give a brief description of Levinson's algorithm below.[2]

$$
\begin{aligned}
E_0 &= \rho_0 && (2.14)\\
\kappa_i &= -\frac{\rho_i + \sum_{j=1}^{i-1} a_j^{(i-1)} \rho_{i-j}}{E_{i-1}} && (2.15)\\
a_i^{(i)} &= \kappa_i && (2.16)\\
a_j^{(i)} &= a_j^{(i-1)} + \kappa_i a_{j-i}^{(i-1)} && (2.17)\\
E_i &= (1-\kappa_i^2) E_{i-1} && (2.18)
\end{aligned}
$$

Solving for the above equation is typically an $O(p^3)$ operation, but Levinson's recursive algorithm has only $O(p^2)$ complexity by exploiting the symmetric, Toeplitz nature of $\boldsymbol{R}$.

2.4 Line Spectrum Frequencies

If we consider the inverse filter $A(z)$ from Equation 2.5, we can form an alternate parameterization of the LPCs which has several useful properties. We split the order p polynomial described by $A(z)$ into two separate $p+1$ order polynomials, the symmetric polynomial $P(z)$ and the anti-symmetric polynomial $Q(z)$ where

$$
\begin{aligned}
P(z) &= A(z) + z^{-(p+1)} A(z^{-1}) && (2.19)\\
Q(z) &= A(z) - z^{-(p+1)} A(z^{-1}) && (2.20)
\end{aligned}
$$

and with substitution, we write the above equations as

$$
\begin{aligned}
P(z) &= 1 + \sum_{k=1}^{p} (a_k + a_{p+1-k}) z^{-k} + z^{-(p+1)} && (2.21)\\
Q(z) &= 1 + \sum_{k=1}^{p} (a_k - a_{p+1-k}) z^{-k} - z^{-(p+1)} && (2.22)
\end{aligned}
$$

So, we can easily derive $A(z)$ back from $P(z)$ and $Q(z)$ via

$$
A(z) = \frac{P(z) + Q(z)}{2} \qquad (2.23)
$$

[2] In describing the algorithm, note that ρ_i refers to the $E[s_n s_{n-i}]$ autocorrelation. It is the i^{th} entry of $\boldsymbol{\rho}$.

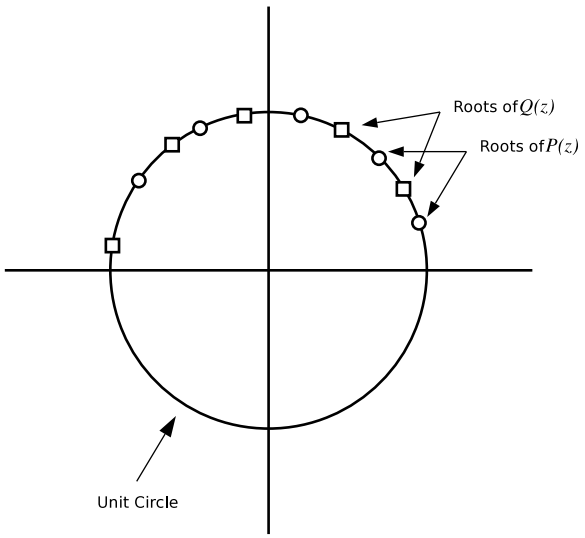

Figure 2.5: The complex roots of polynomials $P(z)$ and $Q(z)$ all lie on the unit circle.

One important property of the polynomials $P(z)$ and $Q(z)$ is that their complex roots all lie on the unit circle (shown in Figure 2.5). So, after solving for the complex roots of these polynomials, since $|z| = 1$, we can substitute $e^{j\omega}$ for z to determine each angle ω of the roots. These angles ω completely specify the roots, and they are known as the *line spectrum frequencies* (LSFs) [69, 65].

Below, we list several advantages of LSFs over LPCs for representation of a speech signal.

1. The LSFs ω_p of $P(z)$ and the LSFs ω_q of $Q(z)$ are interleaved with one another, i.e.,

$$0 < \omega_{p_1} < \omega_{q_1} < \ldots < \omega_{p_i} < \omega_{q_i} < \ldots < \pi \tag{2.24}$$

   This interleaving provides an easy way to check for stability of the filter.

2. The LSFs have good interpolation properties and quantize well because they are more evenly distributed than LPCs [51].

3. Two LSFs that are close in value correspond to a peak in the spectrum, and we can interpret this peak as a formant [16]. Thus, the LSFs correlate well with the formant frequencies that identify a speaker.

4. Because the LSFs are angular frequencies, we measure the distance between two LSFs with a Euclidean measure. In addition, we can warp these frequencies using the Bark scale to provide a perceptual enhancement.

These properties are desirable for voice conversion so we have chosen the LSFs to represent a short segment of speech. The only disadvantage is that we must compute the roots of a $p+1$ order polynomial; but, Rothweiler's method performs adequately for low orders ($p < 24$) [59].

The following simple MATLAB program can be used to calculate the LSFs from the LPCs (a is a column vector of linear predictive coefficients).

```
p       = a + flipud(a);
q       = a - flipud(a);
p_omega = angle (roots (p));
q_omega = angle (roots (p));
```

2.5 Feature Extraction

In the previous two sections, we have described two features, LPCs and LSFs, that we can extract from a waveform. Under the assumptions of our model, we can only extract these features for short segments of time when the first and second order statistics of the waveform do not vary with respect to time. Thus, we should not use a fixed-size frame for the short time segment. We must extract features only during periods of stationarity when the vocal cords are opening and closing. This idea is known as *pitch-marking* and these periods are *pitch periods*. Thus, we should set our frame's length to the size of each pitch period. This method of extraction is called *pitch-synchronous frame-based windowing*.

2.6 Applying the Source-Filter Model to Voice Conversion

Having established a simple model for speech, we now discuss how to apply it to voice conversion. In Figure 2.6, we present the typical application of the source-filter model to voice conversion. We notice that speaker A's vocal tract is a little bit longer than speaker B's tract while speaker B's vocal cords are larger than speaker A's. Thus, voice conversion is highly nonlinear because of the variety of shapes for a speaker's vocal tract and vocal cords.

First and foremost, we must convert the characteristics of the source speaker's vocal tract to the target speaker's. This step involves learning the nonlinear mapping of the features

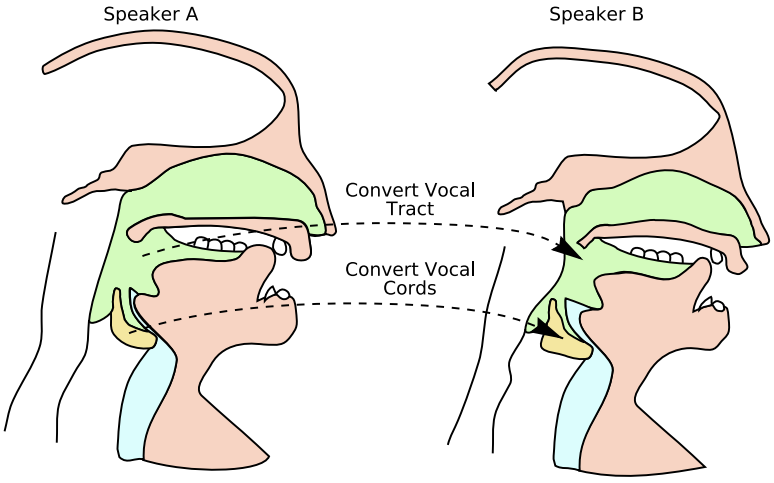

Figure 2.6: Voice Conversion of Speaker A to Speaker B using the Source-Filter Model

that represent the vocal tract. In the next section, we discuss several previous methods for learning this mapping.

In order to have high quality conversion, we must convert both the excitation at the vocal cords and the spectral properties of the vocal tract. Early research used just the excitation of the source speaker with converted spectral features, but Arslan [5] later determined that performance improved significantly by mapping the excitation as well. Another study by Kain and Macon [29] determined that the excitation played an important role in identifying a speaker. When using the source speaker's residual, 52% of the subjective results were closer to the target speaker than the source speaker; but, when using the target speaker's residual, 100% of the subjective results were closer to the target speaker.

Researchers have designed various methods for converting the source speaker's excitation using both codebooks and methods similar to spectral conversion techniques. In Kain's thesis [28], he devised a new technique called *residual prediction* to predict the target speaker's excitation from the LP spectrum. This method is somewhat counter-intuitive since the LP model is based on the filter and excitation being independent of one another, but empirical results from Kain's thesis demonstrate that the method works well.

2.7 Research Background

Most existing voice conversion systems employ the methodology above and that described in §1.2. Below we give advances in the significant fields of the alignment of phonemes and spectral conversion.

2.7.1 Previous Methods for Time Alignment of Phonemes

Researchers have either used *dynamic time warping* or *Hidden Markov Modeling* [54] for time alignment of speech in a parallel corpus. Most studies on voice conversion use dynamic time warping for alignment because of the simplicity of the algorithm and because they focus mainly on new methods for converting the vocal tract features and the excitation. In early work [2], Abe et al. used dynamic time warping of the source and target speech for alignment; they did not mention which features they used. Brugnara [12] and Arslan [5] both performed time alignment with Hidden Markov Modeling. Stylianou [68], Kain [28], Orphanidou [49], and Gillett [21] all used dynamic time warping for time alignment.

2.7.2 Previous Methods for Spectral Conversion

Most researchers have contributed ideas for spectral conversion because it is believed that the performance of the resulting conversion is highly correlated with spectral conversion. Early work in [15, 36, 71] determined that the most relevant features for voice conversion are the formant frequencies, formant bandwidths, spectral tilt, and the glottal waveforms.

Codebook Mapping

Results from speaker adaptation and normalization methods [62, 81] were used in one of the first voice conversion systems developed [2]. Abe's work used vector-quantized codebooks to map the acoustic features. The mapping was simply a linear combination of the source speaker's codebook; histograms determined the weights to apply to each source codebook entry. Arslan [4] extended this work by mapping not only the LSFs, but also the excitation; in addition, he improved the method by which the transformation weights were estimated. He named his method for updating the weights *Codebook Weight Update by Gradient Descent*; it is an optimization technique that iterates until the energy of the weight errors falls below a threshold. He integrated this method into his larger framework for voice conversion called

Speaker Transformation Algorithm using Segmental Codebooks (STASC) [3]. Türk extended the STASC framework by integrating subband voice conversion using wavelets and selective pre-emphasis [75, 76]. Orphanidou [49] utilized the Generative Topographic Mapping [9, 70] in reducing the dimensionality when mapping codebooks, but she didn't benchmark her results against previous work and only showed results for one word.

Mixture Modeling

Stylianou modeled the acoustic probability space of the source speaker with a Gaussian Mixture Model (GMM) in [67, 68]. He then found the cross-covariance of the target speaker with source speaker and the mean of the target speaker using least squares optimization of an overdetermined set of linear equations. In his work, he demonstrated the theoretical superiority of the GMM by showing that vector-quantization methods for voice conversion are a special case of the GMM in which only the mean of a cluster is mapped. Toda [74] implemented the GMM algorithm for voice conversion within their STRAIGHT analysis-synthesis framework [32]. Kain extended Stylianou's work by modeling the joint probability density function of both the source and target speakers [28, 29, 30]. Although this method increases the complexity during EM training, it obviates the need to perform the least squares optimization as with Stylianou's method. Modeling the joint probability density allows the system to capture all possible correlations between the source and target speaker's spectrum. Gillett [21] pushed Kain's work further by rejecting poorly matched data before training with the EM algorithm and by also rejecting unlikely data after estimation. He called these techniques *pre-GMM estimation rejection* and *post-GMM estimation rejection* and using these techniques increased the performance index of his system.

Other Methods for Spectral Conversion

In [38, 39], Lee used an orthogonal vector space transformation to convert voiced speech and a non-linear neural net predictor to model the excitation; he converted the excitation using mapping codebooks. In [6], Baudoin investigated many of the then current techniques for voice conversion including codebooks, GMMs, neural networks, and linear multivariate regressions. Results were that full covariance GMMs objectively performed the best by having the lowest distance between the converted spectrum and the original target speaker's spectrum. Gutiérrez-Arriola mapped voice quality features by a linear regression

method in [22, 23]. Watanabe used radial basis functions for performing the spectral mapping [78], and Narendrath used neural networks [48] with success. Salor [61] employed a least mean square adaptive filtering technique to "filter" the target speaker's features from the source speaker's.

Although mapping with codebooks may be less expensive computationally, most researchers consider the method less robust than mixture modeling because it only estimates the probability distribution of the two speakers' features with a discrete set of vectors. Thus, this method loses much information when mapping.

2.8 Mapping the Excitation

For mapping the excitation in this thesis, we follow Kain's method of residual prediction in which we predict the excitation from the LP envelope. This method is discussed at length in both [28] and [21] so we do not address its theoretical details in this thesis.

2.9 Proposed Method

We extend the spectral mapping aspect of voice conversion by modeling the joint probability space of both speakers with a mixture of Probabilistic Principal Component Analyzers (PPCAs). Previous methods that used the GMM to model the space are constrained to only two possible selections for representing covariance structure — diagonal and full covariance matrices. With diagonal structure, the training time is quick but conversion performance is sacrificed. With full covariances, we can model the underlying second order statistics with improved conversion performance but incur the penalty of longer training time.

