DEVELOPING SPARSE REPRESENTATIONS FOR ANCHOR-BASED VOICE CONVERSION

A Dissertation

by

CHRISTOPHER BRYANT LIBERATORE

Submitted to the Graduate and Professional School of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee, Ricardo Gutierrez-Osuna
Committee Members, Yoonsuck Choe
Dylan Shell
Byung-Jun Yoon
Head of Department, Scott Schaefer

December 2021

Major Subject: Computer Science

Copyright 2021 Christopher Bryant Liberatore

ABSTRACT

Voice conversion is the task of transforming speech from one speaker to sound as if it was produced by another speaker, changing the identity while retaining the linguistic content. There are many methods for performing voice conversion, but oftentimes these methods have onerous training requirements or fail in instances where one speaker has a nonnative accent. To address these issues, this dissertation presents and evaluates a novel "anchor-based" representation of speech that separates speaker *content* from speaker *identity* by modeling how speakers form English phonemes.

We call the proposed method Sparse, Anchor-Based Representation of Speech (SABR), and explore methods for optimizing the parameters of this model in native-to-native and native-to-nonnative voice conversion contexts. We begin the dissertation by demonstrating how sparse coding in combination with a compact, phoneme-based dictionary can be used to separate speaker identity from content in objective and subjective tests. The formulation of the representation then presents several research questions. First, we propose a method for improving the synthesis quality by using the sparse coding residual in combination with a frequency warping algorithm to convert the residual from the source to target speaker's space, and add it to the target speaker's estimated spectrum. Experimentally, we find that synthesis quality is significantly improved via this transform. Second, we propose and evaluate two methods for selecting and optimizing SABR anchors in native-to-native and native-to-nonnative voice conversion. We find that synthesis quality is significantly improved by the proposed methods, especially in native-to-

nonnative voice conversion over baseline algorithms. In a detailed analysis of the algorithms, we find they focus on phonemes that are difficult for nonnative speakers of English or naturally have multiple acoustic states. Following this, we examine methods for adding in temporal constraints to SABR via the Fused Lasso. The proposed method significantly reduces the inter-frame variance in the sparse codes over other methods that incorporate temporal features into sparse coding representations.

Finally, in a case study, we examine the use of the SABR methods and optimizations in the context of a computer aided pronunciation training system for building "Golden Speakers", or ideal models for nonnative speakers of a second language to learn correct pronunciation. Under the hypothesis that the optimal "Golden Speaker" was the learner's voice, synthesized with a native accent, we used SABR to build voice models for nonnative speakers and evaluated the resulting synthesis in terms of quality, identity, and accentedness. We found that even when deployed in the field, the SABR method generated synthesis with low accentedness and similar acoustic identity to the target speaker, validating the use of the method for building "golden speakers".

DEDICATION

Dedicated to those who believed in me, even when I didn't.

ACKNOWLEDGEMENTS

I am forever indebted to my advisor, Dr. Ricardo Gutierrez-Osuna, for giving me the opportunity to study and become a researcher. I have grown in ways that I haven't imagined and had experiences far beyond research. These opportunities opened my mind to new ideas and experiences, and I find myself understanding and loving my fellow man more than I had thought possible. Our differences and unique backgrounds, perspectives, and thoughts, all make us who we are, but we are alike in more ways than we are not.

I am grateful for our collaborators at Iowa State University, Drs. John Levis and Evgeny Chukharev-Hudilainen for their support, opinions, critiques, and reviews.

My time at Texas A&M was blessed by having the company of wonderful colleagues and labmates: Sandesh, Jin, Virendra, Avinash, Anshul, Michael, Guanlong, Adam, Tian, Roger, Dennis, Shaojin, Nitin and Raniero. Your friendship and warmth made my life immeasurably better and I am indebted to you for it.

I wish to express thanks to my colleagues at Air Force Research Laboratory: Todd, Vince, Clare, Olga, and Ben, who were all instrumental recruiting and tutoring me. I look forward to more years of collaboration.

I am also grateful to my undergraduate advisors and professors, Dr. Scott Gordon and Dr. Michelle Norris. Without your guidance and encouragement, I would not have found myself with the background I needed to go into machine learning and research. To Mike Vollmer, thank you for even suggesting that afternoon in the Spring of 2012 after

Dr. Gordon's AI class that, maybe, we had the ability to be successful in graduate school. That simple suggestion changed the course of my career.

To my dear friends outside of academia, Chris, Brian, Laird, Sam, Michael, Micaela, Adam, Guanlong—your support and encouragement were unwavering. Thank you for putting up with my needless mathematical rants, my fixation on unimportant things I found interesting, or reminding me that "it's just regression". To Brian, I request forgiveness: I did not figure out how to name an algorithm "HUSKY".

Finally, I would also like to thank my family. Without your unwavering love, support, and encouragement, I would not be where I am now. I was blessed with your guidance growing up, and your willingness to help me learn and grow gave me opportunities I recognize were special and important. To my father, Stephen, thank you for teaching me the importance of hard work and dedication. To my mother, Tamar, thank you for teaching me the importance of unconditional love. To my sisters, Joy and Hannah, thank you for keeping me both honest and humble—your sass never goes unappreciated.

This dissertation was written during the COVID-19 pandemic. This difficult time helped highlight the important things in life. I am grateful for those who dedicated their time and efforts to keeping our lives safe and healthy.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supervised by a dissertation committee consisting of Professors Ricardo Gutierrez-Osuna, Yoonsuck Choe, Dylan Shell of the Department of Computer Science and Engineering, and Professor Byung-Jun Yoon of the Department of Electrical and Computer Engineering.

The computer aided pronunciation system in Chapter 7 was built in 2016 and 2017 with Shaojin Ding. Other people who contributed to this research include Sandesh Aryal, Seth Posley, Zelun Wang, Shaojin Ding, and Guanlong Zhao.

All other work conducted for the thesis (or) dissertation was completed by the student independently.

Funding Sources

This dissertation was supported by two National Science Foundation research grants (1619212 and 1623750), the SMART Scholarship for Service Program, and Air Force Research Laboratory (under SEED and SLAKE contracts). The opinions expressed herin are solely the author's and do not necessarily reflect the views of the funding sources.

TABLE OF CONTENTS

Page

LIST OF FIGURES

Page

LIST OF TABLES

Page

1. INTRODUCTION

Speech is the convolution of two components: a source signal generated by the glottis (e.g. “voice box”) and a filter controlled by the articulations of the vocal tract. The identity of a speaker is contained in both the source signal (the speaker’s pitch and range) as well as their filter (affected by the size and articulations of their vocal tract). Linguistic content (*what* a speaker said) is a combination of these features. Many speech processing tasks are interested in separating these components and evaluating them; for example, Automatic Speech Recognition (ASR) treats the speaker’s identity effects on the speech signal as noise to be ignored, focusing on the linguistic content. Alternatively, speaker identification systems use the signal to identify the speaker, but treat the linguistic content as less important.

Voice Conversion (VC) methods are concerned with modifying the speaker identity of an utterance while retaining the linguistic content. To change the identity, the two major components of the speech signal must be converted from that of the source speaker to that of the target speaker. First, the pitch (i.e. fundamental frequency) is modified to be in the same range as the target speaker. This is typically achieved through log mean and variance scaling [1-3]. However, converting the spectral envelope (i.e. the filter) is far more involved. Previous work in spectral conversion used statistical regression (i.e. Gaussian Mixture Models) on parallel corpora to learn a mapping between a pair of source and target speakers [2]. These methods can successfully convert spectral envelopes from a source to target speaker, but due to the statistical nature of the regression, they suffer from “over-smoothing” effects, resulting in a “muffling” effect in the synthesized

speech. Different methods have been proposed to solve this problem and increase spectral detail [4-6], but they require increasing amounts of training data to build a voice conversion model. Sparse-coding-based spectral conversion methods have also been proposed as a solution to this spectral detail problem [7, 8]. These methods decompose speech as a sparse, linear combination of exemplars from a speaker (e.g. speech frames from a number of utterances). This decomposition, when combined with aligned exemplars from a target speaker, can be used to perform voice conversion. Typically, these methods build source and target dictionaries using time-aligned data from the two speakers. In practice, it is not always practical to collect parallel utterances from two speakers to learn a mapping between two speakers. Furthermore, there are instances where one of the speakers may be unable to pronounce parallel utterances in the same manner as the other (e.g., when one speaker has a non-native accent), further confounding existing VC methods. In these instances, alternative methods for performing voice conversion without requiring parallel data, or instances where data collection is limited, would be useful.

One alternative to using aligned utterances to build voice conversion models between speakers would be to learn "anchors" of speakers' voices, representing how a speaker forms particular sounds (e.g. phonemes) in the acoustic space. In this dissertation, we use this rationale to represent a speaker as a collection of canonical productions of different phonemes, where each phoneme represents an anchor for that speaker. Though two different speakers will produce different spectral envelopes for the same phoneme, they will agree upon the linguistic content of that sound. Additionally, using phoneme-

oriented anchors allows for more compact speaker representations, resulting in models that are less sensitive to limited training data. In a similar manner to other sparse-coding methods, we use these anchor-based models to represent their speech relative to these anchors, representing each speech frame as a sparse, linear combination of the speaker's anchors. Given different anchor models from two speakers, we show that the learned weights possess speaker-independent properties, effectively separating *who* said an utterance (in the form of the anchors) from *what* was said (in the form of the weights).

This dissertation concerns the implementation and optimization of an anchor-based voice conversion system based on the above intuition. This representation leverages sparse coding algorithms to represent a source speaker's spectral envelope as a linear combination of their anchors; the learned sparse codes are then used in combination with a target speaker's anchor set to estimate the target speaker's spectral envelope. In addition to proposing an implementation of this system, we also optimize the components of the system and evaluate it in both native-to-native and native-to-nonnative voice conversion tasks.

The specific aims of this dissertation are:

1. *Develop a framework for anchor-based voice conversion.* We use sparse-coding methods to perform spectral envelope conversion as part of a voice conversion system. Initially, we will use the centroids of phonemes to build this framework—one anchor per phoneme.
2. *Use the residual to improve spectral detail*. Because of the compact anchor set, the converted utterances will lack spectral detail. We propose and develop a method

to use the source residual to increase the detail of the target speaker's spectral envelope.

3. *Select optimal anchor sets*. A single anchor per phoneme may not represent some phonemes adequately (e.g. stops); alternatively, some phoneme anchors may be unnecessary or redundant (e.g. affricates or diphthongs). We propose and study two different techniques for optimizing the anchor sets—selecting the appropriate number of anchors and optimizing the anchors for two different speakers.
4. *Add temporal smoothness constraints to the representation*. Adding temporal constraints to the objective function will ensure that the sparse codes are temporally smooth, increasing the interpretability of the weights and potentially improving the voice conversion quality.
5. *Case Study: Use the anchor-based voice conversion to build Golden Speakers*. Prior work suggests that pronunciation training could be best accomplished by training a second-language learner with their own voice without a non-native accent. In this aim, we use the anchor-based voice conversion method "in the field" a pronunciation-training tool to allow non-native speakers of English to hear their voice, but without an accent.

1.1. Dissertation Outline

The remainder of this dissertation is organized accordingly. First, it presents an overview of speech production and perception, how this relates to voice conversion and accent conversion and related literature, and an overview of the proposed voice conversion framework which will be used to answer the above research aims. Chapter 3 presents in

detail the implementation of a Sparse, Anchor-Based Representation of Speech (SABR), and objective and subjective evaluations of its ability to separate voice identity from content. These findings were published at the 2015 Interspeech conference [9]. In Chapter 4, we propose and evaluate a method for using the sparse coding residual and a frequency warping method to improve the synthesis quality of SABR. The residual warping method was presented at ICASSP 2018 conference [10], expanded into a journal chapter in 2021 with a more thorough analysis of frequency warping methods and the effects on native-to-nonnative VC, and submitted to IEEE/ACM Transactions on Audio, Speech, and Language Processing in 2021. Chapter 5 presents two methods for optimizing SABR anchors for use in native-to-native and native-to-nonnative voice conversion. This chapter was submitted to IEEE/ACM Transactions on Audio, Speech, and Language Processing, once in 2019 and revised in 2020. The ARS algorithm was published in Interspeech, 2021. In Chapter 6, we present a modification to the sparse coding objective function from Chapter 3 to include temporal constraints. This chapter was submitted to Interspeech 2019 and ICASSP 2019.

In Chapter 7, we perform a case study where we apply the proposed SABR system in a Computer-Aided Pronunciation Training (CAPT) system and evaluate the performance of SABR using speech collected from nonnative speakers in a pedagogical context. This chapter is part of a journal chapter published in the journal Speech Communication [11], in which SABR was used to generate synthesis for learners to practice their accent; the full chapter comprises a discussion of the application, the signal processing backend, and the learning outcomes of participants who used the system. Here,

we focus on the signal processing aspect of the CAPT tool and the performance of SABR on the voices of the learners who used the tool. Chapter 8, concludes this dissertation with a review of the findings and directions for future work.

2. BACKGROUND AND RELATED WORK

2.1. Speech Production

The Acoustic Theory of Speech Production describes speech acoustics as the consequence of four components of the vocal tract [12]. The first component is a *sound source*, which is either a periodic signal generated by the vibration of the glottis (voice box) or a turbulent airstream. The second component is the *vocal tract filter*, which generates resonances that modulate the source signal. The third component, *energy losses*, affect the acoustic structure of speech sounds. And the final component, *radiative effect*, arises from the fact that the speech signal radiates from the mouth. The first two components are largely responsible for the time-varying components of speech sounds and can be in general evaluated independently from each other. For brevity, we will focus on these two components in this discussion. A cross-sectional illustration of a vocal tract model (VocalTractLab, [13]) can be seen in Figure 1, with an illustration of the vocal tract cavity and common articulators.

The source of speech production begins at the glottis, or vocal folds, and can either release air from the lungs without vibrating (producing turbulent airflow for *voiceless* sounds) or vibrating (producing a periodic signal, perceived as pitch, for *voiced* sounds)[14]. The frequency of this periodic signal, or the pitch, varies over time for emphasis or part of speech[1]. The source signal is then modified by the vocal tract filter. This component is called the *filter* because the cross-sectional shape of the vocal tract

[1] While these two classes of source signals do not consider the full variety of source sounds that the glottis can produce, they account for the phonetic differences we consider in this work.

cavity creates resonant frequencies, amplifying or attenuating different frequencies of the source signal from the glottis. Moving the articulatory features of the vocal tract (e.g. the jaw, the tongue, the lips) create different resonances, changing the output acoustics. The time-varying movement of these articulatory features creates a time-varying filter, creating different resonances in the acoustic signal.

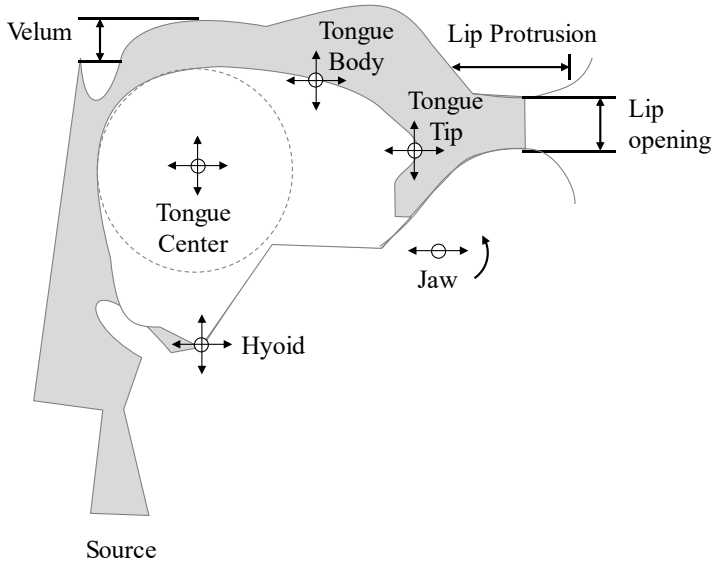


Figure 1: Cross section of a vocal tract model to illustrate vocal tract features and articulators.
The vocal tract cavity is represented by the shaded section. Also shown are articulators for the physical vocal tract model VocalTractLab [13], reprinted with permission from [15].

The combination of these two components form the basis of phoneme categorization [16]. Phonemes represent the smallest component of speech sounds and linguistic content. Phonemes are categorized into two broad categories: *vowels*, which have voicing and unimpeded airflow, and *consonants*, where the airflow is restricted in some manner and voicing can vary [17]. From an articulatory perspective, vowels are

categorized by the location of the tongue tip (i.e. how low and how far forward the tongue tip is) and the body of the tongue must allow for nonturbulent airflow. Because airflow is unimpeded, the vocal tract cavity forms a filter with strong resonances known as *formants*. Perceptually, vowels are distinguished by the frequency locations of the first two or three formants [18, 19]. Consonants are categorized by *place* and *manner* of articulation. The *place* and *manner* of articulation refer to the place of the tightest constriction in the vocal tract, and the type of constriction. The *place* of restriction is typically classified according to the physiological feature on the hard or soft palate (e.g. roof of the mouth) where the narrowest cross-section of the vocal tract cavity occurs. The *manner* of articulation refers to the airflow that results from this cavity. Constants can be either *voiced* or *unvoiced* (i.e., the glottis can be producing a periodic source signal with a pitch, or turbulent airflow which lacks such) [17].

For the most part, these two features—the source signal and the vocal tract filter—can be treated independently, and many speech processing tasks try to separate the two components for easier analysis. However, both of these components carry an element of speaker identity that makes it more difficult to separate identity from content. The pitch, or fundamental frequency of the source signal, also affects *speaker identity*, as different speakers have different distributions and ranges of pitch. In practice, simple log-scale mean-variance scaling is enough to transform the pitch range of one speaker to that of another [3]. Modifying the filter is more complex, as both the linguistic content and speaker identity are encoded in the filter. The front cavity hypothesis [18] suggested that the *front* of the vocal tract cavity encodes linguistic content, and the *back* of the vocal tract

cavity contributes mainly speaker identity. However, these two features interact in the spectrum in complex, non-trivial manners, and therefore the overall vocal tract filter also carries speaker identity. For speech processing tasks which require the modification of the identity of a speaker (such as voice conversion) the filter must also be changed to match the desired target speaker identity.

2.2. Speech processing methods

2.2.1. *Analysis and synthesis of speech*

Speech processing methods can be broken into two broad categories of algorithms: *analysis* and *synthesis* of the speech signal. *Analysis* is the decomposition of the speech signal into its constituent parts—typically a representation of the pitch and spectral envelope. These two components correspond to the components in the source-filter model—the pitch representing the source signal, and the spectral envelope representing the resonances of the vocal tract, which also implicitly represents the vocal tract cavity. Analysis methods typically assume that the glottis and vocal tract cavity are fully decoupled, meaning that the source signal and the resonances can be treated independently. Principally, these methods first estimate the pitch and its harmonics (source signal), then estimate the resonances (the spectral envelope). Once these components are extracted, further analysis or modification can be performed [20].

Synthesis algorithms operate in the opposite direction of analysis algorithms, typically simulating the source and filter components to generate speech acoustics. Formant synthesizers are the most basic of the acoustic synthesis techniques, consisting of individually adjustable filters which simulate resonances in the vocal tract. Different

resonance frequencies and bandwidths correspond with different phoneme units and if these resonances are updated at short intervals (e.g. 5ms), continuous speech can be generated when this filter is driven with a model of the source (e.g. a voiced or unvoiced source signal) [19]. Similarly, Linear Prediction Coefficient (LPC) synthesizers use the source-filter model to generate speech acoustics, but as opposed to formant synthesizers, short frames of speech can be used to estimate LP coefficients. These components have been used in the past for VC as they parameterize the spectral envelope with a handful of coefficients that can be easily transformed. As with the formant synthesizer, when an LPC synthesis filter is excited with an appropriate source signal, intelligible speech can be generated [21].

Alternatively, some synthesis methods do not directly generate the source and filter components to generate realistic synthesis, but still leverage this aspect of speech to generate realistic acoustics. Concatenative synthesis uses samples of speech acoustics combined with a desired sequence of phonemes to stitch together an arbitrary utterance. The database of speech samples can be arbitrarily large and contain very low level (i.e. phoneme) or high-level (syllable or word) units, effectively capturing a wide variety of intonations. To ensure the speech sounds natural, temporal and pitch scaling of adjacent speech samples is necessary. One method for doing this, Pitch Synchronous Overlap and Add (PSOLA) [22], finds the parameters for the spectral envelope at each mark in the pitch sample. The method is pitch synchronous as it generates a window of acoustics for each pitch mark in the source signal. Given a window of the source signal, centered on the pitch, and the corresponding filter at the pitch time, the single pitch impulse could be run

through the filter and an estimate of the acoustics for that window could be generated. Each overlapping window was then added together to generate a full time-domain signal.

Later synthesis methods, such as STRAIGHT, used more complex representations and did not rely on an estimation of the source signal—just an estimate of the corresponding pitch frequency, spectral envelope parameters, and a representation of "aperiodic components" (i.e. features that cannot be explained by periodic features). Recently, vocoders built on deep neural network architectures have shown themselves to be even more effective at generating high-quality speech synthesis using spectral envelope and pitch parameters [23].

2.2.2. *Separating speaker identity and content*

Different speech tasks require the separation of speaker-dependent cues (e.g., identity) from speaker-independent cues (e.g., linguistic information) from the speech signal. In automatic speech recognition (ASR), speaker variability is viewed as unwanted noise (i.e. linguistic content); in VC, one seeks to modify speaker-dependent cues while retaining the linguistic content of the utterances. Several techniques have been developed to remove the influence of speaker identity in speech, but there are two broad categories for removing identity from speech. The first is directly transforming the spectrum to minimize speaker dependencies using techniques such as vocal tract length normalization [24, 25] and speaker adaptation [26]. These methods transform the parameters of a model (e.g., an ASR) to be closer to that of a specific speaker, then use the transformed model in a speaker-specific task [27, 28]. The second class of methods is to project speech into a

latent, linguistic space, where the speaker-specific variations from the speech signal are removed.

One approach to projection is to map acoustics into the articulatory feature space. Though speakers have different vocal tract parameters, the articulatory configuration for the same linguistic content will be agreed upon between two different speakers, allowing for speaker-independent representation [29, 30]. As an example, Frankel et al. [31] trained multi-layer perceptrons to estimate phonological articulatory features (e.g. place, manner, nasality, etc.) from the cepstrum. When they combined the estimated articulatory features with acoustic features, word error rate dropped from 67.7% to 59.7% in a speaker-independent phoneme classification task. Arora and Livescu [32] used canonical correlation analysis (CCA) of simultaneous acoustic and articulatory recordings to capture the common factor (i.e. linguistic content) in these two views. The authors learned CCA transforms from a group of speakers and used them to extract linguistic features from acoustics in a speaker-independent fashion. CCA features improved the accuracy by up to 23% in a speaker-independent phoneme recognition task.

An alternative to using articulatory features is to use linguistic information learned from speech recognition systems. The Kaldi ASR system [33] uses a 4-layer Deep Neural Network (DNN) to classify windows of speech into different subphoneme states (known as “senones”). These subphoneme states are similar to the aforementioned articulatory configurations, representing a latent space where linguistic content is learned over speaker identity. Because the network is trained on many different speakers, it can learn speaker-independent representations of speech content [34, 35].

2.3. Voice Conversion

In this section, we discuss standard frameworks of VC systems and a review of prior work in VC. We also discuss prior work in Accent Conversion (AC) and its relationship with previous VC systems and speech production as a whole.

2.3.1. *Voice conversion systems*

VC methods take an input speaker utterance and modify it such that the *speaker identity* of the utterance is changed while the *linguistic content* is retained [36]. To do this, the two primary components of the speech signal must be transformed to change the identity of the source speaker to that of the target speaker: the pitch and the spectral envelope. First, the pitch (i.e. source signal) is modified to be in the same range as the target speaker. This is typically through log mean and variance scaling [1-3]. Given the source and target speakers' log pitch means and variances μ_s, μ_t, σ_s, and σ_t, scaling a source speaker's pitch F_0^S to the target speaker's range follows:

$$F_0^T = \exp\left(\frac{\log(F_0^S) - \mu_s}{\sigma_s}\sigma_t + \mu_t\right).$$

As stated previously, converting the spectral envelope (i.e. the filter) is more involved. Representations of the spectrum tend to have far more features than the pitch and this detail is required for the resulting speech to be rich and high quality. Full spectra are high-dimensional (for many conversion systems, 512 dimensions or higher) and these features are often highly-correlated, making it difficult for statistical conversion methods to capture spectral detail [2]. Efforts must be made to retain spectral detail and to capture the identity of the target speaker [5-8, 37, 38].

Figure 2 contains an overview of the components of a voice conversion system. This figure illustrates the STRAIGHT vocoder framework [39], but this formulation is similar to many other VC systems and toolkits (e.g. WORLD [40]). The STRAIGHT framework includes two modules: an *analysis* module, that separates a speech signal into source, filter, and aperiodic parameters, and a *synthesis* module, that takes these three parameters and synthesizes a speech signal from it. In the *analysis* module, the three components STRAIGHT generates are:

- the *pitch* (in Hz if voicing is present, 0 if unvoiced)
- the *spectral envelope*, without the pitch harmonics
- and the *aperiodicity* (AP; the magnitude of spectral energy that cannot be explained by periodic components, i.e. the pitch).

To perform VC, one must convert the three components such that they retain the linguistic content of the source speaker, but have the identity of the target speaker. Pitch is converted typically using log mean and variance scaling [2, 5], as mentioned previous. In practice, aperiodicity (AP) is less important to voice identity, but band scaling is a common technique for converting AP from a source to target speaker [41].

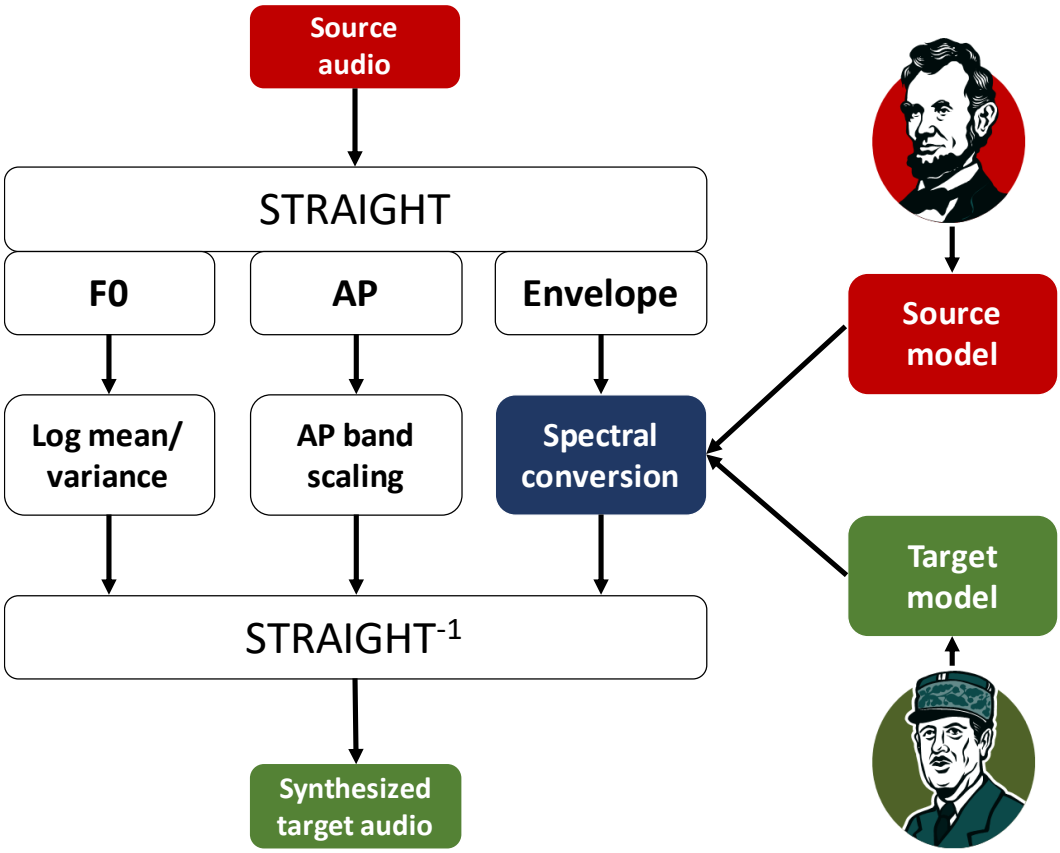


Figure 2: overview of voice conversion system using STRAIGHT. This dissertation focuses on developing a spectral conversion method, shaded in blue.

2.3.2. *Spectral conversion methods*

Because speaker identity and speaker content are tied in the spectral envelope, spectral conversion methods in VC typically rely on learning how source and target speakers form the same linguistic content (e.g., how they form a particular phoneme), then using an encoding of the linguistic content extracted from the source speaker, drive the target speaker's acoustic model [36]. Some VC methods will explicitly learn an encoding based on linguistic content (e.g., using phoneme labels [42]), whereas others use time-aligned, parallel utterances between source and target speakers, relying on the alignment to match the spectrum of the source speaker to that of the target speaker [2, 3]. Once this

alignment is learned, regression can be used to estimate the target speaker's spectrum from a sample from the source speaker.

One of the earliest spectral conversion algorithms used vector quantization (VQ) to learn a source-target mapping of spectral features. Abe *et al.* [43] used codebooks learned from time-aligned source and target training data to learn a mapping from the source speaker's pitch and spectrum to the target speaker. In perceptual tests, listeners identified the synthetic speech as being much more like the target speaker. However, the VQ method resulted in discontinuous trajectories in the pitch and spectral parameters, resulting in distortions and lower quality synthesis.

To deal with the discontinuity issues associated with the VQ method, Stylianou *et al.* [2] proposed using a statistical regression algorithm using Gaussian Mixture Models (GMMs). This method greatly improved quality of these conversions, allowing for smoother parameter trajectories between the GMM mixture centers. Using 3.5 minutes of parallel source and target training data, this method outperformed vector quantization methods in objective and subjective tests. However, this statistical regression method introduced a problem where the spectrum of the target speaker lacked the variance of the target—known as "oversmoothing." One technique for solving this problem was proposed by Toda et al. [6], called Maximum Likelihood Parameter Generation. In addition to the GMM model, the authors added delta and delta-delta features to the source and target datasets to not just perform spectral conversion, but to estimate the *trajectory* of the target spectrum. Combining this with a maximum-likelihood estimation algorithm, the estimated trajectory of the target spectra could be used to increase the spectral variance of the method

to more closely match the trajectories of the target speaker. However, these solutions increased the amount of training data required for high-quality conversions, making it difficult to use these conversion methods in "real world" settings.

While the use of statistical regression is one source of oversmoothing in statistical regression, another source is the use of compressed spectral features in GMM conversion. Methods which have attempted to solve the oversmoothing issue have focused on using full spectra in the conversion process, as opposed to a compressed representation as the GMM methods often use. Frequency Warping and Amplitude Scaling (FW+AS) is one method for using full spectra in VC [4, 5, 37, 38]. Frequency warps are functions that build an invertible transformation of a source spectrum to align it with the energy of a target spectrum, and can be thought of as "stretching" or "squashing" the spectrum between two frequencies. These methods operate by using frequency warping to adjust the formant locations of the source speaker to be closer to the target spectrum. Because frequency warps cannot account for all spectral differences between source and target speakers, another module is required to match the target speaker's spectral energy (a so-called "Amplitude Scaling" module) is included to adjust the warped source spectrum to be closer to the target speaker [4, 5, 37] The net result of these transforms is that all the spectral detail is retained, but the distribution of the details can be changed, preserving more spectral detail than other conversion methods [44]. Partial Least Squares (PLS) methods have also been explored to perform VC on full source spectra, with the same motivation to solve the statistical oversmoothing problem. In [45], the authors examined the use of a kernel-based Partial Least Squares method to learn projections of the source

speaker to the target speaker's spectrum while using the full spectrum and not compressing it. In this method, the authors used a kernel representation of the aligned source and target training data to learn a PLS transform on the original source spectrum. In subjective and objective tests, the authors found that their proposed PLS method performed significantly better than the GMM-based baseline comparison method.

2.3.3. *Exemplar-based voice conversion*

Another technique for solving the GMM oversmoothing issue came with the use of Exemplar-Based VC methods. Exemplar-based methods perform VC using sparse coding and an exemplar dictionary for both the source and target speakers. In the training phase, source and target speaker dictionaries are built from time-aligned source and target spectra, A and B. During conversion, the source utterance X is decomposed into a set of sparse codes H (usually via Nonnegative Matrix Factorization; NMF) using the source speaker's exemplar dictionary. Then, an estimate of the target speaker's utterance $\hat{Y}$ is obtained by multiplying the source speaker's sparse codes with the target speaker's dictionary [9]. An overview of these methods is shown in Figure 3. These exemplar-based methods work well with limited training data [8, 35, 38, 46-52] and are more robust to noise [53] than methods such as GMMs.

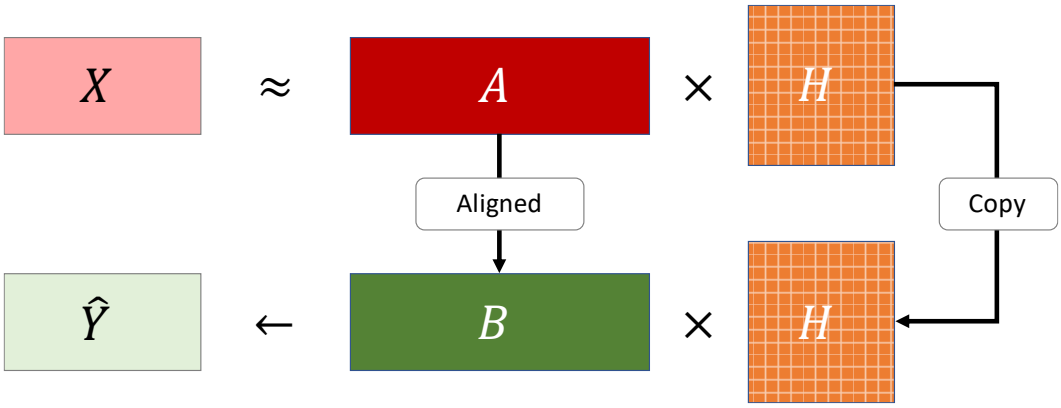


Figure 3: overview of exemplar-based VC methods.

One of the first uses of exemplar-based voice conversion was presented in [8], leveraging the use of Nonnegative Matrix Factorization in noisy environments. The authors proposed using parallel source and target dictionaries to learn both an encoding of a source speaker's utterance and then to estimate the spectrum of a target speaker. One of the motivations for the use of this method was to be able to use full spectrum exemplars to perform the synthesis, as opposed to a compressed version of the spectrum as was typical in statistical conversion methods. The authors found that their proposed method outperformed a GMM-based synthesis method considerably. Aihara *et al.*[48] proposed a technique for performing many-to-many voice conversion by using a multi-speaker dictionary to represent unseen source and target speakers. Dictionaries for unseen speakers were assembled from linear combinations of speakers in the multi-speaker dictionary. In perceptual studies, using this additional data allowed the many-to-many exemplar VC method to significantly outperform a one-to-one GMM method in objective and subjective experiments.

Even for time-aligned, parallel source and target training data, parallel utterances typically having different encodings for source and target speakers. Synthesis methods which rely on these encodings can be affected by these mismatches, potentially lowering synthesis quality or changing speaker identity. One way to constrain these differences is to add phoneme information in the dictionary design and objective functions. Aihara *et al.* [47] proposed an "activity mapping" method for exemplar-based VC in which the source and target dictionaries were given an additional phonetic label, ensuring that exemplars from the source and target were used in coding and synthesis. The authors found that by including this constraint, spectral distortion between the estimated target speaker's spectrum and the ground truth was lowered, and listeners significantly preferred synthesis from the constrained dictionaries over the unconstrained dictionaries. Similarly, Ding *et al.* [54] proposed adding a Phoneme-Selective Objective Function to an exemplar-based VC system, which used a joint L1-L2 group-sparsity penalty to first select a phoneme subdictionary to perform sparse coding and VC. Including the constraint significantly improved synthesis quality in objective and subjective tests. In [55], Sisman *et al.* sought to avoid the parallel training requirement for exemplar dictionaries, instead learning dictionaries based on phonetic features. Differently than prior studies, the authors created a joint dictionary of spectral and phonetic posteriorgrams (PPGs), the latter derived from a deep-learning-based ASR system. At runtime, the authors extracted PPGs from the source speaker's utterance and performed sparse coding using the *target* speaker's PPGs, under the assumption that PPGs were speaker independent. This activation matrix was then applied to the target speaker's spectral dictionary with a residual compensation

component. The resulting synthesis outperformed a baseline method in subjective and objective tests, further demonstrating that including phoneme information in exemplars improves synthesis quality over time-aligned training data.

