Towards Integrated Acoustic Models for Speech Synthesis

Prasanna Kumar Muthukumar

CMU-LTI-15-019

Language Technologies Institute,
School of Computer Science,
Carnegie Mellon University,
5000 Forbes Avenue,
Pittsburgh, PA 15213.
www.lti.cs.cmu.edu

Thesis Committee:
Alan W Black (chair)
Bhiksha Raj
Richard Stern
H. Timothy Bunnell

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
in Language and Information Technologies

Carnegie Mellon University

Language Technologies Institute
School of Computer Science

Towards Integrated Acoustic Models for Speech Synthesis

Prasanna Kumar Muthukumar

Prasanna Kumar Muthukumar
Towards Integrated Acoustic Models for Speech Synthesis
Advisor: Alan W Black

Committee: Bhiksha Raj, Richard Stern, and H. Timothy Bunnell
Carnegie Mellon University
School of Computer Science
Language Technologies Institute

Dedicated to my mother, Suganthi who supported me the most throughout my years in grad school but unfortunately never got a chance to see this thesis in its final form.

Abstract

All Statistical Parametric Speech Synthesizers consist of a linear pipeline of components. This view means that the synthesizer consists of a top-down structure where data fed into the synthesizer goes to front-end, then to the prediction algorithm, then to the waveform generation, and so on until the speech is finally constructed. Each component in this pipeline naïvely receives a stream of numbers from the preceding component, and spits out a stream of numbers for the next one in line, with little to no knowledge of what happens in the larger scheme of the pipeline. In this thesis, I argue against this “Markovian” structure, and instead propose the idea of an *Integrated* structure. In an integrated structure, every component in the system influences, and is in turn influenced by every other component in the system. This thesis describes four sets of experiments that move towards this idea. The first involves using lexical information to improve waveform generation algorithms. The second tries to increase the interaction between prediction algorithms and waveform generation. The third is an attempt to derive phonemes and phonetic information automatically from the speech rather than from the text. The last, and probably hardest, describes an idea for an evaluation metric that pays attention to multiple components of the synthesizer, rather than focusing on just a single one.

Contents

1 Speech Synthesis is not a solved problem

Computers that speak are no longer a novelty. We today possess assistants such as Siri and Cortana, GPS systems that chastize us for making that errant right turn, and IBM Watson which can answer questions better than the best trivia aficionado. With the level of near ubiquity that speech synthesis has attained, we might be tempted to ask if the problem of getting a computer to produce human-like speech has been solved. The answer to this question is of course a loud, resounding, unequivocal ... yes but not really.

If the aim of the speech synthesizer is to exist as a mere information provider or naïve automated receptionist, the state of the art speech synthesis systems are more than capable of doing so. In fact, I would go out on a limb and claim that the speech synthesis technology for this existed more than a decade ago. As early as the late 1990's, computers had progressed to the point where synthesizers could snip and paste segments of speech from large corpora to create speech that could fool humans into thinking that the utterances were pre-recorded. *Understandable* speech synthesis has existed for even longer. The synthesizer built for Prof. Stephen Hawking is extremely outdated yet is highly intelligible, and allows the eminent physicist to delight the world with his wit.

Today, all one needs to do to build an excellent speech synthesizer capable of conveying information is to first go out and find a voice talent with a suitable voice. Once the voice talent is found, he or she is then locked in a recording studio for several hours while their speech is carefully recorded. The larger the amount of speech thus recorded, the better the quality of the synthesizer will be. Perfectly understandable voices can be built with a mere 30 minutes of speech. High quality voices will require data of 4-5 hours or more. Once this recording is complete, a *unit-selection* synthesizer, which creates new speech by slicing and concatenating chunks of the speech recordings, can be built. This does not require a great deal of computational power either. There exist open-source synthetic-voice building toolkits that are both well-documented and computationally efficient. Running these on even the most expensive of the Amazon cloud's virtual servers will cost less than 20$, no great expense even for the poor computer science graduate student. Once all these steps are done, one is left with a system that is perfectly capable of speaking in highly natural sounding voice that can fool the casual listener into thinking that they are listening to pre-recorded speech.

As if all this evidence was not overwhelming enough, there currently exists an obvious trend in the speech synthesis community where systems use larger and larger amounts of recording data for training. In the last few years, improvements in speech synthesis quality caused by the ability to train from larger corpora have vastly outpaced the improvements caused by more advanced algorithms. In fact, the speech synthesizer of today is not radically different from the speech synthesizer of 10 years ago. There are a few notable differences but the ideas are fundamentally unchanged. All of these are indications of a field which is close to its saturation point where the most intimidating research problems have been solved and all that is left is the grunt work of engineering. Extrapolating from current trends, we can even make a prediction that the best system in 2025 will probably be similar to best system of today , but it will be an all consuming behemoth that keeps sucking in speech data much like Hawking's black holes.

The wise among us will already be asking if it is still worth investing money, resources, and graduate student years (and tears) into this problem. The answer to this question though is a loud, resounding, uniequivocal yes, and with no reservations whatsoever.

Constraining the modern speech synthesizer to merely exist as an information provider or naïve receptionist would be a travesty both of computer science's capabilities and of our own imaginations. We need speech synthesizers that are capable of full human conversations, not simple dialog turns. We need speech synthesizers that can emote, not robotic servants that simply utter catch phrases. Human speech is capable of delighting us, electrifying us, or even terrifying us. Why should we not expect synthesizers to do the same?

To expect to reach this level of conversational speech synthesis merely by throwing data at the problem is about as absurd as trying to empty Lake Erie with an ice cream scoop. Large as the amounts of data we use today might seem, they are dwarfed by the space of possible styles and emotions in normal human conversation. To get an idea of the magnitude of this range, let us assume that our goal was to build a synthesizer whose only job was to read books of classic fiction with the right style. The number of possible scenarios for conversations in these books can range from proto-linguistic sounds uttered by early hominids to conversations between life forms who have transcended the limits of the human body. It is safe to assume that the amount of speech data that can be recorded will *never* span the breadth of imagination of every fiction writer combined.

Reaching human conversation quality synthesis will either require unimaginably large amounts of training data in a bewildering variety of possible styles, or newer training algorithms that are extremely efficient in learning from smaller amounts of data. There is no guarantee that training current synthesizers with humongous quantities of data will result in high quality conversational quality synthesis. However there *is* a guarantee that an algorithm for full conversational speech exists. After all, humans have the algorithm in the their heads. (Although one might suspect otherwise after sitting through certain LTI colloquium talks).

In the quest for true conversational speech synthesis, it is necessary to first solve the intermediate problem of building fully natural sounding voices with limited amounts of data. It is possible today to build synthesized voices with 4-5 hours of carefully recorded training data which can be mistaken for pre-recorded speech. With an hour or less of training data though, the synthesizers sound substantially worse, and it is this exact scenario that we shall be tackling in this thesis. The hope is that by searching for algorithms that can produce natural speech despite low-resources, we might someday stumble upon that holy grail algorithm that can produce full conversational speech from a limited set of examples. This thesis therefore describes my efforts at making synthesized speech more natural by examining one of the fundamental problems in the modern speech synthesizer: *fragmentation*.

2 The Need for Integrated Acoustic models

The modern speech or language processing system is usually depicted as a pipeline, a linear stream of components daisy chained together to perform some function. This view of the system arises from the engineering approach to solving complex problems by breaking them down into smaller, more manageable, and easier to optimize parts. The resulting system balkanization has produced great progress in the fields of speech and language technology, and without this breakdown it would be hard to imagine how to approach many of these research problems. However, considering the maturity of this research field, it is worth going back to re-examine this *Markovian*[*] view of the speech and language processing to see if it is still appropriate.

While nearly all speech and language systems suffer from the problem of excessive fragmentation, in this thesis I will only try to address this problem in speech synthesizers. The modern speech synthesizer is composed of a text processing front-end and a waveform generating back-end. The front-end performs an extremely complex series of tokenizations, normalizations, and decompositions to convert input text into a set of phonemes and stress patterns. To highlight the difficulty of this process, consider the number "1990". This number can either be spoken as "Nineteen-ninety" in "July 26, 1990" or "One thousand nine hundred and ninety" in "1990 dollars" depending on the context in which it occurs. The afore-mentioned example is fairly easy for humans to solve, but there are innumerable instances where it is equally hard for humans. Consider the word "Cthulu". The word originated in print, and so there is no standard pronunciation of it. The synthesizer cannot choose to skip the word either. It *has* to say the word even if it has to take a guess. These problems can be greatly mitigated by using sophisticated hand-written rules which can cover the majority of cases even for languages like English which have a lot of counter-intuitive pronunciations (cough, tough, though, through). Learning these rules using modern machine learning algorithms has been less successful though. Hard enough as this problem is in English, it gets much harder when dealing with other languages which have fewer resources.

[*] A Markov process is one in which the current state is independent of earlier states conditioned upon its immediate past states. It is possible for our pipeline to be non-Markovian but still disjoint. However, I would still like to use the term Markovian here because it makes the problem more immediately obvious to the scientist.

While the front-end is undeniably important, for this thesis I focus on the back-end. The back-end is no less complex but is much easier to work with in a machine learning framework. The back-end of the modern speech synthesizer performs the important role of generating speech based on the phonemes provided by the front-end.

2.1 A Brief History of Speech Synthesis Back-end

There are many schools of thought when it comes to waveform generation given a list of phonemes. The two most popular (and arguably most successful) are *Concatenative Synthesis*(Hunt and Black, 1996) and *Statistical Parametric Synthesis*(Zen et al., 2009). Other techniques do exist but are either interesting only as historical context, or exist on the fringes worked on by the most iconoclastic of researchers.

Concatenative systems became popular in the 1990's with the advent of fast computers and cheap data storage. Large corpora could easily be stored and indexed in a computer's disks. The corpus is then segmented into *units*, which vary in length between that of sub-phone segments to that of an entire word. Speech is constructed by concatenating units corresponding to the list of phonemes input to the back-end. The hardest problem in these systems is the selection of appropriate units. Good selection strategies coupled with a reasonably large corpus will produce speech almost indistinguishable from the original. Smaller corpora or bad selection techniques will produce speech disjoint like the mythical sphinx which was composed of the head of a human, the body of a lion, and the wings of a bird.

Statistical Parametric Speech Synthesis was born out of the marriage between modern machine learning techniques and much older *parametric* synthesis techniques. Parametric synthesis tries to produce speech by creating a model of the human speech production apparatus. While the magician can impress his or her audience by producing objects out of thin air, air is a luxury unafforded to the parametric synthesizer. Instead, speech has to be conjured out of the random noise spewed by the computer. A classic example of the parametric synthesizer is Dennis Klatt's DECTalk(Klatt, 1987). These kind of models have parameters that can be controlled, akin to the many muscles controlling the human speech apparatus. In the same way that the muscles can morph the mouth and tongue in a bewildering variety of shapes, the parameters of the parametric synthesizer's models can be changed to produce the appropriate phone of speech. In classical parametric synthesis, the parameters were discovered by experts through extensive hand tuning. This proves to be exceedingly difficult though, as evidenced by Coleman and Slater, 2001 where the authors mention that even for them, it was easier to look up the parameters in the tables from Klatt's papers rather than attempt to estimate these from natural speech. These early parametric speech synthesizers were based on studies of human speech perception. Studies were conducted to see what aspects of a signal caused it to be speech-like, and then synthesizer models would try to replicate those. *Formant synthesis* would be a good example of this. The simplest formant synthesizers work by producing pure tones at three or four frequencies that correspond to the peaks of the spectrum for any particular sound. This tricks the ear into thinking that is is hearing a particular phone of speech.

There is yet another class of synthesizers where the focus is more on understanding human speech production rather than building Text-To-Speech (TTS) systems. This thesis focuses on the TTS aspect, but research into models of human speech production is still active and thriving. This field is called *Articulatory* speech synthesis to emphasize that it tries to synthesize speech by explicitly modeling the articulators of human speech. Birkholz et al., 2006 is an example of this. Articulatory synthesis rarely produces speech that is on par in quality with modern TTS systems, but is nevertheless important because of the insights it provides into human speech production.

With the advent of modern machine learning techniques, it was soon realized that rather than hand set the parameters of the models in parametric synthesis, these could be estimated statistically from data. This gave rise to *Statistical* Parametric Speech Synthesis. There have been attempts to use the same models that Klatt and others built for classic parametric synthesis in statistical parametric synthesis, most notably in Anumanchipalli et al., 2010. But these models are less successful owing to joint constraints of being a feasible model of human speech production, as well as working well in a statistical framework. More success has been found by regressing to a more simplistic source-filter approximation similar to one described in Fant, 1971. The exact parameterization varies from synthesizer to synthesizer, but is usually either Mel Cepstra(Tokuda et al., 1994), Line Spectral Pairs(Itakura, 1975), STRAIGHT spectrum(Kawahara, 2006), or some combination of these.

2.2 The pipeline of the Statistical Parametric Speech Synthesizer

The pipeline of the modern statistical parametric speech synthesizer looks like Figure 2.1. This thesis provides a basic description of each component in the pipeline, and describes my efforts at integrating the various components into a unified monothic acoustic model.

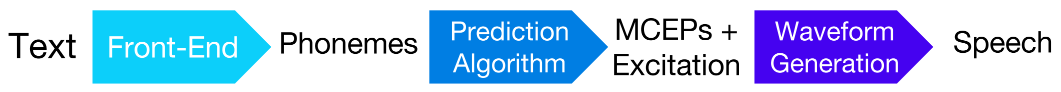

Fig. 2.1: Speech synthesis pipeline

The input to the pipeline is text. The front-end of the system, which for the purposes of this thesis has been abstracted to a single block, extracts the lexical information such as the underlying tokens, the stress patterns, and the phonemes. These are used as input features by the prediction algorithm to predict parameters for the waveform generation component. The prediction algorithm is usually a Classification And Regression Tree (CART)(Black, 2006), a Hidden Markov Model (HMM)(Zen et al., 2009), and sometimes a Gaussian Process. The waveform generation algorithm uses the predicted parameters to construct the speech signal. The waveform generation algorithm is typically either the Mel-Log Spectral Approximation filter(Imai, 1983) in the case of MCEPs, the LPC filter in the case of Linear Predictive Coding coefficients(Makhoul, 1975), or the STRAIGHT algorithm(Kawahara, 2006) in the case of the magnitude spectrum. Various other important components exist in the synthesizer but we will omit them to focus on the parts most essential to this thesis.

2.3 Steps toward Integration

The pipeline-like structure described in the earlier section does not exist in the human speech production system. The various parts of the human speech production apparatus are deeply interconnected with each responding to the feedback from other components. Obvious examples of this are the Lombard effect(Lombard, 1911) and other speech in noise(Langner and Black, 2005). When the human listening mechanism notices the presence of high levels of noise, the brain adjusts the articulators of speech production to make the speech sound more intelligible. This kind of dynamic feedback and adaptation is absent in the modern speech synthesizer. By moving towards a more integrated model of synthesis, we could make great strides in improving naturalness and ultimately making synthesized speech sound more human-like.

Intuitively speaking, the hardest components to integrate together would be the two furthest ones in the pipeline. As the expression goes, one should eat the metaphorical frog first. So, the ideas in Chapter 3 describe a technique for making use of front-end information in improving the excitation modeling in the waveform generation section. The waveform generation algorithm typically assumes a source-filter formulation of the speech signal. The source part of the speech signal is supposed to represent the sound produced by the vocal cords. We model this using the Liljencrants-Fant model. The fitting process for this model is usually treated as agnostic to the underlying lexical content in the speech signal. The technique we describe in this chapter uses the synthesizer to make an informed guess on where the fitting process should be initialized.

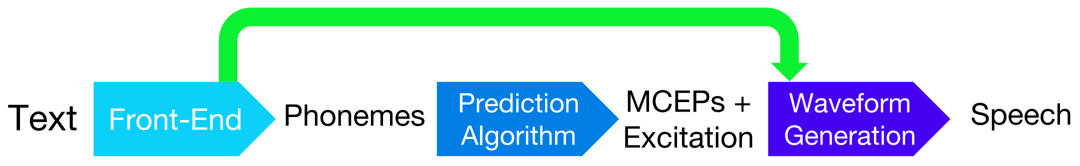

Fig. 2.2: Integrating waveform generation with lexical content

Components that are interconnected are not necessarily connected in the most optimal way. Chapter 4 talks about various ways in which the prediction algorithms used in synthesis can be tied together better with the parameters of the speech model in parametric synthesis.

The prediction algorithms used in Statistical Parametric Speech Synthesis today come in large part from contributions of machine learning community who have little to no knowledge of speech or audio. The waveform generation algorithms on the other hand come from the speech signal processing community who possess a deep understanding of the nature of the speech signal but rarely influence the development of modern machine learning techniques. The result of this reduced interaction between the two communities is that the techniques and algorithms developed by one do not necessarily work well with the other. Speech signal processing algorithms for instance produce parameters are often fairly high dimensional, real or complex valued, and non-sparse. Machine learning algorithms on the other hand often expect the data to be binary with strong assumptions about the independence of data points. The gap between the two has been more detrimental to speech synthesis than other speech research communities because synthesis is unique in being a regression problem. Chapter 4 talks about how these two essential components of the pipeline can be bridged together using an additional post-filter component. While post-filters might appear to be an unelegant solution to a complicated problem, the chapter also describes a more elegant solution to jointly train the Prediction algorithm and the post-filter together.