By modeling covariance structure with a mixture of PPCAs, we provide an entire range of covariance structure that incrementally includes more covariance information. As we incrementally include more information, results indicate that the objective performance of the system incrementally improves. Subjective listening tests also indicate that the quality of conversion for incremental amounts is not perceptually significant. Thus, we present a wider array of options for voice conversion to the end user; and the user chooses the parameters of the model to fit his or her needs.

Chapter 3

Spectral Conversion

THE OBJECTIVE of spectral conversion is to find a statistical mapping between those features of the source and target speakers which best represent their distinct vocal tracts. After finding this mapping, we filter an excitation with the transformed features. The excitation models the opening and closing of the glottis and theoretically should have no correlation with the filter that models the vocal tract [55].

In this chapter, we first discuss some pre-processing steps taken before training. We then highlight a model for representing the probabilistic acoustic space of both speakers followed by the method for voice conversion. Lastly, we discuss a different method that provides a more parsimonious model of the space with a number of resulting benefits.

3.1 Time Alignment and Pre-Estimation Rejection

Before training the voice conversion model, it is necessary to pre-process the source speaker's waveforms to align with the target speaker's. We use the common method of *dynamic time warping* (DTW) to align the waveforms [54].

First, we trim both speakers' waveforms with an algorithm that removes silence both at the beginning and the end so that the DTW algorithm has a better initial alignment. The features that we align on are the cepstral coefficients [54] and the log energy of the residual. The objective is to find the best path through the features of the source speaker and the target speaker; DTW uses a dynamic programming strategy to find this optimal path. We then use this path to select only the frames of speech which match the target speaker's speech. Thus, the DTW algorithm either copies or removes features in the source speaker's waveform. Note that in the implementation, from the Auditory Toolbox [64], we only allow slopes of $\frac{1}{2}$, 1, and

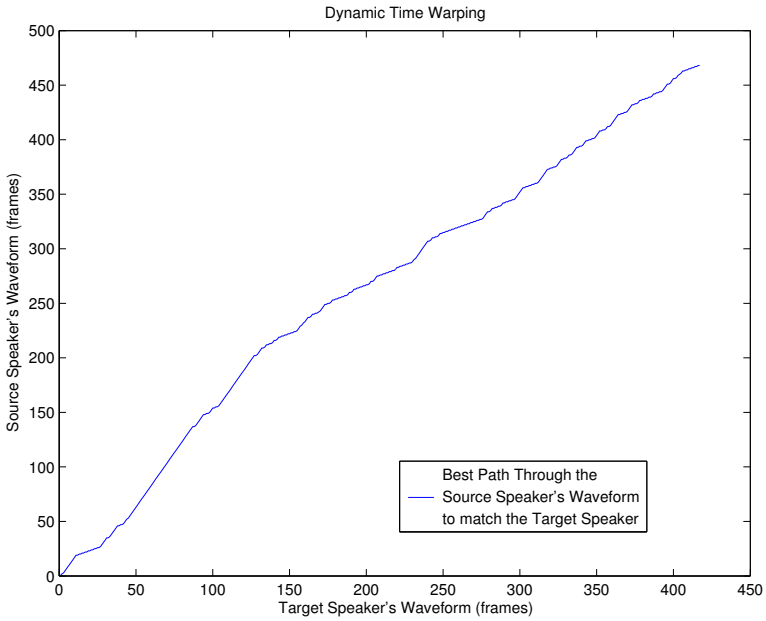

Figure 3.1: Dynamic Time Warping for Time Alignment

2; i.e., at a certain location in the grid for DTW, the next possible position could only be two spaces up and one to the right, to the top right, or two spaces to the right and one to the top. We illustrate the best path that DTW finds through the source speaker's waveform in Figure 3.1.

As an additional measure, we reject any of the poorly aligned data from the training set — a step called *pre-estimation rejection* coined by Gillett [21]. This step slightly improves the performance of the system because we only accept the best matching features. It eliminates the differences in the way that speakers pronounce the same words or voiced/unvoiced misalignments. In our case, this pre-processing step rejects on average about 42% of the training data.

3.2 Model Training

For training the voice conversion system, we model the source and target speakers' features as the vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ respectively. We predict the target vector from the source vector[1] after extracting the speakers' features from their waveforms via pitch-synchronous frame-based windowing as discussed in §2.5.

[1] In regression analysis, $\boldsymbol{y}$ is called the set of response variables and $\boldsymbol{x}$ is the set of predictor variables [31].

After feature extraction, we model the joint probability space of the source and target feature vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ for all N frames of speech. We predict the best estimate of the target vector $\boldsymbol{y}$ given the source vector $\boldsymbol{x}$ with a conditional expectation $E[\boldsymbol{y} \mid \boldsymbol{x}]$.

3.2.1 The Gaussian Mixture Model

For determining the joint probability density $p(\boldsymbol{x}, \boldsymbol{y})$, we consider the concatenation of the source and target feature vectors as the d-dimensional vector $\boldsymbol{z}$ for ease of notation.

$$\boldsymbol{z} = \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix} \tag{3.1}$$

The joint density $p(\boldsymbol{z})$ is highly nonlinear so we model it with a finite number of *locally linear* models. In a finite mixture model, the probability density function $p(\boldsymbol{z})$ is modeled as a linear combination of M basis functions or component densities $p(\boldsymbol{z}|j)$ [7, 46].

$$p(\boldsymbol{z}) = \sum_{j=1}^{M} p(\boldsymbol{z}|j)P(j) \tag{3.2}$$

Each $P(j)$ is a *mixing coefficient* or *prior* probability of component j occurring. The following constraints are imposed on the model:

$$\sum_{j=1}^{M} P(j) = 1 \tag{3.3}$$

$$0 \leq P(j) \leq 1 \tag{3.4}$$

$$\int p(\boldsymbol{z}|j)\, d\boldsymbol{z} = 1 \tag{3.5}$$

Using Bayes' Theorem, we determine the *posterior* probabilities $P(j|\boldsymbol{z})$.

$$P(j|\boldsymbol{z}) = \frac{p(\boldsymbol{z}|j)P(j)}{p(\boldsymbol{z})} \tag{3.6}$$

$$= \frac{p(\boldsymbol{z}|j)P(j)}{\sum_{j=1}^{M} p(\boldsymbol{z}|j)P(j)} \tag{3.7}$$

These posterior probabilities represent the probability that a particular component j generated the sample $\boldsymbol{z}$.

If each component density is distributed $p(\boldsymbol{z}|j) \sim \mathcal{N}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$ so that

$$p(\boldsymbol{z}|j) = \frac{1}{(2\pi)^{\frac{d}{2}}|\boldsymbol{\Sigma}_j|^{\frac{1}{2}}} e^{\left(-\frac{1}{2}[\boldsymbol{z}-\boldsymbol{\mu}_j]^T\boldsymbol{\Sigma}_j^{-1}[\boldsymbol{z}-\boldsymbol{\mu}_j]\right)} \tag{3.8}$$

where $\boldsymbol{\mu}_j$ and $\boldsymbol{\Sigma}_j$ denote the mean and covariance matrix of the j^{th} component of the mixture model, then the finite mixture model is called a mixture of Gaussians or a GMM.

For our application, each component of the GMM models a cluster of sounds acoustically close in the feature space. It is a strong assumption that each cluster's density is Gaussian, but it works well in practice. This assumption is supported by the Central Limit Theorem in the asymptotic case of infinite training data; we discuss more on this controversial issue in the Conclusion (§5.2).

For modeling the probability space, previous researchers chose the Gaussian Mixture Model (GMM) because of its ability to model the probability acoustic space of a speaker with a semi-parametric, continuous representation. Combining the advantages of parametric and non-parametric probability models, the GMM has been used successfully for both speaker identification [56] and voice conversion [68, 66, 28, 21]. It is not necessarily *non-parametric* because the size of the model does not increase with the size of the data set; yet, it is not *parametric* because the model does not assume a specific functional form as it is a linear combination of Gaussian densities. Rather than model the joint probability space of the speakers with a discrete set of vectors as do codebook techniques [2, 4, 5], the GMM models this space as a continuous probability density. The benefit is that the GMM can model a complex, globally nonlinear manifold well with a collection of *locally linear* models that exploit the tractability of operations in the Gaussian domain.

3.2.2 Estimation of Hidden Parameters with the EM algorithm

The parameters, $p(\boldsymbol{z}|j)$ and $P(j)$, in Equation 3.2 are called *hidden* parameters because they are the underlying causes for the shape of the probabilistic manifold. If our set of training data has prior class labels, then these parameters' values are set to the values of the labels. Since we do not have these labels, we need a way to estimate their values and in order to do so, we use the classic expectation-maximization (EM) algorithm [17].[2] The aim is to maximize the

[2]The original intent of the EM algorithm was to calculate maximum likelihood even if observations for all variables were not available. Since its original formulation, it has become quite popular for maximizing parameters of hidden or *latent* variable models by considering them as the incomplete data.

likelihood function $\mathcal{L}(\mathcal{Z}|\boldsymbol{\theta})$ of the parameters $\boldsymbol{\theta} = \{\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j, P(j) : j = 1, \ldots, M\}$ of the GMM. The algorithm consists of two steps: the first where we find the expected likelihood of the posterior component labels, and the second where we maximize the likelihood function using these estimates. A synopsis of the algorithm follows for the case of estimating a GMM's parameters.

1. *Initialization*: Using the data set $\mathcal{Z} = \{\boldsymbol{z}_n : n = 1 \text{ to } N\}$, set an initial guess for the parameters $\boldsymbol{\theta} = \{\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j, P(j) : j = 1, \ldots, M\}$.

2. *E-step*: Estimate the posterior probability $P(j|\boldsymbol{z}_n)$ for every data point $\boldsymbol{z}_n$ in the data set and every component j of the GMM using an equation similar to Equation 3.7 where

$$P(j|\boldsymbol{z}_n) = \frac{\mathcal{N}(\boldsymbol{z}_n; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)P(j)}{\sum_{j=1}^{M} \mathcal{N}(\boldsymbol{z}_n; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)P(j)} \tag{3.9}$$

3. *M-step*: Re-estimate the parameters for each component of the mixture using the posterior probabilities $P(j|\boldsymbol{z}_n)$ calculated in the *E-step*.

$$\hat{P}(j) = \frac{1}{N} \sum_{n=1}^{N} P(j|\boldsymbol{z}_n) \tag{3.10}$$

$$\hat{\boldsymbol{\mu}}_j = \frac{\sum_{n=1}^{N} P(j|\boldsymbol{z}_n)\boldsymbol{z}_n}{\sum_{n=1}^{N} P(j|\boldsymbol{z}_n)} \tag{3.11}$$

$$\hat{\boldsymbol{\Sigma}}_j = \frac{\sum_{n=1}^{N} P(j|\boldsymbol{z}_n)[\boldsymbol{z}_n - \hat{\boldsymbol{\mu}}_j][\boldsymbol{z}_n - \hat{\boldsymbol{\mu}}_j]^T}{\sum_{n=1}^{N} P(j|\boldsymbol{z}_n)} \tag{3.12}$$

4. Repeat steps 2 and 3 until the algorithm converges; i.e., repeat until the negative log-likelihood $-\ln \mathcal{L}(\mathcal{Z}|\boldsymbol{\theta})$ difference between iterations falls below a prescribed threshold. (Minimizing the negative log-likelihood is the same as maximizing the log-likelihood function [7].)

Note that Equation 3.12 is the limiting operation computationally because it involves a complete re-estimation of the j^{th} sample covariance matrix weighted by the j^{th} posterior component probability. Thus, the EM algorithm's complexity with fully populated covariance matrices is $\mathcal{O}(NMd^2)$.

An important issue is the initialization of the EM algorithm. Although it is guaranteed to converge to a local minimum, we should give it some initial help by choosing "smart" initial parameters. In our work, we initialize the means of each cluster via a K-means clustering algorithm and set the prior probabilities to have equal likelihood. This initialization should

help the algorithm converge more quickly while increasing the chances of reaching the global maximum likelihood.

In order to regularize each covariance matrix, we add a small disturbance $\varepsilon \boldsymbol{I}$ to each covariance matrix at each iteration as done in [56]. Although this disturbance could potentially decrease the convergence rate, it is helpful for increasing the chances of having invertible covariance matrices.

As an additional post-processing step similar to that described in [21], using Equation 3.2, we determine the probability that our GMM generates one of the data points in the training set. Then, we reject the data points that have low probability and re-estimate the model once again with the EM algorithm. This post-GMM estimation rejection improves resulting conversion performance although it increases the time to train the model.

3.3 Spectral Conversion with a GMM

Having estimated the parameters of the GMM, we can now estimate the target speaker's feature vector $\boldsymbol{y}$ from the source speaker's feature vector $\boldsymbol{x}$. The joint covariance matrix $\boldsymbol{\Sigma}_j$ for the j^{th} Gaussian component is partitioned as follows.

$$\boldsymbol{\Sigma}_j = \begin{bmatrix} \boldsymbol{\Sigma}_j^{xx} & \boldsymbol{\Sigma}_j^{xy} \\ \boldsymbol{\Sigma}_j^{yx} & \boldsymbol{\Sigma}_j^{yy} \end{bmatrix} \tag{3.13}$$

Below we summarize the different partitions of the joint covariance matrix $\boldsymbol{\Sigma}_j$:

- $\boldsymbol{\Sigma}_j^{xx}$ is the $\frac{d}{2} \times \frac{d}{2}$ component auto-covariance of the source vector $\boldsymbol{x}$.

$$E\left[[\boldsymbol{x} - \boldsymbol{\mu}_j^x][\boldsymbol{x} - \boldsymbol{\mu}_j^x]^T\right] \tag{3.14}$$

- $\boldsymbol{\Sigma}_j^{xy}$ is the $\frac{d}{2} \times \frac{d}{2}$ component cross-covariance of the source vector $\boldsymbol{x}$ with the target $\boldsymbol{y}$.

$$E\left[[\boldsymbol{x} - \boldsymbol{\mu}_j^x][\boldsymbol{y} - \boldsymbol{\mu}_j^y]^T\right] \tag{3.15}$$

- $\boldsymbol{\Sigma}_j^{yx}$ is the $\frac{d}{2} \times \frac{d}{2}$ component cross-covariance of $\boldsymbol{y}$ with $\boldsymbol{x}$. It is the transpose of Equation 3.15.