2.4. Accent Conversion

Accent conversion (AC) seeks to synthesize speech with the voice quality of a nonnative speaker (L2), but the accent of a native speaker (L1). As such, AC is closely related to voice conversion. While traditional VC uses regression methods to convert prosodic and segmental cues, AC has the additional, difficult task of correcting for mispronunciations in the L2 speaker (e.g., phoneme substitutions, additions, and deletions) [56, 57]. In some cases, the L2 speaker may not have desired phonemes in their inventories, so estimating those phonemes becomes necessary [34, 58, 59].

In early work, Yan et al. [60] used an HMM synthesis method to transform vowels of three major regional English accents (British, Australian, and General American). The authors built statistical distributions of the first three formants of English vowels from the three accents and developed an accent synthesis system that would transform the formants of one English accent to another. In an ABX test, 78% of Australian-to-British accent conversions were perceived as having a British accent, and 71% of the British-to-American accent conversions were perceived to have an American accent. In both cases, changing prosody alone (pitch and duration) led to noticeable changes in perceived accent, though not as significantly as formant modifications. Some studies have attempted to blend L2 and L1 spectra instead of replacing them entirely. Huckvale and Yanagisawa [61] reported improvements in intelligibility for Japanese utterances produced by an

English text-to-speech (TTS) after blending their spectral envelope with that of an utterance of the same sentence produced by a Japanese TTS. In [62], the authors proposed a voice morphing strategy, separating spectral detail (carrying linguistic content) and spectral slope (carrying speaker identity). AC was achieved by replacing the spectral detail of an L2 speaker with that of a native L1 speaker. In perceptual studies, listeners rated the AC utterances as being much more native-sounding, but the morphing technique affected the identity of the synthesis.

Accent Conversion algorithms have also been examined in the articulatory domain. In [63], Felps and Gutierrez-Osuna built a joint model of articulatory and acoustic data from an L1 and an L2 speaker and used it to identify mispronounced diphones in an L2 utterance. These misidentified segments were replaced with other L2 diphone segments whose articulatory configuration was similar to the reference L1 articulations. However, this method performed poorly when particular L1 diphones were not in the L2 speaker's inventory. To address this issue, Aryal et al. [58] used Gaussian Mixture Models (GMM) to build a statistical articulatory synthesizer, which was then able to synthesize phonemes not observed in the L2 speaker. The authors normalized the L1 articulatory features into the space of the L2 speaker, then drove the GMM with the normalized L1 articulatory features. This method significantly reduced the perceived nonnative accents while preserving the voice quality of the L2 speaker.

2.4.1. *Accent conversion vs. voice conversion*

Accent conversion is closely related to the problem of voice conversion (VC) [36]. Voice conversion transforms utterances from a source speaker to appear as if a (known)

target speaker had produced them. To be successful, the conversion should match multiple identity cues of the target speaker, including but not limited to vocal tract configurations, prosody, pitch range, accent/dialect, and speaking rate. Ideally, the only information retained from the source utterance is its linguistic content, i.e., what has been said. Accent conversion goes one step further, since it attempts to capture both the linguistic content and the pronunciation of the source utterace, and combine it with the voice quality of the target speaker (i.e., those aspects associated with the target speaker's physiology), to create a new voice that sounds like the target speaker speaking with the source speaker's pronunciation. Therefore, accent conversion is a more challenging problem than voice conversion since ground truth for the output voice (i.e., the L2 learner's voice with a native accent) is not available.

VC methods are potential ways to avoid collecting articulatory data and still perform AC, as an L2 speaker's model could be driven using L1 speech, resulting in native prosody and pronunciation, but with the L2 speaker's voice identity. However, additional modifications would be necessary to account for the segmental differences arising from the accent of the L2 target speaker. In [64], Aryal et al proposed an alternative "acoustic similarity" source-target alignment for use in a GMM-based voice conversion method. Using Vocal Tract Length Normalization (VTLN), they warped the target L2 spectrum to the L1 speaker's space and then paired the L1 and L2 frames using a Mel-Cepstral Distortion metric. In perceptual studies, the authors found that using the alternative frame pairing combined with voice conversion reduced the accent of the synthesized utterances and captured the voice quality of the target speaker. Similarly, Zhao et al. [34, 57]

presented another alternative alignment method to account for these differences. As opposed to using VTLN to learn source and target frame parings, the authors proposed using a Phonetic Posteriorgram (PPG) extracted from the Kaldi ASR system. The PPG representation is akin to a latent phonetic space that is used in speech recognition; because this recognizer was trained on hundreds of speakers, the representation is presumed to be speaker-independent, and two acoustic frames which have similar PPG vectors likely share the same phonetic content. The authors used this property of the PPG to pair L1 and L2 frames in such a way that minimized the KL-divergence of the L1 and L2 PPG data. In perceptual tests, they found that this method had even higher acoustic quality than the previous GMM-based AC methods and further reduced the accent present in synthesis. Notably, participants were also able to identify the AC utterances as coming from the target L2 speaker a significant portion of the time. These results showed that VC-based acoustic-only models could be used to perform accent conversion so long as the regression method between the L1 and L2 speakers accounted for pronunciation differences. However, the substantial amount of training data (300 utterances for GMM, 100 utterances for PPG) required to build these acoustic mappings made them infeasible in a computer aided pronunciation training context.

2.5. Other Considerations in Conversion Algorithms

2.5.1. *Frame Pairing Methods*

To train many voice conversion systems, a method for learning training source and target pairs is required. However, this is difficult as it is not always possible for two speakers to produce identical training utterances, either because of convenience or due to

other issues, such as one speaker having an accent. Even in cases where the speakers did produce the same training sentences, the utterances may differ in terms of timing and rhythm, so techniques are required to build paired source and target training data. One technique for aligning two samples with similar content, but different timing, is the Dynamic Time Warping (DTW) algorithm [65]. This algorithm uses dynamic programming to learn a minimum error alignment between pairs of source and target training data.

In speech, DTW is often used to align two parallel utterances spoken by two different speakers [1, 2, 6, 8]. This is done in the spectral domain, as similar phonetic content will have similar spectral representations even between two speakers. However, this assumption is somewhat strong, and breaks down when the source and target speaker have different productions of the same content. This can happen in instances where one speaker produces different phonetic content than a native speaker due to a non-native accent [34, 66], even when both speakers were asked to produce the same utterance.

2.5.2. *Complexity Considerations*

For VC systems to be used in real-time or in-the-field applications, the complexity of these systems must be kept in mind. While Deep Learning and existing exemplar-based voice conversion systems have remarkable performance, both types of VC require a significant amount of processing power for conversion and synthesis [67, 68]. While many DNN systems offload processing onto client-server frameworks[69], such systems are not always feasible, especially for systems that require immediate feedback or when stable internet connections are not always available [70]. Research into performing many speech

processing techniques on limited-resource devices (such as mobile hardware) shows a need for lightweight algorithms in a variety of tasks [71-74].

3. SABR: SPARSE, ANCHOR-BASED REPRESENTATION OF THE SPEECH SIGNAL[*]

3.1. Overview

In this chapter, we present the primary algorithm of the dissertation: SABR, Sparse, Anchor-Based Representation of Speech. We discuss the intuition of the method, how to build SABR models, and how to use them in VC. In experiments, we show how the SABR method performs in speaker-independent representation and voice conversion tasks. The methods in this chapter addresses the first aim of this dissertation and are the basis for the following chapters. This chapter was originally presented at Interspeech, 2015 [9], and has been modified to fit the structure of this dissertation.

3.2. Introduction

Many VC methods require parallel data or significant amounts of training data to model source and target speakers [1, 6, 34, 75]. In practice, it is not always practical to collect significant amounts of training data. Additionally, there are instances where one of the speakers may be unable to pronounce parallel utterances in the same manner as the other (e.g., when one speaker has a non-native accent). Instead of requiring parallel training data and relying on alignment algorithms to ensure a good mapping between the source and target speakers, "anchoring" the source and target speaker's identities by modeling how they form phonemes would be a way to alleviate issues with these VC

[*] Reprinted with permission from "SABR: Sparse, Anchor-Based Representation of Speech" by C. Liberatore, S. Aryal, Z. Wang, S. Polsley, and R. Gutierrez-Osuna, 2015. *Interspeech 2015*, p.608-612, Copyright 2015 by *International Speech Communication Association*.

methods. By modeling how a source speaker forms an utterance next to these phoneme anchors, an estimate of how the target speaker would form the same utterance could be built.

In this chapter, we present SABR (Sparse, Anchor-Based Representation), an analysis technique that builds speaker models and decomposes the speech signal on this intuition. Specifically, SABR models speaker's voices with a set of speaker-dependent acoustic anchors and decomposes an utterance as a nonnegative weighted sum of these anchors using Lasso regression [76]. As we will show, by selecting the phoneme centroids of each speaker as anchors the resulting weights become speaker-independent and can be used for VC. We illustrate the ability of the model to separate speaker and linguistic information in two experiments. First, we show that SABR weights outperform conventional spectral features (MFCCs) on a speaker-independent phoneme discrimination problem. Second, we show that, by combining SABR weights derived from a source speaker with acoustic anchors from a target speaker, our technique can be used as a low-resource voice conversion method–one that does not require training a specific model for each source-target pair. Both of these experiments motivate the use for SABR in voice conversion, and the formulation presents research questions we answer in the following chapters.

The rest of the chapter is organized as follows. First, we present the SABR model and how to use its components for voice conversion and speech recognition applications. Then, we describe details on the corpus and acoustic features used to evaluate the model, and then we present experimental results on phonetic classification and voice conversion

(subjective and objective comparison). The chapter concludes by discussing the implications of the results, future improvements to the method and its potential application to other speech areas.

3.3. Method

3.3.1. *Anchor-based representation*

Our proposed method represents the speech signal as a collection of speaker-dependent acoustic anchors (derived from phonetic labels) and a matrix of interpolation weights, one set of weights per acoustic frame. In this fashion, as the weights capture the similarity of each acoustic frame to various phonetic anchors, they also capture the linguistic content of the utterance, including the effects of coarticulation. Formally, SABR represents utterance X_S as:

$$X_S \cong A_S W_S \tag{1}$$

where each column in matrix X_S represents an analysis window (i.e., a vector of MFCCs), A_S is a matrix of anchors for speaker S, and W_S is the utterance's weight matrix. If there are M acoustic frames in an utterance, N acoustic features, and P speaker anchors, then $X_S \in \mathbb{R}^{N \times M}$, $A_S \in \mathbb{R}^{N \times P}$, and $W_S \in \mathbb{R}^{P \times M}$.

3.3.2. *Anchor selection*

Several methods may be used to select the acoustic anchors in A_S, including unsupervised (e.g., k-means clustering), supervised learning (e.g., orthogonal least-squares [77]), or time-aligned source and target utterances (e.g. exemplar-based VC, see [8, 46, 50]). However, for the weight matrix W to be speaker-independent the acoustic anchors must be consistent across speakers. For this reason, SABR uses the acoustic

centroid for each phoneme in the speaker's corpus as anchors –one anchor per phoneme, resulting in a compact set of parallel anchors for the source and target speaker. This results in the sparse weights capture the linguistic content of the utterance (i.e., which phones were produced, when and how) whereas the acoustic anchors capture the identity of the speaker (i.e., voice quality and dialect/accent). Phoneme centroids as anchors also makes the decomposition interpretable. Because only phoneme labels are required, source and target anchor sets *do not need to be trained from parallel utterances*. This aspect makes it especially attractive for native-to-nonnative conversion, as alignment effects (e.g. disfluencies in the nonnative speaker's training data) are less of an issue for training SABR models.

To ensure that correct training data are used to select anchors, we include a voicing constraint on the data used to train SABR anchors. For a given phoneme k, only training data that match the voicing of the phoneme will be considered when computing the centroid (e.g. if a vowel centroid is being computed, only spectral frames which have pitch present and the correct phoneme label will be considered when computing the centroid). We enforce this voicing constraint by also examining the pitch (F_0) during the centroid computation process. Removing incorrectly-voiced frames results in centroids that more accurately reflect low-frequency energy of a phoneme, resulting in improved synthesis quality and lower VC error. This can make a significant difference with speakers who may have difficulty consistently forming the right voicing for each phoneme, e.g. nonnative speakers [78].

3.3.3. *Sparse representation*

Given a set of acoustic anchors A_S, obtained from a phonetically transcribed corpus for the speaker, and a new utterance X, we seek to find a set of weights that minimize the reconstruction error $\|X - A_S W\|$. A straightforward approach is to use the least-squares solution:

$$A_S^+ X_S = W_S \tag{2}$$

where A_S^+ is the pseudoinverse of A_S. This solution, however, does not exploit the sparse nature of the speech signal, in which only a few anchors in A_S may be required to accurately reconstruct a given acoustic frame. Moreover, the pseudo-inverse solution allows the weight vector to take negative values, which affects the interpretability of the solution.

For these reasons, SABR enforces a sparse non-negative constraint on the solution by using Lasso regression [76]:

$$\min_{\alpha} \left|\left|X - A_S W\right|\right|^2 + \lambda \left|\left|W\right|\right|_1 \; s.t. 0 \leq W \leq 1 \tag{3}$$

where λ is a parameter that penalizes solutions with large L1 norm. Combined with the constraint that all entries in W be nonnegative, the λ penalty term promotes sparsity (i.e., most of the entries in W are zero). For this dissertation, we use the LARS Lasso solver in the SPAMS sparse coding toolbox [79, 80].

3.3.4. *Voice conversion with SABR*

SABR provides a simple means of performing voice conversion. Given an utterance X_S from a source speaker, we first derive a set of interpolation weights (W_S)

relative to the source speaker's anchors (A_S) via eq. (3). Then, given a target speaker with acoustic anchors A_T, the target speaker's utterance X_T can be estimated as:

$$\widehat{X_T} = A_T W_S \tag{4}$$

As weights W_S contain phonetic information, the resulting spectrum is an estimation of the utterance said by the source speaker, but with the target speaker's voice quality. An overview of the SABR VC algorithm is shown in Figure 4.

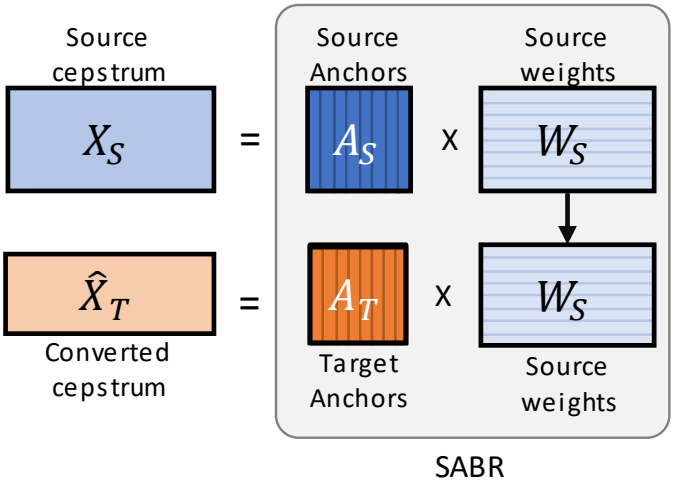


Figure 4: overview of the SABR voice conversion system.

3.4. Experiment design

We evaluated SABR on speech from the ARCTIC speech corpus [81] which includes phonetic transcriptions for each utterance. We chose the four native English speakers in ARCTIC as the basis for our comparison: BDL (male), CLB (female), RMS (male), and SLT (female). For each speaker, we used utterances in the "A" set to compute the SABR anchors, and utterances in the "B" set for testing purposes.

For each utterance, we used STRAIGHT [39] to extract aperiodicity, fundamental frequency and spectral envelope, then computed 25 MFCCs (25 filterbanks, 8 KHz cutoff, 15ms window, 1ms shift) from the STRAIGHT spectral envelope. We assigned each frame a phonetic label based on the ARCTIC transcription, then used $MFCC_{1-24}$ and their deltas as acoustic features, ignoring $MFCC_0$ as it contains the speech energy.

3.5. Results

3.5.1. *Sparsity penalty evaluation*

In an initial experiment, we evaluated the average Mel Cepstral Distortion[2] (MCD) between 100 target utterances and their respective voice-conversions for each combination of source and target speakers (12 pairs). As a baseline, we also calculated the within-speaker reconstruction error. Results are shown in Figure 5. As expected, MCDs are lower when reconstructions are within-speaker than between-speakers. Additionally, the MCD is minimized at $\lambda = 0$ in the within-speaker case, which indicates that sparsity offers no benefits in this case. In contrast, the MCD in the cross-speaker case (i.e., voice conversion) is minimized at $\lambda = 0.025$, which suggests that sparsity does improve generalization across speakers. For this reason, the remaining analyses in the chapter were conducted using the sparsity penalty $\lambda = 0.025$.

[2] Since voice conversions follow the timing of the source speaker, they are time-aligned to the target utterance (via dynamic time warping) prior to computing the MCD.

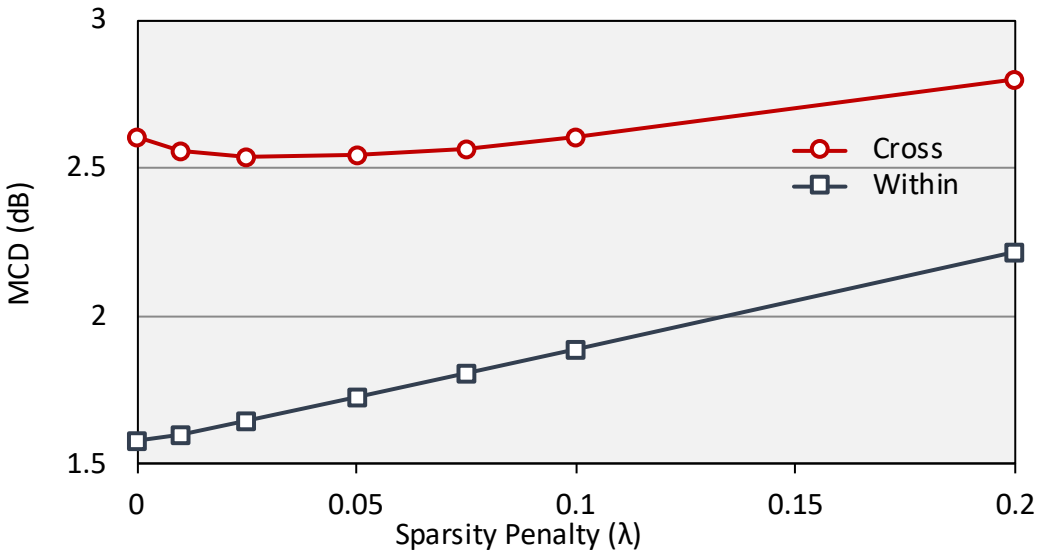


Figure 5: Reconstruction error (MCD) within (solid line) and across speakers (dashed line).
A minimum MCD exists at $\lambda = 0.025$ in the case of cross-speaker reconstruction (i.e., voice conversion).

3.5.2. *Phoneme classification*

In a first set of experiments, we evaluated the extent to which SABR captures phonetic information in a speaker-independent manner. For this purpose, we compared SABR weights against conventional MFCC features on a phone recognition problem. Namely, we built four phoneme classifiers for each of the four ARCTIC speakers:

- **MFCC-W**: within-speaker phoneme classifier on MFCC features, ignoring MFCC energy.
- **SABR-W**: within-speaker classifier on SABR weights (40 weights: ARCTIC phone set, excluding pause and silence frames)
- **MFCC-X**: cross-speaker classifier on MFCC features, trained on three speakers and tested on the fourth speaker

- **SABR-X**: cross-speaker classifier on SABR weights, also trained on three speakers and tested on the fourth speaker

Within-speaker classifiers were trained using 500 utterances from each speaker's training set and evaluated on test utterances from that same speaker using 8-fold cross-validation. In turn, cross-speaker classifiers were trained on the same 500 utterances from each of three speakers and tested on utterances from the excluded fourth speaker. Results are shown in Figure 6. Classification performance for the MFCCs degrades significantly when comparing within-speaker (43%) and between-speaker (23.9%), whereas classification performance for SABR features remains relatively stable: 36% versus 34.6%. Moreover, whereas MFCC features outperform SABR features by a large margin (43% versus 36.1%) in the case of within-speaker phoneme recognition, in the between-speaker case SABR features outperform MFCC features by a larger margin (34.6% versus 23.9%). These results suggest that SABR features are relatively speaker-independent. Results on the voice conversion task (discussed next) corroborate this conclusion.

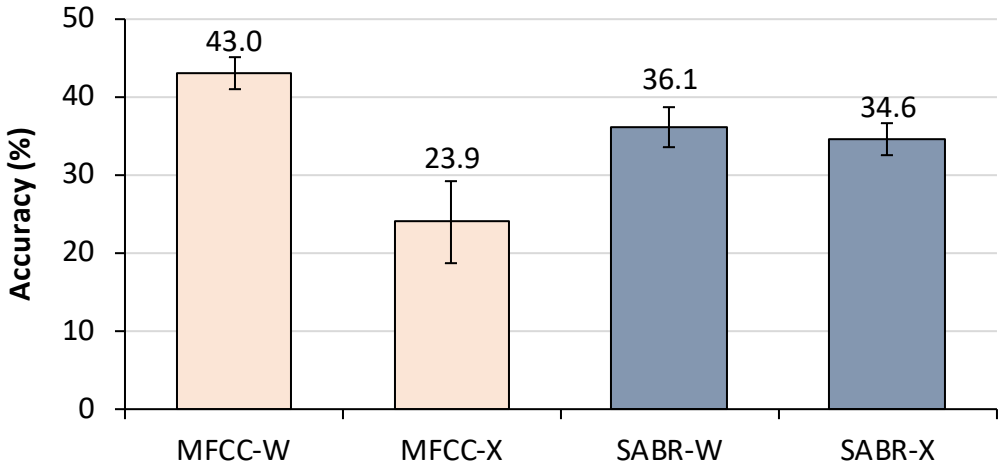


Figure 6: Phoneme classification performance, comparing SABR features against MFCC features.
Performance for MFCC features degrades significantly from within-speaker to cross-speaker tasks, whereas SABR features remain stable and outperform MFCCs in the cross-speaker task.

3.5.3. *Voice conversion performance*

In a second set of experiments, we evaluated the ability of SABR to separate voice-quality and phonetic information using objective and subjective measures on a voice conversion task. For a particular source-target speaker pair, we used eq. (4) to reconstruct the STRAIGHT spectral envelope of the target speaker, combined it with the source energy ($MFCC_0$) and source pitch contour (scaled to match the range of the target speaker), and resynthesized the utterance with STRAIGHT.

3.5.4. *Objective evaluation*

First, we compared SABR against a baseline voice conversion system based on Gaussian mixture models (GMM) [2]. To control for model complexity, we limited the GMM to 40 mixtures—the same number of SABR anchors. Prior to building the voice conversion model, we selected 200 training utterances using a greedy forward-selection

method that maximized the entropy of the phonetic transcriptions of the utterances. Using these 200 utterances, we then build pairwise GMMs for each pair of source and target speakers (12 pairs of speakers) and computed SABR anchors for the four speakers. Results are shown in Table 1; using the 200 carefully-selected training sentences, the GMM method outperformed the SABR method on test utterances (an average MCD of 2.26 versus 2.53, respectively), likely due to the fact that each GMM was optimized for each pair of speakers and had additional free parameters (e.g. full diagonal matrices).

For this reason, we also compared the two voice-conversion models with decreasing corpus size: 100, 50, 25, and 20 training utterances selected from the corpus using the same greedy forward-selection strategy. Results are also shown in Table 1: whereas the GMM performance decreases as the number of training utterances is reduced, the SABR performance remains relatively stable, validating one of the aims of the SABR method.

Table 1: Voice conversion performance for SABR and GMM.
The top row shows the number of training utterances. Entries are the average MCD.

Training	20	25	50	100	200
GMM	2.66	2.59	2.40	2.31	2.26
SABR	2.59	2.59	2.57	2.56	2.53

3.5.5. *Subjective evaluation*

In a final experiment, we conducted a listening test to compare the voice similarity between the SABR voice conversions and the respective source and target speakers. To account for the loss of quality due to the sparse nature of SABR synthesis, we resynthesized *source* and *target* utterances using the speaker's own phonetic anchors. Participants were presented with 48 pairs (*source-VC* and *VC-target*) for all 12 possible

speaker combinations, randomly ordered, then were asked to (1) determine if the utterances were from the same or a different speaker, and (2) rate how confident they were in their assessment using a seven-point Likert scale (1: not confident at all, 3: somewhat confident, 5: quite a bit confident, and 7: extremely confident). Following prior work [82], participants' responses and confidence ratings were then combined to form a voice similarity score (VSS) ranging from -7 (extremely confident they were from different speaker) to +7 (extremely confident they were from the same speaker).

The results of this subjective test are shown in Figure 7. Participants were "quite" confident that the converted utterances had the same voice as the target speaker ($VSS = 4.6, s.e. = 0.4$) and had a different voice from the source speaker ($VSS = -5.9, s.e. = 0.3$). This suggests that the phonetic anchors in SABR analysis successfully capture the speaker's voice identity.

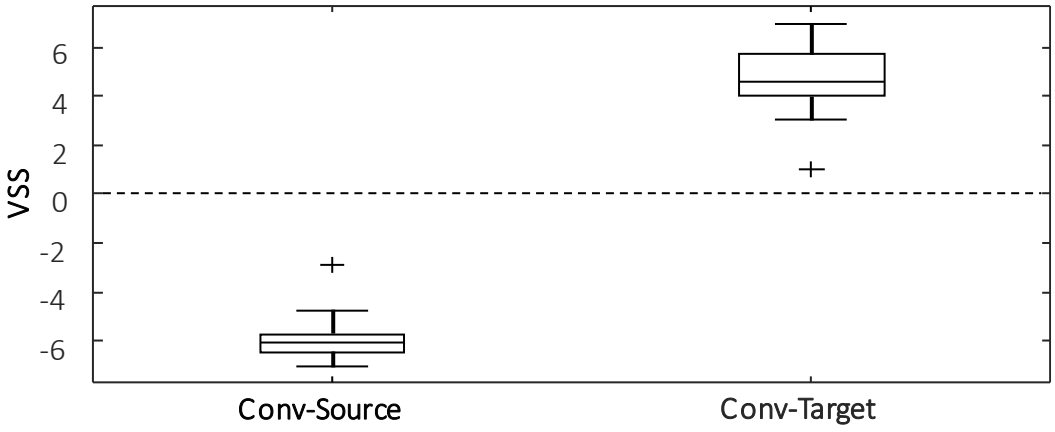


Figure 7: Voice similarity assessment results for SABR VC.
The plot is shown on a 7-point Likert scale, rating voice similarity.

3.6. Conclusion

In this chapter, we presented the presented SABR, an analysis technique that can be used to separate voice quality and linguistic contributions to the speech signal. SABR uses sparse regularization to represent speech frames as a linear non-negative combination of acoustic anchors. By using speaker-dependent phoneme centroids as anchors, the resulting weights generalize well across speakers. In particular, our results show that SABR weights yield similar phoneme recognition performance in within-speaker and between-speaker conditions, and that they outperform conventional MFCCs in the cross-speaker condition.

SABR provides a straightforward method for voice conversion: an utterance from a source speaker can be converted into one for a target speaker by extracting SABR weights relative to the source anchors, and combining them with anchors from the desired target speaker. *More importantly, voice conversions can be performed without having to train a specific model for each pair of source and target speakers*. Indeed, subjective listening tests show that SABR voice conversions have the same voice quality as the target speaker. Objective measures also show that SABR is more resilient to small training corpora than a baseline GMM voice-conversion technique, *validating one of the original goals of the representation.*

The SABR method shown here shows promise for building a speaker-independent representation for use in VC, but the results suggest multiple directions for improvement. In the following chapters, we will discuss the remaining three aims of this dissertation:

- **Improving synthesis quality:** the sparse coding residual in eq. (1) ignores a substantial amount of spectral energy (~1.5 dB, see Figure 5) that contains much of the spectral detail and affects synthesis quality in voice conversion. In the next chapter, we examine ways to use this residual to improve voice conversion synthesis quality while still reaching the target speaker's voice identity.
- **Selecting optimal anchors:** Building anchor sets from the centroid of the source and target training data may not be optimal. In Chapter 5 we examine optimal ways of building SABR anchors using two different optimization methods.
- **Adding temporal constraints:** The lasso method in eq. (3) computes SABR weights on a frame-by-frame basis, without considering any temporal context. Including this information could reduce the noise in the SABR encoding (see Figure 8), resulting in better speaker representation and higher synthesis quality. In Chapter 6, we propose and evaluate temporal constraints for SABR objective function.

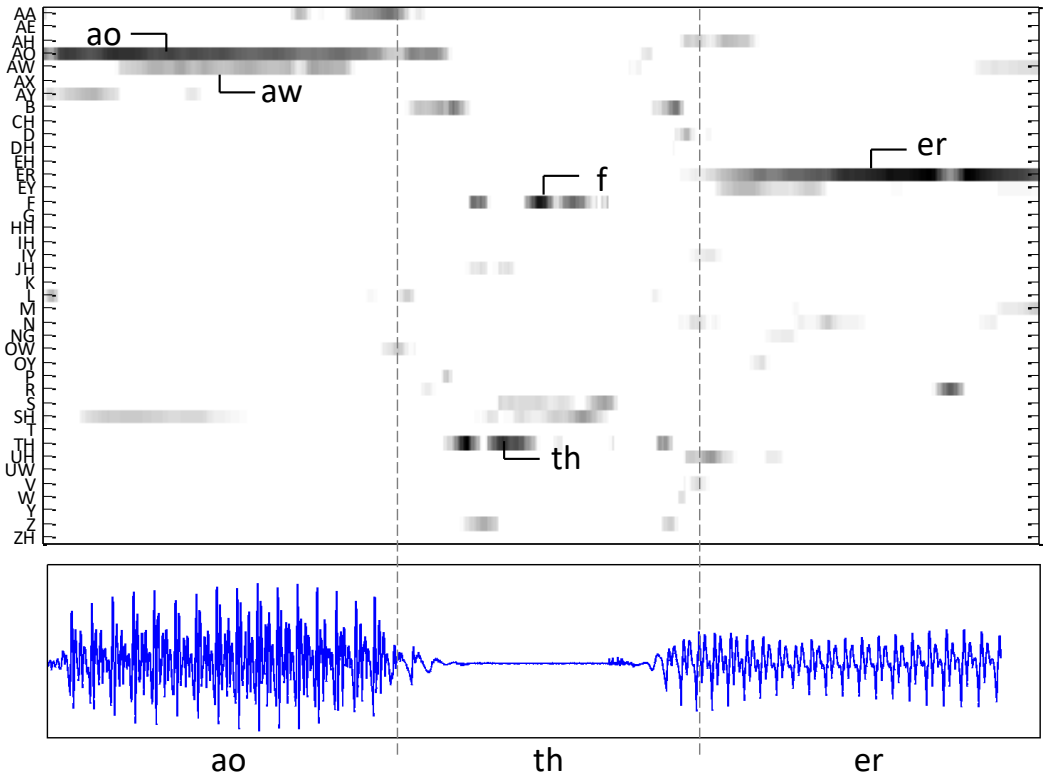


Figure 8: Example SABR weight matrix with phonetic transcription.
The weights capture the word "author" and are interpretable, particularly for vowels where the weights closely match the transcription.

4. NATIVE-NONNATIVE VOICE CONVERSION BY RESIDUAL WARPING IN A SPARSE, ANCHOR-BASED REPRESENTATION[*]

4.1. Overview

In this chapter, we propose and evaluate a method for using the source speaker's residual in the SABR VC method to improve synthesis quality. The method presented in this chapter uses a technique known as "frequency warping" to transform the source residual to the target speaker's space, and uses it in synthesis to improve overall synthesis quality. This chapter addresses the second aim of this dissertation.

The first version of this chapter was presented at ICASSP 2018 [10]. We expanded upon this study, examining both native-to-native and native-to-nonnative conversion, as well as multiple variants of the proposed transform, and submitted it to IEEE/ACM Transactions on Audio, Speech, and Language Processing in 2021. This chapter has been modified to reflect the relevant literature and structure of this dissertation.

4.2. Introduction

In the previous chapter, we presented a sparse, anchor-based representation of speech (SABR) for use in voice conversion. While our experiments showed that the representation could be used to perform voice conversion, the synthesis quality was low due to the compact nature of the SABR model. This is due to the large sparse residual that

---

[*] Reprinted with permission from "Native-Nonnative Voice Conversion by Residual Warping in a Sparse, Anchor-Based Representation" by C. Liberatore 2021. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, Vol. 29, p. 3040-3051, Copyright 2021 by *IEEE*. Parts also reprinted from "Voice conversion through residual warping in a sparse, anchor-based representation of speech" by C. Liberatore, G. Zhao, and R. Gutierrez-Osuna 2018. *ICASSP 2018*, p. 5284-5288, Copyright 2018 by *IEEE*.

occurs during the encoding process—this residual contains a significant amount of energy and spectral detail, and discarding it has a detrminental effect on synthesis quality.

To alleviate this problem, we propose a residual transformation method, SABR+Res, that uses linear combinations of frequency warping transforms to convert the source residual to be closer to the target speaker. Frequency warping transforms are used in instances where there is a desire to retain spectral detail, but the energy of the spectrum needs to be redistributed. SABR+Res builds a linear combination of these transforms to convert the source residual to be closer to the target speaker. We evaluate the proposed transform using four frequency warping functions (piecewise linear [10], bilinear [83], dynamic [83, 84], and correlation frequency warping [44]) from which to learn our anchor-based frequency warps. After determining the optimal frequency warping method for the proposed algorithm, we conduct subjective and objective experiments to compare the proposed SABR+Res transform against two baseline voice conversion techniques: Exemplar-Based Voice Conversion with Residual Compensation (ERC) [51] and Weighted Frequency Warping (WFW) [4]. Our objective experiments show that SABR+Res using Dynamic Frequency Warping (DFW) provides the lowest VC error. In subjective tests, we show that the proposed SABR+Res method both significantly improves upon the synthesis quality compared to the basic SABR synthesis, is closer to the target speaker identity, and reduces the accentedness of native-to-nonnative synthesis. Finally, we also show that listeners prefer the quality of SABR+Res syntheses over WFW and ERC. We argue that this robustness in native-to-nonnative conversion is due to the

fact that SABR+Res relies on phoeneme-based representation and not time-aligned training utterances.

This chapter is organized as follows. First, we review relevant literature to the domain of frequency warping and how it has been applied in voice conversion. Then, we discuss the proposed SABR+Res algorithm and details of frequency warping functions. In experiments, we evaluate our proposed algorithm against two baseline methods in native-to-native and native-to-nonnative contexts. We end with a discussion of the results and conclusions of the algorithm.

4.3. Related Work

In this section, we discuss what frequency warping functions are and how they are used in speech processing. Then, we discuss why the properties of frequency warping make them useful for VC problems and how they have been applied to VC. Finally, we discuss how residuals affect synthesis quality in exemplar-based voice conversion and how residuals have been used to increase synthesis quality in these methods.