While most of this thesis has focused on improving the synthesis back-end, we must remember the old computer science adage of GIGO: Garbage In Garbage Out. The

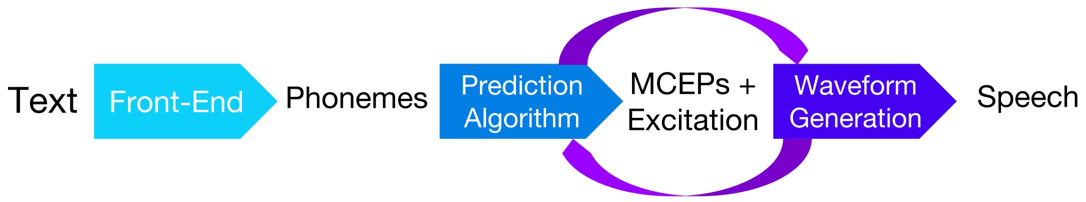

Fig. 2.3: Tying together prediction and waveform generation

quality of the speech produced by the synthesis back-end will always be restricted by the quality of the phonemes it receives as input. Chapter 5 shows the results of preliminary experiments in automatically deriving phonemes for a corpus that is deeply rooted in synthesis techniques rather than being derived from linguistics.

Synthesizers need a way to segment the speech into manageable units that can be either stored in the case of unit-selection systems, or modeled in the case of statistical parametric synthesizers. This segmentation is usually based on the phonemes of a language that are obtained from the text. The phonemes are then used to derive phones of speech through forced alignment of the phonemes to the speech signal. The phonemes of a language are always defined by linguists who expend plenty of time and effort in carefully studying the language. Despite this, the phonemes thus created are still inadequate or sometimes incorrect owing the sheer difficulty in this task. The problem gets exponentially harder when dealing with languages with limited resources. Chapter 5 describes a technique to derive these automatically by transforming the speech signal into a representation that is designed to highlight the phone specific characteristics of the signal. By clustering these features appropriately, we can derive a new phonemic representation of the speech which is more suitable to the task of speech synthesis. This technique can be applied with zero knowledge of the characteristics of the language.

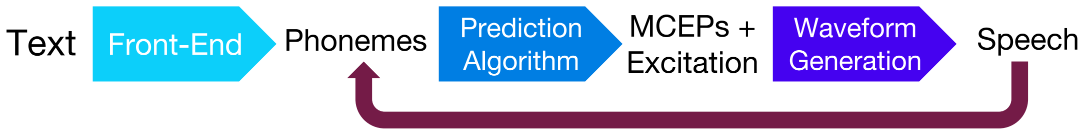

Fig. 2.4: Deriving phonemes from speech

Most of the work in this thesis talks about approaches that have rarely been explored in the past. While this makes the research ideas particularly interesting, this also highlights the problematic absence of a *good* objective metric, especially when working outside the confines of mainstream speech synthesis research. Chapter 6 explains the problems with current objective metrics in speech synthesis, and proposes a new objective metric that measures end-to-end speech synthesis quality rather than focusing on specific components.

Objective metrics available today focus on one end or the other of the speech synthesis pipeline. Word Error Rate focuses on the *text* part of the synthesizer caring only about intelligibility with little attention paid to naturalness or similarity to the speaker. Mel Cepstral Distortion, on the other hand, obsessively looks for similarity between the synthesized and natural speech at the raw signal level with little leeway for alternate and perfectly valid ways of speaking the same utterance. Chapter 6 proposes the building a general model of natural human speech by training HMMs on samples of speech from several different speakers. It also describes the use of auditory models as a means of ensuring that the metric thus used reflects the perception of human listeners.

2.4 Could Deep Learning be a possible solution?

Deep learning proponents will argue that the right solution to integrated models is to throw away every component, and create a massive neural network with text and speech on either end. While there may be some merit to these claims, there is little evidence at this point of time that current training algorithms are capable of doing this. Even in speech recognition research, where deep learning has made tremendous improvements in system performance, an end-to-end neural network does not perform on par with systems that have more isolated components(Graves and Jaitly, 2014). That being said, there is no theoretical reason that prevents neural networks from being capable of this. So we only suggest a cautious skepticism rather than a dogmatic rejection.

3 Integrating Excitation Models with the TTS Front-End

Of all the components in the pipeline of the speech synthesizer, the component that most influences the naturalness of the synthesized speech is the waveform generation section. Bad waveform generation algorithms have haunted the dreams of many a pronunciation model researcher as any errors caused by these models can easily nullify even the greatest of improvements in the front-end of the synthesizer. Excellent waveform generation models might not be sufficient for natural sounding speech, but they certainly are necessary.

A classic joke involves a farmer asking a physicist for help in raising chickens faster and better. The physicist provides a solution, but with the unfortunate requirement that chickens be perfectly spherical, uniformly dense, and live in a vacuum. The situation of the speech synthesis researcher mirrors that of the farmer when asking the signal processing expert for help in dealing with speech signals. The characteristics of the speech signal defy many of the fundamental assumptions made by standard signal processing textbooks. Speech is not stationary. Subsequent samples of speech are not independent as the occurrence of a word from a particular topic typically indicates that several more words (or often the same word) from that topic are likely to follow it. Nor is speech identically distributed; the characteristics of vowels are wildly different from those of fricatives for instance. This non-IID (Independent and Identically Distributed) nature of the speech signal makes it impossible to apply the vast majority of algorithms in machine learning and statistics.

An additional complication is that the noise present in speech signals is neither additive, nor white, and certainly not Gaussian. The most common type of noise in speech signals is either convolutional due to reverberation, or is speech from other speakers which tends to have a nearly identical distribution. The speech signal problem almost seems as if it was created out of malice towards the signal processing expert.

There has been a tremendous amount of research into speech signal processing though, especially at Bell labs owing to the requirements of the telephone industry. Many simple and elegant solutions have been developed to decompose and reconstruct speech for telephone applications, and these have made their way into all aspects of speech technology. The two most common analysis techniques, Linear Predictive Coding(Makhoul, 1975), and Mel Frequency Cepstral Coefficients(Davis and Mermelstein, 1980) are derived from work originally done for speech coding in telephones. Newer techniques like Perceptual Linear Predictive Coding (PLPs)(Hermansky, 1990) and Power Normalized Cepstral Coefficients (PNCCs)(Kim and Stern, 2012) were explicitly designed for non-speech coding tasks, but these still owe a great deal of credit to the early work done by telephone channel researchers.

The waveform generation algorithms in speech synthesis too are tremendously influenced by speech coding. The most commonly used parameterization is still the Mel Cepstrum albeit using Keiichi Tokuda's extraction method (Tokuda et al., 1994). Linear Predictive Coding and its variants are also relatively popular even if they have died out from the speech recognition community. LPC offers little noise-robustness, but this is not a problem for speech synthesis as current datasets are usually quite clean. LPC has the advantage of being much more computationally efficient which makes it attractive when building low-footprint synthesizers.

The major difference between speech analysis algorithms for synthesis as opposed to every other speech processing task is the necessity of having a sophisticated excitation model. Speech analysis algorithms typically use the source-filter formulation defined in Fant, 1971 where the filter represents the vocal tract and the source represents the excitation of the vocal cords, the glottal flow. Most speech applications can get away with disregarding the source model, and focusing on the vocal tract. Synthesis however needs to model the entire speech signal as the human ear will easily notice the smallest discrepancies in the speech signal.

The particular source model that we will be considering in this thesis is the Liljencrants-Fant model(Fant et al., 1985), LF model in short. There have been numerous other models proposed each with its own set of advantages and disadvantages. Fujisaki and Ljungqvist, 1986 has a comprehensive list of the many different models that have been created over the years. I do not make the argument that the LF model is the correct model of the glottal source. However it certainly is the most popular, and the technique described in this chapter is not specific to the LF model.

3.1 The Liljencrants-Fant Model

The LF model was developed in the 1980's to model the glottal flow *derivative* (the excitation is associated with the derivative of the glottal flow and not the flow itself). The model itself consists of the following equation:

$$e(t) = \begin{cases} E_0 e^{\alpha t} sin(\omega_g t) & t < T_e \\ \frac{-E_0}{\varepsilon T_a}.[e^{-\varepsilon(t-T_e)} - \varepsilon e^{(T_c - T_e)}] & T_e < t < T_c \end{cases}$$

Figure 3.1 shows the shape of the LF model for typical values of the parameters. T_p, T_e, T_a, T_c are explained in the figure. E_0, α and ε can be determined from the positive peak. The glottal frequency ω_g can be determined from the fundamental period T_0.

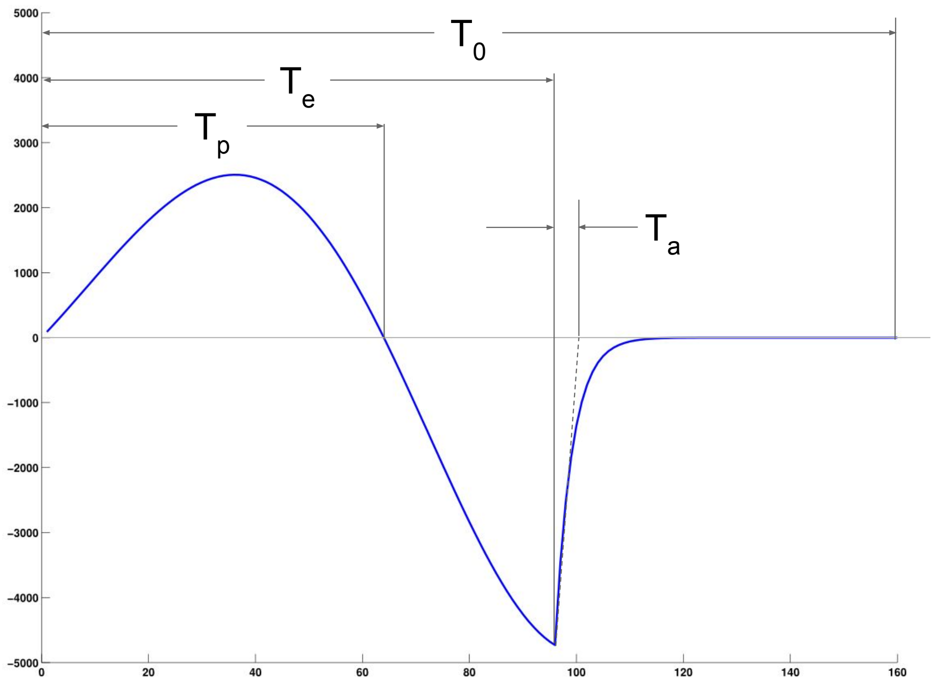

Fig. 3.1: The LF model

Before we throw the LF model head-first into the excitation modeling problem, we must remember that the LF model is intended to work with the glottal flow derivative. As mentioned earlier, we use a source-filter formulation to model the speech signal, and ideally the residual between the filter and the speech signal should resemble the true glottal flow. This however is not the case when Mel Cepstra or LPCs are used as the filter. The residual is close but not identical to the true glottal flow derivative. It is however possible to slightly modify the LPC extraction procedure to get a residual that is much closer to the true glottal flow.

3.2 Iterative Adaptive Inverse Filtering

Iterative Adaptive Inverse Filtering was first described in Alku, 1992. The technique works by making an initial estimate of the vocal tract spectrum, using this to make an initial estimate of the glottal source spectrum, and then iteratively refining its estimates of each making use of estimates of the other. The final glottal source signal thus obtained is much more accurate than could have been obtained in a single pass over the speech signal. The specific technique that was used is more or less identical to Raitio et al., 2011. A block diagram of this procedure is shown in Figure 3.2.

We take the LPC parameters and the residual from the dark magenta blocks in Figure 3.2. The LPC parameters represent the vocal tract, and the residual represents the source signal. The LPC filter coefficients are converted to Line Spectral Pairs(LSPs) (Itakura, 1975) since they are more suited to quantization and interpolation.

3.3 Fitting LF model parameters

There are numerous ways of fitting the LF model to the residual signal. Cabral et al., 2007 is a good example of this. In an earlier work, we tried fitting the model in the frequency domain(Black et al., 2011), however in later experiments, we found it better to do the fitting in the time domain instead.

The first problem in fitting is to identify the locations of the pitch pulses. This is done by first windowing the IAIF residual signal into 70ms frames with a 5ms frame shift. This size of frame is larger than usual, but is necessary to determine if the frame is voiced. A model LF shape is generated by using default values for the LF model. This shape is then convolved (in reverse) with the frame in question. The result of this convolution indicates the presence or absence of pitch pulses. The fundamental frequency F0 can also be estimated based on the locations and spread of the pulses.

Once a potential pitch pulse is identified, the default shape is centered on the pulse, and the temporal parameters T_p, T_e, T_a are adjusted to fit the residual pitch pulse better

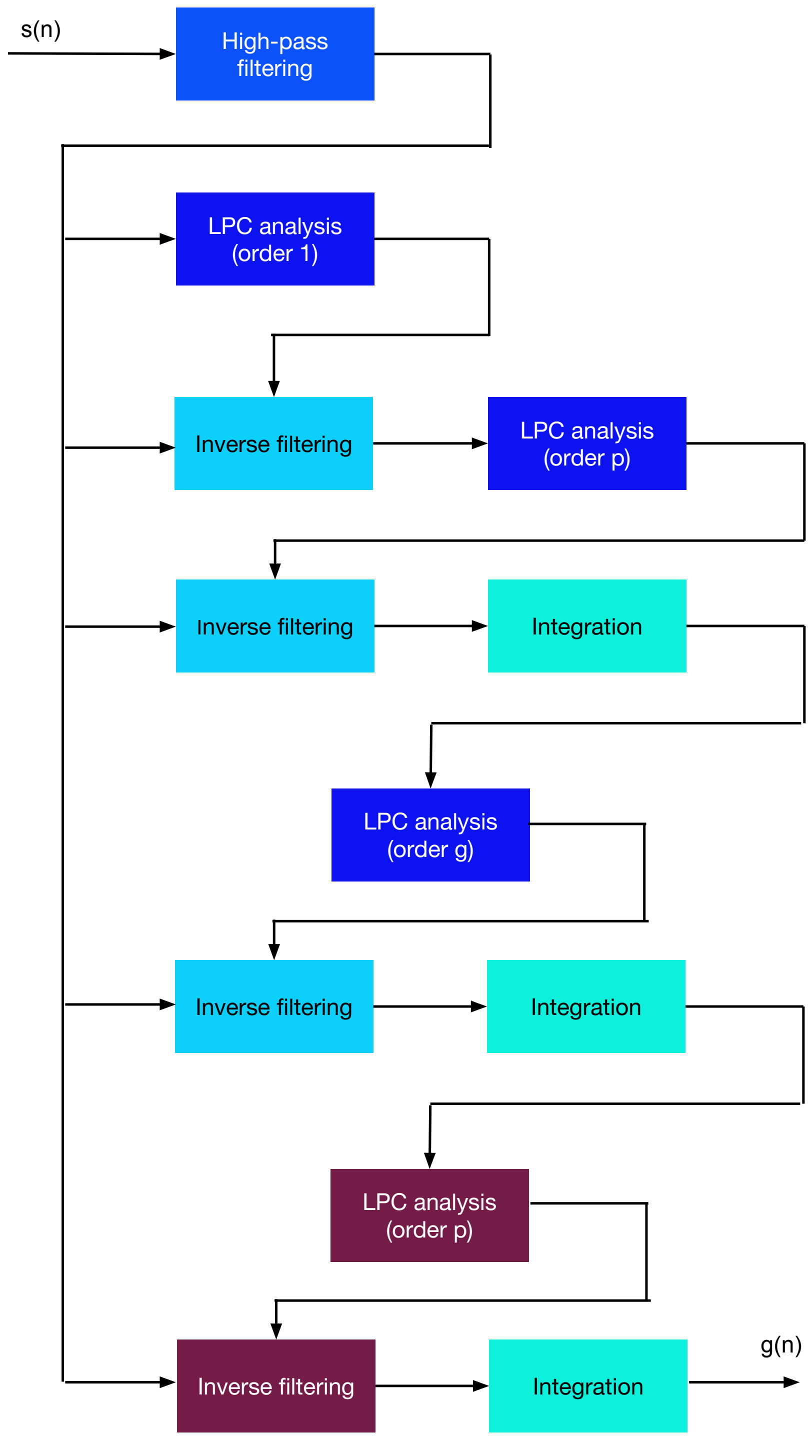

Fig. 3.2: Iterative Adaptive Inverse Filtering

while the fundamental period T_0 is held constant. The amplitude parameter E_0 is also adjusted in a similar fashion. This approach is identical to coordinate descent (Nesterov, 2012) except for the fact that we take steps rather than a full line search.

The LF model was only intended to model the glottal flow for fully voiced sounds. The IAIF residual, on the other hand, is extracted from real speech which runs the whole gamut between fully voiced sounds like vowels and fully unvoiced sounds like fricatives. There is therefore a residual between the LF model shape and the pitch pulses in the IAIF residual. We extract this *sub-residual* by subtracting the fully fit LF model from the pitch pulses in the IAIF residual. The sub-residual is then modeled using a low-order LPC (we used a 4^{th} order lpc). Since this sub-residual is mostly noise-like, the input during synthesis to this low-order LPC is just white noise. Figure 3.3 shows an example of the fit model and a pitch pulse from the original residual. The original glottal flow derivative is shown in black, the fit model in blue, and the sub-residual in red.