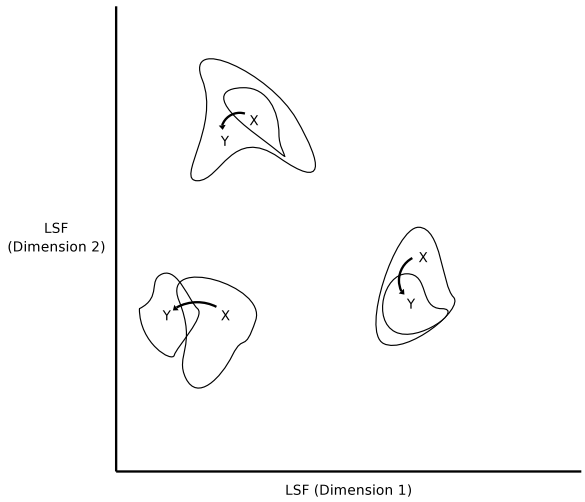

Figure 3.2: Voice Mapping

- $\mathbf{\Sigma}_j^{yy}$ is the $\frac{d}{2} \times \frac{d}{2}$ component auto-covariance of the target vector $\boldsymbol{y}$.

$$E\left[[\boldsymbol{y} - \boldsymbol{\mu}_j^y][\boldsymbol{y} - \boldsymbol{\mu}_j^y]^T\right] \tag{3.16}$$

In the case of one component, a single Gaussian, the expectation $E[\boldsymbol{y}|\boldsymbol{x}]$ is the conditional mean of a joint Gaussian given by

$$\begin{aligned} E[\boldsymbol{y} \mid \boldsymbol{x}] &= \int \boldsymbol{y}\, p(\boldsymbol{y}|\boldsymbol{x})\, d\boldsymbol{y} && (3.17) \\ &= \boldsymbol{\mu}_y + \mathbf{\Sigma}_{yx}(\mathbf{\Sigma}_{xx})^{-1}\left(\boldsymbol{x} - \boldsymbol{\mu}_x\right) && (3.18) \end{aligned}$$

Extending to the mixture case as previously done in [20], the expectation is

$$E[\boldsymbol{y} \mid \boldsymbol{x}] = \sum_{j=1}^{M} P(j|\boldsymbol{x}) \left[\boldsymbol{\mu}_j^y + \mathbf{\Sigma}_j^{yx}(\mathbf{\Sigma}_j^{xx})^{-1}\left(\boldsymbol{x} - \boldsymbol{\mu}_j^x\right)\right] \tag{3.19}$$

where we weight the j^{th} conditional mean by the probability of component j given the information from the source vector $\boldsymbol{x}$.

Note that the relationship in Equation 3.19 should not be misconstrued as a "function" as in previous literature [6, 28, 68] but rather as only a statistical correspondence between the two feature vectors $\boldsymbol{x}$ and $\boldsymbol{y}$. The misnomer of "function" implies causation when we are merely finding the best estimate (minimum or Bayesian MSE [33]) for $\boldsymbol{y}$ given prior knowledge of $\boldsymbol{x}$.

As an example, we show the mapping in Equation 3.19 with the two-dimensional probability space in Figure 3.2. In this artificial example, we notice 3 classes around which both speaker's features have formed. These classes could possibly represent acoustic classes in the feature space of both speakers. In order to map, we simply find the class conditional expectation of $\boldsymbol{y}$ times the probability that the class generated the data point $\boldsymbol{x}$ and sum over all classes in the model.

3.3.1 Decorrelation Assumption

Assuming that the individual random variables of the source and target vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ respectively are decorrelated is a constraint that previous researchers [68, 43] have placed on $\boldsymbol{x}$ and $\boldsymbol{y}$ to speed up training and conversion time. In order to benefit from these savings, researchers have elected either to impose the structure of each submatrix of $\boldsymbol{\Sigma}_j$ or of the entire covariance matrix $\boldsymbol{\Sigma}_j$ to be diagonal. When assuming this decorrelation between the various components of the feature vectors, the computational time for training with EM is significantly reduced to $O(NMd)$; in addition, we see later that conversion time is reduced.

When constraining the entire covariance $\boldsymbol{\Sigma}_j$ to be diagonal, this case is quite similar to voice conversion with codebook methods since the cross-covariance $\boldsymbol{\Sigma}_j^{yx}$ has only zero entries. Thus, the conversion from Equation 3.19 becomes the following in this case

$$E[\boldsymbol{y} \mid \boldsymbol{x}] = \sum_{j=1}^{M} P(j|\boldsymbol{x})\boldsymbol{\mu}_j^{y} \tag{3.20}$$

This conversion is simply the sum of each target component mean $\boldsymbol{\mu}_j^{y}$ weighted by the posterior probability of the j^{th} Gaussian component. It is only different from the codebook case in that the EM algorithm uses the covariance $\boldsymbol{\Sigma}_j$ in the iterative updates whereas the codebook method simply uses an iterative K-means clustering procedure without any covariance information.

When each submatrix is diagonal, the inverse computations become less complex, being equivalent to vectorizing each submatrix and calculating the dot product of these "vectors" rather than multiplying diagonal matrices. With the matrix $\boldsymbol{\Sigma}_j$ partitioned as in Equation 3.13, its inverse $\boldsymbol{\Sigma}_j^{-1}$ is also partitioned in the following way [33].

$$\boldsymbol{\Sigma}_j^{-1} = \begin{bmatrix} \boldsymbol{\Sigma}_j^{A} & \boldsymbol{\Sigma}_j^{B} \\ (\boldsymbol{\Sigma}_j^{B})^{T} & \boldsymbol{\Sigma}_j^{C} \end{bmatrix} \tag{3.21}$$

$$\boldsymbol{\Sigma}_j^A = \left(\boldsymbol{\Sigma}_j^{xx} - \boldsymbol{\Sigma}_j^{xy}(\boldsymbol{\Sigma}_j^{yy})^{-1}\boldsymbol{\Sigma}_j^{yx}\right)^{-1} \quad (3.22)$$

$$\boldsymbol{\Sigma}_j^B = -\boldsymbol{\Sigma}_j^{xx}\boldsymbol{\Sigma}_j^{xy}\boldsymbol{\Sigma}_j^C \quad (3.23)$$

$$\boldsymbol{\Sigma}_j^C = \left(\boldsymbol{\Sigma}_j^{yy} - \boldsymbol{\Sigma}_j^{yx}(\boldsymbol{\Sigma}_j^{xx})^{-1}\boldsymbol{\Sigma}_j^{xy}\right)^{-1} \quad (3.24)$$

3.3.2 Weakness of the Decorrelation Assumption

Although the above two methods can significantly reduce computational time, we should note that this restriction is quite inappropriate because we shouldn't assume that the feature vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ are completely decorrelated. This assumption implies that in the d-dimensional feature space, all of the covariance structures are aligned with the feature space axes. By making this assumption, we have no way of modeling covariance relationships between features. In addition, observation of the covariance matrices during experiments indicate that they are densely populated and that correlations do exist between the elements of the feature vectors. Another way to view the diagonal constraint is that it is a primitive method of reducing the dimensionality of covariance structure from d^2 to d by imposing a constraint on its structure.

Since only these two extremes, diagonal or full covariance matrices, are available, a need exists for a method which can fill in the "spectrum" of options between the extremities. With only these two extremes, the end user must make a difficult tradeoff between the time that it takes to train the system and the resulting performance desired.

3.4 Voice Conversion with a Mixture of Probabilistic Principal Component Analyzers

Although the GMM has become quite popular recently for modeling complex probability densities, one of its shortcomings is that an increase in the dimensionality of the feature space increases the complexity of the model. Each covariance matrix $\boldsymbol{\Sigma}_j$ in the mixture becomes excessively large, and estimation of each sample covariance matrix and its inverse is less tractable to compute as dimensionality increases.

Probabilistic Principal Component Analysis (PPCA), a method developed independently by both Tipping and Bishop [72, 73] and Roweis [60],[3] solves the inflexibility of GMMs

[3]Roweis named his method Sensible PCA. In this thesis, we refer to it as Probabilistic PCA.

by performing a pseudo *local Principal Component Analysis* on each component. We explain this point more below, but first provide a brief introduction to PCA.

3.4.1 Principal Component Analysis

PCA is a widely used technique originally formulated by Pearson in 1901 [52] for dimensionality reduction in large multivariate data sets. It maximizes the total variance of the data set $\{\boldsymbol{z}_n : n \in 1, \ldots, N\}$ by projecting the d-dimensional data onto q orthonormal *principal axes* where $q < d$. After some derivations found in [14, 24], we find that these principal axes are the q prevailing eigenvectors of the sample covariance matrix $\boldsymbol{S}$ where

$$\boldsymbol{S} = \frac{1}{N} \sum_{n=1}^{N} [\boldsymbol{z}_n - \boldsymbol{\mu}][\boldsymbol{z}_n - \boldsymbol{\mu}]^T \tag{3.25}$$

The prevailing eigenvectors are those which have the largest corresponding eigenvalues. One can determine how many principal components to retain by estimating the ratio of the variance of the projection to the total variance using the following equation

$$\frac{\sum_{j=1}^{q} \lambda_j}{\sum_{j=1}^{d} \lambda_j} \tag{3.26}$$

where each λ_j is an eigenvalue of the sample covariance matrix.

Although PCA performs well for linearly distributed data sets, it does not generally extend well to more complicated practical situations because of its global linearity and lack of a proper generative or probabilistic model. A generative model for PCA allows us to use the powerful technique in a mixture of locally linear models like we described with GMMs. Probabilistic Principal Component Analysis provides this generative model.

3.4.2 Probabilistic Principal Component Analysis

PPCA's statistical model is similar to factor analysis (FA) [25] in that both models assume that a set of latent or hidden variables $\boldsymbol{f}$ are responsible for generating the d-dimensional data set $\boldsymbol{z}$. The following equation gives PPCA's statistical model.

$$\begin{aligned} \boldsymbol{z} &= \boldsymbol{W}\boldsymbol{f} + \boldsymbol{\mu} + \boldsymbol{\epsilon} && (3.27) \\ \boldsymbol{f} &\sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}) && (3.28) \\ \boldsymbol{\epsilon} &\sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Psi}) && (3.29) \end{aligned}$$

We place the constraint that the latent variables $\boldsymbol{f}$ (*common factors*) are independent and Gaussian with unit variance, and the noise variables $\boldsymbol{\epsilon}$ (*specific factors*) are independent and Gaussian with covariance $\boldsymbol{\Psi}$. In the PPCA model, we restrict $\boldsymbol{\Psi}$ to have isotropic variance $\sigma^2\boldsymbol{I}$. In addition, the factors $\boldsymbol{f}$ are independent of the noise $\boldsymbol{\epsilon}$. The $d \times q$ matrix $\boldsymbol{W}$ contains the factor loadings, and the parameter vector $\boldsymbol{\mu}$ permits the data to have non-zero mean. Under these assumptions, the observations $\boldsymbol{z}$ are Gaussian with mean

$$\begin{aligned} E[\boldsymbol{z}] &= E[\boldsymbol{W}\boldsymbol{f} + \boldsymbol{\mu} + \boldsymbol{\epsilon}] && (3.30) \\ &= \boldsymbol{W}E[\boldsymbol{f}] + E[\boldsymbol{\mu}] + E[\boldsymbol{\epsilon}] && (3.31) \\ &= \boldsymbol{\mu} && (3.32) \end{aligned}$$

and model covariance $\boldsymbol{C}$.

$$\begin{aligned} \boldsymbol{C} &= E[[\boldsymbol{z} - \boldsymbol{\mu}][\boldsymbol{z} - \boldsymbol{\mu}]^T] && (3.33) \\ &= E[[\boldsymbol{W}\boldsymbol{f} + \boldsymbol{\epsilon}][\boldsymbol{W}\boldsymbol{f} + \boldsymbol{\epsilon}]^T] && (3.34) \\ &= \boldsymbol{W}E[\boldsymbol{f}\boldsymbol{f}^T]\boldsymbol{W}^T + E[\boldsymbol{\epsilon}\boldsymbol{f}^T]\boldsymbol{W}^T + && (3.35) \\ &\quad \boldsymbol{W}E[\boldsymbol{f}\boldsymbol{\epsilon}^T] + E[\boldsymbol{\epsilon}\boldsymbol{\epsilon}^T] \\ &= \boldsymbol{W}\boldsymbol{W}^T + \boldsymbol{\Psi} && (3.36) \end{aligned}$$

With this model, the common factors $\boldsymbol{f}$ account for the statistical dependencies between the individual variables of $\boldsymbol{z}$, and the specific factors $\boldsymbol{\epsilon}$ explain small disturbances in each individual random variable of $\boldsymbol{z}$. In our model for voice conversion, the common factors

capture the correlations between LSFs, and the specific factors capture the sensor noise about each individual LSF. Capturing these correlations allows us to eliminate the redundancies in the LSFs. Additionally, in our work, we use PPCA because we assume that the noise about one LSF should not be significantly greater than the noise about another LSF. This assumption is strong, but inspection of full covariance structure of each component in the GMM indicates that the diagonal elements corresponding to the variance of each individual LSF do not vary by much.