Frequency warps are functions that build a transformation of a source spectrum to align it with the energy of a target spectrum. These transforms are piecewise and invertible and have the effect of "squishing" or "stretching" a segment of the source spectrum to align its energy more closely to that of a target spectrum [44, 83]. An example frequency warping function is shown in Figure 9, but any invertible function from the source spectrum to the target is a valid frequency warp. Frequency warping functions are often used to perform Vocal Tract Length Normalization (VTLN) between two speakers [84-86]. Because of the piecewise linear nature of the transforms, spectral detail is retained

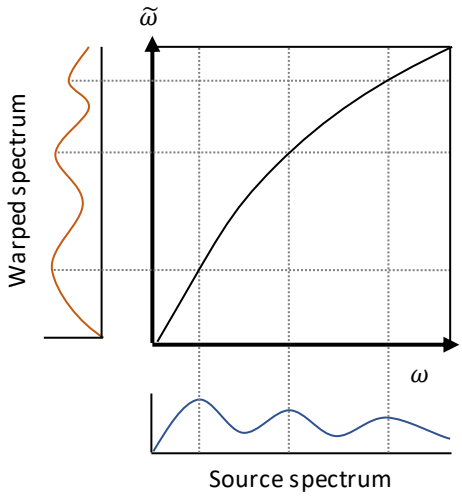


Figure 9: an example of a frequency warping function.
The source spectrum $f(\omega)$ is warped according to $\tilde{\omega}$. Each axis illustrates the effect of warping on an example source spectrum. Horizontal and vertical sample spectra are included to illustrate the change in location of the formants.

and typically these methods retain more spectral detail than statistical conversion methods. This property makes them appealing for use in VC where retaining spectral detail is desirable [4, 5, 37, 38].

To alleviate oversmoothing issues arising from GMM methods (especially when limited training data is available), Frequency Warping and Amplitude Scaling VC methods (FW+AS) were introduced to retain more spectral detail from the source speaker during conversion [4, 37]. Erro *et al.* [5] proposed Weighted Frequency Warping (WFW), which used a GMM to estimate a frequency warping function to transform the source speaker's spectrum to match that of the target speaker. During conversion, instead of using the conditional probability of the GMM to estimate the target spectral envelope, the conditional probability was used to estimate a warping function to transform the source spectrum; to ensure that the spectral energies matched the target speaker, an amplitude

scaling step brought the spectral energy of the warped source closer to that of the target. Their method outperformed a baseline GMM in terms of decreased spectral distortion, and listeners rated the syntheses as having higher acoustic quality. Godoy *et al*. [37] presented a similar method, but removed the requirement for parallel utterances, instead building a GMM, with each phoneme represented by a mixture. The authors proposed a "phonetic GMM", with a single Gaussian mixture for each phoneme label. For each of these Gaussians, the authors computed optimal frequency warping functions between the source and target training data. Amplitude scaling terms were then estimated from the residual of the warped source and target spectrum. The authors found that listeners preferred their method to standard GMM regression, even though it led to higher spectral distortion than a traditional GMM-regression method.

Exemplar-based methods introduce a residual in the encoding process, which can affect the synthesis quality of the output. This residual contains a significant amount of spectral detail and not accounting for it can reduce synthesis quality. Wu et al. [51] proposed a method of encoding this residual called Exemplar Residual Compensation to further improve the synthesis quality of exemplar-based methods. Noting that the sparse residual lowered the overall synthesis quality, the authors proposed a linear transform based on Partial Least Squares to map the source residual to the target residual, and add it to the exemplar-based synthesis method. This had the net effect of further improving synthesis quality by including spectral details which were discarded at the time of the encoding process. The method has the advantage of retaining the spectral detail of the source while matching the voice quality of the target. Listeners preferred the synthesis

quality of the exemplar-based warping method to a GMM-based warping method. This method was used in several other exemplar-based VC methods as a way of mapping residuals during the conversion process [35, 38].

4.4. Methods

The proposed improvement to the SABR synthesis method—SABR+Res—transforms the source residual to be closer to the target speaker, using the SABR anchors and weights to learn the transform for each frame in the source residual. Because the aim of SABR+Res is to retain spectral detail while bringing the source residual closer to the target speaker, we use a class of spectral transforms known as *frequency warping transforms* in this method. These transforms retain spectral detail while adjusting the distribution of the energy in the spectrum.

In this section, we will briefly review the SABR method and the location of the source residual and how we use it during synthesis. Following this, we discuss the proposed residual transform algorithm, SABR+Res, and how it builds the residual transform to be used in synthesis. Then, we discuss frequency warping in the cepstral domain and how we select optimal frequency warping functions which are used by SABR+Res in the residual transformation. Finally, we will discuss how this algorithm is different from other frequency-warping methods used in voice conversion.

To review, given a cepstral representation of a source utterance X_s, SABR decomposes it as:

$$X_S = A_S W_S + R_S, \tag{5}$$

where W_s is a sparse set of weights, A_S is a set of speaker-dependent phoneme "anchors," and R_S is the residual term. For an utterance with T frames, N spectral features, and K anchors, $X_S \in \mathbb{R}^{N\times T}$, $R_S \in \mathbb{R}^{N\times T}$, $A_S \in \mathbb{R}^{N\times K}$, and $W_S \in \mathbb{R}^{K\times T}$.

In the initial version of SABR presented in chapter 3, we discarded the source residual during synthesis. However, as this component is a source of significant spectral energy and detail, discarding it results in lower synthesis quality. To alleviate this, we transform the source residual to be closer to that of the target speaker using a function $F_R(R_S)$ and add it to the estimated target spectrum from eq. (4). Incorporating this into the synthesis, the target speaker's spectral envelope $\hat{X}_T$ is then estimated as:

$$\hat{X}_T = A_T W_S + F_R(R_S). \tag{6}$$

4.4.1. *SABR+Res*

There are two components to the SABR+Res algorithm: training the residual transform and using it during synthesis. For the training step, we compute an optimal frequency warp F_W on each pair of source and target anchors in A_S and A_T. For the k^{th} pair of source and target speaker cepstral anchors A_S^k and A_T^k, the optimal frequency warp F_W^k is:

$$F_W^k = \min_{F_W} \left| T(F_W) A_S^k - A_T^k \right|_2^2 \tag{7}$$

where $T(F_W)$ is a function that performs frequency warping on the cepstrum, following the warp F_W. The resulting vector F_W^k is the frequency warp that minimizes the difference between two cepstral anchors A_S^k and A_T^k. In the following sections, we will discuss how to build and optimize the transforms $T(F_W)$.

Using the optimal frequency transforms learned from eq. (7), we build a transform that warps the source residual R_S to be closer to that of the target speaker. The proposed residual transformation $F_R(\cdot)$ of the t^{th} residual frame R_S^t, given the weights W_S^t and the set of transforms learned from the source and target anchors F_W, is:

$$F_R(R_S^t) = \left(\sum_{k=1}^{K+1} W_S^{k,t} T(F_W^k) \right) R_S^t, \tag{8}$$

where $W_S^{k,t}$ is the weight corresponding to the kth anchor at frame t. The resulting residual warping function $F_R(\cdot)$ performs the conversion in eq. (8) for each frame of the source residual. An overview of the training and synthesis methods are shown in Figure 10.

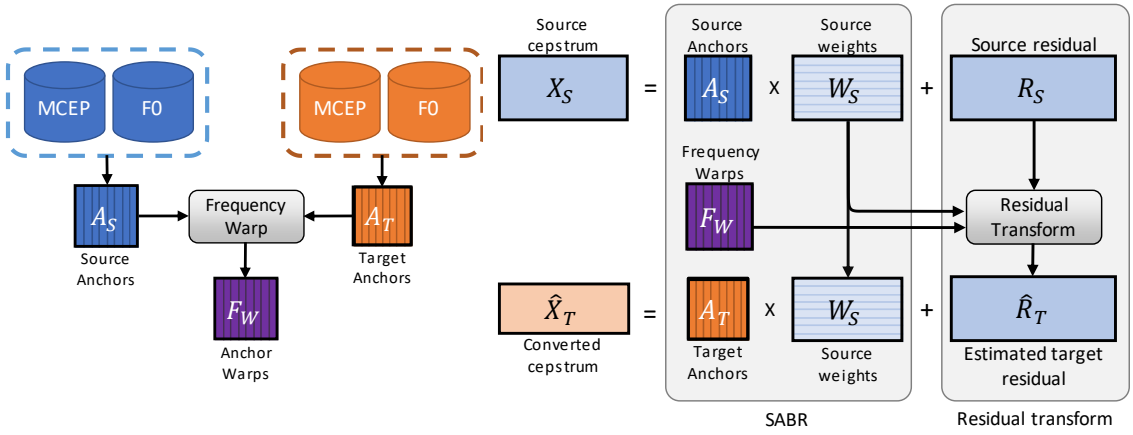


Figure 10: Overview of the training and residual warping method. (a) Training the frequency warps and anchor sets. (b) Synthesizing the audio using the SABR+Res method.

This method was inspired by the covariance mapping component of GMM regression [87]. In contrast with statistical mappings, the proposed method does not

oversmooth the spectrum since each transform $T(F_W^k)$ is built from frequency warps, which retain spectral detail. Because only a few weights are non-zero because of sparsity constraint, spectral detail from the source residual is retained.

4.4.2. *Frequency warps in the cepstral domain*

As stated previously, frequency warps are invertible functions that warp a spectrum of a source speaker $S_S(\omega)$ to be closer to that of a target speaker $S_T(\omega)$ where ω represents the normalized frequency. For two discrete-sampled spectra $S_S(\omega)$ and $S_T(\omega)$ with M frequency bins, the frequency warp $\widetilde{\omega}$ that minimizes the difference between the source and target spectra can be computed following:

$$F_W = \min_{\widetilde{\omega}} |S_S(\widetilde{\omega}) - S_T(\omega)|_2^2. \quad (9)$$

Because the frequency warp $\widetilde{\omega}$ is one-to-one, we can parameterize it as a vector $F_W \in \mathcal{R}^M$, where each entry corresponds to how an input frequency bin (e.g. a set of evenly-spaced normalized frequency bins $\omega = [\omega_0 \dots \omega_M]^T$) is mapped to an output frequency bin $\widetilde{\omega} = [\widetilde{\omega}_0 \dots \widetilde{\omega}_M]^T$ (see Figure 9). Different types of frequency warping functions (e.g. Bilinear or Dynamic Frequency Warping) have different parameters or constraints on $\widetilde{\omega}$, affecting the shape of the warping function and the output warped spectra.

Pangaschan *et al.* [83] showed that arbitrary frequency warps F_W were equivalent to linear transforms on cepstral coefficients. This transform requires two steps: first, the cepstral coefficients must be projected back into the spectral space, and then the spectrum

must be warped according to F_W. For Mel Cepstral (MCEP) coefficients, a transform $T \in \mathcal{R}^{N\times N}$ that performs the frequency warp F_W on cepstral coefficients can be represented as:

$$T(F_W) = Cf(F_W)C^T, \tag{10}$$

where f is a function that builds an "index mapping" (following [88])[3] of size $M \times M$ of the frequency warp F_W, $C \in \mathcal{R}^{N\times M}$ is the linear Discrete Cosine Transform (DCT) and C^T is the inverse DCT. $T(F_W)$ is then a transform of cepstral coefficients according to the frequency warp F_W.

Assuming M spectral coefficients, N cepstral coefficients, the DCT matrix C is:

$$C_{k,m} = [\alpha_k \cos(\pi k)]_{\substack{1\le m\le M \\ 0\le k\le N-1}}, \tag{11}$$

where a_k is a normalization term that ensures each row sums to 1.

4.4.3. *Optimal frequency warps*

Let $W(\Theta)$ be a function that generates a frequency warp $F_W \in \mathcal{R}^M$. The optimal frequency warp F_{opt} between source cepstrum X_S and target cepstrum X_T is represented by:

$$F_{opt} = \underset{W(\Theta)}{\operatorname{argmin}} |T(W(\Theta))X_S - X_T|_2^2. \tag{12}$$

For a given type of frequency warping function $W(\Theta)$, we consider two methods for selecting the optimal warping parameters Θ in eq. (12): grid search and greedy search. In grid search methods, combinations of the parameters in Θ are exhaustively tested and the

[3] The "index mapping" (IM) matrix has one non-zero entry on each row, and maps each source spectral bin to a target spectral bin

frequency warp which minimizes eq. (12) is selected as the optimal parameters for that warp type. However, in instances where all possible parameters are not searchable, a greedy algorithm (e.g. dynamic programming) is used to select the optimal parameters.

In this chapter, we consider four different types of frequency warping methods, which warp the source spectra with different constraints and objective functions. Because we are interested in how these perform not just between native speakers, but between native and nonnative speakers, we also highlight the strengths and weaknesses of each of the warping functions. A summary of the warping functions, the optimization technique used to build the warping functions, and the objective functions are shown in Table 2.

Piecewise linear warping: computed following [5, 84]. This method has two parameters: a slope α and inflection frequency ω_0. The warping function linearly warps the source spectra to the inflection frequency on the slope, and from the inflection frequency to the Nyquist frequency (see Figure 11 (a)). This method has been used to generally warp the vocal tract length of a given source speaker to be closer to that of a target speaker.

Bilinear frequency warping: computed following [5, 84]. This warp functions in a similar manner to piecewise linear, but is a continuous function. The magnitude of the warp is controlled by a parameter α (see Figure 11 (b)). Like piecewise linear warping, this method is also used to more closely match the vocal tract length of a given target speaker.

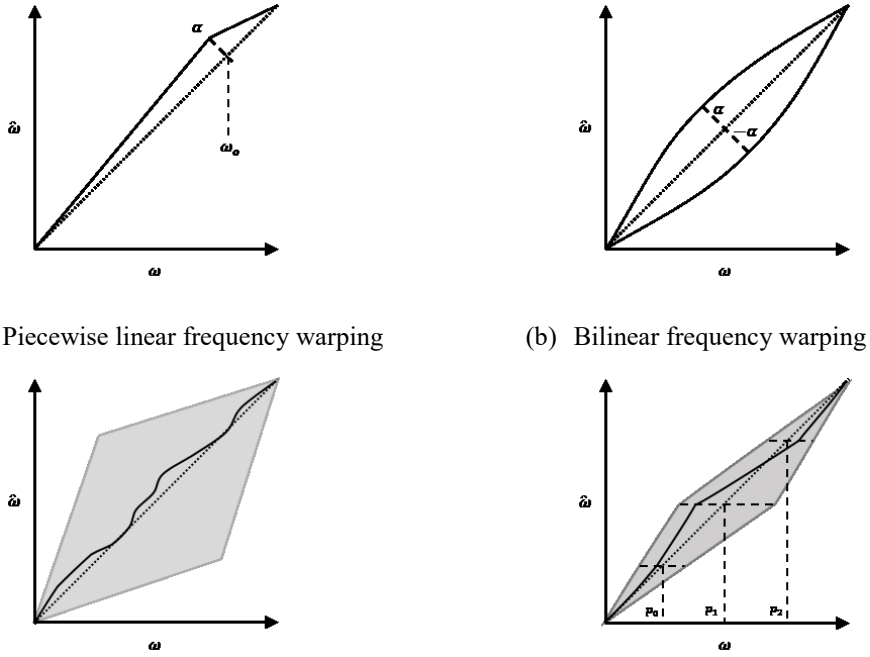
Dynamic Frequency Warping (DFW): this method computes the minimum mean squared error between two spectra to build a warping function using dynamic

programming [83, 84] (see Figure 11 (c)). In the figure, the black line represents a possible frequency warp, and the grey section represents the region of possible warping functions DFW can learn. Because of few constraints on the frequency warp, this function should be more robust to pronunciation differences and disfluencies introduced in nonnative speaker models.

Correlation Frequency Warping (CFW): this method generates a frequency warp which maximizes the correlation between the aligned source and target spectra, as opposed to minimizing spectral distance [44]. The CFW algorithm computes a sequence of warps over n segments centered at frequencies $p_0 \dots p_{n-1}$ (see Figure 11 (d)). In the figure the black line represents a possible frequency warp and the grey section represents the region of warps that CFW can learn. The algorithm greedily selects successive segments that maximize the correlation between the warped source and target spectra. CFW has been shown to be effective at improving synthesis quality compared to DFW in native-to-native conversion.

Table 2: warping functions and optimization methods.

Warp type	Optimization	Objective	
Piecewise linear	Grid search on λ, ω_0	Minimize spectral distortion	[84]
Bilinear	Grid search on α	Minimize spectral distortion	[84]
Dynamic Frequency	Greedy	Minimize spectral distortion	[83]
Correlation Frequency	Greedy	Maximize spectral correlation	[44]



(a) Piecewise linear frequency warping

(b) Bilinear frequency warping

(c) Dynamic frequency warping

(d) Correlation Frequency Warping

Figure 11: frequency warping functions used in this study. For all figures, the source frequency ω is on the x-axis, and the warped frequency $\tilde{\omega}$ is on the y-axis. The dotted line represents no transform between the source frequency ω and the target frequency $\tilde{\omega}$ for illustration purposes.(a) Piecewise linear frequency warping. (b) Bilinear frequency warping. (c) Dynamic frequency warping. (d) Correlation frequency warping, with $n = 4$ segments and three inflection frequencies.

4.4.4. *Relationship to weighted frequency warping*

In this section, we discuss Weighted Frequency Warping (WFW) [4] –briefly discussed in the related work section. As there are similarities to the WFW algorithm and SABR+Res, here we discuss more in-depth the differences between the two methods. First, in contrast to SABR which uses frequency warping methods to transform the *residual*, WFW directly transforms the *source* spectrum, using transforms derived from a GMM trained on aligned source and target data. The second major difference between SABR+Res and WFW is how frequency warps are used to transform spectral information.

WFW computes a linear combination of frequency warps using the set of all frequency warps resulting in a single frequency warp; in contrast, SABR+Res uses a linear combination of transforms, resulting in multiple frequency warps being used simultaneously.

We can illustrate the difference between these transform philosophies by implementing a WFW transform relative to SABR notation. For the t^{th} source residual frame R_S^t, the set of frequency warping functions between each source and target anchor $F = [F_W^1, \dots F_W^K]$, and source weights W_S^t, the WFW residual transform is computed as:

$$\widehat{R_T^t} = CT(FW_S^t)C^T R_S^t. \tag{13}$$

An illustration of these transformation difference is shown in Figure 12. SABR+Res has a similar structure to the corresponding WFW transform; however, in the middle frequency bins, SABR+Res distributes the source residual across multiple frequency warping paths, while WFW maps it to a single frequency warp. This affects synthesis quality when WFW warps residual energy to an incorrect portion of the target spectrum (e.g. between two formants) because of the frequency warps learned between the source and target anchors (something especially important in native-to-nonnative conversion). In contrast, SABR+Res has the flexibility to distribute this energy across multiple frequency warps in the event these functions differ significantly.

(a) (b)

Figure 12: Example of SABR+Res and Weighted Frequency Warping (WFW) transforms between an L1 source speaker and an L2 target speaker.
The transforms are shown in the 40-coefficient Mel frequency scale. (a) SABR+Res linear combination of transforms from eq. (8). (b) WFW frequency warp from eq. (13).

4.5. Experiment design

4.5.1. *Corpus*

To evaluate the proposed residual transform system, we performed a series of experiments using speakers from both the CMU ARCTIC speech corpus [20] and the L2-ARCTIC speech corpus [21]. The L2-ARCTIC speech corpus is a corpus based on the prompts of ARCTIC, but with L2 speakers of English from six L1s: Mandarin, Hindi, Arabic, Spanish, Korean, and Vietnamese. For objective experiments, we used version 1 of the L2-ARCTIC speech corpus, which includes 2 speakers—1 male and 1 female—from all L1s except Vietnamese. In all experiments, the source speakers were American English speakers from ARCTIC. Target speakers were either from ARCTIC or L2-ARCTIC. Including the ARCTIC corpus in the list of target speakers allows for an objective L1-L1 baseline where alignment between the source and target speakers is not a factor. For ease of notation, we refer to ARCTIC—ARCTIC speaker pairs as *A2A*, and ARCTIC—L2-ARCTIC speaker pairs as *A2L2*.

For perceptual experiments, we examined a subset of the A2A and A2L2 pairs to make the perceptual experiments tractable. The pairs used in perceptual experiments are shown in Table 3 and Table 4.

Table 3: A2A perceptual experiment speaker pairs.
Speaker gender is shown in parentheses.

Source	Target
BDL (M)	RMS (M)
BDL (M)	CLB (F)
SLT (F)	SLT (F)
SLT (F)	BDL (M)

Table 4: A2L2 perceptual experiment speaker pairs.
Speaker gender is shown in parentheses.

Source speaker	Target speaker	Target speaker first language
BDL (M)	HKK (M)	Korean
SLT (F)	SKA (F)	Arabic
BDL (M)	YDCK (F)	Mandarin
SLT (F)	EVBS (M)	Spanish

4.5.2. *Implementation details*

For each speaker in ARCTIC and L2-ARCTIC, we used STRAIGHT [39] with 1 ms frame steps and 80 ms window size to extract aperiodicity, fundamental frequency, and spectral envelope. We then computed a 40-dimension MCEP vector ($\alpha = 0.42$, as audio was sampled at 16 kHz). We ignored the first coefficient since it contains energy; given that we desire the transformed utterance to have L1 prosody, during synthesis, we copied the source $MCEP_0$ coefficient to the target.

For synthesis, we converted the source pitch to the target pitch range using log mean-variance scaling [87]. SABR-converted target envelopes were projected from

MCEP back into the STRAIGHT spectrum. Audio was synthesized using the STRAIGHT vocoder with the converted spectral envelope, converted pitch, and source aperiodicity.

To solve for the SABR weights, we used the LARS solver from the SPAMS sparse coding toolbox [89], constraining the Lasso weights to $0 \leq |W|_1 \leq 1$.

4.5.3. *Residual warping comparison*

Initially, we evaluate the proposed SABR+Res residual warping system using the four different frequency warping functions listed in the methods section in an objective experiment. We evaluate these different functions against two other treatments of the source residual:

- *SABR+Identity (baseline)*: the unmodified source residual (i.e., $F_R(R_s) = R_S$). This baseline should be perceptually closer to the source speaker than the target speaker. A successful source residual transform should have a lower VC error than this baseline method.
- *SABR+None (baseline)*: the SABR conversion using just the source and target anchor sets where the source residual is discarded (i.e., $F_R(R_s) = \mathbf{0}$). The performance of this transform will act as a baseline for the identity and speaker quality of the synthesis without compensating for the residual.

4.5.4. *Baseline voice conversion systems*

We also evaluate the performance of SABR+Res against two comparable baseline systems:

- *Weighted Frequency Warping (WFW, baseline):* linear combination of frequency warps, computed from eq. (13), using DFW as the frequency warping function.

- *Exemplar Voice Conversion with Residual Compensation (ERC)*: an exemplar-based voice conversion method which uses time-aligned source and target dictionaries and Partial Least Squares for residual conversion. We use the same factorization (NNMF) and spectral parameters (513-dimensional STRAIGHT spectra) as in the original method [51].

4.5.5. *Objective experiments*

We trained all systems on 20 utterances, following [38]. For SABR models, we examined both parallel and nonparallel training sets to evaluate the performance of the proposed method with nonparallel training data. Training utterances were selected so as to maximize phoneme coverage. Each voice conversion method was evaluated on 50 time-aligned test utterances.

We used Mel-Cepstral Distortion (MCD) as our evaluation criteria. The distortion between two cepstrums X_S and X_T is computed as:

$$MCD(X_S, X_T) = \frac{10\sqrt{2}}{\ln(10)} \|X_S - X_T\|_2^2, \tag{14}$$

where X_S and X_T are vectors of MCEP coefficients [6]. Prior to computing MCD, we set energy ($MCEP_0$) to 0 for both vectors. For the baseline ERC method (which used full spectra), we converted the full spectrum to MCEP and computed MCD in the same fashion.

4.5.6. *Subjective evaluation*

We performed two sets of subjective experiments to evaluate the proposed system. In the first set of experiments, we examined the efficacy of the proposed method against

the two comparison residual methods in synthesis quality, speaker identity, and accentedness experiments. In the second set of experiments, we also compare SABR+Res against the two baseline voice conversion methods from section 4.5.4.

4.6. Results

4.6.1. *Objective results*

First, we examined the objective VC performance of the SABR+Res systems and the baseline methods. The results of the voice conversion objective tests are shown in Table 5. We found that, of the different frequency warping methods tested, the DFW function had the lowest MCD for both A2A and A2L2 speaker pairs and in both parallel and nonparallel training ($p < 0.05$, paired t-test, both A2A and A2L2 pairs). The difference in MCD between parallel and nonparallel training was not significant for any warping functions or speaker pairs ($p = 0.27$, A2A pairs; $p = 0.06$, A2L2 pairs; two-tailed t-tests), providing evidence that SABR+Res method is not significantly affected by a lack of parallel training data.

Table 5: Objective VC results for the residual warping methods. "Source-target testing set" refers to the VC error measured in MCD (dB) of the time-aligned source and target test datasets. Bolded entries show the best SABR+Res warping configuration for parallel and nonparallel training.

			Parallel Training		Nonparallel	
Method	Residual method	Warping function	A2A	A2L2	A2A	A2L2
SABR	SABR+Res	PW Linear	5.26	5.29	5.27	5.38
		Bilinear	5.24	5.30	5.24	5.39
		DFW	***5.09***	***5.22***	***5.12***	***5.31***
		CFW	***5.09***	5.29	5.13	5.39
SABR	SABR+None	None	4.92	4.95	4.95	5.08
SABR	SABR+Identity	None	5.56	5.72	5.59	5.84
Baseline methods						
WFW	N/A	DFW	4.83	5.64	N/A	N/A
ERC	RC	N/A	4.78	5.39	N/A	N/A
Source-target testing set			6.35	7.51	6.35	7.51

In all cases, SABR+Res lowered the MCD significantly as compared to adding the SABR+Identity residual method, providing evidence that the proposed warping function transformed the residual to be closer to the target speaker. Notably, DFW and CFW had very similar performance for A2A speaker pairs on both parallel and nonparallel training utterances, but CFW performed significantly worse on A2L2 speaker pairs, as compared with DFW. This is likely due to mispronunciations or other variations in the L2 speaker's training data which affects the performance of the CFW objective function. Piecewise linear and bilinear warping functions had no statistically significant difference in A2A or A2L2 speaker pairs and in parallel and nonparallel training. The methods also had significantly higher VC error as compared to DFW and CFW, confirming our belief that their simpler transforms were less capable of matching the target speaker than the other warping methods.

SABR+Res significantly outperformed the baseline WFW and ERC systems for A2L2 conversions ($p < 0.01$, both cases, two-tailed t-test). However, baseline system conversions on A2A speakers performed better than the proposed SABR+Res method. This is likely because both the WFW and ERC systems train on parallel source and target data and optimize parameters for conversion, whereas the only parameters learned between the source and target SABR models are the frequency warps in eq. (8).

Note that the proposed SABR+Res method had higher VC error than SABR+None method—that is the normal SABR without any residual. We believe that this is because in sparse coding, the residual is assumed to be uncorrelated with the data represented by the dictionary. By removing the residual, the estimated target spectrum has no

uncorrelated components and is closer to the ground truth. Incorporating the source residual, even transformed to be closer to the target speaker, has the net effect of increasing the MCD while also significantly increasing synthesis quality.

For the remainder of the perceptual studies, we use the DFW frequency warping in the SABR+Res method. While it had the lowest MCD of the four frequency warping techniques we examined, lower MCD does not always indicate perceptually higher quality synthesis. To confirm that DFW was the optimal warping function, we performed a pilot perceptual study, asking participants which frequency warping method they preferred. Participants ($n = 20$) showed a modest preference for the DFW method as compared to the other methods (54-65% preference when compared to the other methods).

4.6.2. *Residual effects*

4.6.2.1. Synthesis quality

We compared the synthesis quality of the three baseline residual warping methods using a Mean Opinion Score (MOS) test, asking participants ($n = 20$) to rate samples on a 5-point scale. For both A2A and A2L2 synthesis directions, participants rated 48 utterances—12 per speaker pair, and 4 per residual method (SABR, SABR+Res, and SABR+Identity). Results are shown in Figure 13.

In both A2A and A2L2 speaker pairs, SABR+Res significantly improved on the baseline SABR+None method (A2A: 3.11, A2L2: 2.64, $p > 0.01$), showing that the proposed residual warping method achieves its goal of improving synthesis quality. However, the SABR+Identity was rated as the highest quality (A2A: 3.58, A2L2: 3.22, $p > 0.01$) and higher than the SABR+Res method. This was expected, as the Identity

residual transform does not transform source residual, so the maximum amount of spectral detail is retained; this lack of transformation will significantly affect the speaker identity of the synthesis.

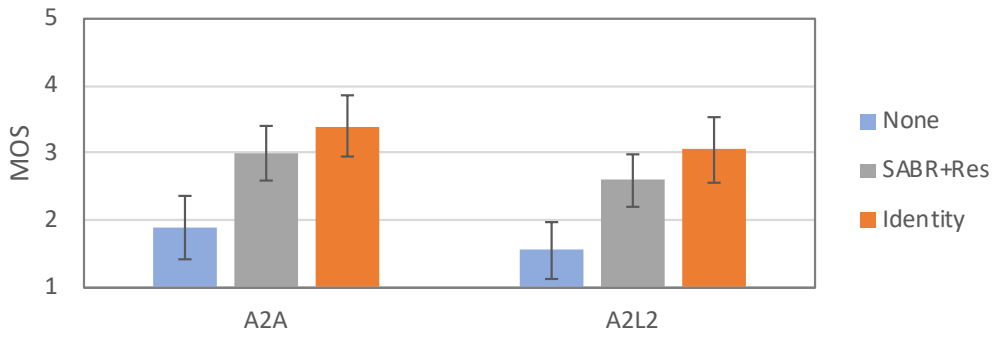


Figure 13: Residual warping method synthesis comparison. Error bars represent 95th percentile confidence intervals.

4.6.2.2. Speaker identity

In a second perceptual test, we compared how well the three residual methods matched the identity of the target speaker using an ABX preference test. We recruited ($n = 20$) participants and presented them with three utterances: an utterance from a target speaker from either A2A or A2L2 speaker pairs (X) and utterances synthesized using two of the baseline residual warping conditions (A, B). The order of A and B was counterbalanced and each utterance had different linguistic content. Participants were asked which utterance—A or B—was closer to X in terms of speaker identity and instructed to ignore accent effects. Participants ($n = 20$) were presented with 148 sets of utterances—48 per pair of methods (24 for A2A conditions and 24 for A2L2 conditions; 6 per speaker pair) and 4 sets of source and target samples to ensure participants were not randomly guessing. The results of this test are shown in Figure 14.

For both A2A and A2L2 speaker pairs, the proposed SABR+Res method was significantly preferred over both the SABR and SABR+Identity methods ($p \gg 0.01$, all conditions), showing that the method was much closer to the target speaker's identity than either method. SABR+Identity was also significantly preferred over the SABR method, to similar degrees as SABR+Res was preferred over SABR (A2A: 72%, A2L2: 65%, $p \gg 0.01$, two-tailed t-test), suggesting that participants associated higher synthesis quality as being closer to the target speaker.

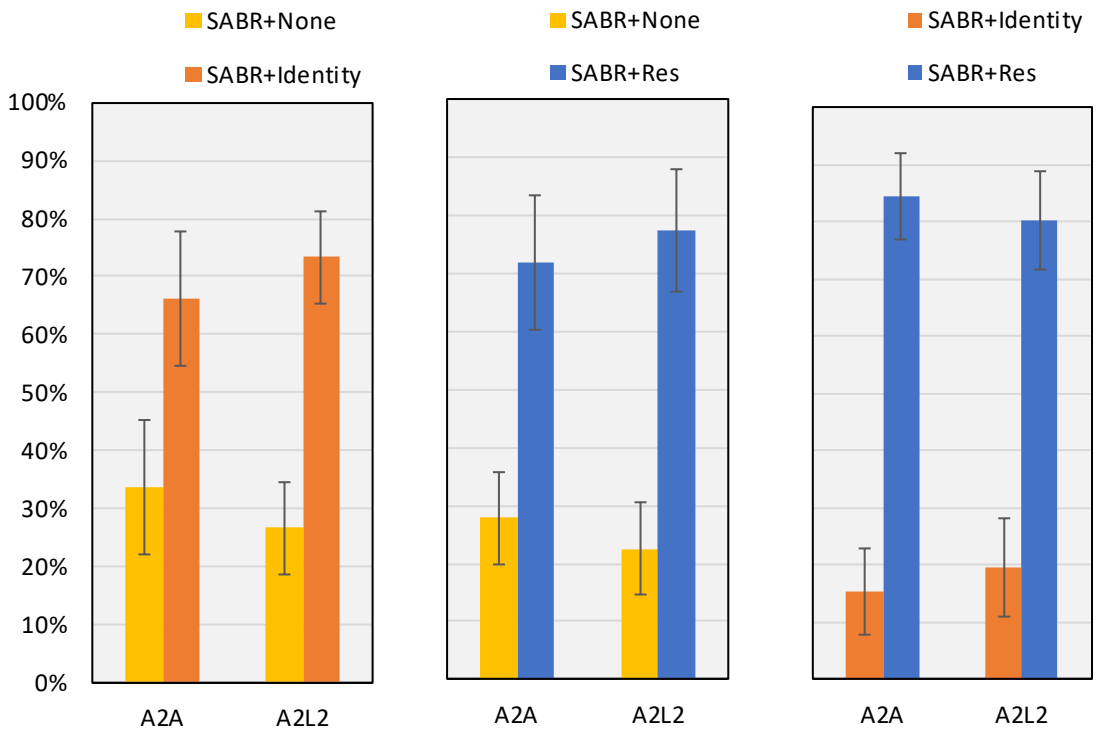


Figure 14: ABX identity test, comparing baseline residual transform methods to SABR+Res.

4.6.2.3. Accentedness

We performed an accentedness test to evaluate how the residual methods affected the accentedness of the synthesis. We asked participants ($n = 20$) to rate the accentedness

of a speaker on a 9-point Likert scale following [90] (1= “no foreign accent”, 9= “very strong foreign accent”) on utterances from the 3 baseline SABR methods (only on A2L2 speaker pairs) as well as utterances from the L1 and L2 speakers. For each condition, participants rated 20 utterances for a total of 100 ratings. Results are shown in Figure 15.

Participants rated the accentedness of the proposed SABR+Res at 1.47, significantly lower than that of SABR (1.88, $p < 0.01$, two-tailed t-test) and far closer to that of the native L1 speakers (1.1). There was no significant difference between SABR+Res and SABR+Ident (1.36, $p = 0.82$, two-tailed t-test), an expected result as both of these methods include the residual from the source native speaker in synthesis. These results demonstrate that not only does SABR encode the accent of the target speaker, the inclusion of the residual warping component further reduces the accentedness of the synthesized speech.

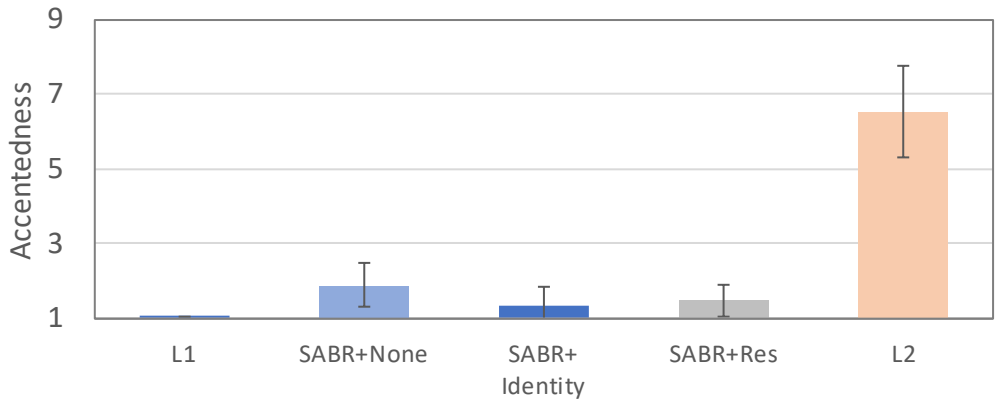


Figure 15: Accentedness ratings for the baseline warping methods.

4.6.3. *Baseline comparison*

4.6.3.1. Synthesis quality

We performed a Mean Opinion Score (MOS) test to compare the synthesis quality of the baseline systems against the proposed SABR+Res method. We asked participants ($n = 10$) to rate the quality of an utterance on a 5-point scale (1 = "low quality"; 5 = "high quality"). Participants rated 124 utterances—40 per synthesis condition (5 per speaker pair, for 8 A2A and A2L2 speaker pairs), and 4 unmodified utterances to ensure the participants were not randomly guessing. Results are shown in Figure 16.