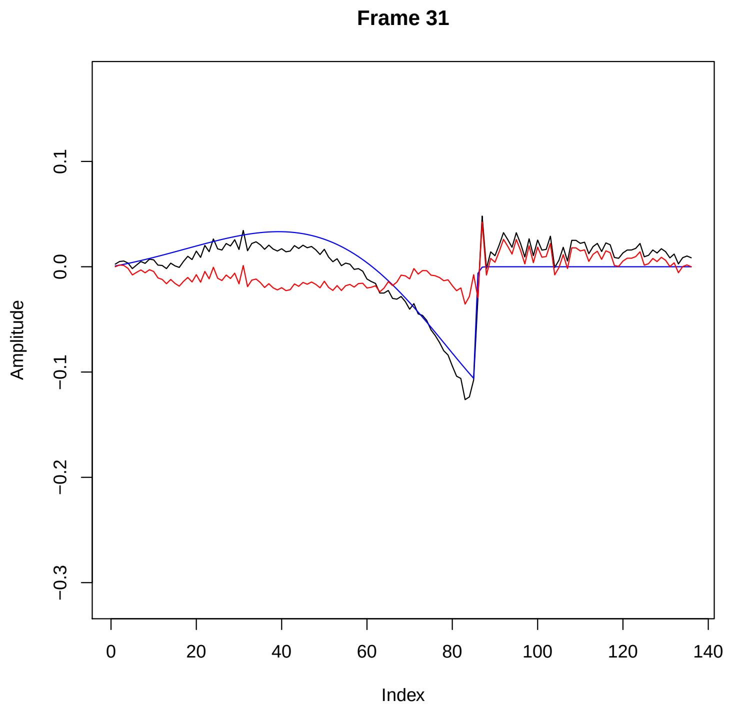

Fig. 3.3: Fitting to the IAIF residual

While many linguists tend to classify speech phones into mutually exclusive voiced and unvoiced categories, phoneticians agree that they are not distinct categories. Instead there is a spectrum ranging from purely voiced to unvoiced sounds. By building a joint model of the IAIF residual using the LF model and a low order LPC, we avoid the need to do a strict classification between voiced and unvoiced sounds.

This model is similar to the one proposed jointly in Vincent et al., 2005 and Vincent et al., 2007. These two papers, and our model model the residual by employing the LF model along with an additional model for the sub-residual. We use a low-order LPC while the two earlier papers use a Harmonic plus Noise Model(Stylianou, 1996). While our model itself may be related to a previous model, the novelty of our work lies in the fitting itself, and the unique way in which the Statistical Parametric Speech Synthesizer is used in the fitting.

We used this excitation model in the Clustergen Statistical Parametric Speech Synthesizer(Black, 2006). However, there is nothing about the technique that is specific to Clustergen. This technique (and every other one in this thesis) can just as easily be used in another statistical parametric synthesizer such as HTS(Zen et al., 2007). The training data used was from the CMU Arctic database(Kominek and Black, 2004). The database consists of multiple speakers, each of whom provided one hour of speech recordings. While we conducted experiments on multiple speakers, in the interest of clarity only the results on the RMS voice will be reported. RMS is an American Male. 20-dimensional LSPs were used for the vocal tract filter model and a 4-th order LPC was used to model the subresidual in addition to the parameters of the LF model.

To test the quality of this model, we conducted listening tests on Amazon's Mechanical Turk using the Testvox interface(Parlikar, 2012) where listeners were asked to choose between two systems. The first system was a synthesizer that used Line Spectral Pairs and the above described (LF) models for the residual. The second system was a baseline system that used an identical vocal tract model but a mixed excitation (ME) residual(Yoshimura et al., 2001).

19 listeners were asked to choose between 10 utterances generated by each system resulting in 190 tests between the two systems. In 116 of those tests, listeners judged our system to be more natural. In 70 of those tests, listeners judged the baseline system to sound more natural. In 4 of the tests, listeners did not have a preference.

A Generalized Estimating Equation (GEE) model was used with listeners repeated over sentences to test the hypothesis that the model intercept is zero, in other words, that the odds of a listener selecting the LF version were equal to the odds of the listener selecting the ME version as more natural sounding. The model coefficient for the intercept was 0.505 with a standard error of 0.2525, a Wald Chi-Square of 4.002 with 1 degree of freedom and a p-value of 0.045. We can thus reject the hypothesis of equal odds and conclude that listeners preferred to LF model versions of the sentences.

3.4 A Synthesis-specific fitting technique

Everything described so far about the IAIF residual extraction, and the LF model fitting could be lifted and placed in a thesis on speech recognition or speaker identification, and it would not look out of place. This is because the fitting process is completely agnostic to all lexical and phonetic information about the speech signal. This is perfectly appropriate for tasks such as speech recognition where no knowledge of the underlying words is known. However synthesis *does* know what the words to be spoken are, and so it is worth reconsidering this idea of trying to cut a synthesis board with a speech recognition hammer.

It was mentioned earlier that we used Clustergen to build a synthetic voice on the vocal tract LSP parameters, the LF model parameters, and the low-order LPCs for the sub-residual. Clustergen works by building Classification And Regression Trees (CARTs) (Breiman et al., 1984) that take lexical information as input, and learning to predict the above mentioned parameters.

Once this is done, we propose a technique where Clustergen *re-predicts* the parameters for the entire training data. The LF model parameters predicted by Clustergen are then used as seed values to perform an additional pass of model fitting. By repeating this process several times, the LF model parameters fit the IAIF residual much much better than before. This improvement occurs due to two reasons. The first is that averaging LF model parameters over several instances of a phone's pitch pulses has a smoothing effect that helps to remove the effect of outliers. The second and more important is that CARTs have full knowledge of the lexical information in that phone of speech, and so can initialize the fitting process based on this information.

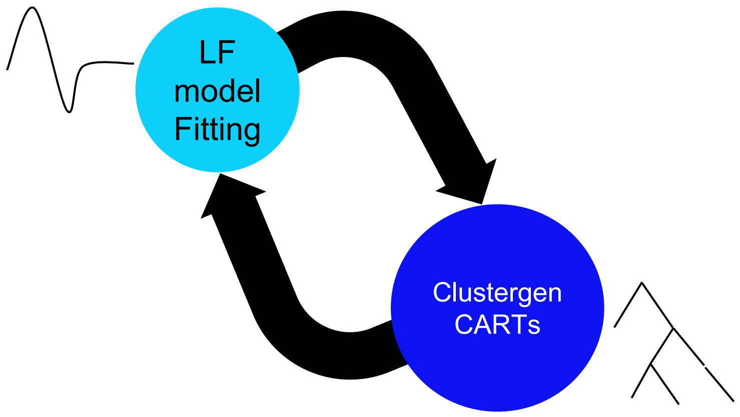

Fig. 3.4: Iterative estimation of LF parameters

One change we had to do in our fitting process was to transform the LF parameters into a different format. This is because Clustergen models the fundamental frequency F_0 independent of the other parameters. Since the LF parameters, T_0, T_e, T_p, T_a implicitly model the F_0, using these parameters directly in the ClusterGen system results in a mismatch between ClusterGen's smooth F_0 contour predictions and the implicit F_0 in the LF parameters. This causes the synthesized speech to sound shaky as if the speaker were about to cry. We can avoid this problem by representing the LF parameters as F_0-independent dimensionless parameters: Open Quotient (OQ), Speed Quotient (SQ) and Return Quotient (RQ) (Fant and Lin, 1988).

$$\begin{aligned} OQ &= \frac{T_e + T_a}{T_0} \\ SQ &= \frac{T_p}{T_e - T_p} \\ RQ &= \frac{T_a}{T_0} \end{aligned}$$

Using a F_0 independent model will also allow us to use models like the Statistical Phrase Accent Model(Anumanchipalli et al., 2011) in the future to impose specific intonations to the speech.

Results

The fitting process was iterated several times with initializations provided by ClusterGen which were fed back to the synthesizer. Listening tests that test the naturalness of the synthesized speech were inconclusive and the difference in quality between iterations was subtle. Even speech researchers who listened to the synthesized speech from the first and last iterations acknowledged that the speech sounded different but had difficulty in making a decision on which one sounded more natural. Empirical evidence suggests that this difficulty arose from the LPC model that is used to model the remainder of the LF fitting process.

There are two objective metrics which were worth considering. The first one was the prediction error. This is the error that we get as a result of limitations of the CARTs that ClusterGen uses. The reasoning was to try to push the fitting process into a space that ClusterGen could model well. The second metric was the RMSE and Correlation of the fitting process itself. These were computed for every pitch period where the LF model was being fit. While a low value for the second metric means the fitting process is going well, a low score for the first metric need not necessarily indicate that the fitting process works well. For example, if a bug in the code caused the parameters of every single pitch period in the database to be identical, then the ClusterGen predictions would be perfect all the time. However, we must not assume that optimizing the fitting metric alone is sufficient. The fitting process is worthless to us unless we can predict the values from the text. Therefore, our modeling technique must be able to do well under both metrics. Table 3.1 shows us that both the metrics indicate that our models get better as the iterations progress. Even in the case where the fitting RMSE starts to increase a little, the correlation still keeps improving.

Tab. 3.1: RMSE and Correlation of Fitting

	LF Fitting		Prediction
Iteration number	RMSE	Corr	RMSE
0	406.89	0.482	4.840
1	405.94	0.479	6.909
5	395.17	0.518	5.611
10	391.57	0.534	5.061
15	389.98	0.543	4.836
20	389.65	0.547	4.722
25	390.06	0.550	4.661
30	390.56	0.551	4.636

3.5 Improving labeling using waveform generation parameters

The parameterization of the speech signal can also be used to improve the labeling of the training data. The phone labels used for the first pass of the training process come from forced-alignment using the EHMM toolkit(Prahallad et al., 2006) which is part of Clustergen. Forced alignment works by finding the segmentation that maximizes the likelihood of the phone HMMs generating the data. In statistical parametric speech synthesis, the regression trees (CARTs) are the ones that are generating speech. We can therefore do a similar forced-alignment like approach that tries to maximize the likelihood of the CARTs generating the training data.

The 'move_label' algorithm described in Black and Kominek, 2009 does something very similar. It iteratively adjusts the phone labels of the training data so as to minimize the prediction error of the CARTs. In theory, this label adjustment should not be necessary because the HMM forced alignment should give us the best possible labels. In practice, this is not the case because the forced alignment is between the phone labels and the

Mel Cepstra, not the speech signal itself. Additionally the Mel Cepstra used for forced alignment while labeling are similar to the ones in Davis and Mermelstein, 1980 while for waveform generation, the Mel Cepstra used are the ones described in Tokuda et al., 1994.

Move_label helps improve synthesis even when both parameters are Mel Cepstra but the extraction methods are different. So, it makes sense to try move_label on coefficients such as LSPs and LF parameters which are vastly different from Mel Cepstra. LSPs and LFs have vastly different magnitudes though, so both were mean and variance normalized.

The better the labels, the easier it should be to predict the LSP and LF parameters. Thus the Mean Squared Error (MSE) of prediction of LSPs and LF parameters on a held out test set is a good metric of how good the labeling is. Table 3.2 reports the distortion thus measured on the LSP and LF parameters after 20 iterations of move_label. LSPD is the distortion on the LSP parameters while LFD is the distortion of the LF ones. The distortions are multiplied by 1000 to give us a cosmetically nicer number.

Tab. 3.2: Experiments with move_label

				Duration	
Optimization	Pass	LSPD	LFD	RMSE	Corr.
Baseline		7.472	4.829	0.907	0.425
LSP	16	7.245	5.015	0.957	0.305
LF	20	7.961	3.702	0.961	0.310
LSP + LF	20	7.379	4.657	0.946	0.313
LSP + LF + Dur	5	7.418	4.777	0.878	0.479

The LSP+LF, which gave the lowest LFD, produced speech which we informally identified as being more smooth than the baseline, but the durations (in text to speech) were different enough to be less desirable. When we added the duration optimization constraint to our move_label optimization it converges quicker, but the spectral quality of the signal is not perceptably different from the baseline (though the durations are better).

3.6 Discussion

Excitation modeling is widely acknowledged to be one of the hardest problems in speech analysis, with the mere mention of its name producing a weary sigh in seasoned veteran researchers. While a myriad of different models have been proposed, most speech synthesis systems only use the most basic and simplistic mixed excitation models. We believe that one reason for this is that these models do not make full use of the synthesis pipeline. We hope that by kindling interest in a more integrated synthesis pipeline, progress will be made faster in this difficult field.

This technique and the above described results were first published in Muthukumar et al., 2013. It is worth noting that despite the positive results reported with the improved excitation model, Clustergen still uses mixed excitation with the Mel Cepstral vocal tract model as the default. While the excitation model reported here is definitely better than simple mixed excitation, the Mel Cepstral vocal tract filter is substantially better than the one based on LPCs. The differences in the quality of the vocal tract filter tend to overwhelm any improvements made in the excitation model. Ideally the improved excitation model should be used along with the Mel Cepstra, but as of now there exists no IAIF-like procedure for it which can give us a good Mel Cepstra-glottal flow decomposition.

4 Integrating Prediction Algorithms with Waveform Generation

The TTS front-end and the excitation models discussed in the previous chapter are worlds apart, both in the synthesis pipeline as well as in the minds of the researchers who work on them. So it is fairly obvious that connecting the two is novel, unusual, and certainly desirable. Less obvious is the case for investigating the connections between neighboring components. At first glance, this might seem absurd. For instance, let's take the prediction algorithm block and the waveform generation block in Figure 2.1. These two are quite clearly connected to each other. Being connected does not mean that they are tightly integrated though. Like warring nations that share a border, the research communities that work on either component rarely interact with each other. So, assumptions abound and rules are bent. The final result of all this cloistered research are two black boxes, completely transparent to the researcher who builds one but inscrutable to the researcher who builds the other.

Why does this gap exist between the machine learning researchers who develop new prediction algorithms and the signal processing experts who invent new ways of analyzing speech? The most likely reason is that many of the issues that caused problems for speech signal processing have also been inherited by machine learning techniques for speech. Speech is not stationary, or Independent and Identically Distributed. Speech is also a time series, and either real or complex valued rather than the simplistic discrete or binary assumptions made by many machine learning algorithms. Also, while the speech recognition researcher often makes the unimaginative analogy of speech synthesis being as easy as squeezing toothpaste out of a tube and speech recognition being the opposite, the mathematically literate will acknowledge that *regression* problems like synthesis are much harder than the generic classification problem.

The inability to handle time-series regression problems is most obvious when looking at the frames of speech predicted by current acoustic models. Speech is labeled using Hidden Markov Models (HMMs). HMMs tend to map several consecutive frames of speech onto the same hidden state. As a result, during prediction the Classification And Regression Trees (CARTs) predict identical frames of speech for consecutive time steps. It is physically impossible for any human to produce two perfectly identical frames of speech, and so when this kind of repetitive structure exists in synthesized speech, the human ear is quick to notice it. This is one of the major causes of buzziness in the modern statistical parametric synthesizer.

Like the politician who puts an ambulance at the bottom of a cliff rather than a fence at the top, the most common fixes to the repetitive frame problem have been *post-filters*. The post-filter works by smoothing the output of the CART so that consecutive frames have a more natural trajectory structure to them. Post-filters vary in sophistication from simple linear interpolators to more context-aware techniques like Global Variance(Toda and Tokuda, 2007). Arguably the most successful post-filter though is MLPG — Maximum Likelihood Parameter Generation (Tokuda et al., 2000). Using the means and standard deviations of Mel Cepstral static and delta coefficients, MLPG makes a maximum likelihood estimation of the true underlying cepstral coefficients.

While it is easy to rant about how post-filters bypass the actual problems in predicting time series data, it is very hard to create a technique that works well without a post-filter. Instead, in this chapter I describe an improved post-filter which works unlike most post-filters in existence today. The problem with techniques like MLPG or Global Variance is that they are derived analytically. MLPG for instance exclusively deals with the means, the deltas, and the delta-deltas for every state. Integrating non-linear combinations of the means across frames or arbitrary new features like wavelets into MLPG will be somewhat non-trivial, and requires revisiting the equations behind MLPG as well as a rewrite of the code.

Using a Deep Neural Network (DNN) as a post-filter can overcome many of these issues. Neural networks are fairly agnostic to the type of features provided as input. The input features can also easily be added or removed without having to do an extensive code rewrite. DNN based post-filters have been previously reported in Chen et al., 2015 and Chen et al., 2014. These simple feedforward Multi-Layer Perceptrons however do not leverage the full power of the neural networks, and only use the same input features that MLPG and the other popular post-filters do. Additionally the only way the feedforward DNNs model inter-dependencies betwen frames is by crude *stacking*, where consecutive frames are concatenated together. While this is somewhat effective, it is difficult to argue that stacking is anything more than a hack.

4.1 Recurrent Neural Networks

A more theoretically sound approach to handle the interdependencies between consecutive frames is to use a *Recurrent* Neural Network(Elman, 1990). RNNs differ from basic feedforward neural networks in their hidden layers. Each RNN hidden layer receives inputs not only from its previous layer but also from activations of itself for previous inputs. A simplified version of this structure is shown in figure 4.1. The actual RNN used in the experiments in this chapter has every node in the hidden layer connected to the previous activation of every node in that layer. However, most of these links have been omitted in the figure for clarity. The RNN structure used is more or less identical to the one described in section 2.5 of Sutskever, 2013.