In order to *generate* a data point $\boldsymbol{z}_0$ from the PPCA distribution, we first sample a point $\boldsymbol{f}_0$ from the distribution $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ of the factors $\boldsymbol{f}$. Then, the factor loadings $\boldsymbol{W}$ transform the generated vector $\boldsymbol{f}_0$. We sample a point $\boldsymbol{\epsilon}_0$ from the distribution $\mathcal{N}(\boldsymbol{0}, \sigma^2\boldsymbol{I})$ of the noise component $\boldsymbol{\epsilon}$ and translate by $\boldsymbol{\epsilon}_0$ and the mean $\boldsymbol{\mu}$.

We can also determine the probability density of PPCA from a conditional probability perspective as follows

$$p(\boldsymbol{z}) = \int p(\boldsymbol{z}|\boldsymbol{f})p(\boldsymbol{f})d\boldsymbol{f} \tag{3.37}$$

$$= \int \mathcal{N}(\boldsymbol{W}\boldsymbol{f} + \boldsymbol{\mu}, \sigma^2\boldsymbol{I})\mathcal{N}(\boldsymbol{0}, \boldsymbol{I})d\boldsymbol{f} \tag{3.38}$$

which when worked through gives the same result mentioned above, i.e., $p(\boldsymbol{z}) \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{C})$.

Because we restrict the noise to have isotropic variance, we can find an analytic solution for its maximum likelihood. After some derivations which are found in [72], the maximum likelihood estimates for the mean vector $\boldsymbol{\mu}$, the factor loadings $\boldsymbol{W}$, and the isotropic noise variance σ^2 are given in the following equations. The MLE for the mean vector is simply the sample mean of the data.

$$\boldsymbol{\mu}_{ML} = \frac{1}{N}\sum_{n=1}^{N}\boldsymbol{z}_n \tag{3.39}$$

The analytic solution for the factor loadings is

$$\boldsymbol{W}_{ML} = \boldsymbol{U}_q(\boldsymbol{\Lambda}_q - \sigma^2\boldsymbol{I})^{1/2}\boldsymbol{R} \tag{3.40}$$

where $\boldsymbol{U}_q$ is a $d \times q$ matrix whose columns are the principal eigenvectors of the sample covariance matrix $\boldsymbol{S}$, $\boldsymbol{\Lambda}_q$ is a $q \times q$ diagonal matrix of the first q eigenvalues of $\boldsymbol{S}$, and $\boldsymbol{R}$ is an arbitrary orthogonal rotation matrix. We can solve for $\boldsymbol{C}$ by using Equation 3.36 and $\boldsymbol{W}_{ML}$.

$$\boldsymbol{C} = \boldsymbol{U}_q(\boldsymbol{\Lambda}_q - \sigma^2\boldsymbol{I})\boldsymbol{U}_q^T + \sigma^2\boldsymbol{I} \tag{3.41}$$

The MLE for the isotropic noise is the following

$$\sigma_{ML}^2 = \frac{1}{d-q}\sum_{j=q+1}^{d}\lambda_j \tag{3.42}$$

and represents the variance that is lost in the projection.

3.4.3 Differences Between Models

Similarity and Difference Between PPCA and PCA

What distinguishes PCA is that it is sensitive to the noise of each variable in $\boldsymbol{z}$. It can capture covariances between the variables of $\boldsymbol{z}$ well, but has no mechanism for capturing individual variances. PPCA is similar to PCA because it captures all of the independent noise with one covariance term σ^2. So, PPCA is more sensitive to the independent noise of each variable in $\boldsymbol{z}$. High noise in one variable of $\boldsymbol{z}$ causes PPCA to consider this variable more important just as PCA does.

The fact that PPCA has the generative model described in Equation 3.27 leads to many practical uses for it and is what distinguishes it from classical PCA. Moreover, PPCA is actually a general case of PCA. We can recover the PCA model by taking the limit as σ^2 goes to 0. We can see how PCA is recovered by inspecting Equation 3.41.

Difference Between PPCA and FA

The only constraint in FA that distinguishes it from PPCA is that in FA, the covariance of the noise $\boldsymbol{\epsilon}$ is a diagonal matrix. The problem with FA is that the diagonal constraint works well for modeling individual noise, but no analytic solution for $\boldsymbol{W}$ exists — it must be estimated iteratively. The fact that PPCA's model has isotropic noise with covariance $\sigma^2\boldsymbol{I}$ allows for an analytic solution for $\boldsymbol{W}$ in PPCA.

PPCA is named after PCA rather than FA because of the sensitivity of PPCA to noise in one variable. FA can capture independent noise with a term for each variable in $\boldsymbol{z}$ with the corresponding ψ_i entry in the diagonal matrix $\boldsymbol{\Psi}$.

Another important difference between PPCA and FA is that in PPCA, we add dimensions incrementally. For example, when retaining two components, the first component is the same that is found if only retaining one component. The same is not true in FA; the first component is not necessarily the same as if only retaining one.

3.4.4 Mixtures of Probabilistic Principal Component Analyzers

Because PPCA has a generative model, we can combine several of these models into a mixture of PPCAs as described in [72]. The form of the mixture is the following

$$p(\boldsymbol{z}) = \sum_{j=1}^{M} \int p(\boldsymbol{z}|\boldsymbol{f}_j) p(\boldsymbol{f}_j|j) P(j) d\boldsymbol{f} \tag{3.43}$$

where we represent the j^{th} component of the mixture with a PPCA model. We find certain similarities between this model and the GMM — the only difference is that we represent each of the M component densities with a single PPCA model as in Equation 3.37 rather than with a multivariate normal like the GMM does.

Local PCA

It is reasonable to ask, why not just perform a local PCA on each component of the GMM? The reason is that PCA does not have a true density model; in other words, we cannot generate data from the PCA model because it lacks a probability model. More importantly, in our case, we need to determine the true probability of a data point for post GMM estimation rejection as we described in §3.2.2.

3.4.5 Difference Among the Diagonal GMM, Full GMM, and Mixture of PPCAs

With the artificial example in Figures 3.3 – 3.5, we illustrate the difference among the models' ability to represent a probability distribution. The data points are the same in all three figures; we generated 500 data points from a GMM of 3 classes. Referring to all Figures 3.3 – 3.5,

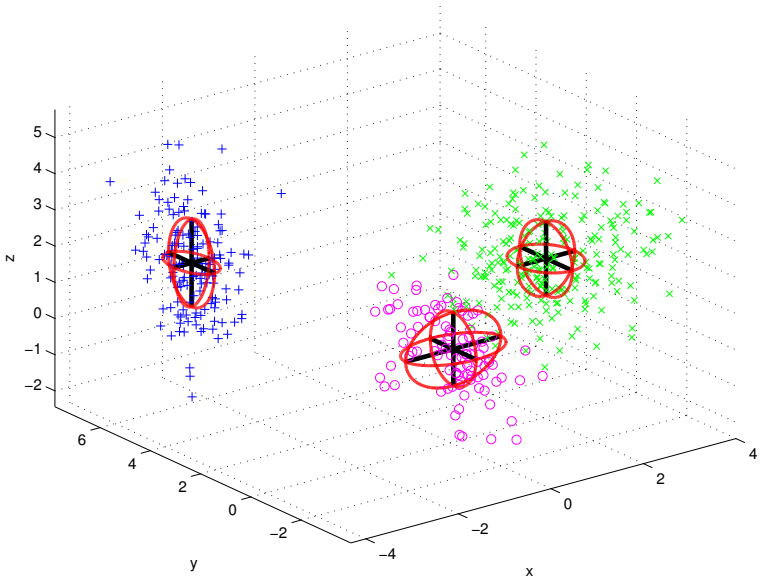

Figure 3.3: 3-D data fit with a Diagonal Covariance GMM

the first class on the left has prior probability $\pi_1 = \frac{1}{5}$ with mean $\boldsymbol{\mu}_1 = (-3.1, 4.7, 2.1)$ and axis-aligned variance $\boldsymbol{\Sigma}_1 = \text{diag}(0.16, 0.64, 1.44)$. The second class in the center has prior probability $\pi_2 = \frac{3}{10}$ with zero mean $\boldsymbol{\mu}_2 = \mathbf{0}$ and unit variance except for the x-axis which has variance $\frac{1}{4}$. We then rotated this second class 60 degrees around the x-axis and 30 degrees around the z-axis. The third class on the right has prior probability $\pi_3 = \frac{1}{2}$ with mean $\boldsymbol{\mu}_3 = (1.2, -1.5, 2.8)$ and unit variance; a majority of the data falls here because of its high prior probability. Each class in each figure has an ellipse of one standard deviation to represent the covariance fit.

In Figure 3.3, the diagonal model performs well in capturing the essence of the first and third classes but has no mechanism for capturing data that is not aligned to the feature axes as is the case with the second class. The full GMM model overfits somewhat for the first and third classes but performs well in capturing the rotation of the second class. Additionally, we see how PPCA has captured the direction of highest variance for each class while providing a more parsimonious fit (one dimension) than the full GMM does. Since the third class has unit variance with only limited generated data, the direction of maximal variance is arbitrary. In most realistic cases, the data does not align with the feature axes as the first and third classes do, but has rotation like the second class.

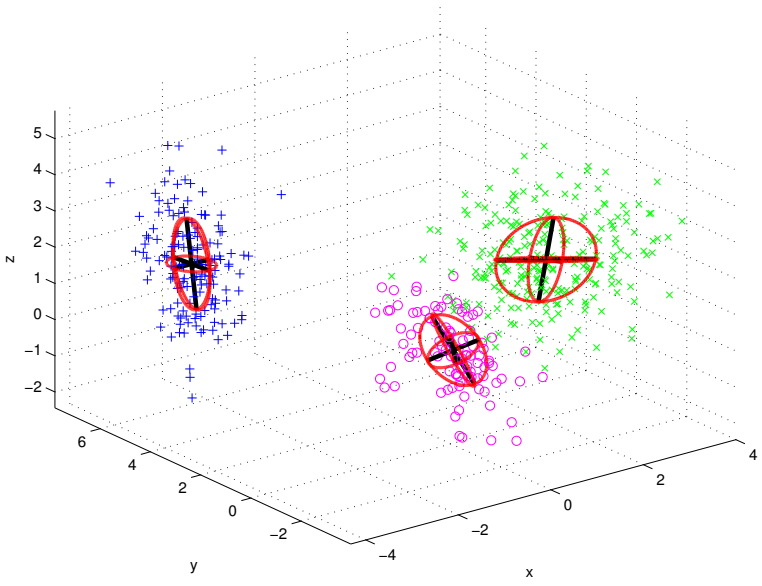

Figure 3.4: 3-D data fit with a Full Covariance GMM

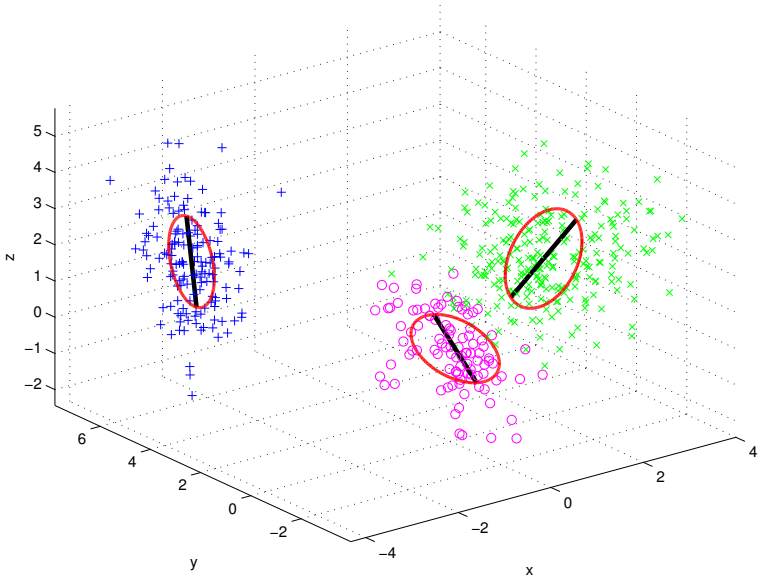

Figure 3.5: 3-D data fit with a Mixture of PPCAs

3.4.6 EM for a Mixture of PPCAs

Having MLEs for $\boldsymbol{\mu}$, $\boldsymbol{W}$, and σ^2 allows us to use these in an EM iterative framework similar to the EM algorithm discussed in §3.2.2. In addition to estimating the usual parameters, we could perform an eigen-decomposition of the sample matrix $\boldsymbol{S}$ and compute the parameters $\boldsymbol{W}$ and σ^2 using Equations 3.40 and 3.42 respectively. Although a simple addition to EM, it is expensive since we estimate the sample covariance with an additional eigen-decomposition.