Participants rated the SABR+Res methods significantly higher quality than either of the baseline methods in A2A and A2L2 speaker pairs ($p > 0.01$, all cases, two-tailed t-test). There was no statistically significant difference between the A2A and A2L2 conditions for the SABR+Res synthesis ($p = 0.41$, two-tailed t-test). For both baselines, the A2L2 speaker pairs were rated as lower quality compared to the A2A pairs ($p > 0.01$).

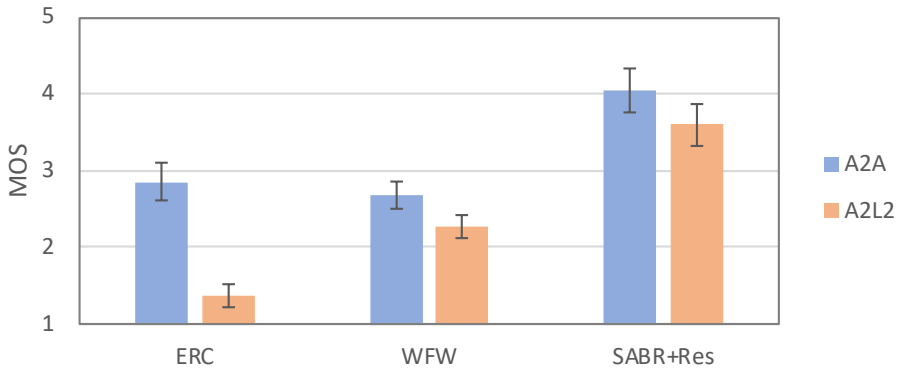


Figure 16: Baseline synthesis quality results (MOS).

4.6.3.2. Speaker identity

In a final perceptual experiment, we performed an XAB speaker identity test to determine how well the baseline and proposed methods were able to capture the target speaker's identity. We recruited ($n = 10$) participants and presented them with three utterances: an utterance from one of the synthesis conditions from either A2A or A2L2 speaker pairs (X) and utterances from the source or target speaker (A, B). The order of A and B was counterbalanced and each utterance had different linguistic content. For each synthesis method, we asked participants to perform 48 evaluations—6 per speaker pair from both the A2A and A2L2 sets, for a total of 144 utterances. We included in the test a set of 4 evaluations where the reference utterance was an unmodified reference from the source speaker to identify participants who evaluated the pairs randomly. Results are shown in Figure 17.

Participants correctly identified the identity of SABR+Res synthesis at a higher rate than the baseline methods for A2L2 speaker pairs ($p > 0.05$, two-tailed t-test) and was significantly higher than WFW in both A2A and A2L2 speaker pairs ($p > 0.05$, two-tailed t-test). However, there was no statistically significant difference between SABR+Res and ERC for A2A speakers ($p = 0.06$, two-tailed t-test). These results further demonstrate that the proposed method is more robust to the alignment and mispronunciation difficulties in native-to-nonnative voice conversion.

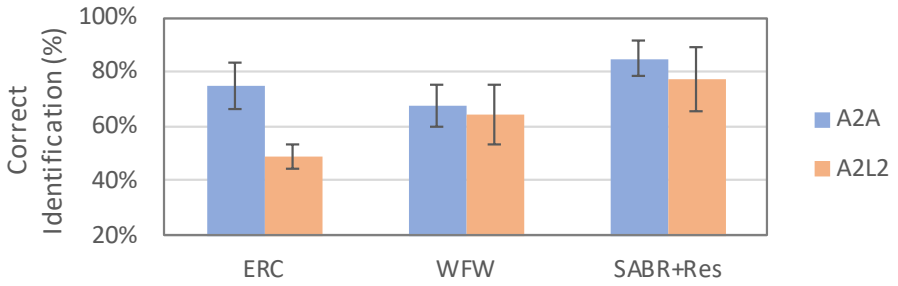


Figure 17: Speaker identity test, comparing SABR+Res to the baseline VC methods.

4.7. Discussion

4.7.1. *Objective results*

The proposed method had significantly lower VC error than the comparison residual methods (SABR+None, SABR+Ident) and the baseline VC methods (WFW, and ERC) for native-to-nonnative conversion, demonstrating that it is not affected by the mispronunciation and time-alignment issues that affect the baseline algorithms. Additionally, there was no statistically significant difference in VC error between parallel and nonparallel training for SABR+Res synthesis. Dynamic Frequency Warping (DFW) had the lowest VC error for both A2A and A2L2 speaker pairs, however, there was no statistically significant difference between any of the three warping functions.

While the baseline methods outperformed SABR+Res in A2A conversion, this is explained by differences in training approaches: the baseline methods train on parallel data, whereas the proposed method does not train directly on time-aligned data, instead learning warping parameters by phoneme label. Introducing parallel training components into SABR+Res (e.g. by training warping functions on time-aligned training data, as opposed to just the anchors) would further reduce the VC error to similar levels as WFW

and ERC for A2A speakers—however, it would remove one of the advantages of the SABR method, namely that it does not require parallel training data.

The flexibility of the SABR+Res method as compared to the WFW method is visible in the spectrum shown in Figure 18; in this figure, the black line represents the SABR+Res spectrum and the red line the WFW spectrum. Differences in the SABR+Res and WFW are most apparent in the third formant at roughly 2.5 kHz. WFW warps the spectrum single warping function, resulting in significant amounts of energy being distributed incorrectly, whereas the multiple transforms used by SABR+Res keep the resulting spectrum near the ground truth, time-aligned spectrum (represented by the blue line). These differences show the advantages of SABR+Res: at frequencies where multiple warps are similar, the source residual will be transformed in the same way it would be transformed with a single warping function. However, when there are dissimilarities, the warping function can distribute the energy among multiple frequencies and lower the overall error. This has the added benefit of reducing VC error by not adding spectral detail where it does not belong.

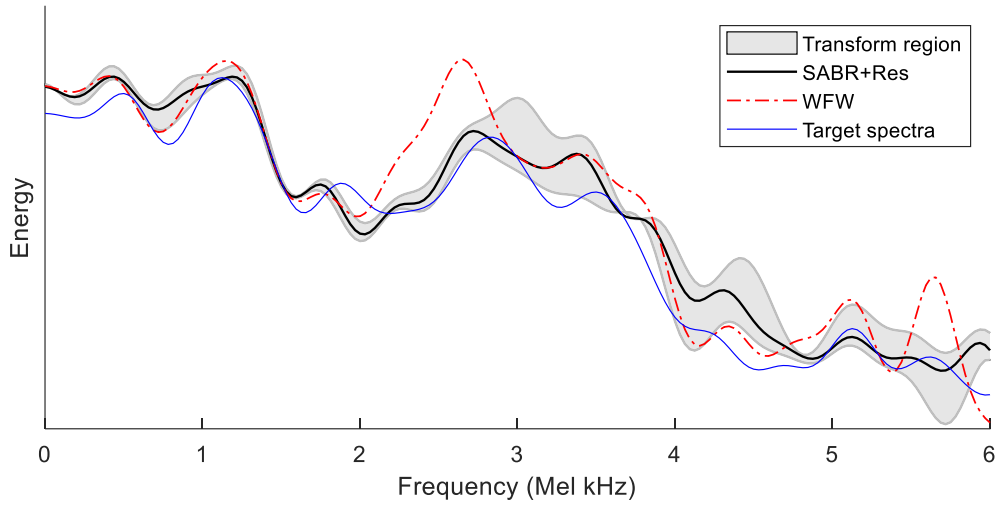


Figure 18: Comparison of SABR+Res transform, WFW transform, and the target spectrum.
The gray area represents the range of possible energies to which SABR+Res can transform the source residual, given the nonzero weights and anchor warps.

4.7.2. *Subjective results*

The first experiment validated our intuition that the proposed SABR+Res method was effective at increasing the synthesis quality of the SABR method while also capturing the target speaker's voice quality, in both native-to-native and native-to-nonnative contexts. The combination of these results and the synthesis quality experiment demonstrate that SABR+Res both improves upon the synthesis quality of the baseline SABR method and significantly improves upon the identity of the speaker.

The experiments in section 4.6.3 demonstrate that SABR+Res is significantly more robust to mispronunciation effects of nonnative speakers over the baseline WFW and ERC methods. Participants rated SABR+Res synthesis significantly higher quality in both A2A and A2L2 speaker pairs, and participants correctly identified the target speaker in A2L2 pairs at significantly higher rates than the baseline methods. These results show

that the phoneme-based representation of SABR is more robust to disfluencies in L1-L2 conversion as compared with baseline exemplar-based VC methods, and that the SABR+Res residual transformation method generates similar synthesis quality for both native-to-native and native-to-nonnative conversion.

An explanation for the poor performance of the baseline systems on A2L2 speakers is that the baseline systems are affected by the time alignment of the source and target training data. For the WFW baseline, the synthesis quality did not differ significantly between the A2A and A2L2 systems because first step of the WFW algorithm warps the source utterance before scaling the amplitude of the warped spectrum, retaining much of the spectral detail and, therefore, synthesis quality. However, the second amplitude scaling step is where mispronunciations and time alignment issues in the A2L2 speakers affects the speaker identity, as this step is unable to correctly adjust the output spectra to match the target speaker's voice identity.

For the ERC baseline, similar alignment issues affect the A2L2 synthesis quality and identity. ERC has two components: a time-aligned source and target dictionary, and a residual compensation PLS mapping component, designed to map the source residual to the target speaker and increase the spectral detail of the synthesis. Disfluencies and mispronunciations in the target speaker's dictionary introduce distortions and overall lower the spectral detail of the synthesis in such a way that the PLS mapping cannot resolve it. The PLS mapping relies on the source and target residuals having similar, consistent structures—if they do not, as in the case with native-to-nonnative conversion, the spectral detail is lost as the mapping can only learn average conversions between the

two systems. This is not the case in A2A systems when relatively few disfluencies affect both the dictionaries and the residual mapping.

4.8. Conclusion

To address the lower synthesis quality that arises from the compact size of SABR's dictionaries, we have proposed and evaluated a method named SABR+Res which transformed the source residual to the target speaker using frequency warping functions and adding it to the estimated target spectrum. We examined four methods for performing frequency warping: Piecewise linear frequency warping, Bilinear frequency warping, Dynamic Frequency Warping (DFW), and Correlation Frequency Warping (CFW). We also compared the proposed residual warping function against two established voice conversion baselines. We tested these systems in parallel and nonparallel training and native-to-native and native-to-nonnative conversion contexts. In objective tests, SABR+Res using DFW as its frequency warp was determined to have the lowest objective VC error and had significantly lower error than the baseline voice conversion methods on native-to-nonnative voice conversion.

Following this, we conducted a series of subjective tests to evaluate the proposed residual method in two contexts. First, we evaluated SABR+Res relative to two other treatments of the source residual (ignoring the residual and not transforming the source residual) in synthesis quality, speaker identity, and accentedness tests. Participants rated SABR+Res as having significantly higher synthesis quality over SABR, as being significantly closer to the target speaker's identity, and as having a more native accent in native-to-nonnative conversion. In a second set of experiments, we compared SABR+Res

against two baseline voice conversion methods. Both synthesis quality and speaker identification rates were similar for the baseline methods and SABR+Res on native-to-nonnative conversion, but in native-to-nonnative conversion, the proposed method performed significantly better than the baseline algorithms. These results validate the use of the proposed residual warping method to both improve the synthesis quality in native-to-nonnative voice conversion contexts, all while using an extremely compact dictionary (39 atoms, one for each English phoneme). Additionally, the proposed method lowered the accentedness of the synthesis as compared to the baseline methods.

In the following chapters, we use the residual warping method here in the synthesis of the SABR method in perceptual studies. We also examine the final two research aims related to the SABR method: selecting optimal anchors for voice conversion, especially in native to nonnative contexts, and adding temporal constraints to the SABR objective function.

5. OPTIMIZING ANCHOR SELECTION FOR SABR VOICE CONVERSION IN NATIVE AND NONNATIVE CONTEXTS*

5.1. Overview

In this chapter, we evaluate two methods for optimizing the anchor sets of the SABR method. The first method focuses on minimizing the residual of the source and target anchor sets, resulting in anchors that more closely match the source or target speaker's distribution and lower residuals. The second method addresses the issue that single anchors were used to represent phonemes, even though phonemes may have multiple acoustic states over the course of the production. Both of these methods address problems in native to nonnative conversion and we also evaluated them in this context. This chapter addresses the third aim of this dissertation.

The ARS algorithm presented in this chapter was accepted at Interspeech 2021. The IRT algorithm will be published at a future venue to be determined.

5.2. Introduction

In the prior chapters, we established how to perform sparse, anchor-based voice conversion and presented a technique for improving synthesis using the sparse coding residual. However, the formulation of the anchors as using a single anchor per phoneme is ill-suited for phonemes with known subphoneme states (e.g., stops or affricates). Further, in instances where the source speaker and target speaker share different accents,

---

* Parts of this chapter are reprinted with permission from "An Exemplar Selection Algorithm for Native-Nonnative Voice Conversion" by C. Liberatore and R. Gutierrez-Osuna 2021. *Interspeech 2021*, p. 841-845, Copyright 2021, *International Speech Communication Association*.

the anchors are learned independently, potentially leading to mismatches in their phonetic content. Finally, anchors are selected only in the context of their phoneme label and not in the context of the other selected anchors, resulting in larger residuals. These issues combine to lower the overall synthesis quality of SABR models and is exacerbated in nonnative voice conversion contexts.

In this chapter, we propose two learning algorithms to address the above limitations of SABR. The first, Iterative Retraining (IRT), performs dictionary learning to optimize the initial source and target anchors, reducing the residual, as well as the VC error. As the name suggests, IRT operates iteratively: it uses the source weights to update the target anchors, and then uses the target weights to update the source anchors, back and forth. In this fashion, IRT reduces the residual as well as the VC error. The second algorithm, Anchor Removal and Selection (ARS), performs clustering and greedily removes or splits anchors to reduce the VC error, allowing multiple anchors to represent a phoneme or the anchor to be removed entirely. Both algorithms can also be used in combination (ARS+IRT), where the output of the ARS exemplar selection algorithm is further optimized by the IRT dictionary learning algorithm. We evaluate both optimization algorithms using a dataset of speech recordings from native and non-native speakers in the ARCTIC [81] and L2-ARCTIC [78] corpora, respectively, and compared them against a state-of-the-art exemplar-based VC baseline [51].

This chapter is organized as follows. First, we review exemplar-selection and optimization literature and how accents can influence these decisions. Second, we present the two anchor optimization algorithms and discuss how both solve different problems

with the SABR algorithms. Then, we present experimental results of the two algorithms against a baseline exemplar-based VC algorithm. We end the chapter with a discussion of the results and our conclusions.

5.3. Related Work

In this section, we review literature relevant to selecting and designing exemplars for exemplar-based voice conversion. Additionally, we discuss difficulties that arise from selecting exemplars and handling model training in native-to-nonnative contexts.

Two constraints arise when selecting exemplars for use in VC. First, they should be chosen to minimize the residual of the reconstructions. Second, exemplars from the two speakers must have similar phonetic content, or the conversions will be distorted or unintelligible [46, 91]. SABR ensures that source and target dictionaries share similar phonetic content by learning one exemplar per phoneme from labeled training data. This results in much more compact dictionaries than other exemplar-based methods [51, 92]. However, as fewer exemplars are included in speaker dictionaries, the residual magnitude increases. Additionally, as fewer exemplars can represent less of the variance of the data set, the resulting synthesis generally has lower quality. Thus, an important task for compact VC methods, such as SABR, is to select exemplars to minimize this residual — and maximize the amount of variance in the data that is explained—while ensuring high-quality synthesis.

Several techniques have been used to include phonetic information in the exemplar selection process. Aihara *et al.* [92] proposed a method for building a phoneme-categorized dictionary, which added a penalty function to the Nonnegative Matrix

Factorization (NMF) objective function so that the conversion algorithm was forced to select target exemplars from the same phonetic class as the source. They found that learning source and target phoneme dictionaries with a cost function that enforced phonemic constraints significantly improved synthesis quality in subjective and objective experiments. Sisman *et al.* [55] expanded on this method by appending a phonetic posteriorgram (PPG) to the selected exemplar dictionaries to encode additional phonetic information. They found that including this phonetic information improved perceptual and objective measures of voice conversion. More recently, Ding et al. [91] found that learning latent phonemic information in source and target dictionaries can significantly improve VC quality in exemplar-based methods. The authors proposed a method to build source and target exemplar dictionaries by learning latent clusters in source and target data using a hard-clustering algorithm. They found that the selected clusters were associated with important phoneme classes, evidence that the proposed method was learning latent phonological information contained in the speech signal. These results suggest that selecting exemplars in such a way as to retain similar phonetic content will significantly improve synthesis quality.

In related work, Zhao and Gutierrez-Osuna [52] examined two methods to select a compact set of exemplars for exemplar-based VC. The first method was a forward selection procedure where the exemplars that reduced the VC error most significantly were added to the dictionary; the second method was a backwards-elimination procedure where the exemplars that contributed the least to the sparse-coding weights were removed. Both procedures were able to outperform a baseline method based on time-aligned dictionaries.

More importantly, the results showed that it is possible to reduce exemplar dictionaries by a factor of five without any significant decrease in VC performance.

5.4. Methods

We propose two approaches to optimize the SABR anchor sets A_S and A_T. Both algorithms begin with initial SABR anchor sets A_S and A_T, collected from labeled training data in the same manner as done in Chapter 3, then optimize them on parallel source and target training data. The first approach, *Iterative Retraining (IRT)*, is a dictionary learning algorithm that balances two criteria: minimizing the VC error and minimizing the residual error on the source and target utterances. The second approach, *Anchor Removal and Selection (ARS)*, is a hybrid clustering and exemplar-selection algorithm that adds or removes exemplars from the source and target anchor sets to decrease the VC error.

5.4.1. *Iterative Retraining*

Iterative Retraining (IRT) is a dictionary learning method based on the Method of Optimal Directions (MOD) [93]. Dictionary learning algorithms are designed to update dictionaries in such ways as to minimize the residual error of the representation. Given an utterance X, a source dictionary A, and an activation matrix W, MOD computes an update ΔA to the dictionary as:

$$\Delta A = (X - AW)(W^{+}), \tag{15}$$

where $(\cdot^{+})$ is the Moore-Penrose Pseudoinverse with a regularization parameter, Γ:

$$W^{+} = \left(WW^{T} + \Gamma\right)^{-1} W. \tag{16}$$

Optimizing the source and target anchors *independently* will reduce their residuals for encoding the source and target speakers, respectively; however, unconstrained, these

updates will not ensure that the anchors are still tuned to be able to perform *voice conversion*. To address this, we add a second update term that also updates anchor sets in the direction of reducing the VC error. Let us denote by $R_{S|T}$ the residual error when representing a source utterance with target weights:

$$R_{S|T} = X_S - A_S W_T, \tag{17}$$

where W_T is computed from and A_T and X_T using eq. (3). Then, we establish a tradeoff between optimizing the anchors to minimize the source residual R_S in eq. (5) and the VC error term in eq. (17) :

$$\Delta A'_S = \left(\alpha R_s + (1-\alpha) R_{S|T}\right) W_S^+, \tag{18}$$

where α is a parameter that balances the two update terms. Source anchors on iteration t are then updated:

$$A_S^{t+1} = A_S^t + \Delta A_S^{'}. \tag{19}$$

Following this, the IRT algorithm iterates, with the source anchors A_S^t being fixed and the target anchors A_T^{t+1} being updated in a similar fashion.

To prevent IRT from overfitting, we split the training data into two non-overlapping subsets: one subset ($X_S^{(1)}$and $X_T^{(1)}$) that is used to update the source anchors, and the other subset ($X_S^{(2)}$ and $X_T^{(2)}$) that is used to update the target anchors. The algorithm then proceeds in an iterative fashion: updating the source anchors based on the target weights (using the first subset), and then updating the target anchors using the source weights (using the second training subset), following eqs. (17)-(19). The overall procedure

Algorithm 1: Iterative Retraining

Inputs:
A_S, A_T: Initial source and target anchor sets
$X_S^{(1)}$, $X_T^{(1)}$: Source-target training data
$X_S^{(2)}$, $X_T^{(2)}$: Target-source training data
α: Source-target residual weighting constant

Initialize: $t = 0, e^0 = \infty,\ A_S^0 = A_S,\ A_T^0 = A_T$

1. Repeat for t=1..end

   /* Update the source anchors using the target weights and source residual */

2. $W_S^{(1)} = \min_{W_S} \left\| X_S^{(1)} - A_S^t W_S^{(1)} \right\|_2^2 + \left\| W_S^{(1)} \right\|_1, s.t. \left\| W_S^{(1)} \right\|_1 \leq 1$

3. $W_T^{(1)} = \min_{W_T} \left\| X_T^{(1)} - A_T^t W_T^{(1)} \right\|_2^2 + \left\| W_T^{(1)} \right\|_1, s.t. \left\| W_T^{(1)} \right\|_1 \leq 1$

4. $R_{S|T} = X_S^{(1)} - A_S^t W_T^{(1)}$ //Target to source, eq. (17)

5. $R_S = X_S^{(1)} - A_S^t W_T^{(1)}$ //Source residual, eq. (5)

6. $\Delta A_S' = \alpha R_S \left(W_S^{(1)} \right)^+ + (1-\alpha) R_{S|T} \left(W_S^{(1)} \right)^+$

7. $A_S^{t+1} = A_S^t + \Delta A_S'$

   /* Update the target anchors using the source weights and target residual */

8. $W_S^{(2)} = \min_{W_S} \left\| X_S^{(2)} - A_S^{t+1} W_S^{(2)} \right\|_2^2 + \left\| W_S^{(2)} \right\|_1, s.t. \left\| W_S^{(2)} \right\|_1 \leq 1$

9. $W_T^{(2)} = \min_{W_T} \left\| X_T^{(2)} - A_T^t W_T^{(2)} \right\|_2^2 + \left\| W_T^{(2)} \right\|_1,\ s.t. \left\| W_T^{(2)} \right\|_1 \leq 1$

10. $R_{T|S} = X_T^{(2)} - A_T^t W_S^{(2)}$ //Source to target, eq. (17)

11. $R_T = X_T^{(2)} - A_T^t W_T^{(2)}$ //Target residual, eq. (5)

12. $\Delta A_T' = \alpha R_T \left(W_T^{(2)} \right)^+ + (1-\alpha) R_{T|S} \left(W_T^{(2)} \right)^+$

13. $A_T^{t+1} = A_T^t + \Delta A_T$

Return: $A_S^t,\ A_T^t$

is outlined in Algorithm 1. IRT iterates for a set number of iterations and returns the t-th iteration anchor sets A_S^t and A_T^t.

5.4.2. *Anchor Removal and Selection*

The second optimization method, Anchor Removal and Splitting (ARS), addresses two issues. First, using single anchors per phoneme may not be enough to represent some phonemes classes, such as stops or affricates, which contain several sub-states. Second,

the phoneme inventory of the L2 speaker may be different from that of the L1 speaker and will likely include mispronunciations of a phoneme. As a result, the source-target anchors may be mismatched, introducing distortions in the VC synthesis. To address these issues, ARS greedily either removes an anchor or "splits" it into sub-anchors, depending on which action reduces the VC error. After the operation is performed, the algorithm iterates again on the new anchor set, until a termination condition is reached, or the error can no longer be reduced.

As a first step, we compute a binary tree of cluster centroids for each phoneme using Ward's method [94]. These clusters are learned by concatenating time-aligned source and target training data: $[X_S^T, X_T^T]^T$. The root node of the binary cluster tree corresponds to the initial anchors for each phoneme. During the *split* operation, a given node is replaced with its two child nodes, which represent to two higher-detail clusters in that phoneme. The *removal* operation is less complex; the operation simply removes the given anchor from the anchor set and all child nodes from that tree.

These two operations represent the two choices that the greedy ARS algorithm can perform. For each anchor k, each operation f in the set of operations $F = \{remove, split\}$ is performed, resulting in a temporary anchor sets $A_S^{k,f}, A_T^{k,f}$. For each anchor-operation pair, the VC error is measured against a validation data set X_S' and X_T'. The temporary anchor with the minimum VC error is used as the input to the next iteration, and each anchor is again tested for the split and remove operation. This process loops either until the VC error stops improving, or after a set number of iterations is reached. The overall procedure is outlined in Algorithm 2.

ALGORITHM 2: ANCHOR REMOVAL AND SPLITTING ALGORITHM

Inputs:
A_S, A_T: Source and target anchor sets
X_S, X_T: Parallel source and target training data sets
X'_S, X'_T: Parallel source and target validation data sets
F: Set of anchor selection functions (removal, splitting)

Initialize: $t = 0, e^0 = \infty,\ A_S^0 = A_S,\ A_T^0 = A_T$

1. $K = |A_S^t|$ //Compute the number of anchors
   /* For each anchor, perform the removal or split function and compute the voice conversion error */
2. For each $k \in K$
3. For each $f \in F$
   /* Perform the operation f on the anchor k for the source and target speakers */
4. $[A_S^{k,f}, A_T^{k,f}] = f(A_S^t, A_T^t, X_S, X_T, k)$
   /* Compute the voice conversion error on the validation data set */
5. $W_S = \min_{W_S} \left\| X'_S - A_S^{k,f} W_S \right\|_2^2 + \|W_S\|_1, \quad s.t. \|W_S\|_1 \leq 1$
6. $e_{k,f}^t = \left| |X'_T - A_T^{k,f} W_S| \right|_2^2$
   // min VC error operation-anchor pair
7. $[k, f] = \min_{k,f}([e_{1,1}^t \dots e_{1,|K|}^t \dots e_{|F|,1}^t \dots e_{|F|,|K|}^t])$
8. $e^t = e_{k,f}^t$
   /* If the operation reduced the VC error, update the anchor sets and iterate again */
9. $if(\mathrm{e}^{\mathrm{t}} < e^{t-1})$
10. $A_S^{t+1} = A_S^{k,f}$
11. $A_T^{t+1} = A_T^{k,f}$
12. $t = t + 1$
13. Go to line 2
14. *else* return

Return: $A_S^t,\ A_T^t$

5.5. Experiments

5.5.1. *Corpus*

To evaluate the proposed anchor optimization algorithms, we performed a series of experiments using speakers from the CMU ARCTIC speech corpus [81] and the L2-ARCTIC speech corpus (v 1.0) [78]. L2-ARCTIC is a corpus based on the prompts of the ARCTIC database, but with L2 speakers of English from six first languages: Mandarin, Hindi, Arabic, Spanish, Korean, and Vietnamese. At the time of these experiments, data

from Vietnamese speakers was not available and were therefore not included in these experiments. We conducted two types of evaluation: objective and subjective. For the objective evaluations, source speakers were American English speakers from the ARCTIC corpus, and target speakers were either from the ARCTIC or L2-ARCTIC corpus. Including the ARCTIC corpus in the list of target speakers provided a native-native VC baseline where alignment between source and target speakers was not a factor. For the subjective experiments, we did not evaluate ARCTIC-ARCTIC pairs, since our goal is to perform accent conversion. For ease of notation, we refer to ARCTIC-ARCTIC speaker pairs as *A2A*, and ARCTIC—L2-ARCTIC speaker pairs as *A2L2.*

5.5.2. *Implementation details*

We used STRAIGHT [39] with 1 ms frame steps and 80 ms window size to extract aperiodicity, fundamental frequency, and spectral envelope from each utterances. We then computed a 25-dimension MFCC vector (25 filter banks, 25 coefficients). We ignored $MFCC_0$ since that contains energy, and we wanted target utterances to have native prosody. Instead, at synthesis we copied the source $MFCC_0$ to resynthesize the target utterance.

To keep the algorithm low-resource, we used 20 parallel, time-aligned utterances to train the SABR models and as input to the ARS and IRT algorithms. We performed time alignment using the MFCC features and dynamic time warping (DTW) [95]. To illustrate the time-alignment difference between native and nonnative speaker pairs, we examined the average difference between the computed DTW trajectories for A2A and A2L2 speaker pairs (see Table 6). On average, A2A pairs had a 124ms (standard deviation:

15ms) alignment difference, whereas A2L2 pairs had a 221ms (standard deviation: 36ms) alignment difference. *These results highlight the challenges of using conventional exemplar-based VC methods, which require accurate alignment, when the target speakers are non-native.*

For synthesis, we converted the pitch of the source utterance to match the pitch range of the target speaker using log mean-variance scaling [6]. Then, we synthesized audio using the STRAIGHT vocoder with the converted spectral envelope, the converted pitch, and the source aperiodicity. To solve for the SABR weights, we used the LARS solver from the SPAMS sparse coding toolbox [89], following the method and constraints described in Chapter 4.

5.5.3. *Accent-conversion systems*

To evaluate the two proposed optimization algorithms, we considered five different accent-conversion systems:

- *SABR*: the default SABR anchors—one anchor per phoneme, selected by computing the centroid of all frames with that phoneme label
- *IRT*: source and target anchors optimized by the IRT algorithm
- *ARS*: source and target anchors optimized by the ARS algorithm
- *ARS+IRT*: a combination of the ARS and IRT algorithms, where the resulting source and target anchor sets from the ARS algorithm are used as initial anchor sets to the IRT algorithm.

- *Baseline:* a state-of-the-art exemplar-based method with residual compensation[4] [51]. Source and target dictionaries were learned from time-aligned data; in this case, the dictionaries were constructed from the same training utterances used by the SABR anchors and optimization methods (i.e. 20, time-aligned source and target utterances).

We did not consider other VC methods (e.g., neural network, GMM) as baselines, since prior work [10, 38, 50] has established that such methods perform worse than exemplar-based methods when limited training data are available.

Table 6: Average time alignment differences when aligning source utterances to target utterances from speakers with different L1s.
Values in parenthesis are the standard deviations of the measurements.

Native L1	ARCTIC alignment error (ms)
Hindi	160 (36)
Arabic	188 (50)
Spanish	205 (64)
Korean	214 (53)
Mandarin	335 (80)
L2-ARCTIC average	221 (36)
ARCTIC average	124 (15)

5.5.4. *Experiments*

Prior to the objective experiments, we performed a cross-validation experiment to tune the parameters of the IRT optimization method. In this experiment, we performed 4-fold cross validation on 80 utterances (20 per split) from the ARCTIC *B* training set. Utterances were selected as to maximize phoneme variability and ensure that SABR has

[4] For consistency with the original implementation, the baseline method operated on the full STRAIGHT spectra, as opposed to the MFCCs used by the other four methods. This gave the baseline method an advantage in terms of acoustic quality.

samples for each phoneme anchor. The optimal number of iterations and α value were selected from this initial experiment.

For the objective experiments, we used all possible pairs of speakers from A2A (12 pairs) and A2L2 (40 pairs). Training was done on the 20 utterances from the ARCTIC *B* set, and testing was done on 200 utterances selected from the ARCTIC *A* set. As the ARS algorithm requires a validation set, we divided each 20-utterance split into a 10-utterance training and a 10-utterance development set. Utterances used in perceptual tests were selected from the ARCTIC *A* set.

For subjective experiments, we selected four speaker pairs from the A2A set (Table 7) and four speaker pairs from the A2L2 set (Table 8) for evaluation. First, we performed a Mean Opinion Score (MOS) test on A2A and A2L2 pairs to measure the synthesis quality of the proposed systems. Second, we performed an accentedness test on only A2L2 speaker pairs to evaluate how the optimization methods affect the accent of the synthesis. Following this, we performed an XAB speaker identity test on just A2L2 pairs to measure the speaker identity performance of the five systems. Finally, we performed an AB preference (ABP) test on A2L2 speakers to determine which methods were preferred in a more direct test.

Table 7: A2A speaker pairs for perceptual experiments.
Source and target speakers are both from the ARCTIC corpus.

Source speaker	Target speaker
BDL (M)	RMS (M)
SLT (F)	CLB (F)
RMS (M)	SLT (F)
CLB (F)	BDL (M)

Table 8: Speaker pairs for the A2L2 perceptual experiments.
Source speakers are from the ARCTIC corpus, target speakers are from the L2-ARCTIC corpus.

Source speaker	Target speaker	First language
BDL (M)	HKK (M)	Korean
SLT (F)	SKA (F)	Arabic
RMS (M)	YDCK (F)	Mandarin
CLB (F)	EVBS (M)	Spanish

We selected two independent objective measures to evaluate the proposed algorithms: the *VC error,* computed from eq. (17), and the *Residual magnitude* from eq. (5), computed in terms of Mel-Cepstral Distortion [6]:

$$MCD(C) = \frac{10\sqrt{2}}{\ln(10)} \|C\|_2^2, \tag{20}$$

where C is a vector of MFCCs. Prior to computing MCD, we set energy ($MFCC_0$) to 0. For the baseline method (which used full spectra), we convert the full spectra to MFCCs.

Additionally, as noted by prior research, higher correlations between pairs of source and target atoms is an indicator of higher synthesis quality [48, 96]. Thus, we added a third objective measure of synthesis quality: the correlation of the source and target anchors sets as:

$$r(a,b) = \frac{\sum_m \sum_n (a_{mn} - \bar{a})\left(b_{mn} - \bar{b}\right)}{\sqrt{\left(\sum_m \sum_n (a_{mn} - \bar{a})^2\right)\left(\sum_m \sum_n \left(b_{mn} - \bar{b}\right)^2\right)}}, \quad (21)$$

where $a, b \in \mathbb{R}^{m \times n}$ are source and target anchors, m is the feature dimension, n is the number of anchors, $\bar{a}$ and $\bar{b}$ are the mean of the anchors. We computed the correlation coefficients for the source and target anchor sets on all coefficients, except energy ($MFCC_0$). For the baseline method (which used full spectra), we converted the source and target dictionaries to MFCCs.

5.6. Results

5.6.1. *Experiment 1: Objective evaluation*

5.6.1.1. Characterizing the Iterative Retraining algorithm

Prior to evaluating the IRT algorithm's performance on test data, we examined the effect of parameter α in eq. (11), which balances the VC error and the residual error. We evaluated the IRT algorithm on all 40 A2L2 speaker pairs. Figure 19 shows the average VC error and residual, iteration by iteration, on the cross-validation dataset[5]. The VC error reaches a minimum in the first few iterations but increases subsequently. In contrast, the residual tends to decrease monotonically[6]. In the remaining analysis, we took the first iteration where the change in residual magnitude was below 0.001 dB as the convergence iteration.

[5] Empirically, we set to $\Gamma = 0.1$ regularization term in eq. (16)

[6] For readability, Figure 19 only shows A2L2 speaker pairs, but A2A pairs reduced the VC error and residual following a similar pattern.

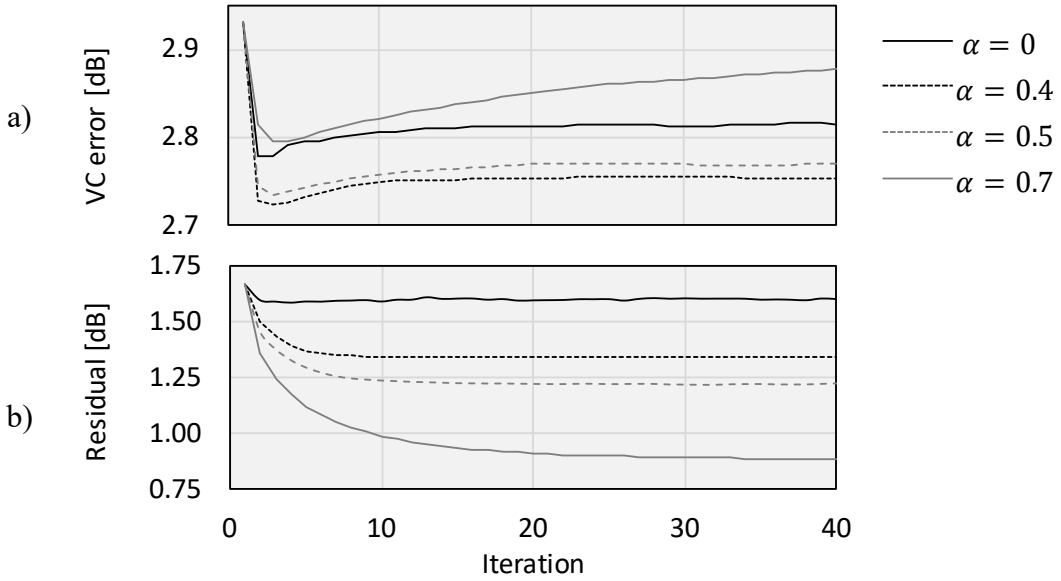


Figure 19: IRT algorithm by iteration on the training set, averaged over the cross-validation folds. (a) VC error of A2L2 pairs. (b) Residual magnitude of A2L2 pairs.