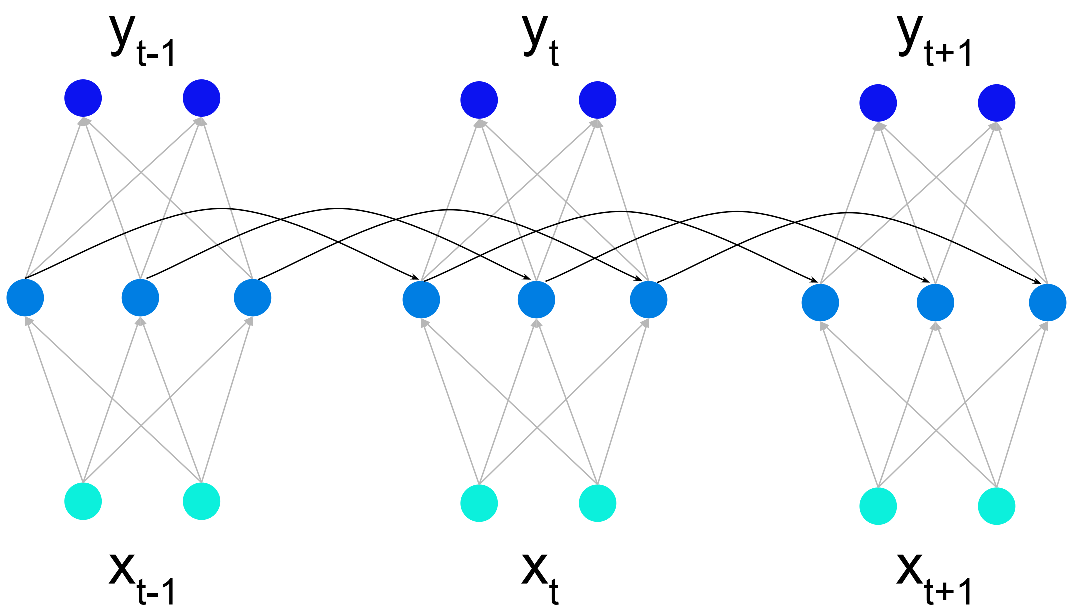

Fig. 4.1: Recurrent Neural Network

4.2 RNNs as post-filters

The first step in training an RNN post-filter is to just build a synthetic voice using a traditional Statistical Parametric Speech Synthesizer. In our case, this was again Clustergen. We then make Clustergen re-predict the entire training data. As a result, we now have Clustergen's predictions of Mel Cepstral Coefficients (MCEPs) which are time aligned with the original MCEPs extracted from speech. The RNN is then trained to learn to predict the original MCEP statics based on Clustergen's predictions of the MCEP statics and deltas. This trained RNN can then be applied to the CART's predictions of test data to remove some of the noise in prediction. We use 25 dimensional MCEPs in our experiments. So the RNN takes 50 inputs (25 MCEP statics + 25 deltas) and predicts 25 features (MCEP statics). The effect of using this RNN post-filter on four different corpora are shown in table 4.1. Each voice had its own RNN. Mel Cepstral Distortion(Kubichek, 1993) is used as the objective metric.

Tab. 4.1: MCD without MLPG

Voice	Baseline	With RNN
RMS (1 hr)	4.98	4.91
SLT (1 hr)	4.95	4.89
CXB (2 hrs)	5.41	5.36
AXB (2 hrs)	5.23	5.12

Tab. 4.2: MCD with MLPG

Voice	Baseline	With RNN
RMS (1 hr)	4.75	4.79
SLT (1 hr)	4.70	4.75
CXB (2 hrs)	5.16	5.23
AXB (2 hrs)	4.98	5.01

This RNN post-filter was initialized with a random start, and only the data for the specific voice was used. It is possible that better results could have been obtained by using all of the available data and building one post-filter for all the voices. However this is an experiment which will be left for the future.

The RMS and SLT voices reported in the tables are from the CMU Arctic speech databases(Kominek and Black, 2004), about an hour of speech each. CXB is an American female, and the corpus consists of various recordings from audiobooks. A 2-hour subset of this corpus was used for the experiments described in this chapter. AXB is a 2-hour corpus of Hindi speech recorded by an Indian female. RMS, SLT, and AXB were designed to be phonetically balanced while CXB is not. The former were also explicitly designed for speech synthesis research while CXB is a generic audiobook.

The RNN was implemented using the Torch7 toolkit(Collobert et al., 2011). The hyper-parameters were tuned on the RMS voice for the experiment described in table 4.1. The result of this was an RNN with 500 nodes in a single recurrent hidden layer, with sigmoids as non-linearities, and a linear output layer. To train the RNN, we used the ADAGRAD(Duchi et al., 2011) algorithm along with a two-step BackPropagation Through Time(Werbos, 1990). A batch size of 10, and a learning rate of 0.01 were used.

Early stopping was used for regularization. L1, L2 regularization, and momentum did not improve performance when used in addition to early stopping. No normalization was done on either the output or the input MCEPs. The hyperparameters were not tuned for any other voice, and even the learning rate was left unchanged.

The results reported in table 4.1 are with the MLPG option in Clustergen turned off. The reason for this is that MLPG requires the existence of standard deviations, in addition to the means of the parameters that the CART predicts. There is no true set of standard deviations that can be provided as training data for the RNN to learn to predict; it only has access to the true means in the training data. That being said, we *did* however apply MLPG by taking the MCEP means from the RNN, and standard deviations from the original CART predictions. We found that it did not matter whether the means for the deltas were predicted by the RNN or if they were taken from the CART predictions themselves. The magnitudes of the deltas were typically so small that it did not influence the RNN training as much as the statics did. So, the deltas were omitted from the RNN predictions in favor of using the deltas predicted by the CARTs directly. This also made the training a lot faster as the size of the output layer was effectively halved. The results of applying MLPG on the results from table 4.1 are shown in table 4.2. Note that MLPG was applied on the baseline system as well as the RNN postfilter system.

4.3 Adding lexical features

The goal of this thesis has been to motivate the use of more integrated acoustic models. However the experiments described with this post-filter so far have not really achieved this goal. There is nothing that the post-filter does which could not have been done (arguably better) by techniques such as MLPG or Global Variance. The power of the RNN comes from its ability to make use of any arbitrary feature that can be provided as input to the RNN. It would be extremely painful to make MLPG use features such as wavelets or prosody information, but adding them to the RNN would barely raise an eyebrow.

We added *all* of Festival's lexical features as additional input features for the RNN, in addition to CART's predictions of f0, MCEP statics and deltas, and voicing. The standard deviations as well as the means of the CART predictions were used. This resulted in 776 input features for the English language voices and 1076 features for the Hindi one (the Hindi phoneset for synthesis is slightly larger). The output features were the same 25-dimensional MCEPs from the previous set of experiments. The results of applying this kind of RNN on various corpora are shown in the following tables. As for the previous set of experiments, each voice had its own RNN. Table 4.3 and table 4.4 show results without and with MLPG respectively. In addition to the voices tested in previous experiments, we also tested this approach on three additional corpora, KSP (Indian male, 1 hour of Arctic), GKA (Indian male, 30mins of Arctic), and AUP (Indian male, 30 mins of Arctic).

Tab. 4.3: RNN with all lexical features. MLPG off

Voice	Baseline	With RNN
RMS (1 hr)	4.98	4.77
SLT (1 hr)	4.95	4.71
CXB (2 hrs)	5.41	5.27
AXB (2 hrs)	5.23	5.07
KSP (1 hr)	5.13	4.88
GKA (30 mins)	5.55	5.26
AUP (30 mins)	5.37	5.09

Tab. 4.4: RNN with all lexical features. MLPG on

Voice	Baseline	With RNN
RMS (1 hr)	4.75	4.69
SLT (1 hr)	4.70	4.64
CXB (2 hrs)	5.16	5.15
AXB (2 hrs)	4.98	4.98
KSP (1 hr)	4.89	4.80
GKA (30 mins)	5.27	5.17
AUP (30 mins)	5.10	5.02

As can be seen in the tables, the RNN *always* gives an improvement when MLPG is turned off. Kominek et al., 2008 reports that an MCD decrease of 0.12 is equivalent to doubling the amount of training data. The MCD decrease in table 4.3 is far beyond that in all of the voices that are tested. It is especially heartening to see that the MCD decreases significantly even in the case where the corpus is only 30 minutes. With MLPG turned on, the RNN always improves the system or is no worse. The decrease in MCD is much smaller though.

With MLPG turned on, the decrease in MCD with the RNN system is not large enough for humans to be able to do reasonable listening tests. With MLPG turned off though, the RNN system was shown to be vastly better in informal listening tests. But the goal was to build a system much better than the baseline system *with* MLPG, and so no formal listening tests were done.

4.4 Post-filters post-MLPG

It is disappointing to see that the use of MLPG on top of the RNN does not give us the same gains as when used on its own. This could be caused by one of two things: either MLPG undoes some of the effects of the RNN, or MLPG and the RNN have very similar smoothing effects. The easy way to test this is to train the RNN on the output of MLPG rather than use MLPG on the output of the RNN. The results of this experiment are shown in table 4.5. The results in column 2 of this table are from table 4.4.

Tab. 4.5: RNNs before vs. after MLPG

Voice	RNN before MLPG	RNN after MLPG
RMS	4.69	4.70
SLT	4.64	4.64

In both of the voices tested, there is little to no difference in the MCD scores. This seems to indicate that the RNN and MLPG do the same kind of smoothing. While MLPG has the advantage of being substantially more computationally efficient, the RNN has the much more important advantage of being agnostic to the input features. There certainly is no dearth of new features which can be added, and so we can safely believe that the RNN will win out in the long run.

4.5 Joint training of the CART and the postfilter

The traditional way to build any postfilter for Statistical Parametric Speech Synthesis is to start off by building the Classification And Regression Trees that predict the parameters of the speech signal, and then train or apply the postfilter on the output of the CART. The drawback of this is that the CART is unnecessarily agnostic to the existence of the postfilter. CARTs and postfilters need not necessarily work well together, given that each has its own set of idiosyncracies when dealing with data. MCEPs might not be the best representation that connects these two. In an ideal system, the CART and the postfilter should jointly agree upon a representation of the data. This representation should be easy for the CART to learn, as well as reasonable for the postfilter to correct.

One way of achieving this goal is to use a fairly new technique called Method of Auxiliary Coordinates (MAC) (Carreira-Perpiñán and Wang, 2012). To understand how this technique works, we need to mathematically formalize our problem.

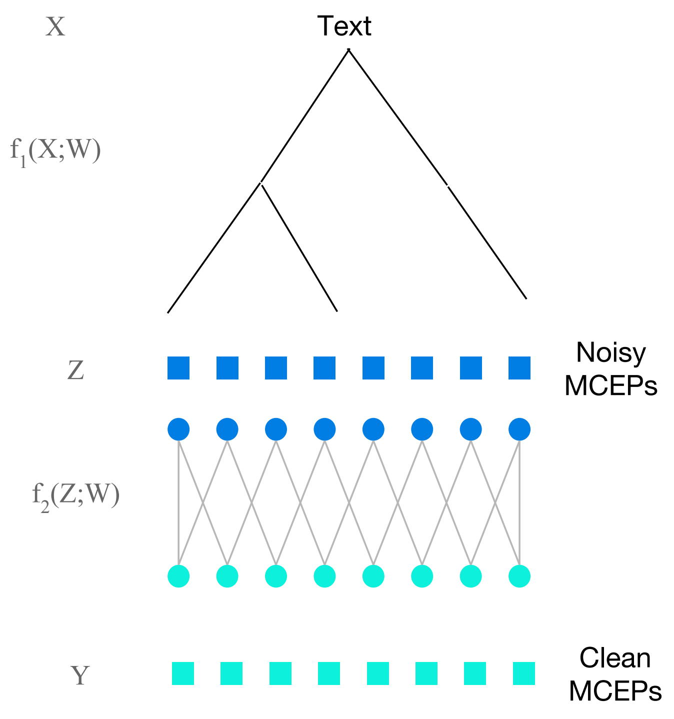

Fig. 4.2: Joint training of the CART and the RNN

Let X represent the linguistic features that are input to the CART. Let Y be the correct MCEPs that need to be predicted at the end of synthesis. Let $f_1()$ be the function that the CART represents, and $f_2()$ the function that the RNN represents. The CART and the RNN both have parameters that are learned as part of training. These can be concatenated into one set of parameters W. The objective function we will want to minimize can therefore be written as:

$$E(W) = \|f_2(f_1(X;W);W) - Y\|^2$$

The MAC algorithm basically rewrites the above equation to make it easier to solve. This is done by explicitly defining an intermediate representation Z that acts as the target for the CARTs and as input to the RNN. The previous equation can now be rewritten as:

$$E(W,Z) = \|f_2(Z;W) - Y\|^2$$
$$s.t.\ Z = f_1(X;W)$$

The hard constraint in the above equation is then converted to a quadratic penalty term:

$$E(W,Z;\mu) = \|f_2(Z;W) - Y\|^2 + \mu\|f_1(X;W) - Z\|^2$$

where μ is slowly increased towards ∞.

The original objective function in the first equation only had the parameters of the CART and the RNN as variables to be used for the minimization. The rewritten objective function adds the intermediate representation Z as an auxiliary variable that will also be used in the minimization. We minimize the objective function by alternatingly optimizing the parameters W and the intermediate variables Z. Minimizing with respect to W is more or less equivalent to training the CARTs and RNNs independently using a standard RMSE criterion. Minimization with respect to Z is done through Stochastic Gradient Descent. Intuitively, this alternating minimization has the effect of alternating between optimizing the CART and RNN, and optimizing for an intermediate representation that works well with both.

We applied this algorithm to the RNN and CARTs built for the SLT and RMS voices built for table 4.1. The results of this are shown in table 4.6. Starting from the second iteration, any further optimization of either the W or the Z variables only results in a very small decrease in the value of the objective function. So, we did not run the code past the second iteration.

We did not try MAC for the RNNs which use lexical features. This is because running the Z optimization on those RNNs would have given us a new set of lexical features for which we would have no way of extracting from text.

Tab. 4.6: MCD Results of MAC

Method	SLT	RMS
RNN from table 1	4.89	4.91
1st MAC iteration	4.86	4.92
2nd MAC iteration	4.89	4.92

On our experiments on the RMS and SLT voices, there was no significant improvement in MCD. There are marginal MCD improvements in the 1st iteration for the SLT voice but these are not significant. Preliminary experiments suggest that one reason for the suboptimal performance is the overfitting towards the training data. It is difficult to say why the MAC algorithm does not perform very well in this framework. The search space for the parameters and the number of possible experiments that can be run is extremely large though, and so it is possible that a more thorough investigation would provide positive results.

4.6 Discussion

The elephant in the room which isn't addressed in this chapter is the idea of completely getting rid of the CARTs and using a deep neural network to go from the text possibly straight to the waveform itself. This approach has been investigated in Zen et al., 2013, Fan et al., 2014, Zen and Senior, 2014, and in several others. Inspite of the huge amount of work put into these papers, it is difficult to argue that DNNs have had the same level of impact that they have had in Automatic Speech Recognition technology. DNN-influenced improvements in synthesis have mostly been fairly moderate. This becomes fairly evident when looking at the submissions to the Blizzard Challenge in the last 3 years. Few of the submitted systems have used deep neural networks in any part of the pipeline, and those that do use DNNs do not seem to have any advantage over traditional well-trained systems.

That being said, it is still be unwise to rule out the possibility of an end-to-end DNN based speech synthesizer. DNNs are extremely powerful models, and like many techniques at the forefront of machine learning research, it might be the case that those models are merely waiting for training algorithms that do not exist yet.

In this chapter, I have tried to walk the fine line between conservatively holding on to existing systems, and fully embracing exciting but untested newer methods. It is likely that the work described in this chapter will eventually give way to either purely DNN based methods or will regress back to older CART based models. But experiments such as those in this chapter are a necessary intermediate step in progressing towards either direction.

5 Integrating Phoneme Representations with Spoken Language

In the two previous chapters, the focus was on integrating features from the front-end with various components in the back-end of the Statistical Parametric Speech Synthesizer. The techniques discussed in both of these are novel ways of passing information from components upstream to components downstream. This begs the question of whether it is possible to flip this idea, and pass information from the back-end upstream to the front-end.

Taking a page from chapter 3, we will again try to connect the components at the extremities of the acoustic model by deriving phonemes from the speech signal itself rather than from the text. The phonemes of a language are defined by linguists who undertake a careful study of the language, and then declare as phonemes those units that they deem appropriate.

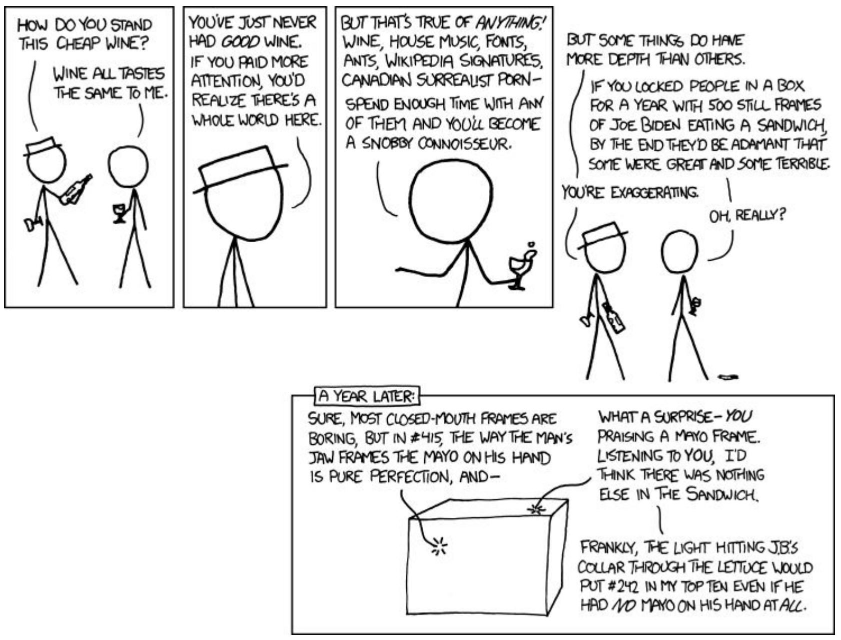
HOW DO YOU STAND THIS CHEAP WINE?
WINE ALL TASTES THE SAME TO ME.
YOU'VE JUST NEVER HAD GOOD WINE. IF YOU PAID MORE ATTENTION, YOU'D REALIZE THERE'S A WHOLE WORLD HERE.
BUT THAT'S TRUE OF ANYTHING! WINE, HOUSE MUSIC, FONTS, ANTS, WIKIPEDIA SIGNATURES, CANADIAN SURREALIST PORN— SPEND ENOUGH TIME WITH ANY OF THEM AND YOU'LL BECOME A SNOBBY CONNOISSEUR.
BUT SOME THINGS DO HAVE MORE DEPTH THAN OTHERS.
IF YOU LOCKED PEOPLE IN A BOX FOR A YEAR WITH 500 STILL FRAMES OF JOE BIDEN EATING A SANDWICH, BY THE END THEY'D BE ADAMANT THAT SOME WERE GREAT AND SOME TERRIBLE.
YOU'RE EXAGGERATING.
OH, REALLY?
A YEAR LATER:
SURE, MOST CLOSED-MOUTH FRAMES ARE BORING, BUT IN #415, THE WAY THE MAN'S JAW FRAMES THE MAYO ON HIS HAND IS PURE PERFECTION, AND—
WHAT A SURPRISE—YOU PRAISING A MAYO FRAME. LISTENING TO YOU, I'D THINK THERE WAS NOTHING ELSE IN THE SANDWICH.
FRANKLY, THE LIGHT HITTING JB'S COLLAR THROUGH THE LETTUCE WOULD PUT #242 IN MY TOP TEN EVEN IF HE HAD NO MAYO ON HIS HAND AT ALL.