Instead of performing these expensive steps, we can actually reduce the amount of computations necessary with a two stage EM algorithm as formulated in [72]. In the first stage, we ignore the latent variables $\boldsymbol{f}$ and compute the expected log likelihood; then we maximize $P(j)$ and $\boldsymbol{\mu}_j$ like we did in Equation 3.10 and 3.11. In the second stage, we increase the likelihood again by maximizing $\boldsymbol{W}_j$ and σ_j^2 with the following equations

$$\hat{\boldsymbol{W}}_j = \boldsymbol{S}_j \boldsymbol{W}_j \left(\sigma_j^2 \boldsymbol{I} + \boldsymbol{M}_j^{-1} \boldsymbol{W}_j^T \boldsymbol{S}_j \boldsymbol{W}_j\right)^{-1} \tag{3.44}$$

$$\hat{\sigma}_j^2 = \frac{1}{d} \operatorname{tr}\left\{\boldsymbol{S}_j - \boldsymbol{S}_j \boldsymbol{W}_j \boldsymbol{M}_j^{-1} \hat{\boldsymbol{W}}_j\right\} \tag{3.45}$$

where $\boldsymbol{S}_j$ is the weighted sample covariance matrix for the j^{th} component and $\boldsymbol{M}_j = (\boldsymbol{W}_j^T \boldsymbol{W}_j + \sigma^2 \boldsymbol{I})$. Although it is convenient to include $\boldsymbol{S}_j$ in the above equation, its computation, an $\mathcal{O}(NMd^2)$ operation, is not explicitly necessary. It is only necessary to evaluate the trace of the j^{th} weighted sample covariance matrix, an $\mathcal{O}(NMd)$ operation; and we evaluate $\boldsymbol{S}_j \boldsymbol{W}_j$ as

$$\boldsymbol{S}_j \boldsymbol{W}_j = \frac{1}{\hat{\pi}_j N} \sum_{n=1}^{N} R_{nj} [\boldsymbol{z}_n - \boldsymbol{\mu}_j] \left\{[\boldsymbol{z}_n - \boldsymbol{\mu}_j]^T \boldsymbol{W}_j\right\} \tag{3.46}$$

where R_{nj} is the *responsibility* of the j^{th} component and $\hat{\pi}_j$ is the j^{th} mixing coefficient as described in [72]. Equation 3.46 is the limiting computation in the above two stage EM formulation with a complexity of $\mathcal{O}(NMdq)$. With this formulation, we can vary the complexity of estimation with q, the amount of information we are willing to keep.

3.4.7 Spectral Conversion with PPCAs

In order to convert the spectrum in the PPCA case, we calculate the expectation of the target vector $\boldsymbol{y}$ given the source vector $\boldsymbol{x}$ for the single PPCA model and then extend the result

to a mixture of PPCAs. To find this expectation, we partition $\boldsymbol{z}$ so that its joint multivariate density is

$$P\left(\begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix}\right) = \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\mu}_x \\ \boldsymbol{\mu}_y \end{bmatrix}, \begin{bmatrix} \boldsymbol{W}_x\boldsymbol{W}_x^T + \sigma^2\boldsymbol{I} & \boldsymbol{W}_x\boldsymbol{W}_y^T \\ \boldsymbol{W}_y\boldsymbol{W}_x^T & \boldsymbol{W}_y\boldsymbol{W}_y^T + \sigma^2\boldsymbol{I} \end{bmatrix}\right) \tag{3.47}$$

where we partition the mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{C}$ from Equation 3.36. Then, using Equation 3.18, the conditional expectation of a joint Gaussian, we find that

$$E[\boldsymbol{y}|\boldsymbol{x}] = \boldsymbol{W}_y\boldsymbol{W}_x^T\left(\boldsymbol{W}_x\boldsymbol{W}_x^T + \sigma^2\boldsymbol{I}\right)^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_x) + \boldsymbol{\mu}_y \tag{3.48}$$

Extending the result from above to the case of a mixture of PPCAs, our statistical mapping is

$$\sum_{j=1}^{M} P(j|\boldsymbol{x})\boldsymbol{W}_y\boldsymbol{W}_x^T\left(\boldsymbol{W}_x\boldsymbol{W}_x^T + \sigma^2\boldsymbol{I}\right)^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_x) + \boldsymbol{\mu}_y \tag{3.49}$$

Another way of deriving this result is to partition the model from Equation 3.27 as follows.

$$\begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix} = \begin{bmatrix} \boldsymbol{W}_x \\ \boldsymbol{W}_y \end{bmatrix}\boldsymbol{f} + \begin{bmatrix} \boldsymbol{\mu}_x \\ \boldsymbol{\mu}_y \end{bmatrix} + \begin{bmatrix} \boldsymbol{\epsilon}_x \\ \boldsymbol{\epsilon}_y \end{bmatrix} \tag{3.50}$$

Then, we determine the expectation as

$$\begin{aligned} E[\boldsymbol{y}|\boldsymbol{x}] &= E[\boldsymbol{W}_y\boldsymbol{f} + \boldsymbol{\mu}_y + \boldsymbol{\epsilon}_y|\boldsymbol{x}] && (3.51)\\ &= \boldsymbol{W}_y E[\boldsymbol{f}|\boldsymbol{x}] + \boldsymbol{\mu}_y && (3.52) \end{aligned}$$

where using Equation 3.50, we find $E[\boldsymbol{f}|\boldsymbol{x}]$ as

$$\begin{aligned} E[\boldsymbol{f}|\boldsymbol{x}] &= E\left[\left(\boldsymbol{W}_x^T\boldsymbol{W}_x + \sigma^2\boldsymbol{I}\right)^{-1}\boldsymbol{W}_x^T(\boldsymbol{x} - \boldsymbol{\mu}_x - \boldsymbol{\epsilon}_x)\right] && (3.53)\\ &= \left(\boldsymbol{W}_x^T\boldsymbol{W}_x + \sigma^2\boldsymbol{I}\right)^{-1}\boldsymbol{W}_x^T(\boldsymbol{x} - \boldsymbol{\mu}_x) && (3.54) \end{aligned}$$

Thus, the expectation is

$$E[\boldsymbol{y}|\boldsymbol{x}] = \boldsymbol{W}_y\left(\boldsymbol{W}_x^T\boldsymbol{W}_x + \sigma^2\boldsymbol{I}\right)^{-1}\boldsymbol{W}_x^T(\boldsymbol{x} - \boldsymbol{\mu}_x) + \boldsymbol{\mu}_y \tag{3.55}$$

This expression is equivalent to Equation 3.48 because of the following identity

$$\left(\boldsymbol{W}_x^T\boldsymbol{W}_x + \sigma^2\boldsymbol{I}\right)^{-1}\boldsymbol{W}_x^T = \boldsymbol{W}_x^T\left(\boldsymbol{W}_x\boldsymbol{W}_x^T + \sigma^2\boldsymbol{I}\right)^{-1} \tag{3.56}$$

that is proven in Appendix A.

The two above results are similar to the mapping found for GMMs. When choosing which expression to use for spectral conversion, we should determine first whether $q < \frac{d}{2}$ for computational purposes. If it is, then we should use Equation 3.55 for converting. Instead of computing the inverse of the $\frac{d}{2} \times \frac{d}{2}$ matrix $\boldsymbol{W}_x\boldsymbol{W}_x^T + \sigma^2\boldsymbol{I}$ of outer products, we compute the inverse of the $q \times q$ matrix $\boldsymbol{W}_x^T\boldsymbol{W}_x + \sigma^2\boldsymbol{I}$ of inner products with a total complexity of $\mathcal{O}(Mq^3)$ for all components in the mixture. This decrease in complexity may not be that significant since we only compute it once with our method; but, if we update the model on line by including novelty test data as recent research has investigated [63], using this model reduces complexity of conversion. If $q > \frac{d}{2}$, then we should use Equation 3.48 extended to a mixture framework.

3.4.8 PPCA as a General Case of Spectral Conversion with a GMM

We now demonstrate the equivalence of conversion with the PPCA model and the GMM for the special case of retaining all principal components. We only show the equivalence for a single component because extension to the mixture case follows easily.

Because the covariance matrix $\boldsymbol{\Sigma}$ is symmetric, we can factor it with an eigen-decomposition.

$$\boldsymbol{\Sigma} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^T = (\boldsymbol{U}\boldsymbol{\Lambda}^{\frac{1}{2}})(\boldsymbol{U}\boldsymbol{\Lambda}^{\frac{1}{2}})^T \tag{3.57}$$

This decomposition is equivalent to the case when we retain all principal components with $\sigma^2 \to 0$ and the arbitrary rotation matrix $\boldsymbol{R} = \boldsymbol{I}$. So, setting $\boldsymbol{W} = \boldsymbol{U}\boldsymbol{\Lambda}^{\frac{1}{2}}$, we can formulate $\boldsymbol{\Sigma}$ as $\boldsymbol{W}\boldsymbol{W}^T$ when retaining all components.[4] Partitioning $\boldsymbol{W}$ into $\boldsymbol{W}_x$ and $\boldsymbol{W}_y$ gives the following.

$$\boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{W}_x \\ \boldsymbol{W}_y \end{bmatrix}\begin{bmatrix} \boldsymbol{W}_x^T & \boldsymbol{W}_y^T \end{bmatrix} = \begin{bmatrix} \boldsymbol{\Sigma}_{xx} & \boldsymbol{\Sigma}_{xy} \\ \boldsymbol{\Sigma}_{yx} & \boldsymbol{\Sigma}_{yy} \end{bmatrix} \tag{3.58}$$

[4] $\boldsymbol{W}$ in this case is $\boldsymbol{W}_{ML}$ from Equation 3.40 without the noise term σ^2 and $\boldsymbol{R} = \boldsymbol{I}$.

We determine from above that $\mathbf{\Sigma}_{xx} = \boldsymbol{W}_x \boldsymbol{W}_x^T$ and $\mathbf{\Sigma}_{yx} = \boldsymbol{W}_y \boldsymbol{W}_x^T$. Substituting into the expression for the conditional mean of a joint Gaussian from Equation 3.18, we find that

$$E[\boldsymbol{y}|\boldsymbol{x}] = \boldsymbol{W}_y \boldsymbol{W}_x^T \left(\boldsymbol{W}_x \boldsymbol{W}_x^T \right)^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_x) + \boldsymbol{\mu}_y \tag{3.59}$$

and notice that this solution is the same as Equation 3.48 with $\sigma^2 \to 0$.

Thus, PPCA is a more general case of the GMM for spectral conversion. In PPCA, we have the flexibility of removing dimensions from $\boldsymbol{W}$ which in turn adds to σ^2 the average variance not captured in the projection. We notice a relationship: the more dimensions that we take away from $\boldsymbol{W}$, the farther away from the "true" conversion we get. The question now is how far we can go away before it perceptually makes a difference to the human ear. We assess this question with both objective and subjective measurements in the next chapter.

Chapter 4

Evaluation

4.1 The Arctic Speech Corpus

FOR OUR RESEARCH, we specifically chose the Arctic Speech Corpus [34] because it is a recently developed, high quality speech corpus with a "free software" license. It was specifically created for the purpose of speech synthesis. Past research on voice conversion [21] used the Boston University Radio Speech Corpus [50], but because it was recorded in 1995, it is becoming dated. Additionally, its original intention was for the study of prosody of speech. Since this thesis only deals with spectral conversion, the Arctic corpus better suits our purpose. The speakers in the Arctic corpus control their prosody consistently throughout each sentence in the corpus.

The Arctic corpus features three male voice talents: Brian, John, and Alan. Brian has a Midwest American accent, John's is Canadian, and Alan's is Scottish. A female voice is also in the corpus, but we opt not to include her voice because it is easily distinguishable from the others.

The text selected for this corpus includes a wide variety of phonemes with a relatively even distribution. This even distribution provides a sound base for our voice conversion model. The speakers read text from various short stories of late 20^{th} century literature found in Project Gutenberg [26].

4.2 Objective Evaluation

In evaluating our system, we use the objective measure given by Kain [28]. This *performance index* is a ratio of two measures. The first measure, the *transspeaker distance*, is the

spectral distance between the converted speech and the target speech determining how "close" the converted speech is to the target speaker's. The second, the *interspeaker distance*, measures the spectral distance between the source and target speaker. To present the performance index, let us again consider the vector of source speech for the n^{th} frame as $\boldsymbol{x}_n$ and the target speaker's n^{th} vector as $\boldsymbol{y}_n$. We compute the performance index p with the following equation

$$p = 1 - \sum_{n=1}^{N} \frac{d(\boldsymbol{y}_n, \hat{\boldsymbol{y}}_n)}{d(\boldsymbol{y}_n, \boldsymbol{x}_n)} \tag{4.1}$$

where $\hat{\boldsymbol{y}}_n$ is the converted target vector. Since our feature for representing speech is the LSF, the distance measure d between any two LSF vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ with dimension p is Euclidean.

$$\left\{ \sum_{k=1}^{p} (a_k - b_k)^2 \right\}^{\frac{1}{2}} \tag{4.2}$$

We interpret a performance index close to 1 as a "good" conversion while one close to 0 is not performing well. Equation 4.1 is close to 1 when the source and target speaker have different spectral properties and when the converted speech is close to the target speaker's speech. Thus, according to the objective measure, it is difficult for the system to perform well if the source and target speaker's spectral properties are already similar.