Figure 20 shows the relationship between the residual and VC error (at convergence iteration) as a function of parameter α. For both A2A and A2L2 speaker pairs, $\alpha = 0.4$ achieved the lowest VC error; however, the MCD at $\alpha = 0.5$ was negligibly higher (0.01 dB) with a significant decrease in residual. For $\alpha \geq 0.7$, the VC error diverged as the IRT algorithm was biased towards minimizing the residual and not minimizing the VC error. Based on these results, in what follows we set $\alpha = 0.5$ and perform IRT until the residual converges (in practice, this occurs at around 20 iterations).

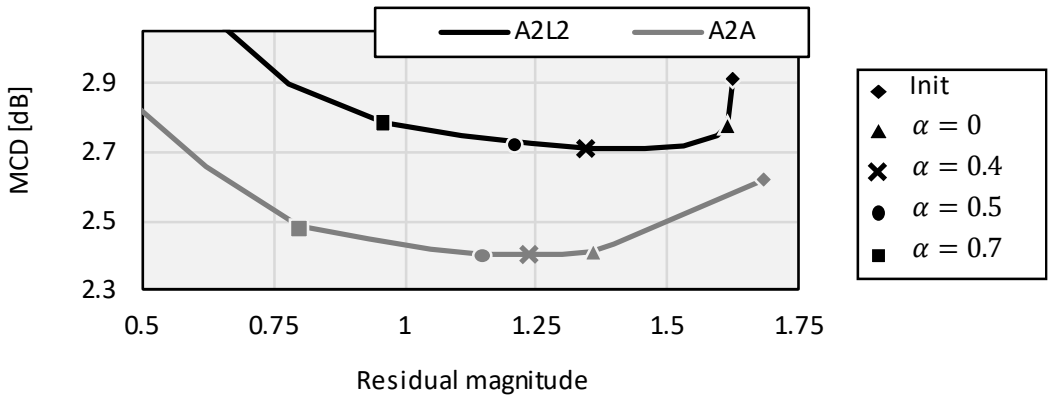


Figure 20: Tradeoff between VC error and residual error for different values of parameter α on the cross-validation dataset.
"Init" refers to SABR models built from phoneme labels.

5.6.1.2. Characterizing the Anchor Selection and Removal algorithm

In this experiment, we evaluated the ARS algorithm on all pairs of A2A and A2L2 speaker pairs. For a set of 20 training utterances, we computed the initial SABR anchors using the centroids of the phoneme labels of the training data, then followed the ARS algorithm. We evaluated the algorithm from two perspectives: the per-iteration results of the metrics from section 5.5.4 and the phonemes the ARS algorithm selected for splitting and removal.

Figure 21 shows the per-iteration results of the ARS algorithm. ARS reduces the VC error (Figure 21 (a)) for both speaker pairs, but more for A2L2 pairs because of pronunciation differences between the source and target speakers. In contrast, residuals decrease similarly for both A2A and A2L2 pairs (Figure 21 (b)). Evidence of time-alignment and accent issues are visible in the number of anchors selected (Figure 21 (c)). Because A2A pairs are generally not affected by accent or time-alignment issues, the ARS algorithm favored the splitting operation, and significantly more splitting operations were

selected over the removal operation. The A2L2 pairs reach an average of 60 anchors, whereas the A2A pairs continue to split and reach an average of 69.8 anchors.

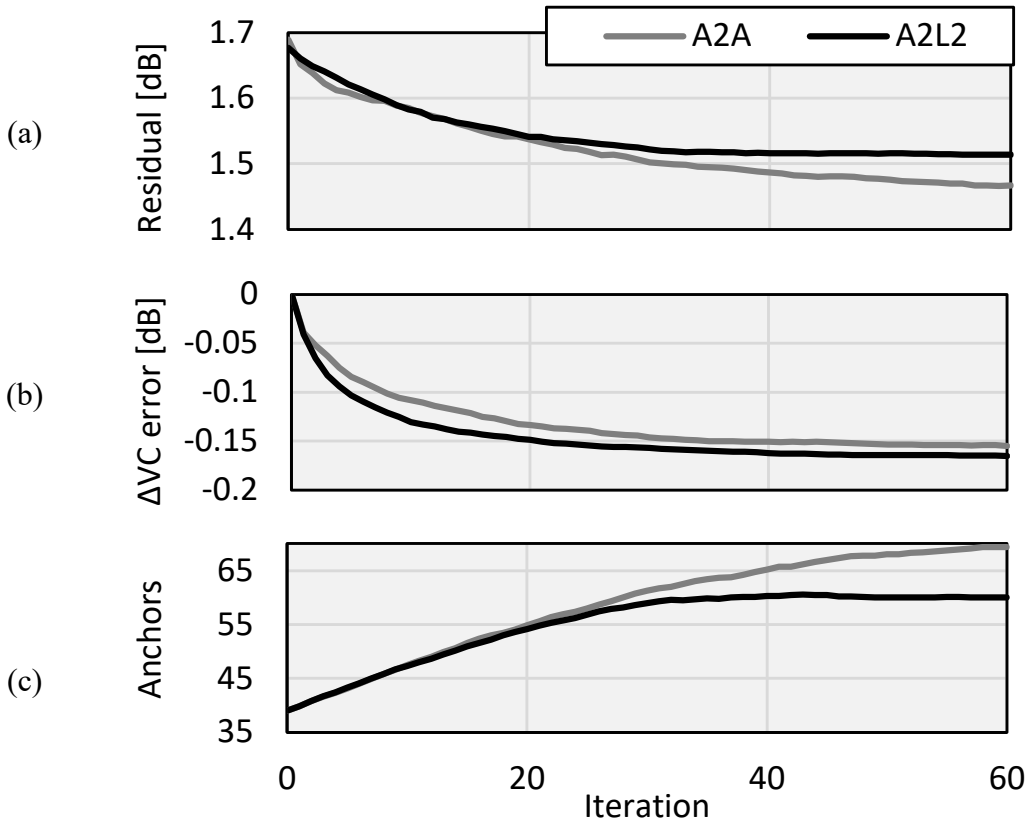


Figure 21: Performance of the ARS algorithm by iteration in terms of (a) VC error delta, (b) source and target residuals, (c) number of source/target anchors. In all figures, light gray represents A2A, black represents A2L2, and the x-axis represents the ARS iteration.

Figure 22 shows the proportion of ARS decisions by iteration for A2A (a) and A2L2 pairs (b). As the ARS algorithm terminates when no further split or remove decision can be made, we include the proportion of speaker pairs that had terminated by that iteration (the "done" label). The *split* decision was more heavily favored for A2A pairs than for the A2L2 speaker pairs; for A2A speakers, it was selected by more than half of all A2A pairs until iteration 48, whereas for A2L2 speakers, the *split* decision fell below

half at iteration 38. This difference is expected and is due to both time-alignment and accent differences between the datasets. Alignment and accent differences also affect the proportion of A2L2 speakers that terminate before the 60th iteration (83%) in contrast with the A2A speakers (55%).

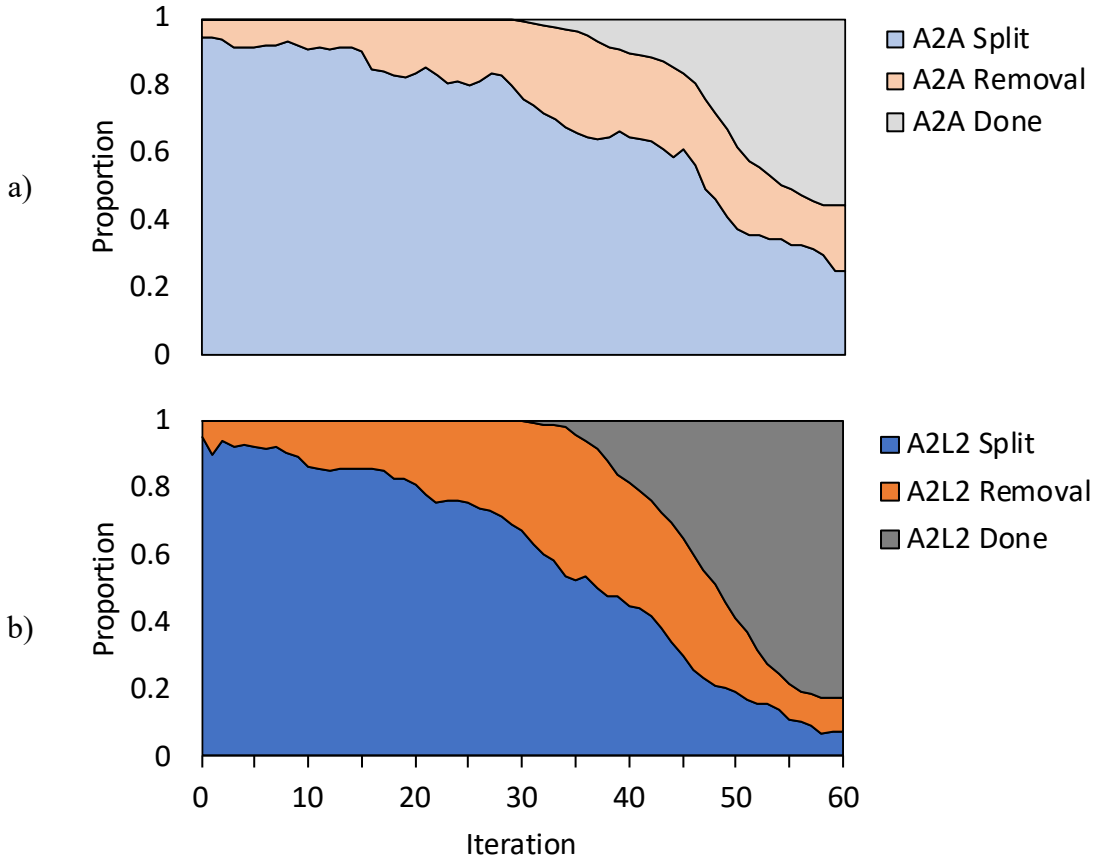


Figure 22: proportion of ARS decisions by iterations on all pairs of (a) A2A and (b) A2L2 speakers.
"Done" means for that pair of speakers, the ARS algorithm

In a final analysis, we investigated which phonemes in the *split* operation contributed to the greatest reduction in VC error. We computed the total amount the VC error decreased when phoneme *k* was split as:

$$\Delta e_k = \sum_{t=1}^{60} S_k^t (e_{t-1} - e_t), \tag{22}$$

where Δe_k is the change in VC error for phoneme k, t is the ARS iteration, S_k^t is an indicator variable that is 1 when phoneme k was selected for splitting on iteration t and 0 otherwise. For e_0, we used the VC error of the initial SABR models. Figure 23 shows the results of eq. (22) computed for all A2A and A2L2 speaker pairs, (a) shows A2L2 speakers, grouped by target L1, and (b) shows A2A speakers. The ten phonemes listed in Figure 23 (a) contain 50% of the reduction in VC error for A2L2 pairs. This is in contrast with the A2A speaker pairs, where the first ten phonemes in Figure 23 (b) represent a reduction of only 40% of the VC error and all phonemes are voiced.

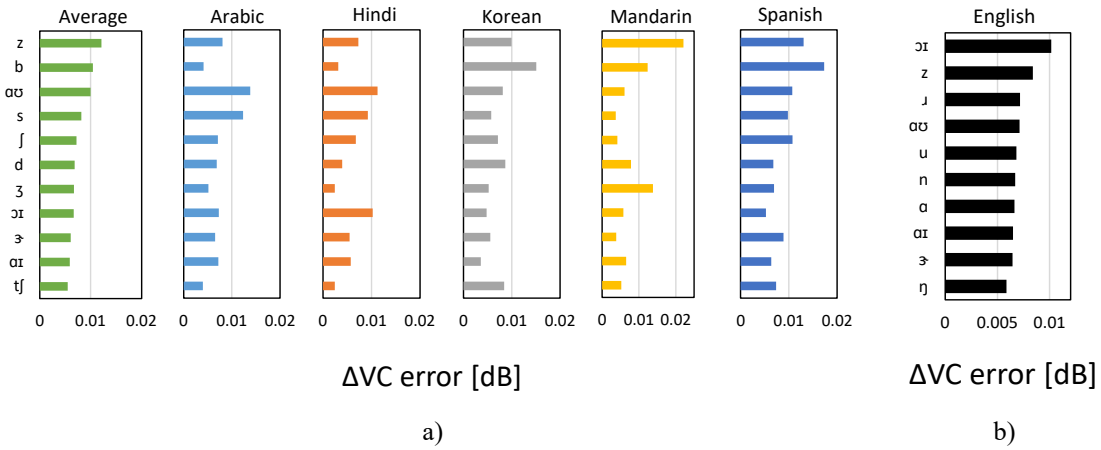


Figure 23: Reduction in VC error for top ten phonemes split by the ARS algorithm. In each figure, the x-axis represents the reduction in VC error from ARS operations on phonemes in that class. (a) A2L2 target speakers. (b) A2A target speakers.

There are a few notable differences between the A2A and A2L2 speaker pairs; first, on A2A pairs ARS favored voiced phonemes and vowels more than on A2L2 pairs. Second, on A2L2 pairs ARS often split phonemes with known voicing substitution errors

(e.g. /s/ and /z/, which have common voicing substitution errors in the L2-ARCTIC corpus [78]). Finally, on A2L2 pairs, the most-selected phonemes were those where the phoneme labels contained multiple states over the production of the phoneme (e.g. diphthongs and stops) or had the same articulation, but different voicing. Additionally, fewer *remove* operations were selected by A2A pairs because source and target anchors had fewer time alignment and pronunciation differences.

5.6.1.3. System comparison

In this section, we compare the performance of the five systems in terms of the objective measures: VC error, residual, and correlation between source and target dictionaries. A2A results are shown in Table 9, and A2L2 results are shown in Table 10.

For both A2A and A2L2 pairs, the three optimization methods had significantly lower VC error than the original SABR model (A2A and A2L2, $p \ll 0.001$, paired t-test). Notably, there was no significant difference in the VC error of the three proposed optimization methods and the baseline model (A2A, $p \geq 0.05$, A2L2, $p \geq 0.35$, paired t-test), a positive result given that the baseline model had dictionaries more than two orders of magnitude larger.

Table 9: A2A objective results summary for anchor optimization methods. Results are for all A2A speaker pairs. Numbers in parenthesis are the standard deviations for each value.

Method	Dictionary Size	VC Error	Residual	Correlation
Baseline	3531 (314)	2.46 (0.13)	0.92 (0.04)	0.78 (0.07)
SABR	39	2.59 (0.13)	1.68 (0.13)	0.81 (0.05)
IRT	39	2.42 (0.12)	1.05 (0.16)	0.74 (0.09)
ARS	69.8 (12.7)	2.46 (0.12)	1.46 (0.13)	0.70 (0.10)
ARS+IRT	69.8 (12.7)	2.39 (0.13)	1.00 (0.08)	0.71 (0.10)

Table 10: A2L2 objective results summary for anchor optimization methods. Results are for all A2L2 speaker pairs. Numbers in parenthesis are the standard deviations for each value.

Method	Dictionary Size	VC Error	Residual	Correlation
Baseline	4720 (367)	2.74 (0.14)	0.96 (0.04)	0.63 (0.06)
SABR	39	2.91 (0.12)	1.67 (0.11)	0.61 (0.09)
IRT	39	2.72 (0.12)	1.24 (0.08)	0.76 (0.07)
ARS	60.0 (8.5)	2.75 (0.12)	1.51 (0.11)	0.66 (0.08)
ARS+IRT	60.0 (8.5)	2.71 (0.11)	1.12 (0.10)	0.74 (0.07)

All systems showed significantly different performance in terms of residual magnitude. Of the proposed optimization methods, ARS+IRT had the lowest residual for both A2A and A2L2 pairs (A2A and A2L2, $p \ll 0.001$, paired t-test). This was for two reasons: IRT on its own significantly reduces the residuals as compared to the original SABR model, and when combined with the larger anchors sets selected by ARS, IRT could reduce the residuals further. Again, notably, ARS+IRT residuals differed from the baseline model by 0.08 dB (A2A) and 0.16 dB (A2L2), even though the ARS+IRT dictionary sizes were significantly smaller than that of the baseline.

In A2A speakers, both the baseline model and the original SABR model had significantly higher correlation than the three optimization methods ($p < 0.01$, paired t-test). This contrasts with the A2L2 speaker pairs, in which the baseline and original SABR model had the lowest correlations ($p < 0.01$, paired t-test). Additionally, in A2A speaker pairs, IRT had significantly higher correlations than ARS+IRT ($p < 0.01$, paired t-test); in A2L2 speaker pairs, ARS+IRT and IRT did not have significant differences in correlation ($p = 0.10$, paired t-test). For A2A speaker pairs, the baseline and original SABR models do not have to contend with pronunciation differences between speakers,

and the optimization algorithms optimize the VC error and residuals more than the A2L2 speaker pairs, which comes at the cost of lower correlations.

In an additional experiment, we examined how many anchors would ARS need to split to match the VC error and residual magnitude of IRT. For this comparison, we only used ARS to split each phoneme into multiple anchors, ranging from 2 to 32, on all A2L2 pairs. The results are shown in Figure 24. ARS has to split each phoneme into 4 clusters (or 156 anchors total) in order to achieve similar VC error and residual as IRT. ARS+IRT improves upon this, reaching lower VC error and residual magnitude than either method or better than the pure clustering condition with 61 anchors. Finally, ARS has to split each phoneme into 16 clusters (624 exemplars) in order to achieve a similar residual magnitude and VC error as the baseline system, which uses 4720 exemplars on average.

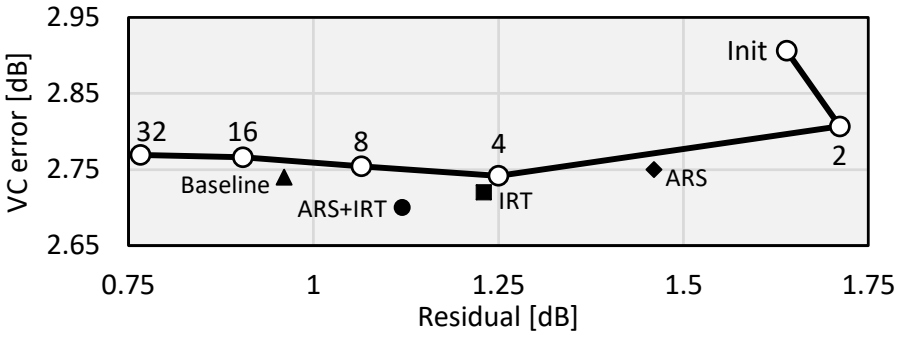


Figure 24: Comparison of the five systems in terms of VC error and residual error. The black line represents the result of splitting each phoneme into an increasingly larger number of clusters (2, 4, 8, 16 and 32). "Init" refers to the initial, SABR-trained anchor set residual and VC error.

5.6.2. *Experiment 2: Perceptual evaluation*

5.6.2.1. Mean Opinion Score

In a first perceptual experiment, we performed a Mean Opinion Score (MOS) test to measure the synthesis quality of the five VC systems. For this purpose, we recruited

participants ($n = 20$) on Amazon Mechanical Turk to rate the quality of an utterance on a 5-point scale (1 = "low quality"; 5 = "high quality"). We included both A2A an A2L2 speaker pairs to evaluate how the effects of accent change the synthesis quality of the methods. For each speaker pair, we asked participants to rate 25 utterances–5 utterances per synthesis method. Following [97], we included eight unmodified utterances from the corpora to verify that participants were listening carefully and not randomly guessing. Results are shown in Figure 25.

A2L2 ratings for the proposed optimization methods (IRT: 2.88; ARS: 2.66; ARS+IRT: 3.00) were approximately twice as high as those of the baseline system (1.39[7]; $p \ll 0.01$, single-tailed t-test), a remarkable result given that they include far fewer anchors in their dictionaries (~61 vs. 4720). This pattern continued in the A2A ratings, where the optimization methods (IRT: 3.48; ARS: 3.18; ARS+IRT: 3.26) were also significantly higher than the baseline (2.20; $p < 0.01$, single-tailed t-test). IRT and ARS+IRT significantly improved the initial SABR synthesis quality for both A2A and A2L2 speaker pairs ($p < 0.01$, single-tailed t-test). Additionally, the IRT and ARS+IRT scores were not significantly different in either synthesis condition (A2A: $p = 0.20$; A2L2: $p = 0.17$, single-tailed t-test). In contrast, the ARS algorithm alone did not significantly improve upon the original SABR MOS for either set of speaker pairs (A2A,

[7] This MOS rating is significantly lower than those reported by the authors of the baseline system. We believe this is due to the difficulty in time-aligning native to non-native utterances, which is critical for the baseline system. In contrast, SABR is less affected by these issues as it does not require time-alignment, and the two anchor optimization methods have built-in mechanisms to address misalignments.

$p = 0.17$; A2L2, $p = 0.13$, single-tailed t-test), suggesting that IRT's reduction of the residual significantly contributes to the quality of the synthesis.

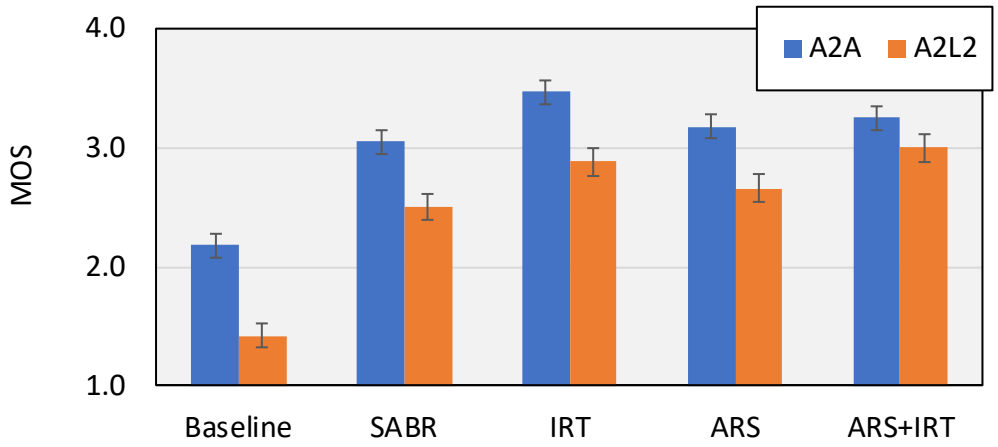


Figure 25: MOS scores of A2A and A2L2 speaker pairs from the baseline and proposed system. Error bars show standard deviation of the ratings.

The baseline method had a significantly larger difference in MOS between A2A and A2L2 versus the SABR and optimization methods ($p < 0.01$, all methods, single-tailed t-test). There was no significant difference in the change in MOS between SABR, IRT, or ARS methods, suggesting that both SABR and the proposed optimization methods are more robust against dealing with time-alignment and mispronunciation effects of native-to-nonnative voice conversion. While the ratings of ARS+IRT had the lowest difference, this is due to the A2A ARS+IRT MOS being lower relative to the A2A IRT MOS than the corresponding A2L2 ratings.

5.6.2.2. Accentedness test

We performed an accentedness test to evaluate how the residual methods affected the accentedness of the synthesis. We asked participants ($n = 21$) to rate the accentedness of a speaker on a 9-point Likert scale following [90] (1= "no foreign accent", 9= "very

strong foreign accent") on utterances A2L2 speaker pairs from: the baseline NMF method, the SABR method without the optimized anchors, the optimized SABR anchor models, and utterances from the L1 and L2 speakers. For each condition, participants rated 20 utterances for a total of 140 ratings. Results are shown in Figure 26.

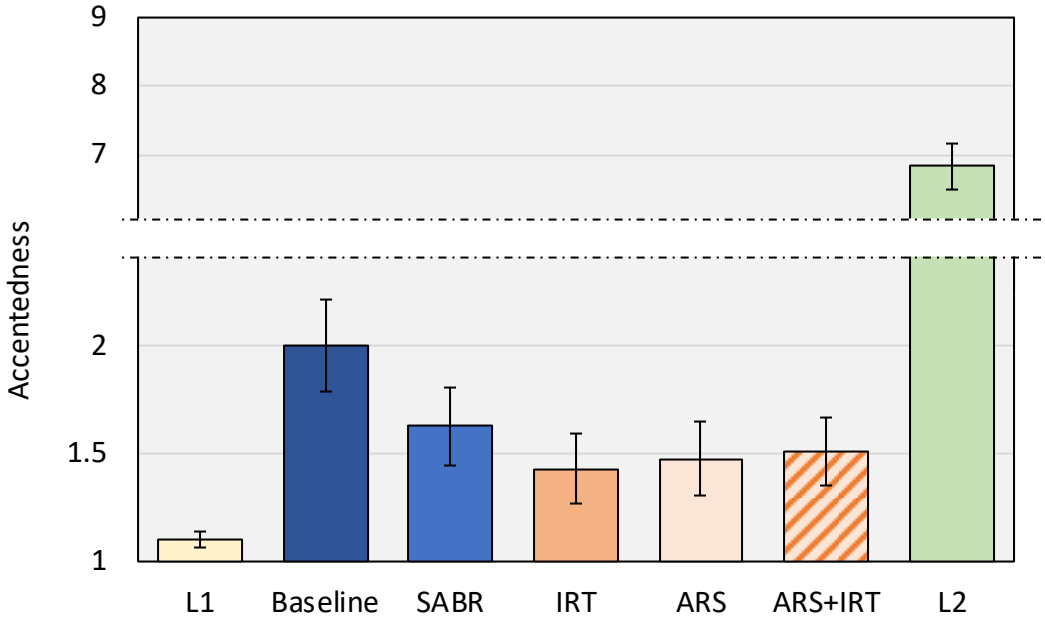


Figure 26: Accentedness ratings of baseline systems and optimized anchor sets.

Overall, participants found all of the VC methods to be significantly closer to a native accent than a nonnative accent. There was no significant difference in the ratings between the SABR and the three optimized anchor sets ($p \geq 0.14$, single-tailed t-test), but IRT and ARS+IRT were rated as having significantly lower accent than the baseline method ($p < 0.05$, single-tailed t-test). Overall, this experiment confirms that the optimized anchor sets do not sound more accented, even though they are optimized on data from the accented target speaker—in fact, they produce even closer ratings to native speakers than the baseline methods.

5.6.2.3. Speaker identity test

In the next perceptual experiment, we performed an XAB speaker identity test comparing synthesis from the baseline method and the three optimization methods (IRT, ARS, ARS+IRT). To ensure the perceptual test was tractable, we did not include the baseline SABR method, as in the MOS trials the SABR synthesis were rated as significantly lower quality than the optimized methods. We recruited ($n = 20$) participants from Amazon Mechanical Turk and presented them with three utterances: an utterance from one of our synthesis methods from the four A2L2 speaker pairs (X), and utterances from the source or target speaker (A, B). The order of A and B was counter-balanced. Following [58], utterances were played in reverse to mask the effects of accent, and allow participants to focus on the identity of the speaker. For each speaker pair, we asked participants to perform 32 evaluations—8 per synthesis method. As before, we included 5 evaluations where the reference utterance was an unmodified reference from the source speaker to identify and remove participants who evaluated the pairs randomly. Results of the XAB test are shown in Figure 27.

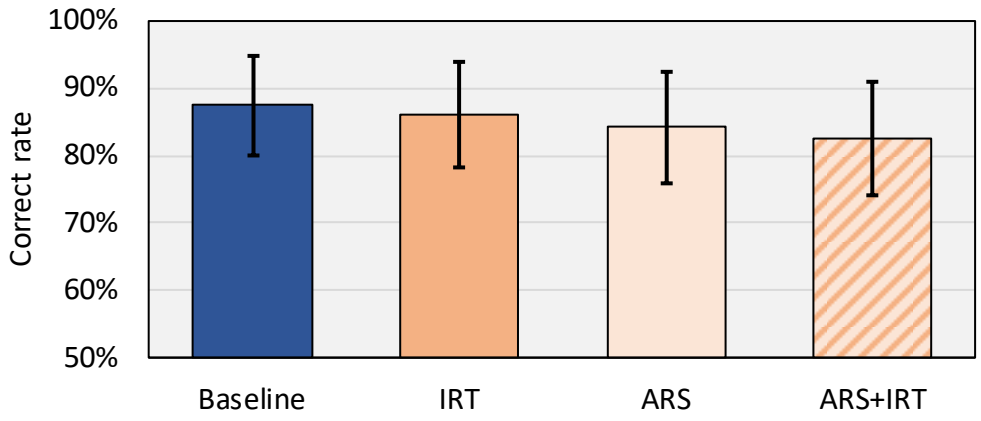


Figure 27: XAB speaker identity test ratings of the baseline VC system and optimized anchor sets.

There was no statistically significant difference between the baseline system and the three optimization methods ($p > 0.05$, two-tailed t-test), despite the fact that the optimization methods used two orders of magnitude fewer anchors than the baseline system. These ABX recognition rates are similar to those reported in related literature [1, 38].

5.6.2.4. Preference tests

To further distinguish differences in MOS ratings between the three optimization algorithms, we performed an additional AB preference test on A2L2 speaker pairs. Subjects ($n = 20$) were presented with two utterances, each from either ARS, IRT, or ARS+IRT, and were asked to determine which utterance was better in terms of acoustic quality. We did not include the baseline or base SABR anchor sets in the comparison because their MOS ratings were significantly lower than the other optimization methods. We used the four source-target speaker pairs listed in Table 10, presenting participants on

32 pairs of utterances from each of the three possible comparisons. Both utterances were from the same source-target speaker pair. As in the previous tests, we included 5 unmodified reference utterances to verify that participants were listening carefully and not randomly guessing. Results are shown in Figure 28. In all cases, listeners preferred the optimization methods that included IRT. IRT was preferred to ARS in 71% of cases ($p < 0.001$, single-tailed t-test), and ARS+IRT was preferred to ARS in 68% of cases ($p < 0.001$, single-tailed t-test). However, listeners did not show a preference for IRT over ARS+IRT ($p = 0.08$, single tailed t-test). This suggests that reducing the residual is more important to improving synthesis quality than the benefits of a larger anchor set.

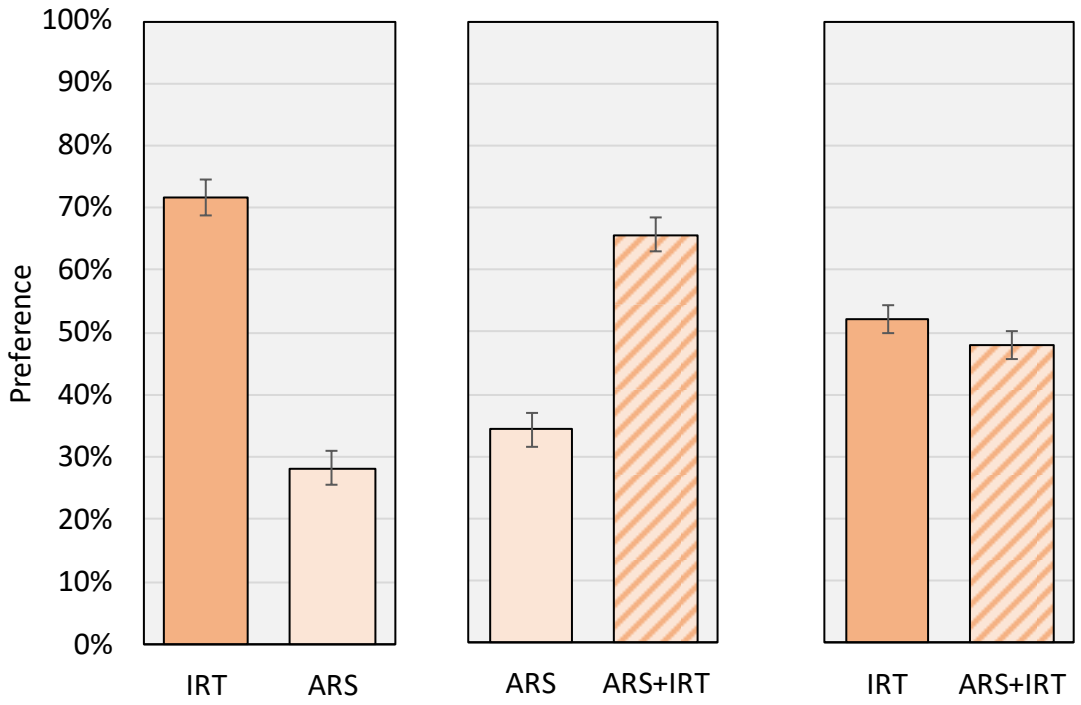


Figure 28: AB preference tests of the optimized methods.

5.7. Discussion

In this chapter, we presented two optimization algorithms for a low-resource exemplar-based voice conversions system (SABR) used to convert utterances between native and non-native speakers. The two optimization algorithms address two issues with the original version of SABR. First, having a compact exemplar set increases the magnitude of the residual error, which negatively affects synthesis quality. Second, one exemplar may not be enough to represent some phoneme classes, so selecting multiple exemplars may improve VC performance. The IRT algorithm optimizes anchor sets to reduce VC error and the residual error. The ARS algorithm either splits or removes anchors if that decision reduces VC error. This allows multiple anchors to represent a phoneme, or the anchor to be removed entirely.

In our experiments, we examined the effect of these algorithms on voice conversion in native-to-native (ARCTIC to ARCTIC, *A2A*) and native-to-nonnative (ARCTIC to L2-ARCTIC, *A2L2*) contexts to highlight the difficulties that time-alignment brings to exemplar-based voice conversion. We performed both objective and subjective experiments on our proposed algorithms as well as using an exemplar-based voice conversion baseline.

5.7.1. *Objective results*

Iterative retraining (IRT) reduced the VC error and residual in both the A2A and A2L2 speaker pairs. Time alignment and pronunciation differences between these datasets are visible in the difference between the $\alpha = 0$ and *Init* conditions in Figure 20; for A2A pairs, the $\alpha = 0$ case significantly reduces both the VC error *and* residual, even though in

this condition IRT only attempts to reduce the VC error (see eq. (18)). This indicates that the optimal direction for updating anchors to account for VC error is also the optimal direction to reduce the residual, something which does not occur with A2L2 speaker pairs. As shown in Figure 19, more than 90% of the reduction in VC error occurred in the first iteration of IRT, suggesting that only a few updates are necessary to significantly reduce VC error; only the residual benefits from continued iterations, with a modest tradeoff in increasing VC error.

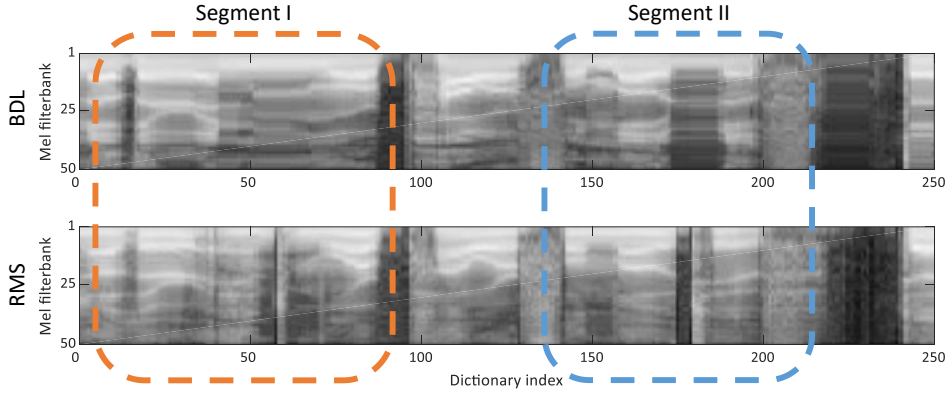
The anchor removal and splitting algorithm (ARS) focused primarily on the *split* operation. Typically, two types of phonemes were selected by the algorithm to be split for A2L2 pairs: those that would benefit from multiple anchors (e.g., stops, diphthongs, and affricates) and those that are known to be difficult for the L2 learner. By analyzing the *split* operation in ARS, it is possible to improve the anchors selected by SABR and other exemplar-based methods. First, non-continuant phonemes (e.g., stops, affricate, and diphthongs) were often selected for splitting. This suggests that allowing for multiple acoustic states (similar to senone states of ASR systems [98, 99]) would improve synthesis quality. Additionally, we found that in both A2A and A2L2 speaker pairs the split operation included many fricatives that *differed* in voicing between the source and target speakers (e.g. the source speaker formed /z/ while the target speaker formed /s/). This suggests that including pitch information during training to ensure that the *voicing* matches between source and target data training data may also improve the performance of SABR. For other exemplar-based methods which require time-aligned data, discarding parallel

source and target data that does not share the same voicing may also improve voice conversion performance.