Fig. 5.1: https://xkcd.com/915/

As anyone who has tried to pick a restaurant for lunch for ten people will understand, getting humans to agree upon a common answer is hard. This problem becomes exponentially harder when the people are experts, and the answer is difficult to validate experimentally. The result is that the ideal set of phonemes even for an extremely well-studied language like English is an issue of contention rivaling modern political squabbles.

This statement isn't meant to belittle the excellent work done by modern linguists though. The International Phonetic Alphabet (IPA), is a comprehensive representation that can represent every single sound that a human can speak(Association, 1999). IPA's biggest strength is in its ability to be a language and speaker independent description of the sounds in a language. This unfortunately is also its biggest shortcoming when it comes to using it in a speech synthesis system. Speech synthesizers are nearly always built specific to a particular voice. The ideal set of phonemes for that voice should be tailored to that speaker's idiolect. By its very definition, IPA cannot do this, nor should it aim to do so.

Further complicating matters is the question of what to do when the language has limited resources. While British English and Mandarin Chinese have been studied extensively enough that a reasonable phoneme set is available, Indian English and Cantonese do not enjoy the same privileges. Javanese for instance has over 82 million native speakers, yet the amount of linguistic resources available for it belies its size. This is not the fault of linguists or linguistics though. The sheer number and diversity of the world's languages vastly exceed the resources available to modern linguists who document them. One potential solution therefore is to try and bypass the knowledge of the linguist, and build a speech synthesizer directly from the available data itself.

In this chapter, we will attempt to solve this problem in its absolute worst case where we know neither the phonemes of the language, nor the language family it belongs to. This assumption is not as ludicrous as it might initially seem. While discovering the language family of an arbitrary corpus is easier for a linguist than coming up with phonemes, it still requires extensive effort which we can avoid if possible. Removing the human from this part of the loop also lowers the chances of making errors. Korean, for instance has been extensively studied for hundreds of years, but there have been numerous debates on which other languages are related to it.

We will be taking our problem to even further extremes though by assuming that we have a corpus of speech data in a particular language but *no text* that corresponds to it. The need for this is obvious when dealing with languages such as Konkani which don't have a standard orthography. Less obvious is the need when dealing with languages like the dialects of Arabic where the written language has little resemblance to the spoken language. The aim of the speech synthesizer is to produce speech akin to human speech, and so the phonemes must be for the spoken language and not the written one.

This distinction between the spoken language and the written language is very important for our task because it is impossible to derive the phonemes for the language based on speech data alone. Deriving *phones*, on the other hand is perfectly feasible. For a speech synthesizer, the phone representation is more appropriate than a phonemic one. So in this chapter, we will only be concerned with phones and not with phonemes.

5.1 Language and speaker independent features

From a purely machine learning point of view, this is a simple clustering problem of speech segments. There is however a problem with doing clustering on features like LPCs or Mel Cepstral Coefficients which are typically used to represent the speech signal. While these features do an excellent job of representing the speech signal, these also capture unnecessary characteristics such as speaker identity, emotion, and language identity. So, doing an unsupervised clustering on one of these features might not give us clusters that are phonetically relevant. Ideally, we would want to apply our clustering process on features that contain little speaker or language specific information.

There has been work in the speech recognition community on trying to find such features. One such set of features which are particular useful for our task are called *Articulatory Features* (AFs). These are based on the notion of the articulatory phonetic features defined in the IPA handbook. Vowels in human speech can be visualized in a chart indexed by the frequency of the first two formants. We can identify vowels as being in two dimensions, high to low(F1), and front to back(F2). Other ways of describing vowels include nasality, length, stress, and tone. Consonants too can be broken down into a set of features that distinguish stops, fricatives, affricates etc... These are the 'articulatory' features by which all human speech sounds can be represented. These are described in detail in Metze, 2005, Black et al., 2012, Kirchhoff, 1999, Kirchhoff et al., 2002, and Stüker et al., 2003. These should not be confused with the features described in Wrench, 1999 which describe how the articulators of speech actually move.

The phones in a language are formed by choosing appropriate combinations of these articulatory features. While different languages and dialects differ in which combinations form phones, the articulatory features themselves are independent of language. This is because the articulatory features are based on the phonetics of human speech production, not on the phonology of specific languages. Articulatory features are also, for the most part, independent of speaker. For this paper, we use the 26 binary articulatory features described in Black et al., 2012.

5.2 Training an Articulatory Feature Detector

Articulatory feature extraction is done by bootstrapping from a speech corpus for which we *do* have transcriptions at the text or the phonetic levels. This bootstrap corpus can be in any arbitrary language, or even a mixture of several languages. It is desirable though to have a corpus with several hours of speech from multiple speakers. We used the Wall Street Journal corpus(Garofalo et al., 1993), which has multiple speakers, but is all standard US English. There are other corpora which might be multi-lingual and therefore more appropriate, but this was chosen because the corpus is extremely well-understood. The license also allows us to release the models trained on this data.

In the English bootstrap corpus, a simple table lookup is used to obtain articulatory feature information corresponding to each phone. While extracting features from data with phone labels is trivial, extracting these directly from the speech signal requires a more sophisticated approach.

Mel Cepstral Coefficients are used as input features from which Articulatory Features are predicted. MFCCs have been extremely successful in all aspects of speech processing, so this seemed liked the best features to start with. The mapping from MFCCs to AFs is learned using a three hidden layer feedforward network with sigmoid activations using the QuickNet toolkit(Johnson et al., 2004). 26 neural networks were trained in this manner (one for each AF).

Each of the neural networks we trained has an accuracy between 85-99% on cross-validation sets. However, we wanted to do some experiments on real data to verify that our predicted AFs were reasonable. To do this, we used the EHMM labeling tool as described in Prahallad et al., 2006 to do a forced alignment between phone labels and articulatory features in the RMS voice of the CMU Arctic database(Kominek and Black, 2004). EHMM gives us an alignment that maximizes the likelihood of the phone labels having generated the articulatory feature data. A synthesizer built using this alignment would only give us sensible results if the articulatory feature extraction works reasonably well. So, this is an independent way of evaluating the quality of our articulatory feature extraction. The synthesizer is compared to one built with alignments obtained using Mel Cepstral features. The Clustergen speech synthesizer is used in all our experiments(Black, 2006). Mel Cepstral Distortion (MCD) on a separate test set is used as an objective metric for evaluating the quality of speech synthesis. This experiment was also replicated on a Hindi speech corpus(Prahallad et al., 2012). The results obtained are shown in table 5.1

Tab. 5.1: MCD Scores

MCD Scores	English	Hindi
EHMM on MCEPs	5.1610	5.255
EHMM on AFs	5.4752	5.665

The synthesizer built with alignments using AFs does not perform as well as the one that used MCEPs. However, the difference in MCD between the two is small enough that it indicates that the articulatory features obtained through the methods described above are reasonable.

5.3 Phone segmentation

Obtaining phonetic information about a speech corpus not only requires a consistent phone representation but also involves *segmenting* the speech corpus at the phone level. The task of doing phone segmentation would be greatly simplified if we had an approximate estimate of the number of phones for each utterance in our corpus. We obtain a first approximation of this number by running a Automatic Speech Recognizer in phone recognition mode. For our task, we used the CMU Sphinx speech recognition system(Huggins-Daines et al., 2006) in *all phone* mode and the iterative procedure developed in Sitaram et al., 2013 to get initial phone transcripts. An English phone recognizer gives us *English* phones and so is unlikely to produce a good phonetic transcription of our foreign language corpus. For example, if we were to run this recognizer on a Hindi corpus, the recognizer would not distinguish between the aspirated and unaspirated stops in Hindi. Or if it were run on a Japanese corpus, the recognizer would try to make distinctions between the liquids 'L' and 'R' while the language itself does not.

While the recognizer makes errors in assigning labels to the phones in an utterance, the segmentation is more important than the labels themselves. Doing a forced alignment using the EHMM tool between the phone labels and the file containing the audio of the utterance will give us a fairly good estimate of the segment boundaries.

5.4 Clustering and Labeling phones

After phone segmentation, we face the difficult problem of assigning labels to these phones. The MCEP parameters that we extract from the speech represent the speech signal in 25ms frames at 5ms shifts. The AFs which are extracted from the MCEPs also are of the same time intervals. However, the phone segments themselves are much longer than the length of a frame. For each phone segment, we *average* the MCEPs and the AFs in that segment so that we have an average MCEP vector, and an average AF vector for that phone segment. Our task is now to find a set of phone labels which gives a consistent description of these average vectors in our hypothetical database. Before we consider possible schemes of label assignment, we have to tackle the issue of evaluating how good our labeling schemes are. It is reasonable to assume that the more consistent and accurate the phone labels are, the easier it is for the synthesizer to model a voice based on that data. A good set of phone labels will therefore result in a good MCD score.

The most straightforward way to automatically assign a consistent set of phone labels is to use some sort of clustering method. As mentioned earlier, it is advantageous to cluster the Articulatory Features, rather than the MCEPs themselves, because of their speaker and language independent characteristics. We build a Classification And Regression Tree (CART)(Breiman et al., 1984) that forms a hierarchical clustering of the AF vectors, where the leaf nodes contain Gaussians modeling the MCEP vectors that correspond to the AF questions that are asked at higher levels of the tree. The clustering algorithm works by minimizing the variance of the MCEP coefficients at the leaf nodes by asking questions about the AF vectors. Overfitting is prevented by specifying a minimum number of MCEP vectors from the training data that must correspond to each leaf of the tree (also called the *stop size*).

Once the tree is built with AF questions and MCEP leaves using our entire corpus, each leaf will now be a fairly reliable cluster that corresponds to an acoustically derived phone. This tree is then used to assign labels to the phone segments in our corpus. We call these labels *Inferred Phones* or *IPs* because they are inferred from purely acoustic information with no human involvement. One important issue in doing this is deciding the stop size of the tree i.e. deciding how many leaves and therefore how many phones we should infer from the corpus. There is no clear answer to this problem because it is difficult even for linguists to give a definitive number for phones of a language. We will instead report results that we obtained with a range of phoneset sizes. In our

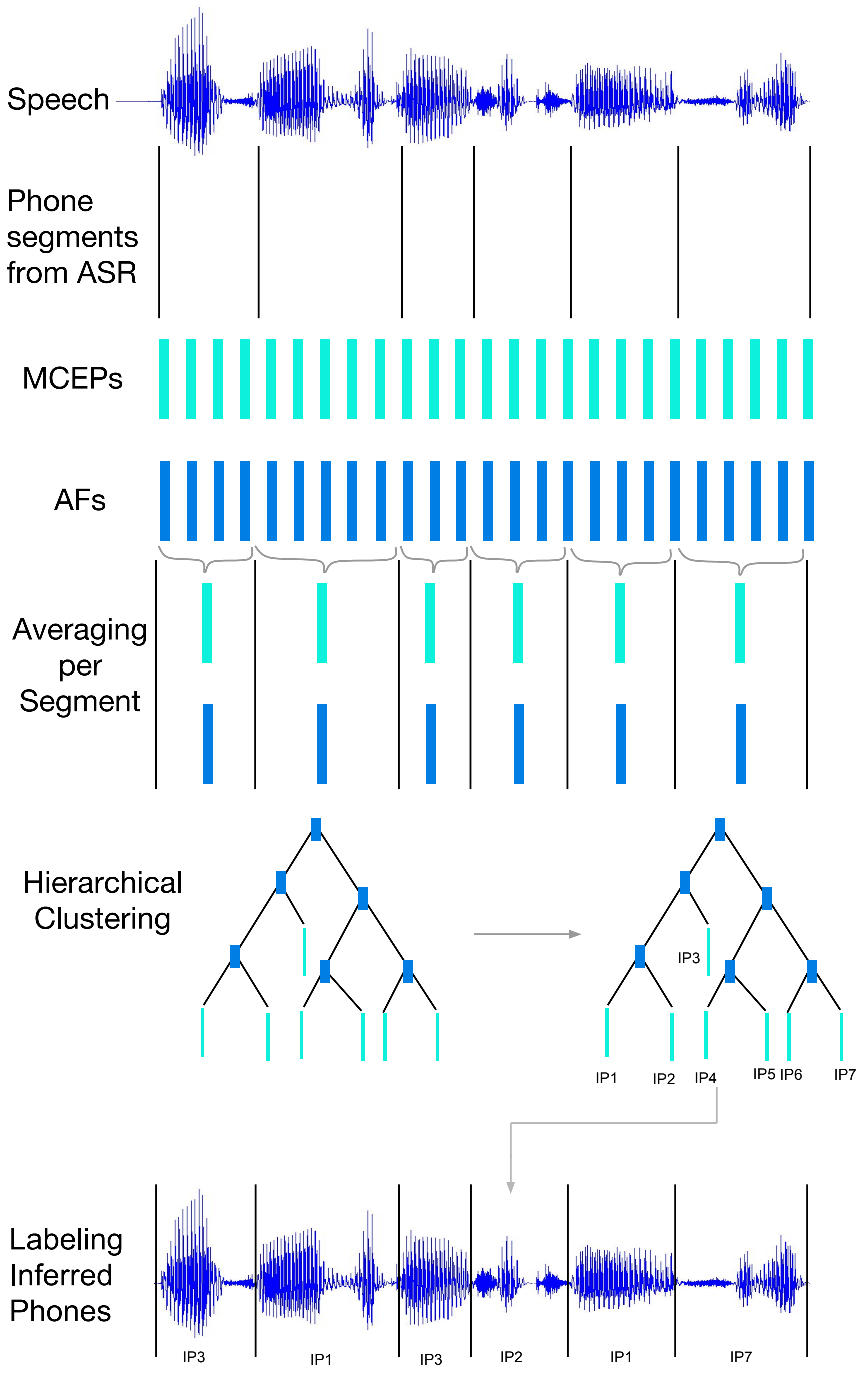

Fig. 5.2: Clustering and Labeling framework

experience, nearly all languages in the world have between 20-200 phones. So, we use that as bounding range within which we will do our experiments.

5.5 Experiments Inferring Phones

We ran experiments on four languages, Hindi, Dari, Iraqi, and English. The results obtained for Hindi, Dari, and Iraqi are shown in Figure 5.3. Comparisons are made between Clustergen synthesis systems built with various numbers of Inferred Phones and the baseline system which was built with phones from ASR. To make a fair comparison of the inferred phones to the ASR phones, we assumed that we did not possess any phonetic feature information about the ASR phones either. Empirically we observed that this assumption increased the baseline MCD by about 0.07. As can be seen, the IP systems in Hindi and Iraqi perform better than the baseline system. In Dari, the IP systems come close to the baseline but do not perform better. However, the trend of the Dari IP systems is similar to that of the other languages.

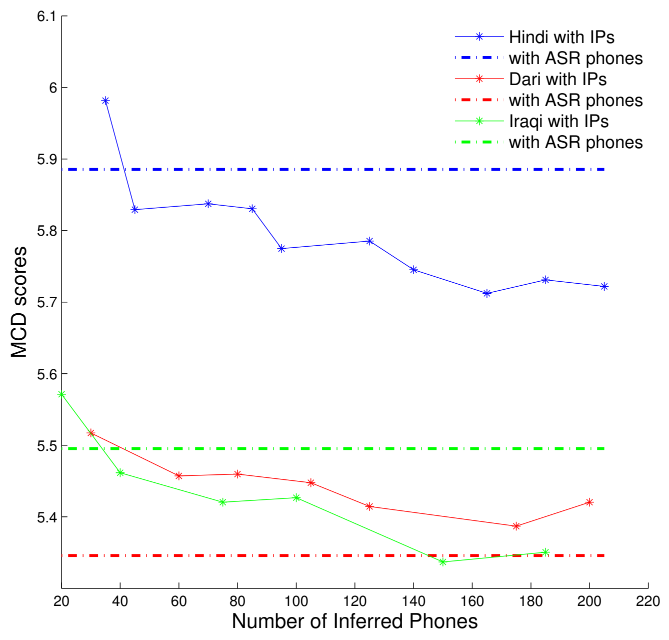

Fig. 5.3: MCD scores for Inferred Phones

As a comparison with a language that we have much more experience with, we also tested an English system using the true phonetic transcription as the baseline to look at the difference in performance between using knowledge-based phonetic transcriptions compared with inferred phones. The phonetic transcription was obtained using the CMUDICT phonetic lexicon(Weide, 2005) on the original text. The baseline MCD was 5.224. The best IP system, with 190 inferred phones, had an MCD of 5.493. This result is quite promising considering that a completely data-driven phoneset only has a 0.27 MCD difference from the baseline system which had phones derived from text. The difference looks especially small considering that the corpus we tested this on was the RMS voice of the CMU Arctic corpus. The text and phonemes in this corpus have been *extremely* well-studied for years, and are difficult to improve upon.