In Figure 4.1, we plot the actual data points from the objective results of our system with a mixture of PPCAs benchmarked against the GMM method; in Figure 4.2, we fit the data points at each time interval with its appropriate "learning curve". As the training time increases, the performance of each system exponentially increases until it plateaus around 14 seconds of training data. We conclude that 14 seconds of training data is enough for the system to give a reasonable *average* performance between 0.24 and 0.28. We expect the average performance to increase slightly with more than 30 seconds of training data.[1] We show the mixture of PPCAs for a range of values for q — the number of dimensions we keep. Note that q varies from one dimension to 39 dimensions because we set the LP model order for each speaker to 20; thus, the joint space of both speakers has dimension 40.

The interesting observation is that for each reduction of q, the performance slightly decreases as we predicted in the previous chapter.[2] The only question is whether this incremental decrease in information is perceptually noticeable to a listener.

[1] Kain's *best* performance index was 0.31 [28], and Gillett's was 0.36 [21] with 120 seconds of training data.

[2] Note that this general trend occurs for each unit decrease of q, but we only show the average for a range of values for q.

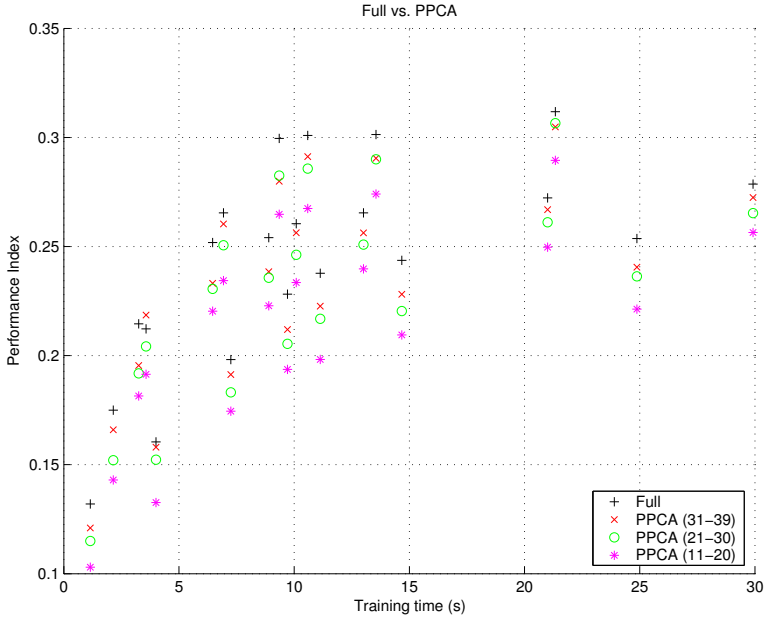

Figure 4.1: Performance Index of Data for a Full GMM and a mixture of PPCAs with varying dimensionality.

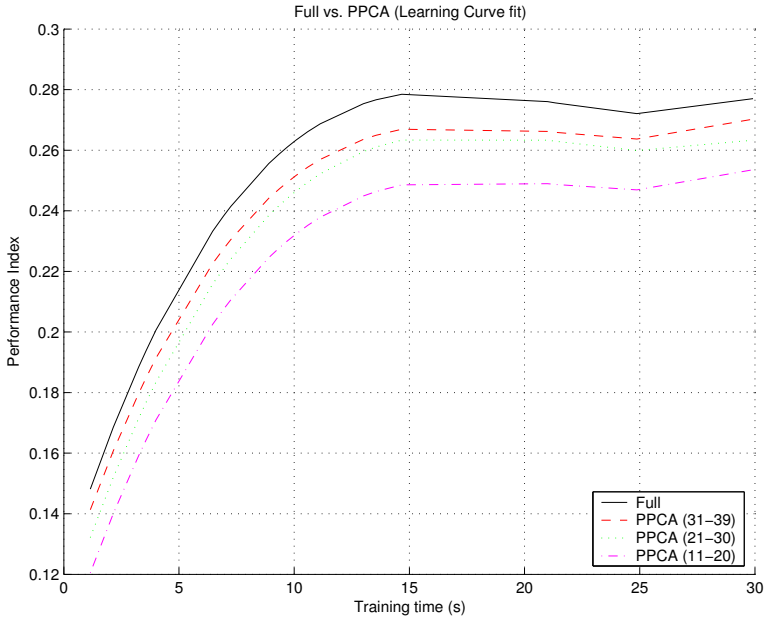

Figure 4.2: Performance Index Learning Curve for a Full GMM and a mixture of PPCAs with varying dimensionality.

Speaker	Alan	Brian	John		
Recognizability	very difficult	difficult	neutral	easy	very easy
Naturalness	very artificial	artificial	neutral	natural	very natural
Quality	very poor	poor	fair	good	excellent

Table 4.1: Sample Question on the Subjective Test

4.3 Subjective Listening Tests

In order to determine if the human ear notices the gradual loss of information, we conducted subjective listening tests with seven listeners. Before starting the test, we trained each listener on each of the three speakers' distinct voice qualities by playing several of their speech files. After this initial training, we quizzed each listener until they identified each speaker correctly for ten consecutive trials.

After quizzing, we began the test, playing the same sentence "Author of the Danger Trail, Philip Steels, etc.", a sentence with rich phonemic content. We played the converted speech in a randomized order varying both the PPCA dimensionality and the speaker. Between each utterance, we gave the listener 10 seconds to write down his / her answer; these time intervals made the entire test last about 40 minutes. We played the same sentence so that listeners could focus intently on the quality of each recording. For each utterance, the listener responded to the questions listed in Table 4.1.

First, the listener identified the speaker; overall, the accuracy was 79.2%. After determining the speaker, the listener provided a subjective answer for three categories.

Recognizability indicates how easy it is for the listener to identify the speaker. We can think of recognizability as a measure of closeness; it is similar to the objective distance measure described with Equation 4.1.

Naturalness describes how much like a human the speech sounds. Artificial implies that it sounds like a computer generated the speech.

Quality is a subjective measure of how clean the speech signal is. If it has artifacts or scratches, we consider it to be of low quality.

We assign a score of 1 to the most negative answers and a score of 5 to the most positive answers.

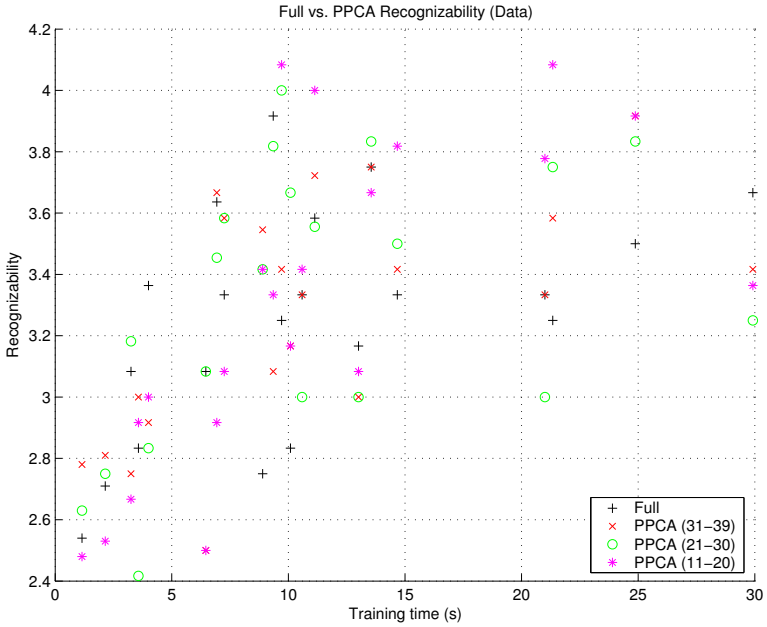

Figure 4.3: Subjective Recognizability Data for a Full GMM and a mixture of PPCAs with varying dimensionality.

4.3.1 Subjective Test Comments

Several of the listeners provided feedback on the test that gives some insight into the results. One noted that Alan and John were difficult to distinguish, but this difficulty became easier as the test progressed. Another mentioned that quality of the recordings improved towards the end, but we suppose that this "improvement" was due to the listener's ear adjusting to the quality of the recordings. Another listener stated that John's voice was easiest to distinguish from the others' because his was deeper; yet another mentioned that Brian's voice was easiest to distinguish because it was higher than the others'. Listeners also struggled to distinguish between naturalness and quality, but as the test progressed, it was easier to separate these two categories.

4.3.2 Subjective Test Results

In Figure 4.3, we plot the mean recognizability data points for each type of model versus the training time; we fit a learning curve to this data in Figure 4.4. Note that recognizability of each system improves from "difficult" to "easy" as the training time increases; but from the listeners' responses, we conclude that it is difficult for them to distinguish the difference between full covariances and reduced dimension covariances.

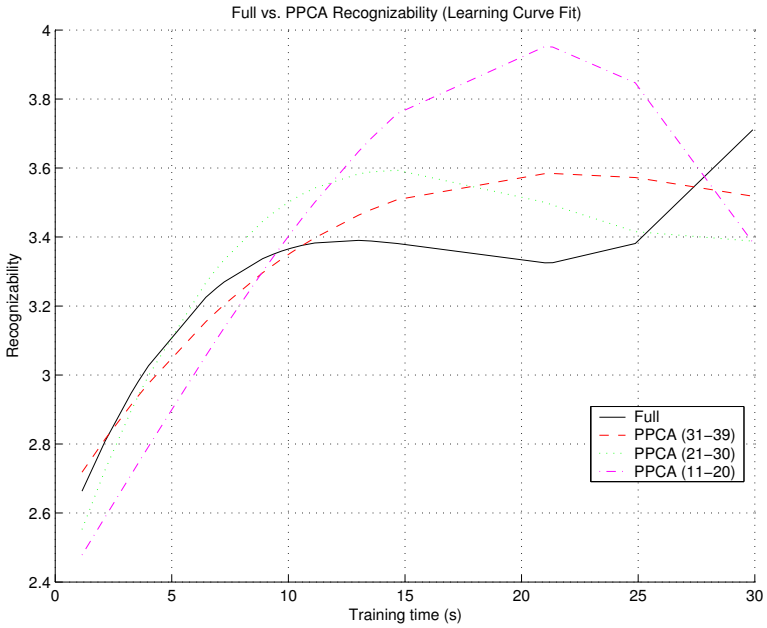

Figure 4.4: Subjective Recognizability Learning Curve for a Full GMM and a mixture of PPCAs with varying dimensionality.

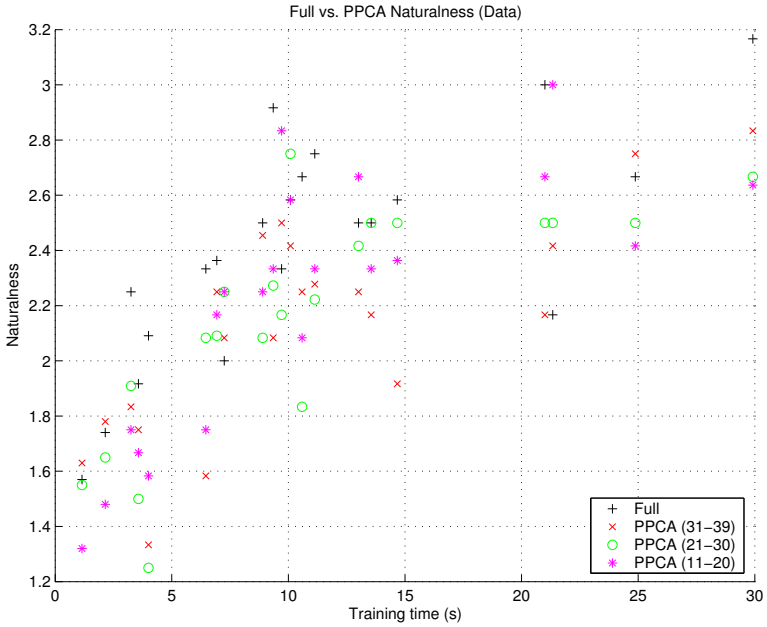

Figure 4.5: Subjective Naturalness Data for Full GMMs and a mixture of PPCAs with varying dimensionality.

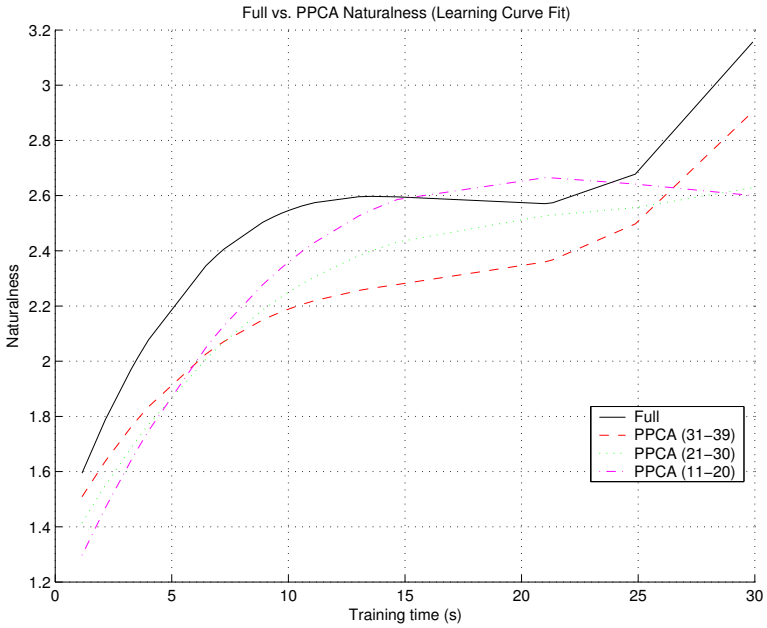

Figure 4.6: Subjective Naturalness Learning Curve for Full GMMs and a mixture of PPCAs with varying dimensionality.