Both methods reduced the overall VC error by a similar magnitude, but IRT significantly reduced the residual when compared to ARS. The joint ARS+IRT method had even lower VC error and residuals than either method individually, and comparable to that of the baseline method. One advantage to IRT is its flexibility; given any parallel source and target data, the algorithm will reduce the residuals of the dictionaries while ensuring they remain conditioned for VC. In contrast, ARS was expensive to compute and favored splitting phonemes as opposed to removing a phoneme. Additionally, a significantly larger dictionary (and correspondingly, more split operations) would be required for ARS on its own to match IRT's residual magnitude and VC error.

5.7.2. *Subjective results*

(a)



(b)

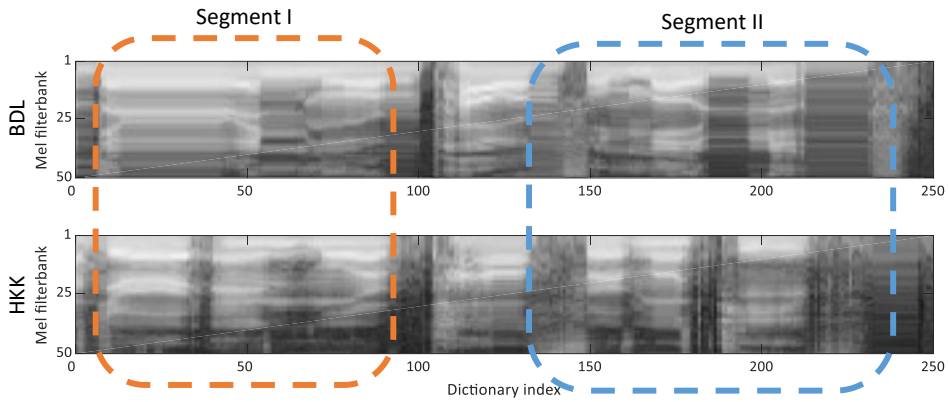


Figure 29: Illustration of time-alignment issues on the baseline system.(a) A2A speaker pair (BDL-RMS). (b) A2L2 speaker pair (BDL-HKK). Segment I (the word "power") shows significant time alignment and pronunciation differences between A2A and A2L2 speaker pair

A possible explanation for the low MOS ratings of the baseline system is that it relied heavily on accurate time alignment between source and target dictionaries. If time-alignment is not ideal or the speakers differ in pronunciation, the synthesis quality of the baseline system would be adversely affected. The time-alignment issues of the nonnative speakers are illustrated in Figure 29, which shows the time-aligned dictionaries of the baseline; (a) shows the alignment of BDL (m) and RMS (m), two A2A speaker pairs, and (b) shows the alignment of BDL and HKK (m, Korean), two A2L2 speaker pairs. Mispronunciation and alignment differences in the A2L2 dictionary is noticeable in segment I (the word "power") and time alignment differences are visible in segment II (the word "motive"). The A2A alignment is noticeably better, with fewer long durations of the same frames duplicated by dynamic time warping. If any of these duplicated frames are selected during the computation of the activation matrix, every single frame will be selected; when the target spectrum is then computed as the average of all the corresponding frames from the target speaker's dictionary. Similarly, mispronunciations

in the target speaker's dictionary will be mismatched to the phonetic content of the target speaker. Both of these issues result in the synthesized spectrum lacking spectral detail and lowering the overall synthesis quality.

In contrast, SABR and the proposed optimization algorithms are less susceptible to the time-alignment issues that affected the baseline system. As a result, the MOS ratings were significantly higher for the proposed systems than for the baseline system. The combination of using a fewer dictionary entries with the residual warping method (Chapter 4) allowed SABR synthesis to avoid the issues to which the time-aligned dictionaries of the baseline method was susceptible. Additionally, the proposed optimization methods were more robust in native to nonnative voice conversion contexts, as the differences in MOS ratings of the A2A and A2L2 pairs were lower than that of the baseline. Importantly, though the baseline method had nearly two orders of magnitude larger dictionaries than SABR or the proposed optimization algorithms, there was no statistically significant difference in listeners' ability to identify the synthesis as coming from the target speaker. The anchor optimization algorithms had the secondary effect of *lowering* the accentedness of the unoptimized SABR models and the baseline NMF method, further showing the optimization did not negatively affect the ability for the models to perform accent conversion.

Prior literature has noted that in higher correlations between anchor sets is associated with higher synthesis quality [35]. However, the A2A SABR anchor set had the highest correlations of any of the models, but did not have the highest synthesis quality. Instead, a combination of high correlation and low residual are indicators of synthesis

quality for SABR methods. As listeners did not show a preference for the IRT or ARS+IRT synthesis over ARS in the AB preference tests, this shows that additional anchors do not contribute as much synthesis quality as do reductions in the residual magnitude. This also suggests that focusing on just one of these components is not enough to guarantee both synthesis quality and reaching target speaker identity.

5.8. Conclusion

In this chapter, we proposed two methods for optimizing the SABR anchor sets (ARS and IRT) and compared each method, and a combination of the two, against the unoptimized SABR anchor sets and another baseline exemplar-based voice conversion algorithm. We examined these methods in both native-to-native and native-to-nonnative voice conversion contexts. In our objective results, we found that the proposed optimization methods resulted in objective performance similar to that of a state-of-the-art baseline voice conversion method while significantly improving upon the initial SABR anchor sets. In perceptual studies, the algorithms significantly improved the synthesis quality of both native and nonnative speakers. The proposed methods were more robust to the effects of native to nonnative conversion than the baseline method. Most notably, the proposed optimization methods were able to outperform the baseline method even though they had two orders of magnitude fewer exemplars.

The experimental results of the two proposed methods proposed here suggest ways to continue to improve the synthesis quality of exemplar-based VC algorithms. First, the results from ARS suggest that matching the voicing of source and target exemplars is important to the synthesis quality of exemplar-based VC systems. For SABR,

incorporating voicing into anchor selection would improve baseline synthesis quality; for systems using time-aligned data (such as the baseline method and the optimization methods) discarding source and target frames with different voicing would alleviate these issues. Second, increasing the number of anchors in a principled way (e.g., using PPG senones to estimate multiple exemplars per phoneme [55] or incorporating manner of articulation constraints as in [46]) provides another route for improving synthesis quality.

In the following chapter, we address the final aim of this dissertation, which is adding temporal constraints to the SABR objective function.

6. ADDING TEMPORAL CONSTRAINTS TO SABR VIA THE FUSED LASSO

6.1. Overview

In this chapter, we present a technique for including temporal constraints in the SABR objective function. This technique is based on the *Fused Lasso*, a modification of the original Lasso that allows for a structured sparsity penalty. We design and test a structured sparsity penalty that forces the Lasso to consider temporally adjacent frames when computing the SABR weights. This chapter addresses the fourth aim of the dissertation, which is to add a temporal constraint to the SABR objective function. This chapter will be published at a future venue to be determined.

6.2. Introduction

In the previous chapters, the SABR objective function we use to get source speaker weights and to perform VC is frame-independent: each spectral frame is computed independent of the prior frame and the next frame. However, because speech is temporal in nature, treating these frames as independent may be suboptimal. In this chapter, we present a modification to the frame-independent SABR objective function, to enforce temporal smoothness constraints with two goals in mind. First, we seek to minimize the frame-to-frame weight differences by considering adjacent speech frames during the encoding process. Second, we wish to reduce the selection of spurious weights that do not contribute to the overall quality of the synthesized utterance—akin to removing noise in the sparse codes. Our proposed modification, known as the "Fused Lasso" (FL) [100], imposes penalties on arbitrary structures within the sparse codes. We then design a structured penalty using the FL that penalizes the magnitude difference between adjacent

weight frames, forcing the algorithm to choose fewer weights and reduce the variance of those weights.

In two experiments, we compare the performance of the proposed FL method with the non-fused Lasso method against two techniques for imposing temporal constraints: Nonnegative Spectrogram Deconvolution (NNSD) [8], and Modified Restricted Temporal Decomposition (MRTD) [101]. Our results show that the Fused Lasso reduces the number of basis vectors selected in an analysis window and increases the smoothness of weights when compared to the baseline methods and the unmodified Lasso. In VC tasks, the FL method had the lowest VC error and selected the fewest number of basis vectors in an analysis window when compared to NNSD and MRTD. These results are encouraging, showing the frame-to-frame penalty leads to a reduction in the number of basis vectors without loss in synthesis quality.

The rest of this chapter is organized as follows. First, we review related VC methods that incorporate temporal information and contrast them with our proposed method. Then, we explain the FL and the associated Generalized Lasso solver, how we use it to enforce temporal constraints, and how to use it in VC. Finally, we examine the different parameters of the FL method and perform objective and subjective experiments on the ARCTIC corpus. We end with a discussion of the results.

6.3. Related work

Several methods exist to incorporate *temporal* information in spectral representations. A common way is to include *delta features*, namely features representing the first and second derivatives of the training data, concatenated to the feature vectors.

To improve upon the trajectories of GMM systems, Toda *et al.* [87] used these features, combined with a Maximum Likelihood Parameter Generation (MLPG) algorithm, to increase the variance of the trajectories. The authors found that the inclusion of this method significantly increased the spectral variance of the synthesis, as well as the synthesis quality. However, more training data was required to build the MLPG model—on the order of several hundred utterances.

In the sparse decomposition domain, several methods have been proposed to incorporate temporal information. Wu *et al*. [8] proposed to use NMF in combination with a temporal reconstruction method to perform "temporal deconvolution" and include temporal information in the coding process. Exemplars were windowed and the matrix factorization was performed on the windowed input data. At runtime, the windowed exemplars were summed in a "deconvolution" step. Through perceptual studies, they found that the method significantly improved the quality of the synthesis. Virtanen [102] included a penalty parameter that enforced temporal sparseness in the NMF decomposition. They found that both the temporal sparseness and frame sparseness had to be treated separately to achieve optimal performance. Including this temporal constraint allowed the source separation algorithm to perform better than other NMF systems on a music classification task.

Similar to sparse coding methods, Temporal Decomposition (TD) [103] is a form of speech coding that represents a signal Y as a linear combination of basis vectors A and "interpolation functions" Φ. TD chooses A and Φ that minimize the residual of $\|Y - A\Phi\|_2$. In principle, the algorithm selects temporal inflection points A in the

spectrogram Y and linearly interpolates between them over Φ, effectively giving a temporally smoothed representation of the signal. As a result, Φ encodes the duration of stationary sounds (e.g. phonemes) whereas A encodes the corresponding acoustics. Extensions to the TD method include non-negativity [104] and monotonicity constraints [101]. Nguyen and Akagi [42] incorporated TD in a VC method by defining phoneme boundaries as event functions and converting the basis vectors A using Gaussian Mixture Models (GMM). The authors found that including the TD marginally improved acoustic quality and speaker identification rates over a baseline GMM system.

Tibshirani *et al.* [100] proposed the Fused Lasso as a modification to the well-known Lasso sparse coding technique to include structural information about the dictionaries. This was useful in instances where sparse codes of noisy data (such as in images) were desired, but there was *a priori* knowledge about the structure of the data. In [105], the authors generalized the penalty in the Fused Lasso to the "Generalized Lasso" and demonstrated different techniques for solving the Fused Lasso for different fused penalty structures. In this chapter, we use a version of the Generalized Lasso that works on smoothing each weight channel individually.

The proposed method differs from other exemplar-based VC methods that include temporal information by explicitly penalizing frame-by-frame encoding changes in the objective function [8, 50, 101, 103]. With this objective, the generated sparse codes are significantly smoother, and instances where spurious changes can occur (e.g. in plosives or fricatives) are reduced.

6.4. Methods

6.4.1. *The Fused Lasso for SABR temporal constraints*

To reiterate, the SABR method from Chapter 3 represents an utterance as a sparse weighted sum of speaker-dependent phonemic anchors in a frame-independent manner [9]. That is, each frame of the input utterance is considered independently from every other frame. For a given source utterance with N acoustic features (e.g., MFCCs) and T frames $X \in \mathcal{R}^{N \times T}$ and a source anchor set $A_S \in \mathcal{R}^{N \times K}$ of K phonemes, SABR uses the Lasso to estimate the "weights" $W \in \mathcal{R}^{K \times T}$:

$$W = \text{argmin} \left|\left|X - A_s W\right|\right|^2 + \lambda_1 \|W\|_1 \; s.t. \; 0 \leq W \leq 1. \tag{23}$$

Eq. (23) does not include any temporal constraints; this lack of constraints allows for frame-to-frame fluctuations in the weights, affecting the interpretability of the weights and potentially introducing distortions in synthesis. The Fused Lasso allows us to add these constraints to the objective function by including a term that penalizes *temporal changes in the weights*:

$$\min_{\boldsymbol{W'}} \|X - A_S W\|_2^2 + \lambda_1 \|W\|_1 + \lambda_2 \sum_{i=2}^{T} (w_i - w_{i-1}). \tag{24}$$

The third term penalizes differences in adjacently indexed weights, but the adjacency penalty can be applied to any pair of weights. To enforce temporal smoothness, we cannot apply eq. (24) to eq. (23) without including a temporal context in the data X and dictionary A_S. To do this, we modify the SABR components to compute the weights over some k-width window centered on t, $[t - k, t + k]$:

$$\boldsymbol{X'} = [X_{t-k}^T \dots X_{t+k}^T]^T \tag{25}$$

$$\boldsymbol{W'} = [W_{t-k}^T \dots W_{t+k}^T]^T \tag{26}$$

$$\boldsymbol{A_s'} = \begin{bmatrix} A_s & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & A_s \end{bmatrix} \tag{27}$$

The spectrum X and weights W are stacked over the temporal window, such that $\boldsymbol{X'} \in \mathcal{R}^{(2k+1)N \times T}$ and $\boldsymbol{W'} \in \mathcal{R}^{(2k+1)K \times T}$; the anchors A_S are diagonally replicated $2k+1$ times to match the number of weights, and $\boldsymbol{A_s'} \in \mathcal{R}^{(2k+1)N \text{ x } (2k+1)K}$. This modification allows the Fused Lasso to penalize temporal differences in the weights:

$$\min_{\boldsymbol{W'}} \|\boldsymbol{X'} - \boldsymbol{A_S'W'}\|_2^2 + \lambda_1 \|\boldsymbol{W'}\|_1 + \lambda_2 \sum_{f=t-k+1}^{t+k} \left\|W_f - W_{f-1}\right\|_1 \tag{28}$$

6.4.2. *Solving via the Generalized Lasso*

Tibshirani *et al.* [105] discussed methods for solving structured penalties such as those in eq. (28) via the "Generalized Lasso". They restructure eq. (28) to have the following penalty:

$$\min_{\boldsymbol{W'}} \|\boldsymbol{X'} - \boldsymbol{A_S'W'}\|_2^2 + \lambda_2 \|D\boldsymbol{W'}\|_1 \tag{29}$$

The difference between eqs. (28) and (29) is the missing first sparsity term; in [105], the authors call eq. (29) the *sparse fused lasso*, as D enforces sparsity only on a structure of W and not the magnitude. Setting $\lambda_1 = 0$ in eq. (28), it is equivalent to eq. (29) when D has the following structure:

$$D = \begin{bmatrix} -\boldsymbol{I} & \boldsymbol{I} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & -\boldsymbol{I} & \boldsymbol{I} & \dots & \mathbf{0} \\ \vdots & & \ddots & & \vdots \\ \mathbf{0} & \dots & -\boldsymbol{I} & \boldsymbol{I} & \mathbf{0} \\ \mathbf{0} & \dots & \mathbf{0} & -\boldsymbol{I} & \boldsymbol{I} \end{bmatrix} \tag{30}$$

where $\mathbf{0} = 0^{K \times K}$ and $\boldsymbol{I} = I^{K \times K}$.

With a final row, we can also penalize the magnitude of W_T:

$$D_{lasso} = [\mathbf{0} \dots \mathbf{0}\ \boldsymbol{I}\ \mathbf{0} \dots \mathbf{0}] \tag{31}$$

When eqs. (30) and (31) are combined into a single structure, they penalize the temporal structure of the weights as well as the weights at time t:

$$D_{fused} = \begin{bmatrix} \beta D \\ D_{lasso} \end{bmatrix} \tag{32}$$

By including a scaling term β, similar to [102], we can treat the penalties D and D_{lasso} separately.

When D_{fused} is invertible, the following substitution $\Theta = D_{fused}\boldsymbol{W'}$ transforms eq. (29) into a Lasso equation on Θ:

$$\min_{\boldsymbol{W'}} \left\| \boldsymbol{X'} - \boldsymbol{A_S'} D_{fused}^{-1} \Theta \right\|_2^2 + \lambda_2 \|\Theta\|_1 \ s.t. \Theta \geq 0 \tag{33}$$

Because of the way we structured D_{fused}, it has at most a single index of 1 and -1 per row, is full rank, square, and invertible. Using $\boldsymbol{A_S'} D_{fused}^{-1}$ as our basis vectors, we can solve eq. (33) using a Lasso solver, and Θ includes the temporally-constrained weights at frame t, W_t.

To extract W_t from Θ, we leverage the structure of $\boldsymbol{W'}$ and D_{fused} and extract it directly from a partitioning of Θ:

$$\Theta = \begin{bmatrix} \beta D \\ D_{lasso} \end{bmatrix} \boldsymbol{W'} \tag{34}$$

$$\Theta = \begin{bmatrix} \beta D \boldsymbol{W'} \\ \mathbf{0} \dots \mathbf{0}\ W_t\ \mathbf{0} \dots \mathbf{0} \end{bmatrix}. \tag{35}$$

This extracted W_t contains the SABR weights, centered on frame t, smoothed with the Fused Lasso temporal constrains. These weights can be used as the source weights in the SABR VC algorithm presented in Chapter 3.

6.5. Experiments

6.5.1. *Experiment Design*

6.5.1.1. Corpus

To validate the effectiveness of the approach, we performed a series of experiments using the ARCTIC speech corpus [81]. We used STRAIGHT [39] with default settings (1ms frame steps, 80 ms window sizes) to extract aperiodicity, fundamental frequency and spectral envelope, then computed a 24-dimension MFCC vector (25 filterbanks, 24 coefficients after ignoring $MFCC_0$ (energy), 8 KHz cutoff) from the spectral envelope. We assigned each acoustic frame a phonetic label based on the ARCTIC transcription. In these experiments, we built SABR models by selecting 25 utterances from the ARCTIC "A" set that maximized the entropy of the phoneme labels. For the objective experiments, we used 200 utterances from the ARCTIC "B" set as our test data.

In a set of preliminary experiments, we determined optimal λ_2 and β (penalties in eqs. (32) and (33)) using Pattern Search on different window sizes. Empirically, we found that λ_2 was roughly proportional to $0.09k$, where k was the window size of eq. (25), and

$\beta \cong 10/\lambda_2$. We used Least Angle Regression (LARS) [80, 89] to solve eq. (33), with the constraint of $0 \leq |W_t|_1 \leq 1$.

6.5.1.2. Comparison methods

We compared the proposed Fused Lasso method (SABR-FL) against the original SABR method and two baseline methods that incorporate temporal information into the VC objective, either explicitly (e.g. through the design of the dictionary) or implicitly (e.g. in the formulation of the objective function). The four methods we compared were:

- **SABR-FL:** the proposed Fused Lasso SABR method. Weights were computed according to eq. (33), the source weights W_t extracted from Θ, and target spectra computed using eq. (4).
- **SABR**: the original SABR method. Weights were computed using eq. (3) and target spectra are computed using eq. (4).
- **Nonnegative Spectrogram Deconvolution (NNSD)**: This method [8] *explicitly* includes temporal information by including previous and future frames in each dictionary entry. Following this method, we built a dictionary of 25 ms (5ms steps) and performed decomposition using the centered window. For comparison, we used the same number of atoms as SABR anchors: one per phoneme; the entries correspond to the SABR centroids, but including the 25ms window. We synthesized the target spectra using the windowed anchors, following the NNSD method. Weights were computed using Least Angle Regression on the windowed data, using the windowed dictionary.

- **Modified Restricted Temporal Decomposition (MRTD)**: MRTD [101] *implicitly* enforces temporal constraints by calculating the points where the spectrum has the highest-magnitude changes and then interpolating between these points. MRTD follows the TD algorithm, except with the constraints on the interpolation functions at each timeframe t of $|\Phi_t|_1 = 1$, $|\Phi_t|_0 = 2$, and monotonicity constraints on Φ. VC was performed by using SABR decomposition and conversion (eqs. (3) and (4)) on the learned basis vectors, similar to the training method presented in [42].

6.5.1.3. Measures of temporal smoothness

Given that our proposed SABR-FL method provides temporal constraints, we used the following two metrics to measure the smoothness of the weights:

Number of anchors: The total number of nonzero weights selected in a window m, centered on time t:

$$n(W; t, m) = \left\| \sum_{f=t-m}^{t+m} W_f \right\|_0 \tag{36}$$

Weight smoothness: the sum of the standard deviation of the frame-by-frame deltas of each weight channel:

$$s(W) = tr(\sigma_{\Delta W}). \tag{37}$$

The rationale behind these measures is as follows: the number of selected anchors n will decrease if spurious, short-duration segments of weights are removed, whereas s will decrease if the frame-by-frame weight deltas is lowered.

6.5.2. *Objective experiments*

In an objective experiment, we examined the smoothing effects of the proposed SABR-FL method and the two baseline methods across all 12 ARCTIC speaker pairs by computing the two smoothness measures (eqs. (36), (37)); we also computed the VC error for each method. While we examined a variety of smoothing windows from 10ms to 40ms, the SABR-FL VC error was minimized at 20ms ($k = 10$ in eqs. (25)-(27)); this error was slightly lower than the SABR baseline (SABR-FL: 2.52 dB; SABR: 2.55 dB). For the number of anchors measure eq. (36), we used $n = 40ms$ windows. Results are shown in Table 11.

Table 11: Summary of objective measures for the Fused Lasso and baselines. MCD is a measure of the VC error between time-aligned source and target utterances; n and s are those in eqs. (36) and (37), respectively.

	SABR-FL	MRTD	NNSD	SABR
MCD (dB)	**2.52**	2.55	2.55	2.52
n	**6.06**	8.21	8.07	8.21
s	0.21	**0.20**	0.25	0.56

The proposed method reduced the VC error by 0.03 dB, but also significantly reduced the number of anchors and increased the frame-to-frame smoothness of the weights. On average, SABR-FL selected 6.06 basis vectors, 26% fewer than the original SABR method (8.21) and the two baseline methods (MRTD: 8.21; NNSD: 8.07). SABR-FL also had frame-to-frame smoothness similar to that of the MRTD and NNSD methods, having 62% lower frame-to-frame variance in the weights as compared to the SABR baseline.

6.5.3. *Subjective experiments*

To determine the perceptual effect of the smoothed representation, we performed two AB preference tests using $n = 31$ participants from Amazon Mechanical Turk [97, 106]. In this test, we asked participants to determine which utterance they preferred, comparing the proposed SABR-FL method against the original SABR and the two baselines. One utterance was from SABR-FL and the other utterance was from one of the other three methods. Pitch was converted using log-mean pitch scaling [8]. Utterances were synthesized using STRAIGHT [39].

Results are shown in Figure 30, along with the 95% confidence interval computed from a t-Test. FL was preferred over both MRTD (68.8% ± 4.9%; $p \ll 0.01$, single-tailed t-test) and NNSD (57.0% ± 6.0%; $p < 0.05$, single-tailed t-test); however, there was no significant difference between the SABR and FL methods (46.5% ± 5.4%; $p = 0.21$, single-tailed t-test).

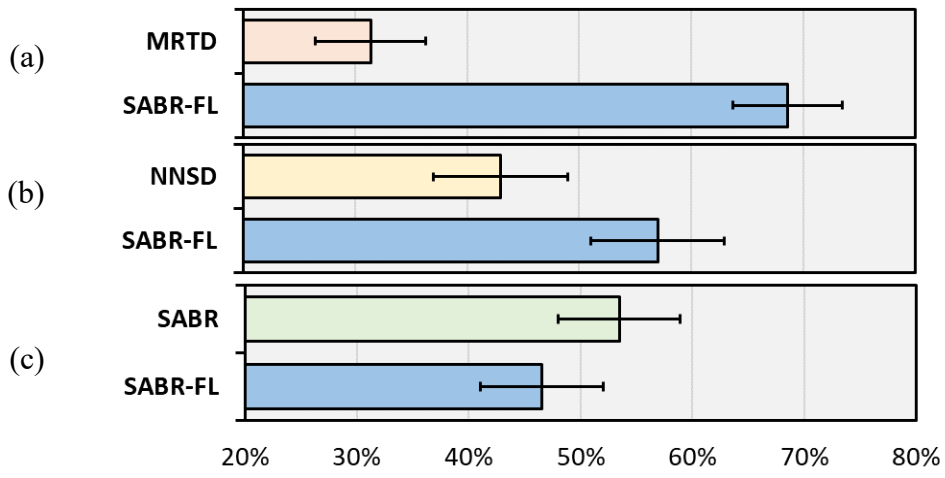


Figure 30: Preference comparison for the proposed Fused Lasso method. All values are shown with T-Test 95% confidence intervals.

6.6. Discussion

Our experimental results show that the proposed Fused Lasso method can significantly reduce the number of basis vectors selected in an analysis window with modest decreases in VC error and increases in synthesis quality compared to other methods that introduce temporal constraints. Though perceptual experiments show little difference between the baseline SABR and the proposed Fused Lasso method, the smoothness induced by the Fused Lasso is substantial—26% fewer anchors are selected in a 40 ms analysis window, and the frame-to-frame variance of the weights is reduced 62%. While both MRTD and NNSD reduced the frame-to-frame variance, they still selected just as many basis vectors as the unsmoothed SABR method. Only the proposed SABR-FL reduced both measures and did not degrade the audio quality when compared to the original SABR method. This suggests that the inclusion of these temporal constraints alone may not be enough to significantly improve the quality of VC utterances; however, the fewer selected anchors and smoother frame-to-frame weights may present future avenues for improving VC quality.

To illustrate the effects of SABR-FL on the weights, a segment of audio (the word "author", spoken by ARCTIC speaker *BDL*) and the corresponding weights for the original SABR are shown in Figure 31. When compared with the Lasso weights in Figure 31(b), the Fused Lasso weights in Figure 31(c) have smoother trajectories in the vowel segments. There are also fewer anchors being selected to represent /TH/ and those weights are smoother than the baseline method.

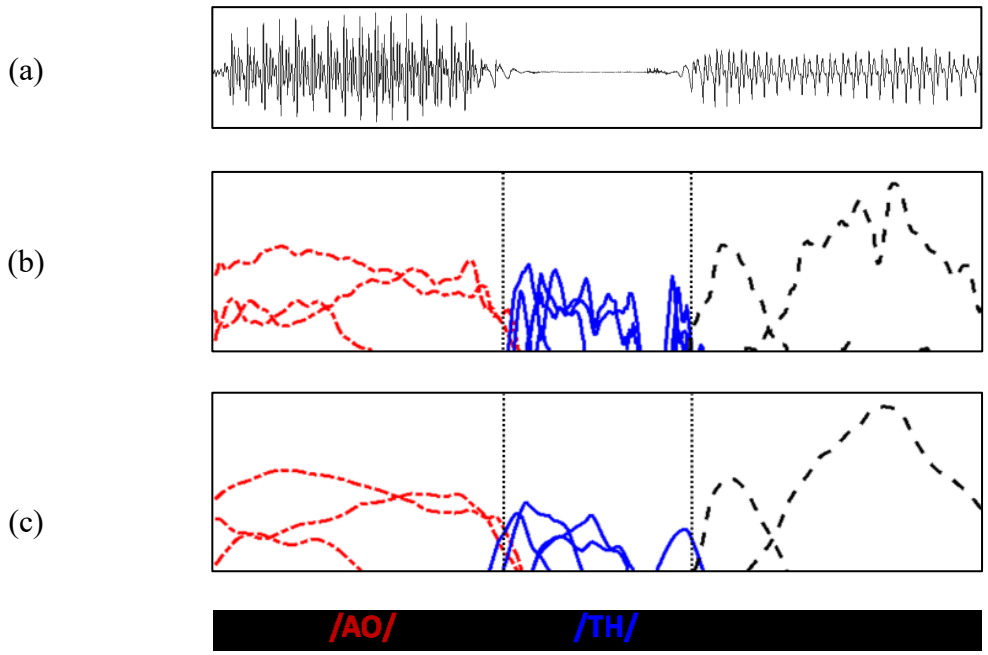


Figure 31: Effect of the Fused Lasso on the sparse representation.
(a) The waveform of the word "author", as spoken by BDL. (b) Relevant weights computed by the original SABR method for the selected audio segment. Dashed red lines represent /AO/, solid blue lines are /TH/, and dashed black lines are /ER/. (c) The corresponding weights from the proposed SABR-FL.

6.7. Conclusion

In this chapter, we derived a method from the Fused Lasso to enforce temporal smoothness in a sparse coding method for VC. We then experimentally tested this method with a frame-to-frame weight penalty function. Our results show that by penalizing weight differences, the number of weights selected as well as the frame-by-frame changes can be significantly reduced. Importantly, in perceptual experiments, we found that the increased smoothness did not degrade the acoustic quality compared with the original SABR method. Some of the original applications for the Fused Lasso included denoising; as Figure 31 shows, SABR-FL operates in a similar manner here, removing weights that can be better attributed to "noise" for the purposes of VC. This change in the sparse encoding

occurred without significant change in subjective or objective measures in VC quality. The Fused Lasso objective potentially allows for more complex temporal penalties and structures, which could operate as a direction for further improvement. We discuss these potential improvements and changes in Chapter 8.

7. BUILDING GOLDEN SPEAKERS WITH SABR*

7.1. Overview

In this chapter, we present a case study on using SABR for a computer aided pronunciation training (CAPT) tool called "Golden Speaker Builder", an accent training tool for nonnative speakers of English. The Golden Speaker Builder uses SABR to synthesize speech of a nonnative learner's voice, but with a native accent, for use in self-imitation for pronunciation training. We discuss the signal processing backend used to build SABR models and use perceptual studies to evaluate the synthesis used in the tool. This chapter is a modified version of [107], submitted to Speech Communication in 2019.

7.2. Introduction

Many second language (L2) speakers of English have difficulty acquiring native pronunciation, despite being fluent in English. Pronunciation training has been shown to help nonnative speakers acquire native pronunciation [108]. Several studies have suggested that learners who practice with voices most similar to their own is more effective in pronunciation training [109-111], the rationale being that listeners would be able to remove voice-related differences and allow learners to focused on accent-specific differences. Probst *et al.* [111] proposed that so-called "Golden Speakers" would be the ideal voice for learners to practice with. Their research indicated that foreign language learners imitating a voice similar to their own would more likely acquire native

* Repreinted with permission from "Golden speaker builder–An interactive tool for pronunciation training" by S. Ding, C. Liberatore, S. Sonsaat, I. Lučić, A. Silpachai; G. Zhao; E. Chukharev-Hudilainen, Evgeny; J. Levis, R. Gutierrez-Osuna, 2019. *Speech Communication*, Vol. 115, p. 51-66, Copyright 2019 by *International Speech Communication Association*.

pronunciation. However, finding optimal "golden speaker" teachers for each individual learner is onerous and infeasible in practice.

Accent conversion algorithms which synthesize the voice of a nonnative speaker, but with a native accent, offer an ideal solution to generating synthetic "Golden Speakers". When combined with Computer-Aided Pronunciation Training (CAPT) programs, these synthetic "Golden Speakers" could be a method for solving the personalized instruction problem [112, 113]. As many CAPT systems do not adjust the practice voice for listeners, forcing them to listen and practice with a single, unmatched voice [109], a system that allows for *personalized* voices may provide learners an even better practice environment.

To address these limitations in CAPT tools, we built the Golden Speaker Builder (GSB), a CAPT program which builds personalized Golden Speaker for learners: their own voice, but with a native accent. GSB allows for users to build incrementally and interactively, selecting different source accents for which to build their personalized accent training model. We use the SABR voice conversion system presented in the previous chapters, where the source speaker is a native L1 speaker of North American English, and the Golden Speaker Builder interface builds models of the L2 learner and synthesizes speech accordingly. The resulting synthesis has the prosody of the native L1 speaker, but the identity of the target L2 learner. This chapter describes the use of an accent conversion system based on the SABR framework and an evaluation of the synthesized L2 speaker voices in subjective tests. The SABR system presented in this dissertation is ideal for CAPT, as it requires both small amounts of training data and models how source and target speakers form English phonemes.

This chapter is organized as follows. First, we review the use of self-imitation in pronunciation training. Then, we discuss the implementation of the Golden Speaker Builder as a web application and how the collected data is then processed into SABR models for synthesis and use in training. We then perform perceptual studies to evaluate how effectively the synthesized Golden Speaker capture the target speaker's identity. We conclude with a discussion of the synthesis and application.

7.3. Related Work

Self-imitation for computer-aided pronunciation training has mainly focused on *prosodic* modification of a learner's voice for use in pronunciation training [114-119]. In an early study by Nagano and Ozawa [120], the authors used a voice conversion algorithm to resynthesize the L2 learner's voice with the prosody of a native L1 teacher's voice. They evaluated the voice conversion method for pronunciation training by comparing a group of students who mimicked their own voices to a control group of students who were trained with a reference English speaker. In a post-training evaluation, the students who used their own voice with the prosody adjustment were rated as being more native-like than the control group. More recently, Bissiri et al. [116, 117] built an automatic pronunciation teaching tool that to teach German lexical stress to Italian speakers. This tool extracted the pitch, speaking rate, and intensity from a reference native German speaker and copied it to the learner's speech signals for feedback in pronunciation training. The authors compared two groups of students—one who were trained on their own resynthesized voice, and another control group who were trained on a reference German teacher's voice. Again, the group using their own voice for accent training were rated as

having a more native-like accent than the control group. Additionally, providing feedback in the learner's own voice also had a motivating effect, with several participants asking to continue the training, whereas participants in the control group showed no particular interest.

De Meo et al. [118] evaluated the effectiveness of two forms of training (imitation and self-imitation) to teach suprasegmental patterns of Italian to Chinese learners. Participants in the self-imitation condition heard their own voice–resynthesized to match the native model, whereas those in the imitation condition followed traditional imitation exercises. Native listeners were then asked to classify learners' post-training productions as belonging to one of four speech acts: requests, orders, granting, and threats. Classification performance was significantly larger for utterances from participants in the self-imitation group. Similar improvements in communicative effectiveness were obtained in a later study with Japanese learners of Italian [119].

These studies show that the *prosodic* modification of accents are effective tools for teaching pronunciation to L2 learners and the effect is robust across several L1-L2 combinations. Here, we present the next step of self-imitation in CAPT—the addition of *segmental* modifications—something which the SABR VC system in the GSB tool allows.

7.4. System description

We developed Golden Speaker Builder (GSB), an online interactive tool that allows L2 learners to build a personalized pronunciation model: their own voice producing native-accented speech (i.e. a "golden speaker"). To build their golden speaker, L2 learners follow three steps. In the first step, the learner records a keyword for each phone

(e.g., for phoneme /ʒ/, the learner records the keyword "vision") under the guidance of an instructor to ensure that the utterance has near-native production. After recording each keyword, the learner segments the phone using a graphical display of the waveform. In the second step, the learner records several sentences, which are used to estimate the learner's pitch statistics. In a final step, the learner selects a native speaker as a source model, and GSB resynthesizes the native speaker's sentences using the recorded phone segments and prosody statistics of learner. The process can be completed in less than thirty minutes and generates a Golden Speaker voice that produces intelligible speech with the voice quality of the L2 learner, and the prosody of the source native speaker normalized to the pitch range of the L2 learner.