5.6 Deriving phonetic features for Inferred Phones

One potential problem that usually occurs with a completely data driven phoneset, as opposed to using phone recognition, is that it is usually impossible to get any phonetic information about inferred phones. The technique described in this chapter has a great advantage here since articulatory features are fundamental to our clustering process. By looking at the articulatory features that correspond to our inferred phones, it is straightforward to derive phonetic information for these.

Each of the leaves of our CART that we had used for clustering is a cluster that corresponds to a particular Inferred Phone. The mean AF of all the AF vectors in that cluster gives us a good estimate of the phonetic information about that Inferred Phone. This information can be used by the synthesizer in prediction.

We repeated our experiments on Hindi, Dari, and English with the inferred phonetic information. For our baseline, we take the best performing system from the previous set of experiments. For Hindi, this would be the IP system with 165 phones. For Dari, the baseline would be the same as the previous one. For English, it would again be the system with perfect phonetic transcriptions. Figure 5.4 shows the results in Hindi and Dari. In both languages, the IP system is able to perform better than the baseline systems.

In the English system, the baseline MCD was 5.224. After adding inferred phonetic information, the best IP system, with 190 phones, had an MCD of 5.431. This brings our system even closer to the best possible phonetic transcriptions.

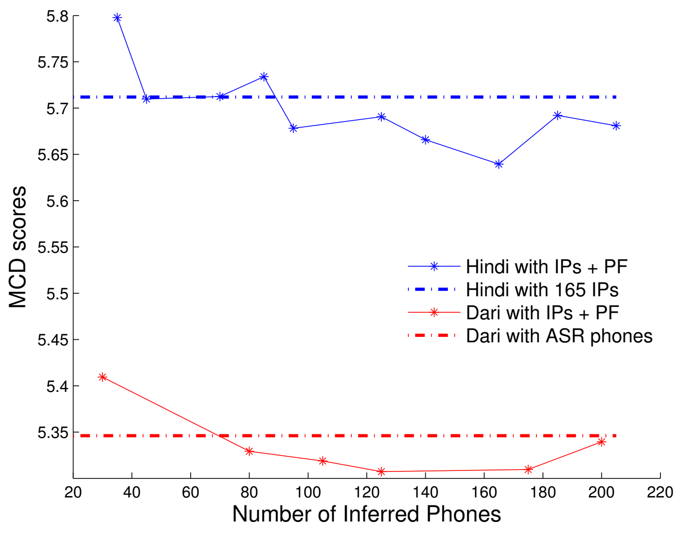

Fig. 5.4: IPs with phonetic information added

5.7 Discussion

While the technique described in this chapter is highly novel, and the results obtained highly promising, there is a missing piece of the equation though. The point of the phoneme or phone representation of text or speech is for them to be useful as an intermediary representation in between the two. The Inferred Phones have an obvious relation to the speech data in hand. The relation between them and the text is less obvious though. Preliminary experiments that tried to predict Inferred Phones based on just the text have not been very successful. While it is too early to rule out the possibility that we might succeed in doing so with more extensive experiments, it does require us to ask why we have problems predicting Inferred Phones but not the original phonemes from text. The answer to this lies in the distinction between the written text and the spoken language. While most people, including this author, have often assumed that speech and text are fairly interchangeable, speech science experts have yelled themselves hoarse in trying to make us acknowledge the differences between the two. Our recent experiments in trying to predict Inferred Phones from text seem to be adding more weight to what those experts say, and it certainly seems to be taking us into interesting new frontiers.

The work described in this chapter was first published in Muthukumar and Black, 2014.

6 An Integrated Objective Metric for Speech Synthesis

It has often been claimed that nearly all problems in machine learning can be formulated as optimization problems. And quite rightly so; all we would need to do is morph the task at hand into an objective function which can be minimized or maximized by the hundreds of algorithms available today. The amount of increase or decrease of the objective function's absolute value is also often a good metric of how well each algorithm works.

Automatic Speech Recognition research, for instance, uses Word Error Rate (WER). Word Error Rate is not a perfect metric, but it certainly is an excellent measure of speech recognition quality. The existence of such a measure has allowed ASR researchers to detect even the most marginal of improvements, and build techniques that capitalize on these. While this obsession with lowering WER may have led to some papers with dubious claims of success based on marginal improvements in WER, it is undeniable that a good objective metric can contribute more to a field's success than most algorithms can.

Speech synthesis has had a long history of borrowing algorithms from the speech recognition community(Dines et al., 2010). So, it is only natural that one of the earliest metrics of speech synthesis quality was also Word Error Rate. While for ASR, WER measures the number of words that the system can recognize in normal human speech, in synthesis WER is used to measure the number of words a human can recognize in synthesized speech. In terms of synthesized speech intelligibility, WER is a near-ideal metric. However, modern speech synthesizers have rendered obsolete the question of intelligibility. Statistical Parametric Speech Synthesizers are as understandable as natural human speech, and in several cases are even more understandable. This is illustrated in the case of the Blizzard Challenge in 2010 where teams were challenged to build the most understandable system in the presence of noise. The system that won by a large margin turned out to sound more robotic and buzzy than the other competing systems. This is because the uncannily consistent nature of synthesized speech greatly enhances understandability at the expense of naturalness.

Modern systems do have problems with *naturnalness* though. Even the best systems produce noticeable clicks, occasional hisses, and prominent buzzes that immediately reveal their synthetic origins to the human listener. Since these are fairly noticeable in plots of the speech signal, it would stand to reason that an objective evaluation of the speech signal parameters would detect the presence of these anomalies. The most successful parameterization of the speech signal are Mel Frequency Cepstral Coefficients, and appropriately the most commonly used metric of speech synthesis quality is distortion in the Mel Cepstral domain(Kubichek, 1993). Mel Cepstral Distortion (MCD for short) measures the Euclidean distance between natural speech in a held-out test set and synthesized speech of the same text produced by a speech synthesizer. Lower MCD scores indicate a higher quality synthesizer and experiments have shown that an improvement of 0.12 is equivalent to the improvement produced by doubling the amount of training data(Kominek et al., 2008).

While MCD has been a very successful metric for a number of years, it is not without its drawbacks. The most obvious is the necessity of a single reference audio sample to compare the synthesized speech to. This is unreasonable because it is impossible even for humans to say the same sentence five times in a row and produce speech identical at the individual sample level for each of the five. MCD also errs by computing the Euclidean distance on a frame by frame basis and then computing the mean over every frame. The result of this is that the metric pays no attention to the transitions between frames.

Another flaw is the choice of features. Mel Frequency Cepstral Coefficients were designed as an approximation of only the vocal tract structure, and a crude one at that. A large part of the identity of the human voice comes from the glottal flow which is completely orthogonal to the vocal tract, and so is completely ignored by MCD measurements. While it could be argued that the Mel Cepstrum is actually affected by the spectral characteristics of the glottal flow a little, we must remember that the Mel Cepstral coefficients became popular in ASR because they were very good at teasing apart the vocal tract characteristics from those of the glottal flow. So, there is no theoretical justification for using MCEPs as representative of the *whole* signal. An obvious example of the flaws of MCD is the comparison of Mixed Excitation systems(Yoshimura et al., 2001) with Pulse-Noise excited systems. Mixed excitation systems are preferred by a huge margin in human listening tests. Yet the MCD of these systems is nearly identical to the Pulse-Noise excited systems.

6.1 Why Speech Coding metrics are unsuitable

An inevitable suggestion received in every discussion of speech quality analysis is the use of PESQ(Rix et al., 2001). PESQ which stands for Perceptual Evaluation of Speech Quality is a highly successful metric for measuring the distortion caused by telephone channels when transmitting speech. Ostensibly, using a telephone channel distortion measure might appear to make sense. Both telephone channels and speech synthesizers produce distorted speech and so using a metric for one to measure another sounds reasonable.

The flaws in this analogy start revealing themselves once we delve deeper into each problem. The goal of the telephone channel is to transmit with near perfect detail the input speech on either end. Hence any arbitrary speech can be input to the channel, and the channel has to produce the *same* signal with minimum distortion. For a speech synthesizer, the speech produced on the output end need not correspond to any exact reference. All that is necessary is for the speech to sound like natural speech without the necessity of being identical at the sample level. To help us understand this difference better, consider the case of the fricative sound [f]. When this sound is put through a telephone channel, the output of the channel has to reproduce this sound exactly identical to the way it was input. A synthesizer though will use white noise as the source signal from which the fricative [f] is generated. This means that the output of the synthesizer is nearly *guaranteed* to be different from every single reference sample, but will still sound perfectly reasonable to human listeners.

Speech duration is another reason why speech coding metrics are inadequate. If we were to take an utterance and increase the length of each phone in the utterance by 10ms, the difference would be barely perceptible to the listener, but would be marked as highly distorted by PESQ and other telephone channel metrics.

PESQ is by no means unique in having these drawbacks. Quackenbush et al., 1988 is a comprehensive description of nearly every single metric of speech quality today. But nearly all of the metrics described in the book suffer from the same problems as PESQ. This is not meant to disparage these metrics though. These are excellent metrics for speech coding tasks. However, the kind of distortions that occur as a result of channel transmission errors are very different from the errors that occur in speech synthesis. So speech coding metrics are of little use to us.

6.2 Building a model of human speech

Objective metrics of speech quality can be classified into two approaches. Metrics like PESQ, WER, and MCD are *full-reference* metrics. This means that there needs to exist a gold-standard or at least a pseudo-gold-standard reference that the output can be compared against. This is problematic for speech synthesis because there are an infinite number of possible ways in which the same utterance can be spoken even if we were to leave out the effects of emotion and intonation. Ideally we would want to strive for a *no-reference* metric which can give us a score of speech synthesis quality without needing a reference.

When we as humans listen to examples of synthesized speech, we can immediately pick the most natural one without the need of any reference. Through years and years of exposure to human speech, we have an excellent idea of what human speech sounds like. So we are in effect comparing the synthesized utterances to our mental model of human speech. Mariniak, 1993 describes an interesting idea for evaluating speech synthesis quality along the same lines. The proposed idea uses speech samples from multiple speakers to derive features appropriate for evaluating naturalness. Synthesized speech can then be evaluated on the basis of these features. Hinterleitner et al., 2010 however mentions that Mariniak's proposal was never really implemented, and only existed as an interesting idea for a long time.

The first attempt at actually building such a metric based on Mariniak's idea is described in Falk and Möller, 2008. Here Hidden Markov Models (HMMs) are used to build a generative model of the speech with training data from several reference speakers. Hidden Markov Models are highly suited to modeling speech due to their inherent ability to deal with sequences of arbitrary length. Conventional objective metrics such has Mel Cepstral Distortion have to rely on techniques such as Dynamic Time Warping (DTW) to first align sequences of varying length before doing a frame by frame comparison. This necessity of using DTW has the effect of adding or removing frames from the objective metric computation. The best alignment of frames is one where there is the least amount of difference between the two sequences. It is easy to think of a case where there is a synthesized speech utterance with frames filled with garbage at every fifth time step. The DTW alignment will discard these frames in the MCD computation even if these will be obvious to human listeners. The MCD in this case will be erroneously low even if the listener perception says otherwise. HMMs, on the other hand, will care about *every* single frame in the synthesized speech and no frames can be discarded.

The naturalness of synthesized speech is estimated by computing the normalized likelihood of the speech given the HMM using the Forward algorithm(Rabiner, 1989). In Falk and Möller, 2008 the authors use 8-state HMMs with 16 Gaussians per state. The Gaussians are assumed to have diagonal covariance. 12th-order MFCCs analyzed in 25ms windows and at 10ms frame shifts were used along with a delta coefficient computed of the zeroth MFCC coefficient (power). The results described are highly impressive with correlations as high as 0.81 in some dimensions. However, in a later paper(Hinterleitner et al., 2010), the authors apply a much more sophisticated version of this approach on data from the Blizzard Challenges of 2008 and 2009. The results on this are substantially less convincing. The technique works fairly well on identifying the differences in quality between utterances produced by the same synthesizer, but fails quite badly when asked to rank synthesizers of many different types as is the case with the Blizzard Challenge data.

However this approach must definitely get credit as being the first integrated objective metric for speech synthesis. WER focuses exclusively on the underlying lexical content in the speech signal and pays no attention to the characteristics of the speech signal itself. MCD on the other hand only cares about how closely the speech signal matches the reference and does not directly care about the underlying text. In the HMM based metric described earlier, the authors consider both the text as well as the speech signal in computing the error. The HMMs are assembled together according to the phonemes derived from the text. The likelihood scores output by these HMMs though is dependent on the speech signal itself, thereby taking into account both the text and speech signal.

The approach we will describe in this chapter, while novel, builds heavily upon this HMM based technique. As a necessary pre-requisite as well as for academic curiosity, we replicated the experiments of Falk and Möller, 2008 albeit on different datasets and different testing conditions. While those authors trained separate models for male and female speakers, we chose to model them jointly. While those authors trained 8-state HMMs with 16 Gaussians per state, we also trained HMMs with 5 states and 2 Gaussians per state. This was to study the effects of model complexity.

The Baum-Welch algorithm was used to train 3 HMMs per phoneme. All of our experiments were on English. Our synthesizer assumes 42 phonemes (40 phonemes from the language, pause, and short silence), and so this resulted in 126 HMMs in total. The models were trained on five voices from the CMU Arctic database(Kominek and Black, 2004): BDL, CLB, JMK, RMS, SLT. BDL and RMS are American males. SLT and CLB are American females. JMK is a Canadian male. Each voice corpus consists of 1 hour of speech, and all of the voices contain the exact same sentences. This gave us a well balanced corpus for training our HMMs on. The features we trained on are more or less identical to the features used in the original idea proposal: 13-dimensional MFCCs which are mean and variance normalized. We did not however use any delta coefficients or F_0. The likelihood estimation code for the HMMs was a modified version of the code available in the EHMM toolkit(Prahallad et al., 2006).

6.3 The trouble with MFCCs

We used the AWB and KSP voices, also from the CMU Arctic Database, as test voices for our HMMs. AWB is a Scottish male while KSP is an Indian male. These voices are recorded under similar recording conditions and with the same set of sentences. The objective of this preliminary test was to test for naturalness, so using the same set of text sentences for training and testing was not a problem. The AWB and KSP voices were each built with a 90-10 train-test split. The synthesizers built for each of these voices were then made to synthesize the held out test set. Table 6.1 shows the results of four sets of such experiments. In the first set, the AWB and KSP voices were built on the whole 1 hour of data. In the second set, the AWB and KSP voices were built with a randomly chosen subset of about 20 minutes. The likelihoods shown in the table have been scaled to be positive. This is simply because it is harder to humans to visualize the rank of negative numbers.

The results are compared to Mel Cepstral Distortion for the same voices. We may have ranted at length about how inappropriate MCD was for measuring the naturalness of synthesized speech, but for the scale of differences shown in the Table 6.1 MCD is quite accurate. So a strong correlation between the HMM likelihood and the MCD would be a good thing. (By strong, we mean a strong *negative* correlation. For MCD, lower is better but for HMM likelihoods, higher is better).

Tab. 6.1: Comparison of MCD and Likelihood scores

Voice	MCD	Likelihood
AWB (1 hr)	4.29	16.97
AWB (20 mins)	4.61	16.93
KSP (1 hr)	4.89	16.54
KSP (20 mins)	5.23	16.42

It might initially appear that the results shown in Table 6.1 look promising. The likelihoods correctly depict the synthesizers trained on the 1 hour corpora as being better than their counterparts trained on the subset. They also correctly describe the KSP synthesizer as sounding less natural than the AWB voice. Before we jump to conclusions and start pulling out the party favors though, let's run an additional experiment where we also compute the likelihoods on the natural speech from these corpora. The results of this are shown in Table 6.2. The first four rows of this table are taken directly from the previous table. Since the likelihoods are computed directly on the reference, there is no MCD that corresponds to this likelihood.

Tab. 6.2: Likelihoods of natural speech

Voice	MCD	Likelihood
AWB (1 hr)	4.29	16.97
AWB (20 mins)	4.61	16.93
KSP (1 hr)	4.89	16.54
KSP (20 mins)	5.23	16.42
AWB (natural)	–	**16.48**
KSP (natural)	–	**16.31**

Rather ironically, our HMMs seem to think that natural speech sounds less natural than synthesized speech does! Strange as this may seem, this result was not entirely unexpected. All statistical parametric speech synthesizers segment speech data into units using HMMs. This is exactly what we did to train our general speech HMM too. Statistical Parametric synthesizers deal with temporal data by assuming that the data was drawn from a Gaussian distribution, and then storing the mean of the Gaussian as representative of all the data. Our general speech model HMMs use Gaussian Mixture Models to model the emission probabilities. Basic probability theory tells us that the highest probability of a Gaussian distribution is at its mean. Since the frames of speech predicted by our synthesizers are *always* the means of the Gaussians, it makes sense that the HMMs think that the synthesized speech is more likely than the natural speech. Note that this is true despite the fact that the HMMs were trained on natural speech, not synthesized speech.