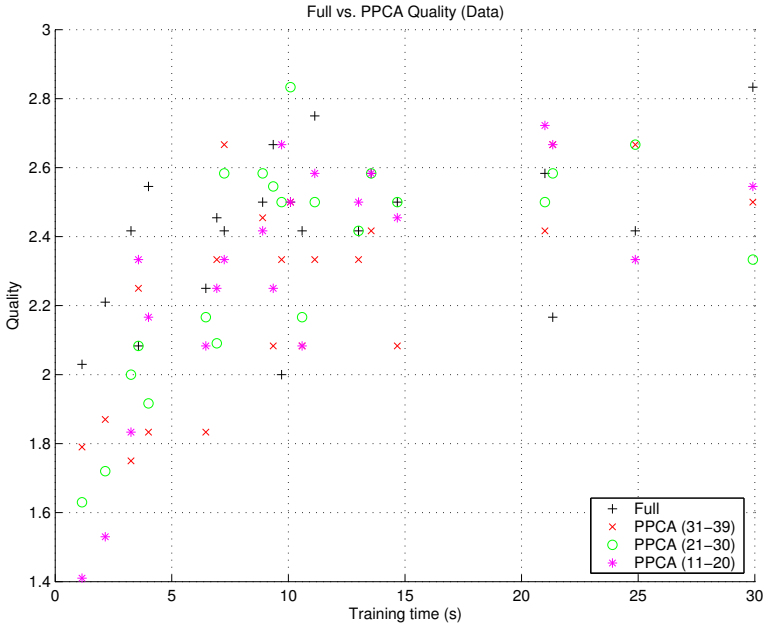

Figure 4.7: Subjective Quality Data for a Full GMM and a mixture of PPCAs with varying dimensionality.

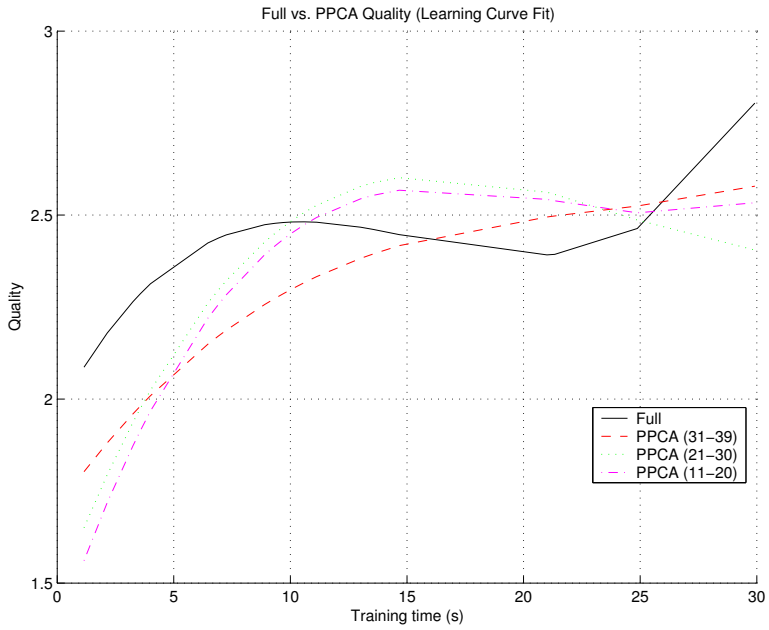

Figure 4.8: Subjective Quality Learning Curve for a Full GMM and a mixture of PPCAs with varying dimensionality.

We plot the mean naturalness data points in Figure 4.5 with the corresponding learning curve in Figure 4.6. In Figure 4.6, it is apparent that the full GMM method performs mostly better than PPCA though its performance is equivalent for some time to the PPCAs when q is between 11 and 20. The fact that lower dimensions perform better than higher dimensions indicate that perhaps a little more testing is needed for this case. This anomaly probably wouldn't occur with additional testing — we expect the different systems' naturalness to be closer.

Lastly, in Figures 4.7 and 4.8, we illustrate the subjective quality data points and learning curve respectively. Again, nothing indicates that one system's quality outperforms the others. All learning curves are relatively close so we can conclude with some degree of confidence that listeners do not notice the difference when we reduce dimensionality. The subjective quality tests indicate that the conversion's quality is only between "poor" and "fair".

Based on these tests of the three subjective categories, it is apparent that each category's performance increases with the amount of training data. Although the performance increases with more training data, incrementally removing "less relevant" information does not affect the end listener perceptually. Thus, in using PPCA, we have the benefit of reduced time

and memory complexity for both training and conversion without affecting the end listener's perceptual experience.

Chapter 5

Conclusion

5.1 Concluding Remarks

IN SUMMARY, we have applied the mixture of PPCAs model to voice conversion. Using a mixture of PPCAs rather than a GMM reduces the time complexity for training from $\mathcal{O}(MNd^2)$ to $\mathcal{O}(MNdq)$ and conversion time from $\mathcal{O}(M\left(\frac{d}{2}\right)^3)$ to $\mathcal{O}(Mq^3)$ if $q < \frac{d}{2}$. This reduction in complexity comes with the penalty of a slight decrease in the objective quality of conversion. Although the objective measure can detect that we have reduced the amount of information, the human ear has difficulty perceiving this reduction. Therefore, we recommend that the user of the system select an appropriate value of q to suit performance needs of the application. Obviously we always want the system to perform with highest quality; but, this high quality only comes with the price of expensive computations. For voice conversion training and execution, the mixture of PPCAs model provides a flexible range of tradeoffs to select from.

5.2 Future Opportunities for Research in Voice Conversion

We have several ways that this system can be improved upon by incorporating the recent advances of variational Bayesian modeling, independent component analysis, and an on line EM algorithm.

By incorporating variational Bayesian techniques with PPCA as described in [8], the system can automatically estimate the appropriate model order for q, the amount of information to retain of each component in the mixture, and M, the number of components in the mixture. This method falls in the *automatic relevance determination* framework of variational

Bayesian techniques. Having this ability implies that we can describe the nonlinear probability density with an appropriate number of components for the mixture and an appropriate projection for each component in the mixture. Incorporating this automatic technique increases complexity slightly but solves the difficult problem of estimating the correct model order and dimension reduction. Incorporating this technique lifts these burdens from the end user of the system.

As discussed previously in §3.2.1, using a Gaussian for each component in the mixture is a controversial issue because we do not have infinite training data. One way to solve this problem is to use a mixture of independent component analyzers as described in [57] where each component's distribution is non-Gaussian. With this approach, perhaps we could describe the density of each component in the mixture more properly.

In our model, we only estimate the probability density from a fixed set of training data. Including novelty test data in the model is expensive with our current method because it requires a complete re-estimation via EM. However, by updating the model on line with the testing data of the source speaker, we could improve performance significantly. Additionally, by using this model, we could realize computational savings for conversion with Equation 3.55 if $q < \frac{d}{2}$.

Appendix A

Equivalent PPCA Spectral Conversion Expressions

We prove that $\boldsymbol{W}^T \left(\sigma^2 \boldsymbol{I} + \boldsymbol{W}\boldsymbol{W}^T\right)^{-1} = \left(\sigma^2 \boldsymbol{I} + \boldsymbol{W}^T\boldsymbol{W}\right)^{-1} \boldsymbol{W}^T$ for any matrix $\boldsymbol{W}$ and any $\sigma \neq 0$. Using the Matrix Inversion Lemma,

$$\left(\sigma^2 \boldsymbol{I} + \boldsymbol{W}^T\boldsymbol{W}\right)^{-1} = \frac{\boldsymbol{I}}{\sigma^2} - \frac{\boldsymbol{W}^T}{\sigma^2}\left(\boldsymbol{I} + \frac{\boldsymbol{W}\boldsymbol{W}^T}{\sigma^2}\right)^{-1}\frac{\boldsymbol{W}}{\sigma^2} \tag{A.1}$$

$$= \frac{\boldsymbol{I}}{\sigma^2} - \boldsymbol{W}^T\left(\sigma^2 \boldsymbol{I} + \boldsymbol{W}\boldsymbol{W}^T\right)^{-1}\frac{\boldsymbol{W}}{\sigma^2} \tag{A.2}$$

$$\sigma^2\left(\sigma^2 \boldsymbol{I} + \boldsymbol{W}^T\boldsymbol{W}\right)^{-1} + \boldsymbol{W}^T\left(\sigma^2 \boldsymbol{I} + \boldsymbol{W}\boldsymbol{W}^T\right)^{-1}\boldsymbol{W} = \boldsymbol{I} \tag{A.3}$$

$$\sigma^2 \boldsymbol{I} + \boldsymbol{W}^T\left(\sigma^2 \boldsymbol{I} + \boldsymbol{W}\boldsymbol{W}^T\right)^{-1}\boldsymbol{W}\left(\sigma^2 \boldsymbol{I} + \boldsymbol{W}^T\boldsymbol{W}\right) = \sigma^2 \boldsymbol{I} + \boldsymbol{W}^T\boldsymbol{W} \tag{A.4}$$

$$\boldsymbol{W}^T\left(\sigma^2 \boldsymbol{I} + \boldsymbol{W}\boldsymbol{W}^T\right)^{-1}\boldsymbol{W}\left(\sigma^2 \boldsymbol{I} + \boldsymbol{W}^T\boldsymbol{W}\right) = \boldsymbol{W}^T\boldsymbol{W} \tag{A.5}$$

$$\boldsymbol{W}^T\left(\sigma^2 \boldsymbol{I} + \boldsymbol{W}\boldsymbol{W}^T\right)^{-1}\boldsymbol{W} = \boldsymbol{W}^T\boldsymbol{W}\left(\sigma^2 \boldsymbol{I} + \boldsymbol{W}^T\boldsymbol{W}\right)^{-1} \tag{A.6}$$

$$\left(\sigma^2 \boldsymbol{I} + \boldsymbol{W}\boldsymbol{W}^T\right)^{-1}\boldsymbol{W} = \boldsymbol{W}\left(\sigma^2 \boldsymbol{I} + \boldsymbol{W}^T\boldsymbol{W}\right)^{-1} \tag{A.7}$$

$$\boldsymbol{W}^T\left(\sigma^2 \boldsymbol{I} + \boldsymbol{W}\boldsymbol{W}^T\right)^{-1} = \left(\sigma^2 \boldsymbol{I} + \boldsymbol{W}^T\boldsymbol{W}\right)^{-1}\boldsymbol{W}^T \tag{A.8}$$

Bibliography

[1] Virtual voices. Guardian Unlimited Online, February 2004.

[2] Masanobu Abe, Satoshi Nakamura, Kiyohiro Shikano, and Hisao Kuwabara. Voice conversion through vector quantization. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pages 655–658. IEEE, 1988.

[3] Levent M. Arslan. Speaker transformation algorithm using segmental codebooks (stasc). *Speech Communication*, February 1999.

[4] Levent M. Arslan and David Talkin. Voice conversion by codebook mapping of line spectral frequencies and excitation spectrum. In *Proceedings of Eurospeech*, pages 1347–1350, Rhodes, Greece, 1997.

[5] Levent M. Arslan and David Talkin. Speaker transformation using sentence hmm based alignments and detailed prosody modification. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pages 289–292, 1998.

[6] G. Baudoin and Yannis Stylianou. On the transformation of the speech spectrum for voice conversion. In *Proceedings of the International Conference on Spoken Language Processing*, Philadelphia, USA, 1996.

[7] Christopher M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1st. edition, January 1996.

[8] Christopher M. Bishop. Variational principal components. In *Proceedings of the* 9^{th} *International Conference on Artificial Neural Networks*, volume 1, pages 509–514, 1999.

[9] Christopher M. Bishop, Marcus Svensén, and C. Williams. Gtm: The generative topographic mapping. *Neural Computation*, 10:215–234, 1998.

[10] Alan W. Black and P. Taylor. The festival speech synthesis system: System documentation. Technical Report HCRC/TR-83, Human Communication Research Centre, University of Edinburgh, Scotland, UK, January 1997.

[11] Mike Brookes. Voicebox: Speech processing toolbox for matlab. April 1998.

[12] F. Brugnara, D. Falavigna, and M. Omologo. Automatic segmentation and labeling of speech based on hidden markov models. *Speech Communication*, 12:357–370, 1993.

[13] Roberto Brunelli and Daniele Falavigna. Person identification using multiple cues. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 955–966, 1995.

[14] Christopher Chatfield and Alexander J. Collins. *Introduction to Multivariate Analysis*. Chapman and Hall, London New York, 1980.

[15] D.G. Childers, B. Yegnanarayana, and Ke Wu. Voice conversion: Factors responsible for quality. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, Tampa, March 1985.

[16] J.R. Crosmer. *Very Low Bit Rate Speech Coding Using the Line Spectrum Pair Transformation of the LPC Coefficients*. PhD thesis, Georgia Institute of Technology, 1985.

[17] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of Royal Statistical Society*, 39:1–38, 1977.

[18] J. Durbin. Efficient estimation of parameters in moving-average models. *Biometrika*, 46(1 and 2):306–316, 1959.