The software architecture of GSB is shown in Figure 32. GSB consists of three components: a web application, a signal processing back-end, and a middleware to connect the signal processing back-end to the web application. The web application provides a graphical interface for the learner, responds to the learner's requests, and stores the learner's data (i.e., login information, speech recordings, and golden speakers) onto a database. The signal processing back-end runs the accent conversion algorithms, which generates synthesized speech for each Golden Speaker model. Finally, the middleware layer provides communication between the web application and the signal processing back-end via an asynchronous task queue. In this chapter, we focus on the signal-processing backend and evaluate the performance of the SABR synthesis in subjective tests.

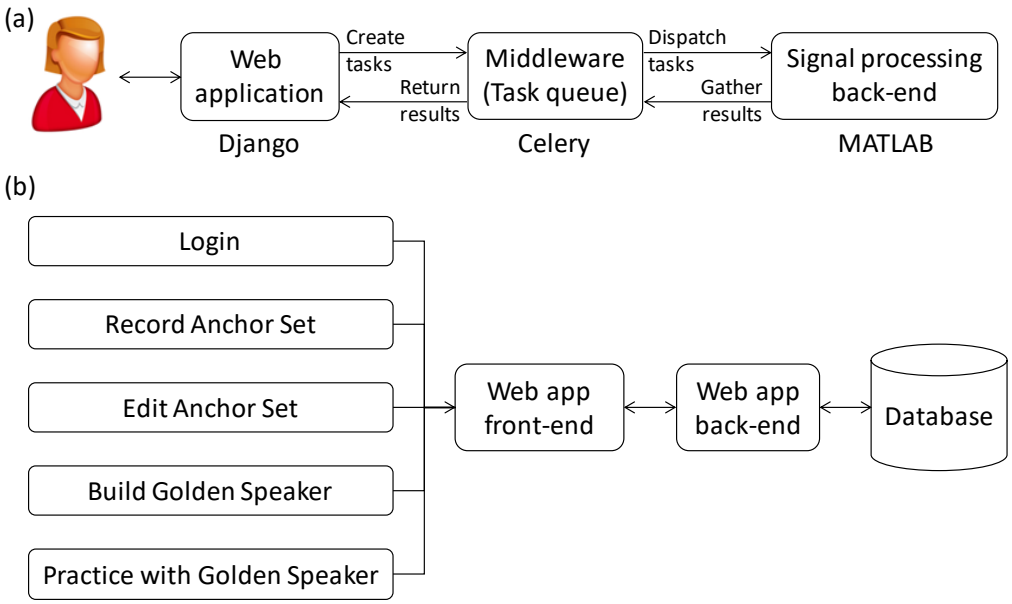


Figure 32 (a) Overall software architecture. (b) Architecture of the web application

7.4.1. *Web application*

We implemented the web application using the Django framework[8]. The web-app front-end was written in HTML5 and Javascript, and decorated with Bootstrap[9], whereas the web-app back-end was written in Python with Django internal modules. User data is managed by an SQLite database engine[10] on a standard Linux file system. We hosted the web application through Nginx[11]. To follow the workflow described below, we provide five functional modules: Login; Record Anchor Set; Edit Anchor Set; Build Golden Speaker; and Practice with Golden Speaker. In this chapter, we focus on the Record

[8] https://www.djangoproject.com/
[9] https://getbootstrap.com/
[10] https://www.sqlite.org/
[11] https://www.nginx.com/

Anchor Set module and the associated signal processing backend. The remaining modules are discussed in [107].

The Record Anchor Set module enables learners to record keywords and prosody sentences, later used to build a Golden Speaker model. As shown in Figure 33, the learner must record a keyword for each of the 40 phones in American English (CMU phone set[12]). Once a user records a keyword, the interface allows the learner to segment the phone segment (or "Anchor") by highlighting the corresponding region of the speech waveform. Separate tabs are used for consonants, vowels, and pitch sentences. Consonants are arranged according to their place and manner of articulation, and vowels are arranged according to their frontness and height (not shown). This arrangement allows the teacher and learner to review the basic organization of speech sounds in English, as the learner records the various keywords.

The "Pitch Sentences" tab includes 30 sentences representative of conversational speech (e.g., "What time does the bus leave for the airport?") that were deliberately selected to provide good coverage of various prosodic contexts, and a free-speech exercise in which the learner first watches a 3-minute short film[13] and then records a 1-2 minute audio summary. Recordings for all the keywords and pitch sentences are saved on the file system, whereas the segmentation information is saved in the database. In a final step, both the recordings and the segmentation information are sent to the signal processing back-end.

---

[12] http://www.speech.cs.cmu.edu/cgi-bin/cmudict

[13] "Spellbound" by Ying Wu and Lizzia Xu; available at youtube.com/watch?v=W_B2UZ_ZoxU

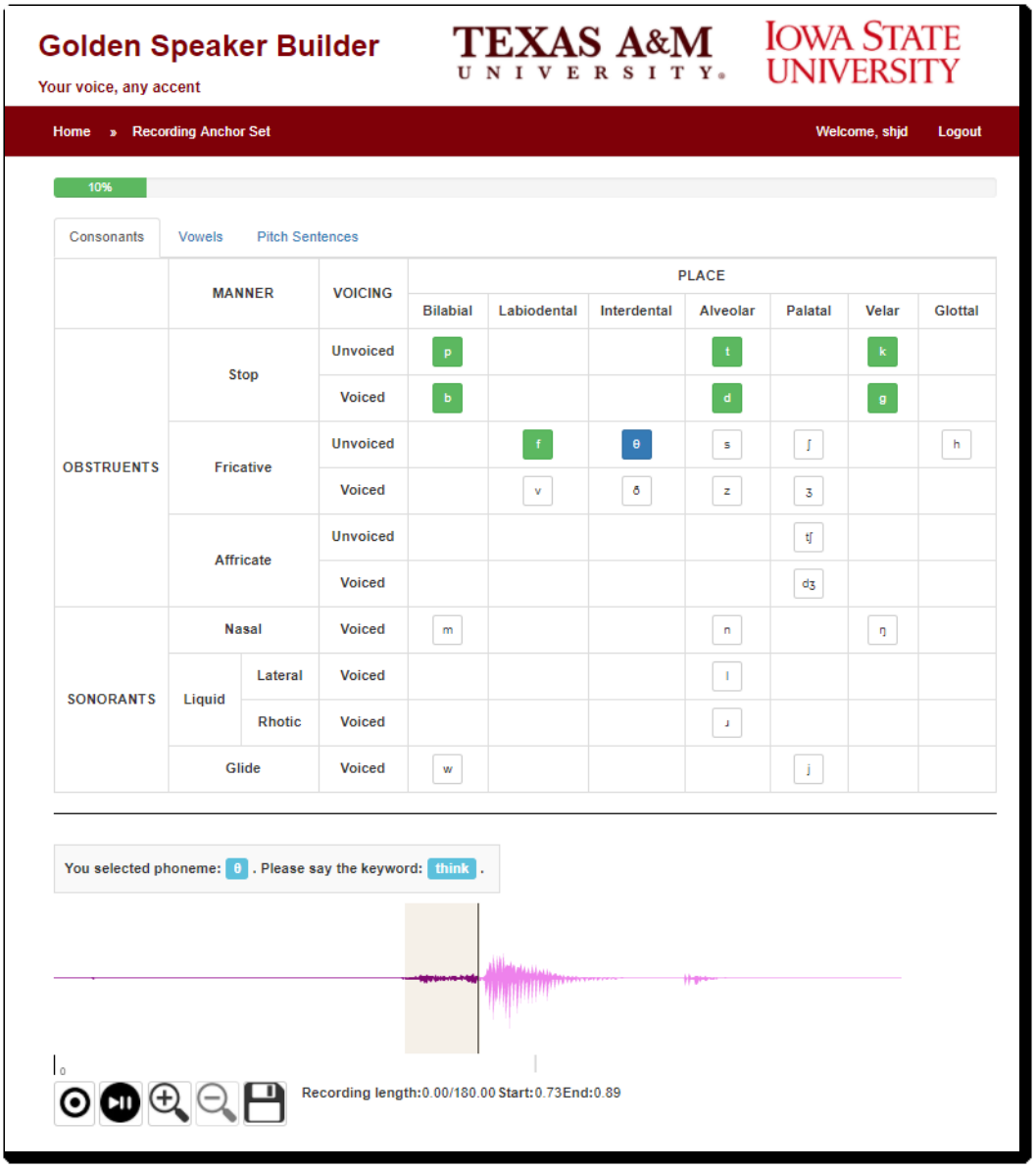


Figure 33. Graphical user interface for recording consonants in American English. In the example shown, the learner has already recorded keywords for all the stop consonants (highlighted in green), has recorded the phone ***/θ/*** (highlighted in blue) and is in the process of selecting the appropriate section in the speech waveform shown at the bottom of the page.

We selected one keyword per phoneme to capture an "ideal" example of that phoneme or its main characteristic, e.g., the dominant allophone of that phoneme. Voiceless aspirated stops are more distinct than unvoiced aspirated stops, and were chosen

preferentially for that reason. Additionally, final stops were avoided, as well as final rhotics and velarized approximants (e.g. “dark L”). The full selection of keywords is shown in Table 12.

Table 12: Golden Speaker Builder keyword selection.
The following is a list of keywords used to build anchor sets for L2 learners in the GSB application. Phoneme names are shown on the left column in ARPABET notation, and the words used to elicit the phoneme on the left.

AA	*father*	CH	*cheat*	HH	*heat*	NG	*sing*	TH	*think*
AE	*ash*	D	*deep*	IH	*if*	OW	*oh*	UH	*push*
AH	*us*	DH	*this*	IY	*east*	OY	*toy*	UW	*boot*
AO	*horse*	EH	*"s"*	JH	*jeep*	P	*poke*	V	*vote*
AW	*ouch*	ER	*earth*	K	*keep*	R	*reads*	W	*weeds*
AX	*sofa*	EY	*ace*	L	*leads*	S	*See*	Y	*yes*
AY	*ice*	F	*feed*	M	*make*	SH	*sheep*	Z	*zoo*
B	*boat*	G	*gust*	N	*no*	T	*tea*	ZH	*vision*

7.4.2. *Signal processing back-end*

To build Golden Speakers, the signal processing back-end uses the SABR technique discussed in previous chapters. In the case of GSB, source speaker anchors (i.e., the teacher’s anchors) are precomputed in advance for each of the native speaker voices, whereas target anchors are obtained from the learner’s recordings. We built the target anchor sets for the learners using the labeled segments in the same manner as the SABR models in Chapter 3. Frequency warping functions were computed on pairs of source and target anchors and the residual warping method in Chapter 4 was used during synthesis. We used STRAIGHT [39] to extract the spectral envelope and compress it to 25 MFCCS (25 Mel-filterbanks, 25 coefficients, 8kHz cutoff). Then, we separate energy ($MFCC_0$) and use the remaining coefficients ($MFCC_{1-24}$) during conversion. After converting these coefficients, we append the source $MFCC_0$ and backproject the MFCCs into the

STRAIGHT spectrum. Finally, we transform the pitch to match the target speaker's pitch range using log mean and variance scaling, as done in prior chapters.

7.5. Experiment design

7.5.1. *Speech corpus*

The speech corpus used for these perceptual listening tests consisted of recordings from L1 speakers (the "teacher" voices), L2 speakers (the "learner" voices) and golden speaker voices of the L2 speakers using the L1 speakers as models. For this purpose, first we recorded two American English speakers (CBL: male; GMA: female) as teacher voices. Each speaker produced 100 utterances from the ARCTIC corpus [121]. To generate SABR models for each teacher, we extracted phoneme labels using the Montreal forced-aligner [122]. Namely, for each phoneme in the GSB "Record Anchor Set" interface ($N = 40$), we extracted a single phoneme anchor corresponding to the centroid of all frames in the 100 utterances that were labeled with the corresponding phoneme.

Next, we recruited 18 L2 learners of American English to participate in the pronunciation training study. Each L2 learner recorded a set of keywords and prosody sentences, from which we built their corresponding SABR models. Then, L2 learners practiced with the 24 training utterances and recorded them pre- and post-treatment. Two of the L2 learners did not finish the study and another one L2 learner did not record their post-test sentences. Consequently, we have speech data from 15 learners (8 males, 7 females). Of these, we used speech data from 6 learners[14] (3 males, 3 females) for the

[14] We randomly selected 6 learners from the original set of 15 learners to ensure that perceptual study participants could complete the test within a reasonable time.

perceptual listening tests reported here. To obtain golden-speaker voices, we paired the 3 male L2 learners with the male L1 teacher voice (CBL), and the 3 female L2 learners with the female L1 teacher voice (GMA).

7.5.2. *Perceptual studies*

For each pair of L1-L2 speakers, we evaluated three types of golden-speaker voices:

- **Golden speaker 1 (GS1)**: These golden-speaker voices used a SABR model for the L2 learner where each phoneme anchor was obtained from the corresponding keyword segment, as originally segmented by the L2 learner –see Figure 2.
- **Golden speaker 2 (GS2)**: Out of concern that extracting a phoneme anchor from a single keyword would prove too limited, this golden-speaker voice used keywords and prosody sentences (forced aligned with Kaldi) to generate SABR models for each L2 learner.
- **Pitch transformation (PT)**: a baseline golden-speaker that only applies a pitch transformation [49, 50] to the L1 teacher voice to match the pitch range of the learner.

We conducted the perceptual listening tests on Amazon Mechanical Turk to evaluate the non-native voice identity, accentedness, and acoustic quality of the three golden-speaker voices. Recordings in each listening test were randomly ordered. We also included 12 calibration utterances in each listening test to detect if listeners were cheating [51]. If so, we removed their responses from the sample.

7.6. Results

7.6.1. *Voice identity*

We evaluated the voice identity of the syntheses using a Voice Similarity Score [112, 123] (VSS) test. Namely, participants listened to pairs of utterances and were required to (1) decide whether the two utterances were from the same speaker, and (2) then rate their confidence in the decision on a 7-point scale, as in [124]. For each utterance pair, one was a testing utterance randomly sampled from one of the three golden-speaker voices; the other was a reference utterance randomly sampled from either the corresponding source or target speaker. The VSS was then computed by collapsing the above two fields into a 15-point scale from -7 (definitely different speakers) to +7 (definitely the same speaker). Thirty listeners rated the VSS of 144 utterance pairs. We used 48 pairs for each golden-speaker voice and 8 pairs for each L1-L2 direction (4 AC-L1, 4 AC-L2). Following Felps et al. [112], we played utterances in reverse to reduce the influence of accents in the perception of voice identity.

Results are shown in Figure 34. For GS1, listeners were very confident that the syntheses and the original L1 recordings were from different speakers (-4.06), but were not quite sure if the syntheses and the original L2 recordings were from the same speaker (0.24). For GS2, listeners were also very confident that the syntheses and the original L1 recordings are from different speakers (-4.41), and they were confident that the syntheses and the original L2 recordings are from the same speaker (2.00). In contrast, listeners rated syntheses from pitch transformation as being from the same speaker as the original L1 recordings (4.46) and as from different speakers than the original L2 recordings (-2.94).

In addition, GS2 showed a statistically significant improvement on capturing the L2 speaker identity over GS1 ($p \ll 0.05$).

These results indicate that PT syntheses are perceived as being very close to the L1 speaker and very different from the L2 learners. By contrast, the two GSB syntheses are rated as being very different from the L1 speaker, and close to the identity of L2 learners, particularly in the case of GS2. Several factors may explain the relatively lower VSS ratings of GS1 and GS2 when compared to the L2 speakers. First, as noted by Munro and Derwing [125], playing utterances in reverse does not entirely eliminate the perception of accent; listeners may treat the GSB syntheses and original L2 speech as being from different speakers. Second, and as we will see in the following sections, the GSB syntheses have lower quality than the original L2 speech, further discouraging listeners from identifying similarities between them.

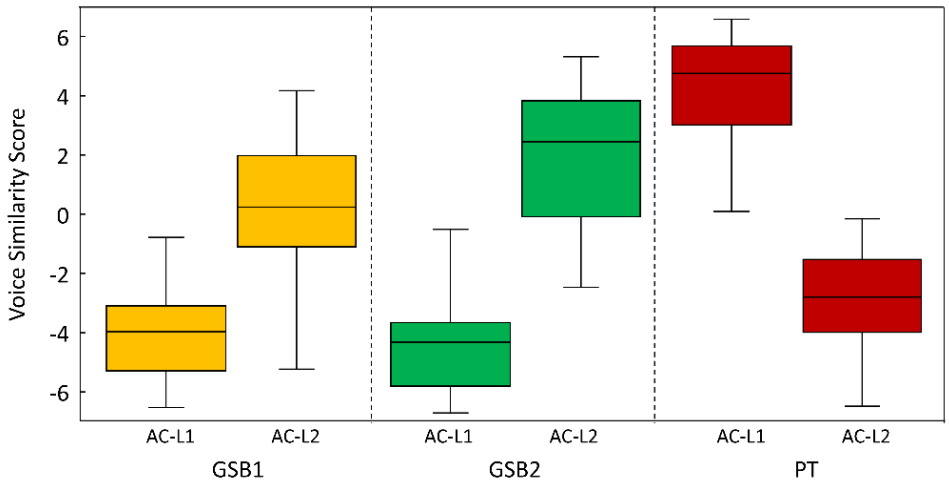


Figure 34. Voice identity ratings for the Golden Speaker voices.
The range is from -7 (definitely different speakers) to +7 (definitely the same speaker)

7.6.2. *Foreign accentedness*

Following Munro and Derwing [125], we used a scaled-rating test to establish the degree of accentedness of individual utterances. Twenty-seven listeners rated the foreign accentedness (1-No foreign accent, 9-Very strong foreign accent) of 150 utterances. The utterances were from either of the three test conditions above, from the source native speakers, or from the target foreign speakers. We used 30 utterances for each golden speaker voice and the target foreign speakers—5 utterances for each of the 6 learners. For the source native speakers, we selected 15 utterances for each of the speakers to ensure a class balance in the test.

Results are summarized in Figure 35 (a). As expected, original utterances from the L1 speakers received the lowest ratings of foreign accentedness (1.11), whereas those from the L2 learners received highest ratings (7.44). Pitch transformation achieved similar ratings as the original L1 utterances (1.17), which is to be expected since pitch-transformed utterances are identical to L1 utterances except for the pitch range. Finally, the two golden-speaker voices were rated as being significantly less accented than the L2 utterances but not as much as L1 utterances (GS1: 2.59, GS2: 2.42), with differences between GS1 and GS2 being not statistically significant ($p \gg 0.05$).

In summary, the three golden-speaker voices showed a significant decrease (~84%) in foreign accentedness compared to the original L2 speech. However, foreign accentedness ratings for GS1 and GS2 were higher than those of the original L1 speech as well as from the PT condition. This can be attributed in part to the anchor-building process;

although keywords and prosody utterances were recorded under the supervision of the teacher, evidence of L2 pronunciation still comes through in the productions.

7.6.3. *Acoustic quality*

We evaluated the acoustic quality of the three golden-speaker voices using a Mean Opinion Score (MOS) test. Twenty-eight listeners rated the MOS (1-bad, 5-excellent) of 150 utterances. We used the same test conditions as in the foreign accentedness test in the prior section.

Results are summarized in Figure 35 (b). As one might expect, listeners rated original utterances from L1 speakers and pitch transformation as having the highest acoustic quality (L1: 4.66, PT: 4.56). Surprisingly, though, listeners gave the L2 recordings a much lower MOS (3.44), despite the fact that they were unmodified recordings, which indicates the presence of interaction effects between ratings of acoustic quality and accentedness. Finally, listeners rated the synthesized golden-speaker voices as having lower quality: GSB1 received a 1.77 MOS and GSB2 received a 2.16 MOS. Differences between these two syntheses were statistically significant ($p \ll 0.05$), which indicates that including pitch utterances in the computation of phonetic anchors was beneficial.

The two GSB syntheses (GS1 and GS2) did not provide as good acoustic quality as the Pitch Transformation (1.77 for GS1, 2.16 for GS2, 4.56 for PT), due to distortions introduced in the accent-conversion algorithm. We anticipated this result, since the pitch transformation technique does not alter the speech spectrogram and distortions are minimal, i.e., due to the STRAIGHT vocoding process. In contrast, the GSB spectral

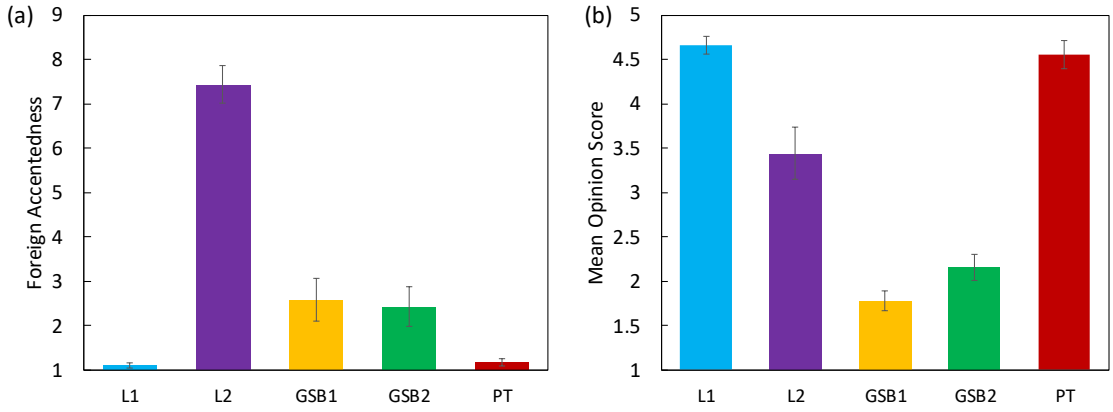


Figure 35. (a) Foreign accentedness ratings. The rating ranges from 1 (no foreign accent) to 9 (very strong foreign accent). (b) Mean opinion score (MOS) of acoustic quality ratings with 95% confidence interval. The MOS scale is from 1 (bad) to 5 (excellent).

conversion algorithms introduce several forms of distortion. First, the conversion takes place in the (compressed) MFCC space, so the spectrum must be reconstructed by back-projecting the converted MFCCs. Additionally, there are minor differences in how the GS1 and GS2 models are built, which also impact the conversion quality. Namely, the anchor set for the source model is derived from labels obtained by forced-aligning a relatively large corpus (100 utterances), whereas the anchor set for the GS1 model is derived from manually-labeled keywords (1 keyword per phone), and the anchor set for the GS2 model is derived by force-aligning 15 utterances. Thus, the difference in corpus size and annotation sources may result in significantly different anchors, affecting the synthetic acoustic quality. We discuss the impact of these model differences in the following section.

7.7. Discussion

7.7.1. *Analysis of the perceptual studies*

The perceptual study indicate Golden Speaker Builder accomplished our goal of building a speaker voice that is suitable for self-imitation pronunciation training: the two GSB golden-speakers have the same speaker identities as L2 learners, a near native English accent, and reasonable acoustic quality. Although the syntheses based on pitch transformation [126, 127] achieve lower foreign accentedness and higher acoustic quality than GSB syntheses, pitch transformation fails to capture the L2 learners' identity, which is critical for self-imitation pronunciation training. The increased identity scores and acoustic quality of the GS2 utterances (when compared to GS1) also provides a direction for improving the GSB syntheses overtime: as additional training data becomes available for each learner's practice sessions with the tool, the original anchor sets can be refined.

Additionally, we found that a compounding factor in evaluating synthesis results is that of the rated acoustic quality. While GS1 and GS2 both had lower MOS than the original L1 speech, the original L2 recordings were also rated significantly lower (3.44 MOS). Since the L1 and L2 speakers were recorded under identical conditions, we suspect listeners regarded disfluencies and foreign accents in L2 speech as being of lower acoustic quality than native speech. Post-test feedback from some listeners supports this explanation: some were unsure if the low intelligibility was due to the speaker or to the overall low acoustic quality.

7.7.2. *Anchor robustness in the Golden Speaker Builder*

Results show that the GS2 syntheses outperform the GS1 models by a significant margin in both acoustic quality and voice similarity measures. In previous chapters, SABR had significantly higher synthesis quality than we observe here, but the target speakers were from the ARCTIC or L2-ARCTIC corpora. As in prior chapters, we examined the correlation between source and target anchors (as higher correlation has been associated with higher synthesis quality [92, 128]) to determine if the relationship between correlation and synthesis quality can be observed here. We compare the two methods for building target anchors (GS1, GS2) against anchors built from ARCTIC speakers. Results are shown in Table 13.

GS2 correlation coefficients are higher than GS1 in all cases, though not as high as those obtained when speaker pairs are from ARCTIC. This can partially explain why the acoustic quality of SABR-based voice conversion on ARCTIC speakers is higher than the values seen for the GSB syntheses. These results provide some directions as to how to improve model building for GSB. First, as ARCTIC phoneme labels—and resulting SABR models—are built using forced-alignment tools, it may not be desirable to use human-annotated phoneme boundaries to build SABR models. This may also impact nonstationary phonemes (e.g. stops, affricates, diphthongs) as human-annotated phoneme boundaries may differ from those selected by a forced-aligner; the resulting anchor may be substantially different than that selected by a human annotator.

Table 13: correlation coefficients in anchor sets between pairs of speakers in the present study (GS1, GS2) and pairs of speakers from ARCTIC.
ARCTIC speaker models had higher anchor correlations in all cases than the corresponding GSB models, which may account for the significant MOS difference between ARCTIC syntheses and those of the GSB models. Additionally, including the pitch sentences (GS2) also improved the correlation of the anchor sets.

	GS1	GS2	ARCTIC
Stop	0.61	0.828	0.924
Fricatives	0.599	0.703	0.844
Affricates	0.381	0.381	0.794
Nasals	0.922	0.929	0.941
Liquids	0.853	0.924	0.972
Glides	0.867	0.922	0.964
Monophthongs	0.869	0.887	0.946
Diphthongs	0.814	0.849	0.944
Overall	**0.805**	**0.86**	**0.935**

7.8. Conclusion

This case study evaluated the use of SABR in a CAPT program to help learners of a second language gain a native accent. SABR was used to generate audio samples the learners could use to practice their accent. Results showed that GSB systems achieve low foreign accentedness while capturing the voice identity of the L2 learner and achieving reasonable acoustic quality.

8. CONCLUSION AND FUTURE WORK

This dissertation proposed and presented a method for performing voice conversion by representing speaker identity as a linear combination of speaker-specific "anchors", which separated speaker identity from content. The first chapter, Chapter 3, presented the sparse, anchor-based framework, and evaluated it in subjective and objective contexts. Chapters 4 through 6 presented three optimizations to aspects of this proposed method. Chapter 7 examined a case study in which SABR was used in a Computer Aided Pronunciation Training context to build "Golden Speakers" for nonnative speakers of English. This chapter summarizes these findings, notes the contributions, and offers ideas for future work.

8.1. Summary

In Chapter 3, we presented the initial formulation of the anchor-based voice conversion framework SABR: Sparse-Anchor Based Representation of speech. One anchor per phoneme in English was used to represent a speaker's "anchors", and the Lasso was used as the sparse coding method from which to learn a nonnegative, sparse set of weights. We demonstrated that this representation could not only separate identity from content, but perform voice conversion with limited training data as compared to a statistical voice conversion baseline.

In Chapter 4, we presented a method for using the sparse coding residual to significantly improve on the synthesis quality of the SABR utterances by performing frequency warping to warp the source residual to be closer to the target speaker. We evaluated four frequency warping functions to determine which gave the best objective

and subjective errors. We also demonstrated that the proposed residual warping method both improved the synthesis quality and captured the target speaker's identity. In a set of baseline experiments, we also showed that the method also outperformed other baseline methods in native-to-nonnative VC.

In Chapter 5, we examined selecting and optimizing SABR anchors to improve synthesis quality and phoneme representation. We two proposed algorithms—Iterative Retraining (IRT) and Anchor Removal and Selection (ARS)—to solve different problems associated with the anchor selection of the original SABR algorithm. The IRT algorithm was a novel dictionary learning algorithm that optimized the anchor sets to minimize the residual, conditioned on the voice conversion error of the anchor sets. The ARS algorithm was meant to address the multiple problems with phoneme-based anchors. First, some phonemes that were not adequately represented by a single anchor (e.g. affricates or diphthong vowels, as they have multiple acoustic states over their production). Second, the "removal" operation would help to optimize anchors for nonnative speakers, as mispronunciations could cause there to be no good sample of a phoneme. The ARS algorithm would greedily split or remove anchors depending on whether or not they reduced the VC error. We found that the proposed optimization methods significantly improved the synthesis quality of both native-to-native and native-to-nonnative voice conversion contexts, outperforming other baseline voice conversion methods with very limited training data (20 utterances). The split and removal decisions of the ARS algorithm focused on phonemes with known mispronunciations (for nonnative target speakers) or multiple acoustic states (for both native and nonnative target speakers), demonstrating that

the intuition of the algorithm was correct: some phonemes need multiple anchors to be represented properly, and in the case of native-nonnative conversion, removing some anchors can reduce the VC error. The IRT algorithm significantly reduced the residual to be closer to that of voice conversion methods with thousands more exemplars.

In Chapter 6, we proposed a modification to the Lasso to incorporate temporal constraints on the SABR voice conversion method. This variant penalized not only the magnitude of the sparse codes at a given frame, but the *frame-to-frame* changes in the codes. We found that there was no improvement in synthesis quality, but there was *significantly* less frame-to-frame coding variance and no change in the objective voice conversion error. Additionally, the proposed method significantly outperformed other methods of incorporating temporal information into an exemplar-based voice conversion system.

Finally, in Chapter 7, we performed a case study in which we evaluated SABR in the context of a computer-aided pronunciation training system. This system, called the "Golden Speaker Builder", generated "Golden Speakers" (ideal voices for teaching nonnative speakers how to gain native pronunciation) for nonnative learners of English. Using SABR, we synthesized the learner's voice using SABR, resulting in training utterances with the learner's identity but the accent of a native speaker. SABR syntheses were rated as being similar to the L2 learner's voice.

8.2. Contributions

This dissertation contains the following main contributions:

- Developed a Sparse, Anchor-Based Representation of Speech, and demonstrating that the phoneme-based exemplars and the use of the Lasso as a sparse coding method can separate speaker identity from content for use in Voice Conversion. This method is significantly more compact than other exemplar-based voice conversion systems while demonstrating similar performance in terms of synthesis quality and speaker identity.
- Developed a novel residual warping algorithm to improve the synthesis quality of exemplar-based VC methods. Experimentally, we demonstrated that the algorithm had better synthesis quality in native-to-nonnative voice conversion when compared to baseline VC methods.
- Developed two novel exemplar optimization algorithms, Anchor Removal and Selection and Iterative Retraining. These methods significantly improved synthesis quality in native-to-nonnative conversion compared to a baseline exemplar-based algorithm.
- Proposed and evaluated a novel temporal constraint framework using Fused Lasso that significantly reduced the temporal variance of the SABR weights without significantly changing the VC synthesis quality.
- Developed a Computer Aided Pronunciation Training system called the "Golden Speaker Builder" using the proposed SABR VC algorithm. The synthesis that sounded more similar to the speaker identity of a learner than a comparison pitch modification method alone.
- Demonstrated that SABR models can be used to perform Accent Conversion (AC) and perform much better than other exemplar-based VC systems on AC tasks.

8.3. Future work

8.3.1. *Using SABR Weights for nonparallel alignment*

In Chapter 5, we proposed two anchor optimization algorithms which required parallel data to perform training, as we used dynamic time warping (DTW) to align the source and target training data. While alternative means of alignment (e.g. PPG) could be used, one area of interest with SABR would be to use the SABR weights to perform alignment. We can exploit statistical properties of the Lasso to perform such an alignment. In [129], Park and Casella presented a Bayesian formulation of the Lasso and solved for the posterior distribution of the Lasso sparse codes β, which follow an exponential distribution with variance σ and mean λ. Applying this distribution to the SABR weights, we can compute Kullback-Leibler divergence on pairs of source and target frames in a similar manner that PPG alignment techniques do and learn. Because the weights will be sensitive to the anchors selected for the source and target speakers, using the posterior distribution of the SABR weights to ensure that both the source and target weights have similar distributions will be necessary to ensure a good alignment.

8.3.2. *Improving the anchor optimization algorithms*

In the perceptual results of the dictionary learning algorithm, we noted that the algorithms significantly improved the synthesis quality of the anchor sets, but it did not necessarily improve the speaker identity. This may be because speaker information is retained in the SABR weights, and therefore explicitly minimizing the differences between the source and target weights could improve the *identity* of the speakers.

8.3.2.1. Adding weight difference penalty to ARS

To add a weight difference penalty to the ARS algorithm, we can simply add a weight difference penalty to the anchor optimization decision to Algorithm 1 from Chapter 4:

$$e_{k,f}^{t} = \gamma_1 \left| |X_T' - A_T^{k,f} W_S| \right|_2^2 + \gamma_2 \|W_S - W_T\|_1 \tag{38}$$

The parameters γ_1 and γ_2 can be used to tune the how much either term contributes to the error contribution. The inclusion of the weight penalty term will incentivize the algorithm to select anchors that have similar encoding in the source and target speaker domains. Because the ARS algorithm is greedy and only uses weights in its decision at each iteration, including this penalty is straightforward. To incorporate weight differences in the IRT algorithm requires a more complex solution, which we present next.

8.3.2.2. Weight difference penalty using the Fused Lasso

With some reworking of the optimization algorithm, the proposed Fused Lasso algorithm in Chapter 6 is flexible enough for us to penalize *encoding* differences in the weights, then use those weights in either the ARS or IRT algorithms.

To use the Fused Lasso to minimize differences in encoding, we must first reformulate the SABR weight calculation as a *joint* encoding:

$$\min_W \left\| \begin{bmatrix} X_S \\ X_T \end{bmatrix} - \begin{bmatrix} A_S & 0 \\ 0 & A_T \end{bmatrix} \begin{bmatrix} W_S \\ W_T \end{bmatrix} \right\|_2^2 + \lambda \left\| D \begin{bmatrix} W_S \\ W_t \end{bmatrix} \right\|_1 \tag{39}$$

Then, we design a penalty structure D to penalize both the magnitude of W and the difference between the weights:

$$D = \begin{bmatrix} I & -I \\ 0 & I \end{bmatrix} \tag{40}$$

As before, we introduce a surrogate variable Θ to retain the Lasso structure of the problem. The penalty value becomes:

$$\Theta = D \begin{bmatrix} W_S \\ W_T \end{bmatrix} \tag{41}$$

And solving for the joint weights is still possible, because D is full-rank:

$$\begin{bmatrix} W_S \\ W_T \end{bmatrix} = D^{-1}\Theta \tag{42}$$

Substituting Θ for W into (39) and expanding the terms:

$$\min_W \left\| \begin{bmatrix} X_S \\ X_T \end{bmatrix} - \begin{bmatrix} A_S & A_S \\ 0 & A_T \end{bmatrix} \Theta \right\|_2^2 + \lambda \|\Theta\|_1. \tag{43}$$

We note that Θ can then be split into two components:

$$\min_W \left\| \begin{bmatrix} X_S \\ X_T \end{bmatrix} - \begin{bmatrix} A_S & A_S \\ 0 & A_T \end{bmatrix} \begin{bmatrix} \Theta_1 \\ \Theta_2 \end{bmatrix} \right\|_2^2 + \lambda \|\Theta\|_1. \tag{44}$$

This equation is now a joint encoding problem of a Lasso form that computes the weights for the source and target speakers as well as the difference between the source and target encoding. We can expand (44) equation to understand the system of equations being evaluated, and in turn, how to use these parameters in coding and decoding:

$$X_s \cong A_S\Theta_1 + A_S\Theta_2 = A_S(\Theta_1 + \Theta_2) \tag{45}$$

$$X_T \cong A_T\Theta_2 \tag{46}$$

In more plain terms, this shows that after learning the sparse codes Θ, we can interpret that the source SABR weights $W_S = \Theta_1 + \Theta_2$ and the target codes are $W_T = \Theta_2$.