This problem is difficult to remove but it can be mitigated by taking a closer look at the features we have used for both the synthesizer as well as in our general speech model HMM. In both cases, we have used Mel Cepstral coefficients (there are subtle differences in the extraction procedures between the two, but for the case considered here they are irrelevant). While Mel Cepstral coefficients are the best available features for synthesis today, they are not necessarily the best for evaluating speech quality. We have already mentioned that these coefficients only capture information specific to the vocal tract, and highlighted that with an example. But Mel Cepstral coefficients are also unsuitable because they were designed with the intention of maximizing reconstruction quality(Tokuda et al., 1994). Reconstruction quality is fairly correlated with the naturalness, but a more ideal choice would be a set of features that pay more attention to the details that the human ear would.

6.4 Auditory Models

There has been a tremendous amount of effort put into understanding the working of the human auditory mechanism. Based on our understanding of this mechanism, several auditory models have been proposed both for use in speech technology systems as well as to enhance our understanding of human hearing. A detailed analysis of the whole hearing mechanism and the various models of audition used in modern speech recognition systems is available in Stern and Morgan, 2012. However in the interest of keeping this thesis as self contained as possible, we provide a brief summary of human hearing and the various models we will be using in this thesis.

The human ear can be sectioned into three regions: the outer ear, the middle ear, and the inner ear. The outer ear is shown in Figure 6.1 borrowed from Gray, 1918. It consists of the Pinna which is the visible part of the ear, which serves to direct sound into the ear canal. The pinna also helps in sound localization to detect both the horizontal as well as vertical direction in which the sound originates. The pinna channels sound into

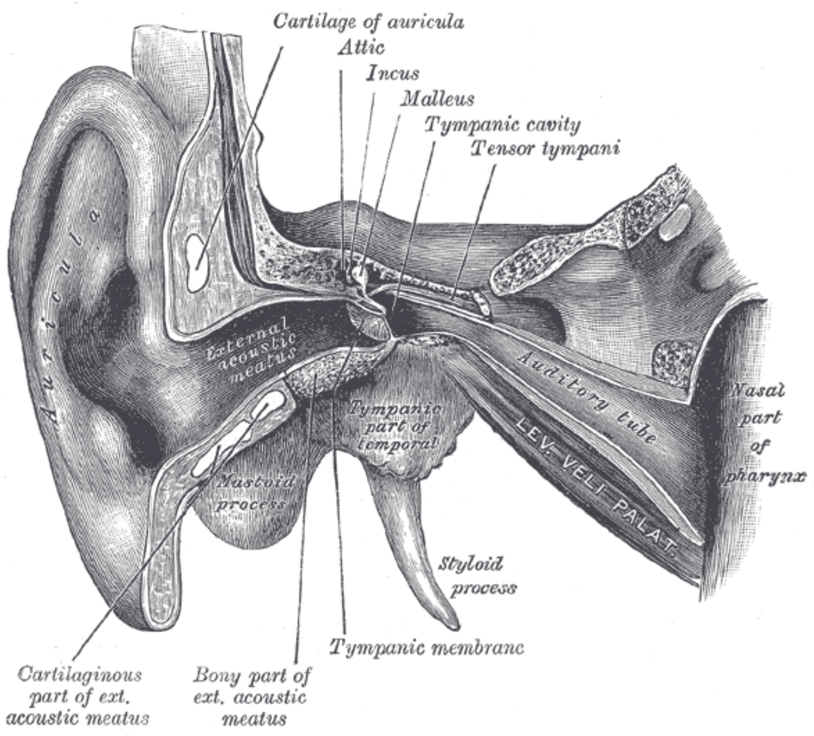

Fig. 6.1: Anatomy of the Outer Ear

the ear canal. At the end of the ear canal is the tympanum, colloquially called the ear drum. This is a flap of skin which vibrates much like the skin of a drum in response to the sounds that are received through the ear canal.

The middle ear is shown in Figure 6.2 (also taken from Gray, 1918). It consists of three small bones also called ossicles. The Malleus, also known as the hammer, is directly connected to the ear drum. The malleus is connected to the Incus, which is sometimes referred to as the anvil. The malleus and the incus convert the vibrations of the ear drum to mechanical movements. They also serve to amplify the sounds substantially. The bones of the middle ear can also protect to a certain extent in response to very loud sounds. The movement of the incus translates to an up-and-down motion of the stirrup shaped Stapes.

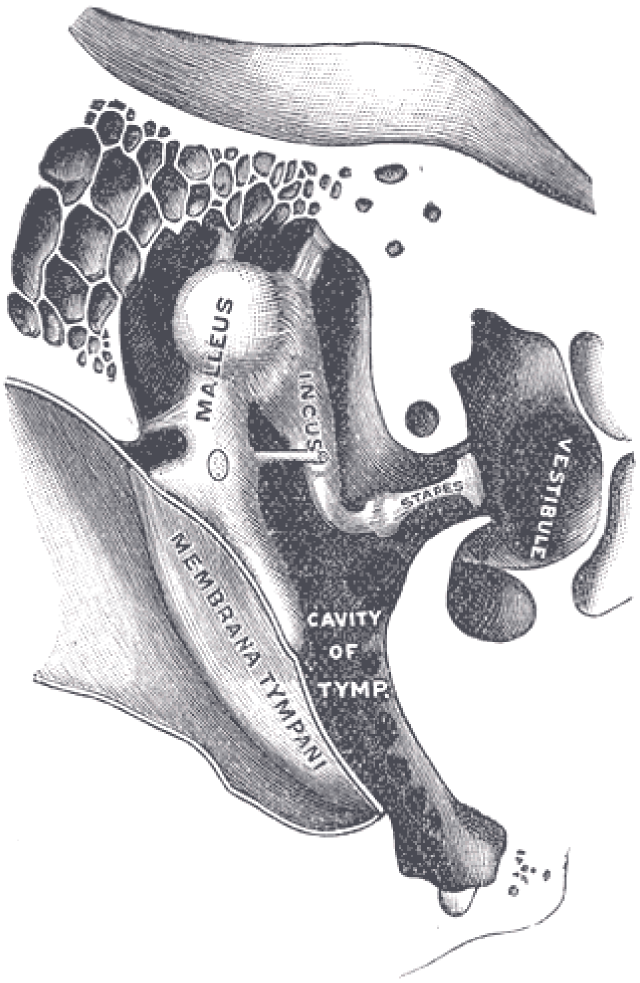

Fig. 6.2: Anatomy of the Middle Ear

The inner ear is shown in Figure 6.3 (again from Gray, 1918). The stapes of the inner ear is connected to the cochlea, which is a coiled organ much like a snail's shell. The cochlea is filled with fluid. The up-down motion of the stapes sets up vibrations in the cochlea.

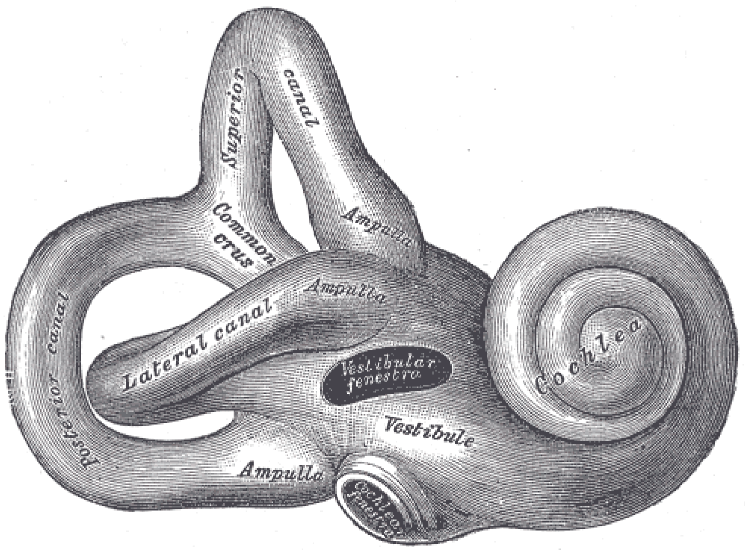

Fig. 6.3: Anatomy of the Inner Ear

Running down the middle of the cochlear spiral is the cochlear duct. The region of the duct closer to the base of the cochlea is called the basilar membrane. The organs of corti which are located in the basilar membrane are responsible for transducing the vibrations of the basilar membrane into nerve impulses. The organs of corti are marked in the cross section of the cochlear duct in Figure 6.4.

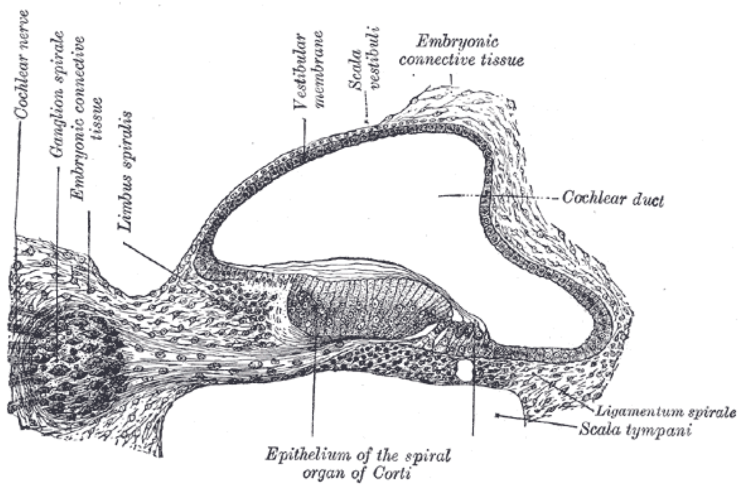

Fig. 6.4: Cross-section of the Cochlear Duct

The basilar membrane is not uniform throughout the cochlea, and has evolved so that the membrane closer to the apex vibrates at lower frequencies while the part of the membrane closer to the base vibrates at higher frequencies. Whenever a particular region of the basilar membrane vibrates, hair cells in the organs of corti fire and send impulses through the cochlear nerve to the brain. These hair cells are visible in the cross section of the organ of corti shown in Figure 6.5.

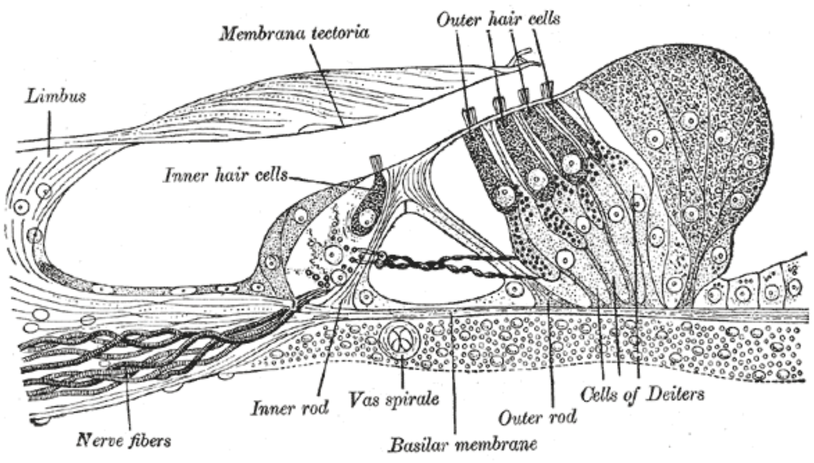

Fig. 6.5: Cross-section of the Organ of Corti

The result of all this sophisticated processing is an exceptionally sound mechanism (pardon the pun) for amplification, automatic gain correction, and frequency resolution.

There have been numerous attempts at creating models that replicate the actions of the human ear. This thesis considers the Lyon Cochlear model(Lyon, 1982; Lyon and Lauritzen, 1985), the Seneff ear model(Seneff, 1990), and Roy Patterson's gammatone filterbank(Holdsworth et al., 1992) coupled with Ray Meddis' hair cells(Meddis, 1986). While there exist several other popular models, we chose these three because well documented implementations exist in the Slaney Auditory Toolbox(Slaney, 1998).

6.4.1 The Lyon Cochlear model

The Lyon Cochlear Model models the workings of the middle and inner ear using a cascading set of filters which feed into Half-Wave Rectifiers which in turn feed a set of interconnected Automatic Gain Control Blocks. The actions of the external ear and the ear canal are believed to only be a simple linear filter and so are ignored. A block diagram of this model is shown in Figure 6.6.

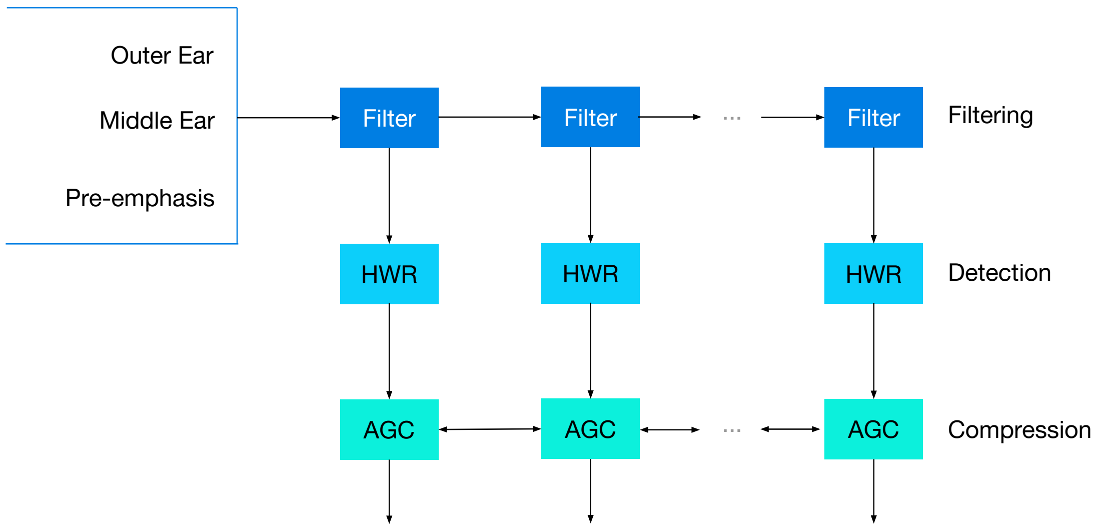

Fig. 6.6: Lyon Cochlear Model

The exact implementation used in the Slaney Auditory toolbox uses notch filters and resonators rather than the components described above for reasons of computational efficiency.

6.4.2 The Seneff Ear model

The Seneff Ear model consists of 40 channels that cover the range from 130-6400Hz. Each channel consists of a three stage processing setup that involves critical band filtering, an approximation of the basilar membrane response, and envelope detection. This is shown in Figure 6.7

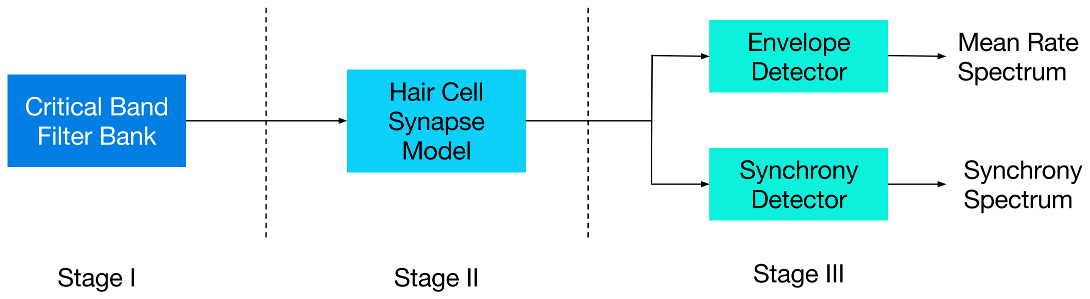

Fig. 6.7: Seneff Ear Model

The second stage of this model is shown in more detail in Figure 6.8. The full details of this model are available in Seneff, 1990.

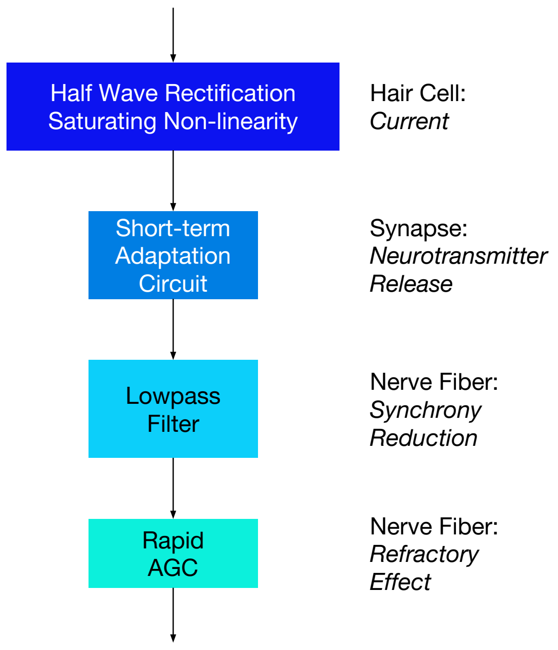

Fig. 6.8: Stage II of the Seneff Ear model

6.4.3 The Patterson Filterbank with Meddis hair cells

The Patterson Cochlear model consists of a Gammatone filterbank spaced at equal intervals on the Equivalent Rectangular Bandwidth (ERB) scale. The ERB Scale is defined by the equation:

$$ERB(f) = 24.7 * \left(4.37 * \frac{f}{1000} + 1\right)$$

The output of this filterbank is then fed into Meddis hair cells. The full details of the Meddis hair cells are beyond the scope of this thesis and so will be omitted. The details of these is available in Meddis, 1986.