[19] M. Fried-Oken and H.A. Bersani. *Speaking Up and Spelling It Out*. Paul H. Brookes Publishing Co., 2000.

[20] Zoubin Ghahramani. Solving inverse problems using an em approach to density estimation. In *Proceedings of the 1993 Connectionist Models Summer School*, pages 316–323, Hillsdale, NJ, 1994. Erlbaum Associates.

[21] Benjamin Gillett. Transforming voice quality and intonation. Master's thesis, University of Edinburgh, The Centre for Speech Technology Research, January 2003.

[22] J.M. Gutiérrez, Y.S. Hsiao, J.M. Montero, J.M. Pardo, and D.G. Childers. Voice conversion based on parameter transformation. In *Proceedings of the International Conference on Spoken Language Processing*, Australia, 1998.

[23] J.M. Gutiérrez, J.M. Montero, J.A. Vallejo, R. de Córdoba, R. San Segundo, and J.M. Pardo. A new multi-speaker formant synthesizer that applies voice conversion techniques. In *Proceedings of Eurospeech*, pages 357–360, Denmark, 2001.

[24] Joseph F. Hair, Rolph E. Anderson, Ronald L. Tatham, and William C. Black. *Multivariate Data Analysis*. Prentice Hall, Upper Saddle River, New Jersey 07458, 4th edition, 1995.

[25] Harry Horace Harman. *Modern Factor Analysis*. University of Chicago Press, 3rd edition, August 1976.

[26] M. Hart. Project gutenberg. http://promo.net/pg, 2003.

[27] J. Hosom, A. Kain, T. Mishra, J. van Santen, M. Fried-Oken, and J. Staehely. Intelligibility of modifications to dysarthric speech. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, May 2003.

[28] Alexander Kain. *High Resolution Voice Transformation*. PhD thesis, OGI School of Science and Engineering at Oregon Health and Science University, 2001.

[29] Alexander Kain and Michael W. Macon. Spectral voice conversion for text-to-speech synthesis. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pages 285–288, 1998.

[30] Alexander Kain and Yannis Stylianou. Stochastic modeling of spectral adjustment for high quality pitch modification. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, 2000.

[31] N. Kambhatla. *Local Models and Gaussian Mixture Models for Statistical Data Processing*. PhD thesis, Oregon Graduate Institute of Science and Technology, January 1996.

[32] H. Kawahara. Speech representation and transformation using adaptive interpolation of weighted spectrum: Vocoder revisited. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, Munich, Germany, April 1997.

[33] Steven M. Kay. *Fundamentals of Statistical Signal Processing: Estimation Theory*. Prentice Hall Signal Processing Series. Prentice-Hall, Englewood Cliffs, NJ 07632, 1993.

[34] John Kominek and Alan W. Black. Cmu arctic databases for speech synthesis. Technical Report CMU-LTI-03-177, Carnegie Mellon University Language Technologies Institute, 2003.

[35] A. Kumar and A. Verma. Using phone and diphone based acoustic models for voice conversion: A step towards creating voice fonts. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pages 720–723, April 2003.

[36] H. Kuwabara and T. Takagi. Quality control of speech by modifying formant frequencies and bandwidth. In 11^{th} *International Congress of Phonetic Sciences*, pages 218–284, August 1987.

[37] P. Ladefoged and J. Ladefoged. The ability of listeners to identify voices. *UCLA Working Papers in Phonetics*, 49:43–51, 1980.

[38] Ki Seung Lee, Dae Hee Youn, and Il Whan Cha. Voice personality transformation using an orthogonal vector space conversion. In *Proceedings of Eurospeech*, volume 1, pages 427–430, Madrid, Spain, 1995.

[39] Ki Seung Lee, Dae Hee Youn, and Il Whan Cha. A new voice transformation method based on both linear and nonlinear prediction analysis. In *Proceedings of the International Conference on Spoken Language Processing*, Philadelphia, USA, 1996.

[40] Norman Levinson. The weiner rms (root mean square) error criterion in filter design and prediction. *Journal of Mathematical Physics*, 25(4):261–278, 1947.

[41] John Makhoul. Linear prediction: A tutorial review. *Proceedings of the IEEE*, 63(4):561–580, April 1975.

[42] H. Matsumoto, S. Hiki, T. Sone, and T. Nimura. Multidimensional representation of personal quality of vowels and its acoustical correlates. *IEEE Transactions on Audio and Electroacoustics*, 21(5):428–436, October 1973.

[43] Athanasios Mouchtaris, Shrikanth S. Narayanan, and Chris Kyriakakis. Maximum likelihood constrained adaptation for multichannel audio synthesis. In *36th IEEE Asilomar Conference on Signals, Systems, and Computers*, Pacific Grove, CA, November 2002. IEEE.

[44] Athanasios Mouchtaris, Shrikanth S. Narayanan, and Chris Kyriakakis. Multichannel audio synthesis by subband-based spectral conversion and paramater adaptation. *IEEE Transactions on Speech and Audio*, August 2002.

[45] Paul Mueller, Athanasios Mouchtaris, and Jan Van der Spiegel. Non-parallel training for voice conversion by maximum likelihood constrained adaptation. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, 2004.

[46] Ian Nabney. NETLAB*: Algorithms for Pattern Recognition*. Springer-Verlag, 1st. edition, 2001.

[47] Ian Nabney. NETLAB: Neural network software. October 2001.

[48] M. Narendranath, H. Murthy, S. Rajendran, and B. Yegnanarayana. Transformation of formants for voice conversion using artificial neural networks. *Speech Communication*, 16:207–216, 1995.

[49] Christina Orphanidou. Voice morphing. Master's thesis, Linacre College, University of Oxford, 2001.

[50] Mari Ostendorf, Patti Price, and Stefanie Shattuck-Hufnagel. The boston university radio news corpus. Technical Report ECS-95-001, Boston University, March 1995.

[51] K.K. Paliwal. Interpolation properties of linear prediction parametric representations. In *Proceedings of Eurospeech*, Madrid, Spain, 1995.

[52] Karl Pearson. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh and Dublin Philosophical Magazine and Journal of Science*, 2(6):559–572, 1901.

[53] Bryan Pellom and John H.L. Hansen. An experimental study of speaker verification sensitivity to computer voice-altered imposters. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, Phoenix, Arizona, March 1999.

[54] L.R. Rabiner and B.H. Juang. *Fundamentals of Speech Recognition*. Prentice Hall Signal Processing Series. Prentice-Hall, Englewood Cliffs, NJ 07632, 1993.

[55] L.R. Rabiner and R.W. Schafer. *Digital Processing of Speech Signals*. Signal Processing Series. Prentice-Hall, 1978.

[56] D.A. Reynolds and R.C. Rose. Robust text-independent speaker identification using gaussian mixture speaker models. *IEEE Transactions on Speech and Audio Processing*, 3:72–83, Jan. 1995.

[57] Stephen J. Roberts and William D. Penny. Mixtures of independent component analysers. In *International Conference on Artificial Neural Networks*, pages 527–534, 2001.

[58] Enders A. Robinson. *Multichannel Time Series Analysis with Digital Computer Programs*. Holden-Day, San Francisco, California, 1967.

[59] Joseph Rothweiler. On polynomial reduction in the computation of lsp frequencies. *IEEE Transactions on Speech and Audio Processing*, September 1999.

[60] Sam Roweis. Em algorithms for pca and spca. In *Neural Information Processing Systems 10*, pages 626–632, 1997.

[61] Ozgul Salor, M. Demirekler, and Bryan Pellom. A system for voice conversion based on adaptive filtering and line spectral frequency distance optimization for text-to-speech synthesis. In *Proceedings of Eurospeech*, pages 2417–2420, Geneva, Switzerland, September 2003.

[62] K. Shikano, K.F. Lee, and R. Reddy. Speaker adaptation through vector quantization. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pages 2643–2646, Tokyo, Japan, 1986.

[63] Yoram Singer and Manfred Warmuth. Batch and on-line parameter estimation of gaussian mixtures based on the joint entropy. In *Advances in Neural Information Processing Systems*, volume 11, pages 578–584, Cambridge, MA, 1998. MIT Press.

[64] Malcolm Slaney. Auditory toolbox. Technical Report 1998-010, Interval Research Corporation, 1998.

[65] F. Soong and B.H. Juang. Line spectrum pair and speech data compression. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 1.10.1–1.10.4, Tokyo, Japan, 1984.

[66] Yannis Stylianou and Olivier Cappe. A system for voice conversion based on probabilistic classification and a harmonic plus noise model. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pages 281–284, 1998.

[67] Yannis Stylianou, Olivier Cappe, and Eric Moulines. Statistical methods for voice quality transformation. In *Proceedings of Eurospeech*, Madrid, Spain, 1995.

[68] Yannis Stylianou, Olivier Cappe, and Eric Moulines. Continuous probabilistic transform for voice conversion. *IEEE Transactions on Speech and Audio Processing*, 6(2):131–142, March 1998.

[69] N. Sugamura and Fumitada Itakura. Speech data compression by lsp speech analysis-synthesis technique. *Transactions on Electronics and Communications in Japan*, 64A:599–606, 1981.

[70] Markus Svensén. *GTM: The Generative Topographic Mapping*. PhD thesis, Aston University, Birmingham, UK, 1998.

[71] T. Takagi and H. Kuwabara. Contributions of the pitch, formant frequencies, and bandwidth to the perception of voice-personality. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, Tokyo, Japan, 1986.

[72] Michael E. Tipping and Christopher M. Bishop. Mixtures of probabilistic principal component analysers. *Neural Computation*, 11(2):443–482, 1999.

[73] Michael E. Tipping and Christopher M. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society*, 21:611–622, 1999.

[74] Tomoki Toda, Jinlin Lu, Hiroshi Saruwatari, and Kiyohiro Shikano. Straight-based voice conversion algorithm based on gaussian mixture model. In *Proceedings of the International Conference on Spoken Language Processing*, 2000.

[75] Oytun Türk. New methods for voice conversion. Master's thesis, Boğaziçi University, 2003.

[76] Oytun Türk and Levent M. Arslan. Subband-based voice conversion. In *Proceedings of the International Conference on Spoken Language Processing*, pages 289–292, 2002.

[77] J. van Santen, L. Black, G. Cohen, A. Kain, E. Klabbers, J. de Villiers T. Mishra, and X. Niu. Applications of computer generated expressive speech for communication disorders. In *Proceedings of Eurospeech*, August 2003.

[78] T. Watanabe, T. Murakami, and M. Namba. Transformation of spectral envelope for voice conversion based on radial basis function networks. In *Proceedings of the International Conference on Spoken Language Processing*, Denver, Colorado, September 2002.

[79] Hui Ye and Steve Young. High quality voice morphing. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, 2004.

[80] Robert Zemeckis and Eric Roth. Forrest gump. PG-13, 1994.

[81] P. Zhan and M. Westphal. Speaker normalisation based on frequency warping. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, pages 1039–1042, 1997.

Biography

The author, Mark McMahon Wilde, was born in Metairie, Louisiana on September 10, 1980 where he was raised by Gregory Emmett Wilde, Sr. and Sharon Rose Wilde with brother Gregory Emmett Wilde, Jr. and sister Elizabeth Rose Wilde.

He attended St. Clement of Rome and Christian Brothers Academy for elementary school, and then became a Blue Jay at Jesuit High School where he played basketball, ran track, wrestled, and studied Latin and Greek. In August of 1998, he matriculated at Texas A&M University (Whoop!!). During this time, he worked four summer internships for Compaq Computer Corporation in both Dallas and Houston while learning the basics of computer science and electrical engineering during the academic years. In May of 2002, he graduated Magna Cum Laude with a Bachelor of Science degree in Computer Engineering. In the fall of 2002, he began Master's studies in Electrical Engineering at Tulane University.

In August 2004, Mark begins the Ph.D. program in Electrical Engineering at the University of Southern California working at the Signal and Image Processing Institute.

Mark McMahon Wilde

cordially invites

The faculty and students of the Tulane EECS Department

and the interested public

to a presentation of

his thesis

Controlling Performance in Voice Conversion with Probabilistic Principal Component Analysis

on

the Fourteenth Day of May, 2004 at 2:00 PM

in

Stanley Thomas Hall, Conference Room 201

Abstract

Voice conversion is the art of transforming one speaker's voice to sound as if it was uttered by another speaker. We have extended the state of the art in voice conversion by representing the joint probabilistic acoustic space of the two speakers with a mixture of Probabilistic Principal Component Analyzers (PPCAs). In past work, when modeling with a Gaussian Mixture Model, only two extremes were possible — one had to restrict the covariance matrix of each component in the mixture to have all its entries or entries only on the diagonal. When using a diagonal covariance for each component of the mixture, the model is not able to capture enough underlying structure; thus, the model results in poorer conversion. In addition, using diagonal covariances asserts the unfounded assumption that the features of each speaker are decorrelated. In contrast, when using a full covariance matrix for each component in the mixture, we can model the underlying structure well. Hence, conversion performance improves, but the tradeoff is that complexity of training and conversion increases an order of magnitude.

When modeling the probability density with a mixture of PPCAs, we are able to present a finer resolution of options to the end user of the voice conversion system. Objective experiments demonstrate that the dimension of the PPCA chosen directly impacts the resulting objective performance with the tradeoff of a savings in both time and memory complexity. Subjective tests imply that this incremental removal of information does not make a difference perceptually to the listener when the increments are small. Thus, the end user can select with a greater degree of freedom how well the system should perform depending on the application.