As such, we have computed weights W_S and W_T, but with the structured sparsity constraints $\Theta = DW$ directly penalizing the framewise differences between W_S and W_T.

8.3.2.3. Adding tuning parameters

The weight difference penalty (41) and sparsity penalties (See (3) in Chapter 3) may benefit from having different tuning constants, γ_1 and γ_2, for penalizing differences and the magnitude of the target weights:

$$\min_W \left\| \begin{bmatrix} X_S \\ X_T \end{bmatrix} - \begin{bmatrix} A_S & 0 \\ 0 & A_T \end{bmatrix} \begin{bmatrix} W_s \\ W_t \end{bmatrix} \right\|_2^2 + \lambda \left\| D \begin{bmatrix} \gamma_1 & 0 \\ 0 & \gamma_2 \end{bmatrix} \begin{bmatrix} W_S \\ W_t \end{bmatrix} \right\|_1 \tag{47}$$

Incorporating γ into (42), the inverse then accounts for the tuning parameter difference:

$$\min_W \left\| \begin{bmatrix} X_S \\ X_T \end{bmatrix} - \begin{bmatrix} A_S & A_S \\ 0 & A_T \end{bmatrix} \begin{bmatrix} \gamma_1 & 0 \\ 0 & \gamma_2 \end{bmatrix}^{-1} \begin{bmatrix} \Theta_1 \\ \Theta_2 \end{bmatrix} \right\|_2^2 + \lambda \|\Theta\|_1 \tag{48}$$

Incorporating that into the original coding equations, we can solve for approximations of X_S and X_T:

$$X_s \cong A_S \left(\frac{\Theta_1}{\gamma_1} + \frac{\Theta_2}{\gamma_2} \right) \tag{49}$$

$$X_T \cong A_T \gamma_2^{-1} \Theta_2 \tag{50}$$

The resulting values of $W_S = \left(\frac{\Theta_1}{\gamma_1} + \frac{\Theta_2}{\gamma_2} \right)$ and $W_T = \gamma_2^{-1} \Theta_2$ can then be used in the dictionary learning algorithms, such as the IRT algorithm.

In the Generalized Lasso [105], the authors report on the use of a penalty parameter similar to the one presented here. According to the authors, the parameter γ_1 (which penalizes the difference between the source and target weights) should be $\gamma_1 < 0.1\gamma_2$.

8.3.2.4. Incorporating frequency warping into iterative retraining

In the evaluated version of the iterative retaining (IRT) algorithm presented in Chapter 5, we did not consider including the warped source or target residuals in the training algorithm. Instead, we optimized the source and target anchor sets after excluding the warped residual. Incorporating the source-target and target-source warped residuals using the method in Chapter 4 during the weight computation step is a natural extension to the IRT algorithm. Estimating the weights from the full residual-warped spectra would allow IRT algorithm to optimize not just on the anchor sets, but to also account for the effects of the residual warping algorithm. In the context of Accent Conversion (e.g., L1 to L2 conversion) including the residual in the iterative retraining has a high-level similarity to back-translation in machine translation [130-132]. In machine translation, including synthetic target sentences translated from source sentences in the training set (i.e., *back translation*) improves overall translation accuracy. Similarly, including synthetic target spectra (i.e., estimated target spectra, including the warped source residual) in the IRT method would serve to optimize the anchors sets in such a way as to account for the residual warping functions during the optimization, and potentially further improve synthesis quality.

One potential challenge to this method, however, is ensuring the source and target weights eventually converge. In preliminary evaluations of the IRT algorithm that

included the warped source spectra in the weight computation process, the source and target weights tended to change non-monotonically, and thus did not show steady reduction in either the VC error or residual metrics. Constraining the weights in such a way as to ensure monotonicity would be necessary to include the warped residuals in the IRT algorithm.

8.3.3. *More complex temporal constraints*

The temporal constraints shown in Chapter 6 take the source and target anchor sets and duplicate them over the temporal window we are evaluating. While this allows for weight consistency frame-to-frame, it doesn't account for the fact that some phonemes change their acoustic states over their production (e.g. affricates, stops). In Chapter 5, we explored adding in multiple anchors per phoneme class, but we did not modify the Lasso penalty to be aware of these multiple classes or the temporal relationship between these multiple anchors. The Fused Lasso can be extended to account for this, penalizing both frame-to-frame weight differences (e.g., difference between the weights for one anchor k_1 at times t and $t-1$) as well as *transitions* between two different anchors in successive time steps.

The implementation of this in the Fused Lasso is equivalent to building a network of penalties between anchors. Given a penalty network a structure such that edges are the penalty magnitude for transitioning between anchors on successive and nodes are the anchors themselves. The penalty matrix D then has a similar structure to an adjacency matrix.

In general, we would want penalties between anchors belonging to different phoneme class to be higher to discourage the frame-to-frame selection changes, and penalties to anchors within the same phoneme class to be lower to encourage selection of those samples. To implement this in the Fused Lasso, consider the formulation of the Fused Lasso presented in Chapter 6. We modify the anchor sets to include *temporal* anchors over the window $[t-k, t+k]$:

$$\boldsymbol{A_s'} = \begin{bmatrix} A_s^{t-k} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & A_s^{t+k} \end{bmatrix} \tag{51}$$

We then modify the penalty structure matrix D (32) to penalize not just frame-to-frame differences of the same weights, but also transitions between the weights. First, we define a structure matrix $C \in \mathbb{R}^{K \times (2k+1)K}$, where K are the number of phoneme classes, and k are the number of temporal frames of that phoneme in the source and target anchor sets (resulting in $A_S' \in \mathbb{R}^{(2k+1)N \times (2k+1)K}$). C is structured such that nonzero entries on the k^{th} row correspond to the anchors belonging to that phoneme:

$$C = \begin{bmatrix} [1 \ldots 1] & \cdots & \mathbf{0} \\ \vdots & \ddots & \vdots \\ \mathbf{0} & \cdots & [1 \ldots 1] \end{bmatrix} \tag{52}$$

For illustrative purposes, let us consider the difference between two time frames t and $t+1$, where there is a single anchor per phoneme, with in-phoneme transition penalty γ_1 and phoneme transition penalty γ_2. This results in a penalty with the structure:

$$\gamma_1 \begin{bmatrix} C \\ -C \end{bmatrix}^T \begin{bmatrix} W_t \\ W_{t+k} \end{bmatrix} + \gamma_2 \begin{bmatrix} C \\ -(1-C) \end{bmatrix}^T \begin{bmatrix} W_t \\ W_{t+k} \end{bmatrix} \tag{53}$$

Which can be simplified to:

$$\begin{bmatrix} \gamma_1 C + \gamma_2 C \\ -\gamma_1 C - \gamma_2 (1+C) \end{bmatrix}^T \begin{bmatrix} W_t \\ W_{t+k} \end{bmatrix} \tag{54}$$

We can expand this intra and cross-phoneme penalty to an arbitrary number of time windows.

$$D = \begin{bmatrix} \gamma_1 C + \gamma_2 C & -\gamma_1 C - \gamma_2 (1+C) & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \gamma_1 C + \gamma_2 C & -\gamma_1 C - \gamma_2 (1+C) & \dots & \mathbf{0} \\ \vdots & & \ddots & & \vdots \\ \mathbf{0} & \dots & \gamma_1 C + \gamma_2 C & -\gamma_1 C - \gamma_2 (1+C) & \mathbf{0} \\ \mathbf{0} & \dots & \mathbf{0} & \gamma_1 C + \gamma_2 C & -\gamma_1 C - \gamma_2 (1+C) \end{bmatrix} \tag{55}$$

In the Generalized Lasso, the inverse of this structured penalty is included if the penalty is full matrix. In this case, the penalty is no longer full rank, but we can approximate it using the pseudoinverse:

$$\min_{\boldsymbol{W'}} \|\boldsymbol{X'} - \boldsymbol{A'_S} D^+ \Theta\|_2^2 + \lambda_2 \|\Theta\|_1 \ \ s.t. \Theta \geq 0 \tag{56}$$

This structure would allow for a highly-tunable SABR encoder, allowing for senone-like phoneme anchors. However, in this modification, the dictionary used by the Lasso solver is $A'_S D^+$, and because D is now significantly larger, the number of atoms in the dictionary is also much larger. The complexity of many Lasso solvers is $O(n^3)$, where n is the number of atoms in the dictionary. Including multiple time frames and penalizing on transitions between anchors result in significant more computation to solve eq. (56)

than the Fused Lasso methods discussed in Chapter 6[15]. Additional optimizations to increase the speed of the solver (e.g. incorporating graph structures into LARS, [89, 133, 134], or incorporating deep learning techniques in the solvers [135], would be required to make this solver tractable.

[15] In initial experiments with this proposed modification, the solver took several minutes to solve a single time window of width 5 (2 frames before and after the current frame being encoded), several thousand times slower than the Fused Lasso method shown in Chapter 6. Modifications to the LARS solver to "short cut" and only examine subsets of the anchor sets could significantly reduce the complexity of the algorithm.

REFERENCES

[1] S. Aryal and R. Gutierrez-Osuna, "Can voice conversion be used to reduce non-native accents?," in *ICASSP*, 2014, pp. 7879-7883: IEEE.

[2] Y. Stylianou, O. Cappé, and E. Moulines, "Continuous probabilistic transform for voice conversion," *IEEE Transactions on Speech and Audio Processing,* vol. 6, no. 2, pp. 131-142, 1998.

[3] T. Toda, H. Saruwatari, and K. Shikano, "Voice conversion algorithm based on Gaussian mixture model with dynamic frequency warping of STRAIGHT spectrum," in *ICASSP*, 2001, vol. 2, pp. 841-844: IEEE.

[4] D. Erro, A. Moreno, and A. Bonafonte, "Voice conversion based on weighted frequency warping," *IEEE Transactions on Audio, Speech, and Language Processing,* vol. 18, no. 5, pp. 922-931, 2010.

[5] D. Erro, E. Navas, and I. Hernaez, "Parametric voice conversion based on bilinear frequency warping plus amplitude scaling," *IEEE Transactions on Audio, Speech, and Language Processing,* vol. 21, no. 3, pp. 556-566, 2013.

[6] T. Toda, A. W. Black, and K. Tokuda, "Voice conversion based on maximum-likelihood estimation of spectral parameter trajectory," *IEEE Trans. on Audio, Speech, and Language Processing,* vol. 15, no. 8, pp. 2222-2235, 2007.

[7] V. Popa, J. Nurminen, and M. Gabbouj, "A study of bilinear models in voice conversion," *Journal of Signal and Information Processing,* vol. 2, no. 02, p. 125, 2011.

[8] Z. Wu, T. Virtanen, T. Kinnunen, E. S. Chng, and H. Li, "Exemplar-based voice conversion using non-negative spectrogram deconvolution," in *ISCA Workshop on Speech Synthesis*, 2013.

[9] C. Liberatore, S. Aryal, Z. Wang, S. Polsley, and R. Gutierrez-Osuna, "SABR: sparse, anchor-based representation of the speech signal," in *Interspeech*, 2015.

[10] C. Liberatore, G. Zhao, and R. Gutierrez-Osuna, "Voice conversion through residual warping in a sparse, anchor-based representation of speech," in *ICASSP*, 2018.

[11] S. Ding *et al.*, "Golden Speaker Builder - An interactive tool for pronunciation training," *Speech Communication,* vol. 115, pp. 51-66, 2019.

[12] G. Fant, *Acoustic theory of speech production: with calculations based on X-ray studies of Russian articulations*. de Gruyter, 1971.

[13] P. Birkholz, B. J. Kröger, and C. Neuschaefer-Rub, "Model-based reproduction of articulatory trajectories for consonant–vowel sequences," *Audio, Speech, and Language Processing, IEEE Transactions on,* vol. 19, no. 5, pp. 1422-1433, 2011.

[14] J. Harrington and S. Cassidy, "The acoustic theory of speech production," in *Techniques in Speech Acoustics*: Springer, 1999, pp. 29-56.

[15] C. Liberatore and R. Gutierrez-Osuna, "Joint Optimization of Anatomical and Gestural Parameters in a Physical Vocal Tract Model " in *IEEE Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2015, pp. 4250-4254.

[16] C. P. Browman and L. Goldstein, "Articulatory phonology: An overview," *Phonetica,* vol. 49, no. 3-4, pp. 155-180, 1992.

[17] K. Johnson, "Acoustic and auditory phonetics," *Phonetica,* vol. 61, no. 1, pp. 56-58, 2004.

[18] H. Hermansky and D. J. Broad, "The effective second formant F2'and the vocal tract front-cavity," in *International Conference on Acoustics, Speech, and Signal Processing*, 1989, pp. 480-483: IEEE.

[19] J. M. Pickett, *The acoustics of speech communication: Fundamentals, speech perception theory, and technology*. Allyn & Bacon, 1999.

[20] P. Taylor, *Text-to-speech synthesis*. Cambridge university press, 2009.

[21] L. R. Rabiner and R. W. Schafer, *Introduction to digital speech processing*. Now Publishers Inc, 2007.

[22] E. Moulines and F. Charpentier, "Pitch-synchronous waveform processing techniques for text-to-speech synthesis using diphones," *Speech communication,* vol. 9, no. 5-6, pp. 453-467, 1990.

[23] A. van den Oord *et al.*, "WaveNet: A Generative Model for Raw Audio," in *9th ISCA Speech Synthesis Workshop*, 2017

pp. 125-125.

[24] E. Eide and H. Gish, "A parametric approach to vocal tract length normalization," in *1996 IEEE International Conference on Acoustics, Speech, and Signal Processing Conference Proceedings*, 1996, vol. 1, pp. 346-348: IEEE.

[25] D. Sundermann, H. Ney, and H. Hoge, "VTLN-based cross-language voice conversion," in *2003 IEEE workshop on automatic speech recognition and understanding (IEEE Cat. No. 03EX721)*, 2003, pp. 676-681: IEEE.

[26] C. J. Leggetter and P. C. Woodland, "Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models," *Computer speech & language,* vol. 9, no. 2, pp. 171-185, 1995.

[27] D. Sundermann, H. Ney, and H. Hoge, "VTLN-based cross-language voice conversion," in *Automatic Speech Recognition and Understanding, 2003. ASRU'03. 2003 IEEE Workshop on*, 2003, pp. 676-681: IEEE.

[28] A. Mouchtaris, J. Van der Spiegel, and P. Mueller, "Nonparallel training for voice conversion based on a parameter adaptation approach," *IEEE Transactions on Audio, Speech, and Language Processing,* vol. 14, no. 3, pp. 952-963, 2006.

[29] C. Qin and M. Á. Carreira-Perpiñán, "An empirical investigation of the nonuniqueness in the acoustic-to-articulatory mapping," in *Eighth Annual Conference of the International Speech Communication Association*, 2007.

[30] S. Panchapagesan and A. Alwan, "A study of acoustic-to-articulatory inversion of speech by analysis-by-synthesis using chain matrices and the Maeda articulatory model," *The Journal of the Acoustical Society of America,* vol. 129, no. 4, pp. 2144-2162, 2011.

[31] J. Frankel, M. Magimai-Doss, S. King, K. Livescu, and O. Çetin, "Articulatory feature classifiers trained on 2000 hours of telephone speech," 2007.

[32] R. Arora and K. Livescu, "Multi-view CCA-based acoustic features for phonetic recognition across speakers and domains," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013, pp. 7135-7139: IEEE.

[33] D. Povey *et al.*, "The Kaldi speech recognition toolkit," in *IEEE 2011 workshop on automatic speech recognition and understanding*, 2011, no. EPFL-CONF-192584: IEEE Signal Processing Society.

[34] G. Zhao and R. Gutierrez-Osuna, "Using phonetic posteriorgram based frame pairing for segmental accent conversion," *IEEE/ACM Transactions on Audio, Speech, and Language Processing,* vol. 27, no. 10, pp. 1649-1660, 2019.

[35] B. Sisman, M. Zhang, and H. Li, "A Voice Conversion Framework with Tandem Feature Sparse Representation and Speaker-Adapted WaveNet Vocoder," in *Interspeech*, 2018.

[36] S. H. Mohammadi and A. Kain, "An overview of voice conversion systems," *Speech Communication,* vol. 88, pp. 65-82, 2017.

[37] E. Godoy, O. Rosec, and T. Chonavel, "Voice conversion using dynamic frequency warping with amplitude scaling, for parallel or nonparallel corpora,"

IEEE Transactions on Audio, Speech, and Language Processing, vol. 20, no. 4, pp. 1313-1323, 2012.

[38] X. Tian, S. W. Lee, Z. Wu, E. S. Chng, and H. Li, "An exemplar-based approach to frequency warping for voice conversion," *IEEE/ACM Trans. on Audio, Speech, and Language Processing,* vol. 25, no. 10, pp. 1863-1876, 2017.

[39] H. Kawahara, "STRAIGHT, exploitation of the other aspect of VOCODER: Perceptually isomorphic decomposition of speech sounds," *Acoustical Science and Technology,* vol. 27, no. 6, pp. 349-353, 2006.

[40] M. Morise, F. Yokomori, and K. Ozawa, "WORLD: a vocoder-based high-quality speech synthesis system for real-time applications," *IEICE Trans. on Information and Systems,* vol. 99, no. 7, pp. 1877-1884, 2016.

[41] T. Yoshimura, K. Tokuda, T. Masuko, T. Kobayashi, and T. Kitamura, "Mixed excitation for HMM-based speech synthesis," in *Seventh European Conference on Speech Communication and Technology*, 2001.

[42] B. P. Nguyen and M. Akagi, "Phoneme-based spectral voice conversion using temporal decomposition and Gaussian mixture model," in *IEEE Conference on Communications and Electronics (ICCE)*, 2008, pp. 224-229.

[43] M. Abe, S. Nakamura, K. Shikano, and H. Kuwabara, "Voice conversion through vector quantization," *Journal of the Acoustical Society of Japan (E),* vol. 11, no. 2, pp. 71-76, 1990.

[44] X. Tian, Z. Wu, S. W. Lee, and E. S. Chng, "Correlation-based frequency warping for voice conversion," in *Chinese Spoken Language Processing (ISCSLP), 2014 9th International Symposium on*, 2014, pp. 211-215: IEEE.

[45] E. Helander, H. Silén, T. Virtanen, and M. Gabbouj, "Voice conversion using dynamic kernel partial least squares regression," *IEEE transactions on audio, speech, and language processing,* vol. 20, no. 3, pp. 806-817, 2012.

[46] R. Aihara, T. Nakashika, T. Takiguchi, and Y. Ariki, "Voice conversion based on non-negative matrix factorization using phoneme-categorized dictionary," in *ICASSP*, 2014, pp. 7894-7898.

[47] R. Aihara, T. Takiguchi, and Y. Ariki, "Activity-mapping non-negative matrix factorization for exemplar-based voice conversion," in *ICASSP*, 2015, pp. 4899-4903.

[48] R. Aihara, T. Takiguchi, and Y. Ariki, "Many-to-many voice conversion based on multiple non-negative matrix factorization," in *Interspeech*, 2015.

[49] R. Aihara, T. Takiguchi, and Y. Ariki, "Multiple non-negative matrix factorization for many-to-many voice conversion," *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP),* vol. 24, no. 7, pp. 1175-1184, 2016.

[50] Z. Wu, E. S. Chng, and H. Li, "Exemplar-based voice conversion using joint nonnegative matrix factorization," *Multimedia Tools and Applications,* vol. 74, no. 22, pp. 9943-9958, 2015.

[51] Z. Wu, T. Virtanen, E. S. Chng, and H. Li, "Exemplar-based sparse representation with residual compensation for voice conversion," *IEEE/ACM Transactions on Audio, Speech and Language Processing,* vol. 22, no. 10, pp. 1506-1521, 2014.

[52] G. Zhao and R. Gutierrez-Osuna, "Exemplar selection methods in voice conversion," in *ICASSP*, 2017: IEEE.

[53] R. Takashima, T. Takiguchi, and Y. Ariki, "Exemplar-based voice conversion using sparse representation in noisy environments," *IEICE Trans. on Fundamentals of Electronics, Communications and Computer Sciences,* vol. 96, no. 10, pp. 1946-1953, 2013.

[54] S. Ding, C. Liberatore, and R. Gutierrez-Osuna, "Learning Structured Dictionaries for Exemplar-based Voice Conversion," in *INTERSPEECH*, 2018.

[55] B. Sisman, H. Li, and K. C. Tan, "Sparse representation of phonetic features for voice conversion with and without parallel data," in *Automatic Speech Recognition and Understanding Workshop (ASRU), 2017 IEEE*, 2017, pp. 677-684: IEEE.

[56] S. Aryal and R. Gutierrez-Osuna, "Data driven articulatory synthesis with deep neural networks," *Computer Speech & Language,* vol. 36, pp. 260-273, 2016.

[57] G. Zhao, S. Sonsaat, J. Levis, E. Chukharev-Hudilainen, and R. Gutierrez-Osuna, "Accent Conversion Using Phonetic Posteriorgrams," in *ICASSP*, 2018, pp. 5314-5318: IEEE.

[58] S. Aryal and R. Gutierrez-Osuna, "Reduction of non-native accents through statistical parametric articulatory synthesis," *Journal of the Acoustical Society of America,* vol. 137, no. 1, pp. 433-446, 2015.

[59] D. Felps, H. Bortfeld, and R. Gutierrez-Osuna, "Foreign accent conversion in computer-assisted pronunciation training," *Speech Communication,* vol. 10, no. 51, pp. 920-932, 2009.

[60] Q. Yan, S. Vaseghi, D. Rentzos, and H. Ching-Hsiang, "Analysis and Synthesis of Formant Spaces of British, Australian, and American Accents," *IEEE*

Transactions on Audio, Speech, and Language Processing, vol. 15, no. 2, pp. 676-689, 2007.

[61] M. Huckvale and K. Yanagisawa, "Spoken language conversion with accent morphing," presented at the Proceedings of ISCA Speech Synthesis Workshop, Bonn, Germany, 2007.

[62] S. Aryal, D. Felps, and R. Gutierrez-Osuna, "Foreign accent conversion through voice morphing," presented at the Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH), Lyon, France, 2013.

[63] D. Felps, C. Geng, and R. Gutierrez-Osuna, "Foreign accent conversion through concatenative synthesis in the articulatory domain," *IEEE Trans. on Audio, Speech, and Language Processing,* vol. 20, no. 8, pp. 2301-2312, 2012.

[64] S. Aryal and R. Gutierrez-Osuna, "Accent conversion through cross-speaker articulatory synthesis," in *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 7694-7698: IEEE.

[65] R. J. Turetsky and D. P. Ellis, "Ground-truth transcriptions of real music from force-aligned midi syntheses," 2003.

[66] S. Aryal, D. Felps, and R. Gutierrez-Osuna, "Foreign Accent Conversion through Voice Morphing," in *Interspeech*, 2013, pp. 3077-3081.

[67] T. Hayashi, A. Tamamori, K. Kobayashi, K. Takeda, and T. Toda, "An investigation of multi-speaker training for WaveNet vocoder," in *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2017, pp. 712-718: IEEE.

[68] R. Skerry-Ryan *et al.*, "Towards end-to-end prosody transfer for expressive speech synthesis with tacotron," in *international conference on machine learning*, 2018, pp. 4693-4702: PMLR.

[69] R. C. Rose and I. Arizmendi, "Efficient client–server based implementations of mobile speech recognition services," *Speech communication,* vol. 48, no. 11, pp. 1573-1589, 2006.

[70] A. Hair, P. Monroe, B. Ahmed, K. J. Ballard, and R. Gutierrez-Osuna, "Apraxia world: A speech therapy game for children with speech sound disorders," in *Proceedings of the 17th ACM Conference on Interaction Design and Children*, 2018, pp. 119-131.

[71] Y. He *et al.*, "Streaming end-to-end speech recognition for mobile devices," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 6381-6385: IEEE.

[72] D. Huggins-Daines, M. Kumar, A. Chan, A. W. Black, M. Ravishankar, and A. I. Rudnicky, "Pocketsphinx: A free, real-time continuous speech recognition system for hand-held devices," in *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, 2006, vol. 1, pp. I-I: IEEE.

[73] M. K. Mustafa, T. Allen, and K. Appiah, "A comparative review of dynamic neural networks and hidden Markov model methods for mobile on-device speech recognition," *Neural Computing and Applications,* vol. 31, no. 2, pp. 891-899, 2019.

[74] G. Venkatesh *et al.*, "Memory-efficient Speech Recognition on Smart Devices," *arXiv preprint arXiv:2102.11531,* 2021.

[75] L. Sun, K. Li, H. Wang, S. Kang, and H. Meng, "Phonetic posteriorgrams for many-to-one voice conversion without parallel data training," in *Multimedia and Expo (ICME), 2016 IEEE International Conference on*, 2016, pp. 1-6: IEEE.

[76] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society,* vol. Series B, pp. 267-288, 1996.

[77] S. Chen, C. F. Cowan, and P. M. Grant, "Orthogonal least squares learning algorithm for radial basis function networks," *Neural Networks, IEEE Transactions on,* vol. 2, no. 2, pp. 302-309, 1991.

[78] G. Zhao *et al.*, "L2-ARCTIC: A Non-Native English Speech Corpus," in *Interspeech*, 2018, pp. 2783-2787.

[79] J. Mairal, F. Bach, J. Ponce, G. Sapiro, R. Jenatton, and G. Obozinski, "Spams: A sparse modeling software, v2. 3," *URL http://spams-devel. gforge. inria. fr/downloads. html,* 2014.

[80] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani, "Least angle regression," *The Annals of Statistics,* vol. 32, no. 2, pp. 407-499, 2004.

[81] J. Kominek and A. W. Black, "The CMU ARCTIC speech databases," in *ISCA Workshop on Speech Synthesis*, 2004, pp. 223-224.

[82] D. Felps, C. Geng, M. Berger, K. Richmond, and R. Gutierrez-Osuna, "Relying on critical articulators to estimate vocal tract spectra in an articulatory-acoustic database " in *Interspeech*, 2010.

[83] S. Panchapagesan and A. Alwan, "Frequency warping for VTLN and speaker adaptation by linear transformation of standard MFCC," *Computer Speech & Language,* vol. 23, no. 1, pp. 42-64, 2009.

[84] M. Pitz and H. Ney, "Vocal tract normalization equals linear transformation in cepstral space," *IEEE Trans. on Speech and Audio Processing,* vol. 13, no. 5, pp. 930-944, 2005.

[85] T. Claes, I. Dologlou, L. ten Bosch, and D. Van Compernolle, "A novel feature transformation for vocal tract length normalization in automatic speech recognition," *IEEE Transactions on Speech and Audio Processing,* vol. 6, no. 6, pp. 549-557, 1998.

[86] M. Pitz and H. Ney, "Vocal tract normalization as linear transformation of MFCC," in *Eighth European Conference on Speech Communication and Technology*, 2003.

[87] T. Toda, Y. Ohtani, and K. Shikano, "One-to-many and many-to-one voice conversion based on eigenvoices," in *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, 2007, vol. 4, pp. IV-1249-IV-1252: IEEE.

[88] X. Cui and A. Alwan, "Adaptation of children's speech with limited data based on formant-like peak alignment," *Computer speech & language,* vol. 20, no. 4, pp. 400-419, 2006.

[89] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online learning for matrix factorization and sparse coding," *Journal of Machine Learning Research,* vol. 11, no. Jan, pp. 19-60, 2010.

[90] D. Felps and R. Gutierrez-Osuna, "Developing objective measures of foreign-accent conversion " *IEEE Transactions on Audio, Speech, and Language Processing,* vol. 18, no. 5, pp. 1030-1040, 2010.

[91] S. Ding, G. Zhao, C. Liberatore, and R. Gutierrez-Osuna, "Improving Sparse Representations in Exemplar-Based Voice Conversion with a Phoneme-Selective Objective Function," in *INTERSPEECH*, 2018, pp. 476-480.

[92] R. Aihara, T. Takiguchi, and Y. Ariki, "Parallel Dictionary Learning for Voice Conversion Using Discriminative Graph-Embedded Non-Negative Matrix Factorization," in *INTERSPEECH*, 2016, pp. 292-296.

[93] K. Engan, S. O. Aase, and J. H. Husoy, "Method of optimal directions for frame design," in *IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, 1999, vol. 5.

[94] J. H. Ward Jr, "Hierarchical grouping to optimize an objective function," *Journal of the American Statistical Association,* vol. 58, no. 301, pp. 236-244, 1963.

[95] D. J. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series," in *KDD workshop*, 1994, vol. 10, no. 16, pp. 359-370: Seattle, WA.

[96] B. Sisman, M. Zhang, and H. Li, "Group sparse representation with wavenet vocoder adaptation for spectrum and prosody conversion," *IEEE/ACM Transactions on Audio, Speech, and Language Processing,* vol. 27, no. 6, pp. 1085-1097, 2019.

[97] S. Buchholz and J. Latorre, "Crowdsourcing preference tests and how to detect cheating," in *Interspeech*, 2011, pp. 3053-3056.

[98] M.-Y. Hwang and X. Huang, "Subphonetic modeling with Markov states-Senone," in *Acoustics, Speech, and Signal Processing, 1992. ICASSP-92., 1992 IEEE International Conference on*, 1992, vol. 1, pp. 33-36: IEEE.

[99] K. Kintzley, A. Jansen, and H. Hermansky, "Event selection from phone posteriorgrams using matched filters," in *Twelfth Annual Conference of the International Speech Communication Association*, 2011.

[100] R. Tibshirani, M. Saunders, S. Rosset, J. Zhu, and K. Knight, "Sparsity and smoothness via the fused lasso," *Journal of the Royal Statistical Society: Series B (Statistical Methodology),* vol. 67, no. 1, pp. 91-108, 2005.

[101] P. C. Nguyen, O. Takao, and M. Akagi, "Modified restricted temporal decomposition and its application to low rate speech coding," *IEICE Transactions on Information and Systems,* vol. 86, no. 3, pp. 397-405, 2003.

[102] T. Virtanen, "Monaural sound source separation by nonnegative matrix factorization with temporal continuity and sparseness criteria," *IEEE TASLP,* vol. 15, no. 3, pp. 1066-1074, 2007.

[103] B. S. Atal, "Efficient coding of LPC parameters by temporal decomposition," in *ICASSP*, 1983, vol. 8, pp. 81-84: IEEE.

[104] S.-J. Kim and Y.-H. Oh, "Efficient quantisation method for LSF parameters based on restricted temporal decomposition," *Electronics Letters,* vol. 35, no. 12, pp. 962-964, 1999.

[105] R. Tibshirani and J. Taylor, "The solution path of the generalized lasso," PhD, Stanford University, 2011.

[106] Amazon. (2017, 1 March). *Amazon Mechanical Turk*. Available: www.mturk.com

[107] S. Ding *et al.*, "Golden Speaker Builder, an interactive tool for pronunciation training," *Speech Communication,* 2019.

[108] R. Hincks, "Speech technologies for pronunciation feedback and evaluation," *ReCALL,* vol. 15, no. 1, pp. 3-20, 2003.

[109] K. Nagano and K. Ozawa, "English speech training using voice conversion," in *Proc. of Int. Conf. on Spoken Language Processing (ICSLP)*, Kobe, Japan, 1990.

[110] M. Peabody and S. Seneff, "Towards automatic tone correction in non-native mandarin," in *International Symposium on Chinese Spoken Language Processing*, 2006, pp. 602-613: Springer.

[111] K. Probst, Y. Ke, and M. Eskenazi, "Enhancing foreign language tutors–-in search of the golden speaker," *Speech Communication,* vol. 37, no. 3, pp. 161-173, 2002.

[112] D. Felps, H. Bortfeld, and R. Gutierrez-Osuna, "Foreign accent conversion in computer assisted pronunciation training," *Speech Communication,* vol. 51, no. 10, pp. 920-932, 2009.

[113] K. Hirose, F. Gendrin, and N. Minematsu, "A pronunciation training system for Japanese lexical accents with corrective feedback in learner's voice," in *Eighth European Conference on Speech Communication and Technology*, 2003.

[114] K. Hirose, F. Gendrin, and N. Minematsu, "A pronunciation training system for Japanese lexical accents with corrective feedback in learner's voice," presented at the Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH), 2003.

[115] M. Peabody and S. Seneff, "Towards Automatic Tone Correction in Non-native Mandarin," in *Chinese Spoken Language Processing*, 2006, pp. 602-613.

[116] M. P. Bissiri and H. R. Pfitzinger, "Italian speakers learn lexical stress of German morphologically complex words," *Speech Communication,* vol. 51, no. 10, pp. 933-947, 2009.

[117] M. P. Bissiri, H. R. Pfitzinger, and H. G. Tillmann, "Lexical Stress Training of German Compounds for Italian Speakers by means of Resynthesis and Emphasis," presented at the Proceedings of the 11th Australian International Conference on Speech Science & Technology, University of Auckland, New Zealand, 2006.

[118] A. De Meo, M. Vitale, M. Pettorino, F. Cutugno, and A. Origlia, "Imitation/self-imitation in computer-assisted prosody training for Chinese learners of L2 Italian," presented at the Proceedings of the 4th Pronunciation in Second Language Learning and Teaching Conference, 2012.

[119] E. Pellegrino and D. Vigliano, "Self-imitation in prosody training: A study on Japanese learners of Italian," presented at the Proceedings of the Workshop on Speech and Language Technology in Education (SLaTE), Leipzig, Germany, 2015.

[120] K. Nagano and K. Ozawa, "English Speech Training Using Voice Conversion," presented at the First International Conference on Spoken Language Processing (ICSLP), Kobe, Japan, 1990.

[121] J. Kominek and A. W. Black, "The CMU Arctic speech databases," presented at the Fifth ISCA Workshop on Speech Synthesis, 2004.

[122] M. McAuliffe, Michaela Socolof, Sarah Mihuc, Michael Wagner, and Morgan Sonderegger, "Montreal Forced Aligner: Trainable Text-Speech Alignment Using Kaldi," presented at the Interspeech, 2017.

[123] J. Kreiman and G. Papcun, "Comparing discrimination and recognition of unfamiliar voices," *Speech Communication,* vol. 10, no. 3, pp. 265-275, 1991.

[124] B. W. Pelham and H. Blanton, *Conducting research in psychology: Measuring the weight of smoke*. Cengage Learning, 2012.

[125] M. J. Munro and T. M. Derwing, "Foreign accent, comprehensibility, and intelligibility in the speech of second language learners," *Language learning,* vol. 45, no. 1, pp. 73-97, 1995.

[126] P. Martin, "WinPitch LTL II, a multimodal pronunciation software," in *InSTIL/ICALL Symposium 2004*, 2004.

[127] Genevalogic. (2006). *SpeedLingua*. Available: http://home.speedlingua.com

[128] R. Aihara, T. Nakashika, T. Takiguchi, and Y. Ariki, "Voice conversion based on non-negative matrix factorization using phoneme-categorized dictionary," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 7894-7898: IEEE.

[129] T. Park and G. Casella, "The bayesian lasso," *Journal of the American Statistical Association,* vol. 103, no. 482, pp. 681-686, 2008.

[130] M. Fadaee and C. Monz, "Back-Translation Sampling by Targeting Difficult Words in Neural Machine Translation," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 436-446.

[131] V. C. D. Hoang, P. Koehn, G. Haffari, and T. Cohn, "Iterative back-translation for neural machine translation," in *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, 2018, pp. 18-24.

[132] R. Sennrich, B. Haddow, and A. Birch, "Improving neural machine translation models with monolingual data," *arXiv preprint arXiv:1511.06709,* 2015.

[133] J. Ni, Q. Qiu, and R. Chellappa, "Subspace interpolation via dictionary learning for unsupervised domain adaptation," in *CVPR*, 2013, pp. 692-699: IEEE.

[134] J. Mairal and B. Yu, "Supervised feature selection in graphs with path coding penalties and network flows," *Journal of Machine Learning Research,* vol. 14, no. 8, 2013.

[135] C. J. Rozell, D. H. Johnson, R. G. Baraniuk, and B. A. Olshausen, "Sparse coding via thresholding and local competition in neural circuits," *Neural computation,* vol. 20, no. 10, pp. 2526-2563, 2008.