6.5 Experiments and Results

We repeated our HMM experiments described in the earlier sections with a separate set of HMMs for each of the three auditory models described earlier. The default setting for the Lyon Cochlear Model uses a cascaded set of 86 filters. This model outputs a stream of numbers for every single sample in the input. To make this more manageable, we decimate the output of this so that the output is equivalent to a 5ms frame shift. The output of this model is therefore 86 x number of 5ms frames in the utterance.

Empirically we found that the HMMs worked best when the result was scaled to be in the interval from 0-0.3.

As mentioned earlier, the Seneff Ear model uses 40 channels. As with the Lyon model, the output of the model was decimated so that it was equivalent to a 5ms frame shift. The output of this model is therefore 40 x number of frames, and was scaled to be between 0 and 1.

The default setting for the Patterson filterbank uses 80 filters. This model too had to be decimated in the same way as the previous two. The output of this is passed through the Meddis haircells and then low-pass filtered. The output of this model is 80 x number of frames. Like the Lyon ear model, the output worked best when scaled to be between 0 and 0.3.

The HMMs trained for each of the models were identical to the settings used for the MFCCs. Separate HMMs were trained for the beginning, middle, and end of each phoneme (40 English phonemes + pause + short silence). The training again was on the same five voices from the CMU Arctic Database: BDL, CLB, JMK, RMS, and SLT.

For the testing cases though, in addition to the AWB and KSP voices, we also computed the scores on nine other voices: TATS, CXB, F1A, F2B, F3A, M1B, M2B, M3B, M4B. These span a wide variety of sizes and recording styles. Table 6.3 lists the MCD and the likelihood scores of these voices with HMMs trained on MFCCs, Lyon model features, Seneff model features, and the Patterson and Meddis features.

Tab. 6.3: Different types of evaluation on 11 different voices

Voice	MCD	LyonEar	Meddis	Senn	MFCCs
TATS	4.956173	172.438	193.162	62.237	-33.856
CXB	4.995510	184.120	195.238	64.703	-32.809
F1A	5.095489	185.083	194.053	57.131	-33.638
F2B	4.617769	188.097	193.90	61.159	-33.247
F3A	4.811825	186.714	193.430	55.054	-33.350
M1B	4.946825	183.790	194.043	60.327	-33.117
M2B	5.064132	188.196	195.457	66.332	-33.291
M3B	4.706359	186.501	195.292	60.187	-32.943
M4B	4.815951	187.251	193.524	61.772	-33.064
AWB	4.297440	184.093	192.142	63.091	-33.031
KSP	4.892248	178.322	193.742	61.339	-33.461

The above table is not particularly interesting. To get a more useful sense of the numbers in table 6.3, let's look at the correlations between MCD and the other numbers. We might have strongly laid out a case against MCD in the earlier sections but MCD is still known to be correlated with naturalness of synthesized speech. Table 6.4 lists the correlations between four likelihood scores and *negative* MCD. For MCD scores, lower is better while for likelihood scores, higher is better. So the correlations are computed against the negative of the MCD.

Tab. 6.4: Correlation between MCD and Likelihood scores

Feature	Correlation with MCD
MFCCs	0.3770
Lyon	0.1713
Meddis	-0.5664
Seneff	0.0044

The MFCC results seem to be the most correlated with the MCD scores. This is not unexpected. After all, computing the likelihood with MFCCs is closely tied to computing distortion in the Mel Cepstral domain. The Lyon model seems to have the highest correlation, and the Seneff model seems almost uncorrelated. For some reason, the Meddis model seems to have a strong negative correlation. The p-values for all of these correlations are well below 0.01.

The true test of an objective metric's performance is not how well the metric is correlated to other metrics but rather how correlated it is to *human* evaluations. Human evaluations of synthesized speech are generally quite expensive. However there exists plenty of data from the various Blizzard Challenges that have been run over the last ten years. This data is particularly interesting as it contains synthesized speech from a wide variety of unit-selection, Statistical Parametric, and Hybrid systems from all over the world. The Mean Opinion Score (MOS) of these examples as rated by human listeners is also publicly available.

The particular dataset we considered is the Blizzard Challenge from the year 2013. This dataset was chosen at random and there was no reason why this experiments could not have been run on other years too. The intelligibility test results and the similarity-to-speaker test results are not directly relevant for our task as we are most interested in testing for naturalness. So these were removed from our experiments. The 2013 challenge focused on audiobooks and so had sections with paragraph length utterances. We did not want to start off with an unusual task for the first set of experiments; so these too were removed.

The remaining data consisted of a set of utterances from broadcast news, audio books, semantically unpredictable sentences, and neutral sentences. These were spoken by every system that participated that year. There were about 560 utterances in total, and each was assigned a MOS score by multiple listeners. We computed the average MOS score for each utterance of each synthesizer. This was the target that our objective metric has to match.

We computed the likelihood scores for each of those utterances for the HMMs trained on MFCCs, Lyon Ear features, Seneff Ear features, and the Patterson and Meddis features. The correlations between the human listener ratings and the likelihood scores of each set of HMMs is reported in the tables below. Table 6.5 reports the results for the 5-state 2-Gaussian case while table 6.6 reports the same for the 8-state 16-Gaussian case.

Tab. 6.5: Correlations for 5-state 2 Gaussian HMM models

Feature	Correlation with MOS
MFCCs	-0.2019
Lyon	0.1850
Meddis	0.1985
Seneff	0.0910

Tab. 6.6: Correlations for 8-state 16 Gaussian HMM models

Feature	Correlation with MOS
MFCCs	-0.2214
Lyon	0.0214
Meddis	0.1535
Seneff	0.0590

The results support our suspicions that MFCCs are not particularly well correlated with human listener ratings. In both the simple as well as the more complex HMM models, the MFCC models had the strongest *negative* correlations among all of the tested models.

The Seneff model is consistently uncorrelated while the Meddis model exhibits a reasonable correlation. The Lyon Ear model is relatively correlated in the 5-state case but is uncorrelated in the 8-state case. This could perhaps be explained by overfitting, but more rigorous testing is necessary before any conclusions can be made.

The correlations listed in table 6.5 might not initially appear very promising but we must remember the difficulty of the task at hand. Hinterleitner et al., 2010 who we cited earlier in this chapter uses a much more complex approach using voice activity detection, feature selection, and linear regression; and yet, the results shown do not have much higher correlations either. It is likely that applying more complex composite algorithms on top of these scores will yield better correlations between the MOS scores and our predictions. However the point of this chapter was to investigate integrated objective metrics using auditory models, and for that purpose using more complex algorithms to do late fusion is unfair.

6.5.1 Within-synthesizer correlations

One interesting idea is to treat each speech synthesis system as a separate speaker, and then compute scores (and correlations) within the utterances produced by each system. Perhaps this might help us make better sense of the data. The results of these are reported in table 6.7. The results reported are for the 8-state 16-Gaussian case.

Tab. 6.7: Correlations per system

System	MFCCs	Lyon	Meddis	Seneff
A	-0.2446	0.0177	0.2362	0.0240
B	0.2859	0.2086	-0.1705	0.1547
C	-0.0136	-0.1413	-0.2622	-0.0179
F	0.0840	0.0562	0.0664	0.0640
H	-0.0280	0.1441	0.1005	-0.0418
I	-0.0391	0.0773	0.1164	0.1869
K	0.1154	0.1026	0.1928	0.1032
L	-0.2691	-0.0155	0.0810	-0.0328
M	-0.0161	-0.1108	0.1109	0.0632
N	-0.1389	0.0423	0.1866	0.1756
P	0.2642	0.1846	0.1933	0.1788

It is difficult to tease apart any interesting trends from the above table though. So it might be worth separating the columns above based on the type of synthesizer. The errors made by unit-selection synthesizers differ from those made by statistical parametric systems. Perhaps the correlations appear random due to these differences.

The results released by the Blizzard Challenge organizers are anonymized. However the teams that participate tend to identify themselves in the papers that get published as part of the challenge. Not every team that participates does this though, and so the next few tables will report the results for systems that do self-identify.

Despite carefully perusing through the results of the above tables, it is difficult to pinpoint any interesting trend in the data. Perhaps a more extensive set of experiments on more datasets is necessary in the future.

Tab. 6.8: Correlations for Unit Selection Synthesizers

System	MFCCs	Lyon	Meddis	Seneff
B	0.2859	0.2086	-0.1705	0.1547
L	-0.2691	-0.0155	0.0810	-0.0328
N	-0.1389	0.0423	0.1866	0.1756

Tab. 6.9: Correlations for Statistical Parametric Synthesizers

System	MFCCs	Lyon	Meddis	Seneff
C	-0.0136	-0.1413	-0.2622	-0.0179
H	-0.0280	0.1441	0.1005	-0.0418
I	-0.0391	0.0773	0.1164	0.1869
P	0.2642	0.1846	0.1933	0.1788

Tab. 6.10: Correlations for Hybrid synthesizers

System	MFCCs	Lyon	Meddis	Seneff
K	0.1154	0.1026	0.1928	0.1032
M	-0.0161	-0.1108	0.1109	0.0632

6.6 Discussion

One of the core parts of the approach discussed in this chapter is the use of a HMM to model our time series data. Over the last four or five years though, there has been a massive shift in the way time series data have been modeled. Deep learning techniques have surpassed traditional HMM based techniques in almost every task. One major advantage that deep learning techniques hold is their ability to model arbitrary complex functions. Additionally they do not have to be constrained by assumptions such as that of the emission probabilities being produced by a Gaussian Mixture Model. The Markovian assumption which is key to the power of the HMM is actually blatantly wrong for speech data. Speech has interesting long and short term dependencies. For instance, if the word 'stock' was said once in a particular sentence, it is highly likely that the word will occur again in the near future. Speech is not just dependent on the past either, the sound at any instance of speech is influenced both by past sounds as well as future sounds. It is impossible for the HMM to model such relations. Deep learning techniques such as the use of Long Short Term Memory cells(Hochreiter and Schmidhuber, 1997), and their bidirectional variants are fully capable of modeling all the complex time relations of the speech signal. However, training deep learning models is difficult because of the huge choice of hyperparameters and the unintuitive way in which the training process behaves. Unlike HMMs, the models are also very difficult to interpret and understand. Speech synthesis evaluation is a hard problem even for humans, and so the labels are not necessarily very accurate. So in summary, the problem at hand has noisy training data of limited size, and with feature sets that have not been tested before. So it stands to reason that our first choice of model was a HMM which is really well understood. Bidirectional LSTMs though would be the next logical technique to try.

Evaluating the quality of speech synthesis is probably the hardest problem addressed in this thesis. The idea of using auditory models might still be at the early stages, but it really opens up the idea of trying to evaluate the quality of synthetic speech by caring about how humans hear. An interesting idea worth exploring is to study what humans pay attention to when evaluating the *aesthetic* quality of sound. We obviously prefer some voices to others, and we obviously prefer sounds like violins to sounds like jackhammer noises. There has been a lot of effort into quantifying such preferences in the field of computational music. So in the future, it is worth borrowing ideas from those fields to apply to speech synthesis.

7 Where do we go from here?

At the beginning of this thesis, we argued strongly against pipelines which treated each component as being independent of the rest. We hypothesized that moving towards more integrated pipelines would yield better systems. We also therefore described one set of experiments in each of the four preceding chapters that would explore one aspect of this idea. Before we consider directions to move in the future, it is worth reviewing all the results presented to see how well they support our hypothesis.

In chapter 3, we described a technique that uses phoneme information from the text to better fit the LF model to the residual. In listening tests as well as in objective metrics, the use of the LF model outperformed existing mixed-excitation systems. In objective measurements, the LF model that used phoneme information was shown to fit better than the model which did not. So this was clearly a victory for the integrated model front.

In chapter 4, we used an RNN to bridge the gap between the prediction algorithm and the waveform generation algorithms by acting as a post-filter that connected the two. The RNN improves results substantially over the existing systems, and can still work in addition to the current most successful post-filter, MLPG. The use of the RNN is unique not only because we used information from the front-end to help the post-filter, but also because it still leaves open the possibility of adding more features as input to the RNN. This should count as additional support for considering integrated models.

We discussed the idea of deriving phonemes directly from the audio in chapter 5. While the idea of an Inferred Phoneme (IP) certainly falls under the category of integrated models, the true motivation behind this approach was primarily to investigate zero resource speech synthesis. The IPs work very well if we were to only care about integration. However, a lot more testing is required to see how useful they will be in zero resource speech synthesis.

Chapter 6 on evaluation metrics presents the most difficult-to-interpret results of this thesis. The use of auditory models seems logically sound, and the HMM based approach definitely is a metric that cares about multiple aspects of the synthesizer. However it is hard to conclusively say that the approach succeeded or that it failed. On the one hand, we do get reasonable correlations with human ratings. But on the other, the correlations are not very high and the analysis of the per-system correlations did not yield very interesting results. The only strong conclusion we can make based on these results is that evaluating speech synthesis quality is really really hard.

7.1 Future Directions

We have had a whirlwind run-through of many different ways in which the components of the acoustic model of a speech synthesizer can be integrated to form a more cohesive system. These however were all considered in isolation. The ideal speech synthesizer framework should instead look something like the one shown in Figure 7.1.

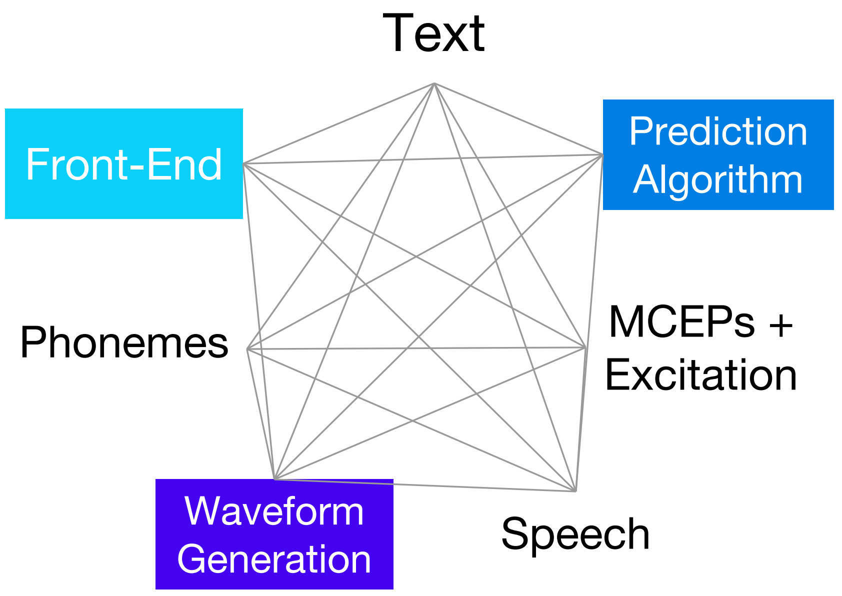

Fig. 7.1: A truly integrated statistical parametric synthesizer

Every component in the above framework is connected to every other component in the synthesis system. The decisions that each component makes is influenced by every other component in the network, and in turn each component's decision will influence every other component in the network. Like a strong football team, each part builds upon the strengths and weaknesses of the others. Any weak part of the system is substantially strengthened through the influence of the other parts.

Building a joint model of every aspect of the speech synthesis system could have other benefits apart from pushing for more natural sounding synthetic speech. Akin to a generative model, we could train the system on a wide variety of speech data and make interesting inferences from the resulting system. Along the same lines as Chapter 5 where we inferred phonemes based on speech, we could also infer a text representation for languages without an orthography. We could even infer a standardized writing system for languages which are highly diglossic.

Taking this joint generative model to its extreme, the lines between speech recognition and speech synthesis could get blurred. After all in this model, speech recognition would simply be the inference of text given speech. Synthesis would simply be the inference of speech given text. We could even add the missing features of F_0, prosody, and durations into the mix.

The more machine-learning-literate among us will have immediately noticed that the structure of this integrated generative model of speech synthesis closely resembles that of a Markov Random Field (MRF)(Kindermann, Snell, et al., 1980). A Markov Random Field is an undirected graphical model that defines the relationship between the various variables in a probability density function. We could think of each of our components in our synthesis system as a variable in an MRF. This would allow us to easily test the effectiveness of including or removing connections between components as it suits our needs. A fully connected MRF will be extremely powerful but inference over all the variables will be extremely difficult. A more sparsely connected MRF will be computationally more efficient, but it might push us back into our old habits of thinking of the synthesis system as more of a pipeline. Moving to a purely graphical model type of representation will also allow us to add additional variables for factors like room acoustics, emotional characteristics, and gender. Also possible is the addition of latent variables that can help us represent other factors which we might not have taken into consideration while building the system.

Building upon the latent variable idea, we can consider dispensing with the components of the system altogether and replacing them with latent variables. One can even imagine adding layers of latent variables in a structure like the Deep Boltzmann Machine(Salakhutdinov, 2009) that has been receiving a lot of attention in the last few years because of the recent meteoric rise of deep learning techniques. The system would take as input, text and the raw waveform, and then build a massive deep generative model with hundreds if not thousands of latent (hidden) variables with no further details necessary from any expert. This idea did not originate with me though; Pallavi Baljekar and others have publicly discussed much more detailed and thoroughly planned versions of this much earlier. There is no known published work describing these ideas though.

This is probably due to the nascent nature of these ideas. I expect to see experimental results from these ideas very soon from the researchers who work on them.

The idea of getting rid of every carefully designed component of the speech synthesizer in favor of purely inferred latent variables might seem outlandish to us. We must however remember the state of speech recognition and natural language processing systems in the early days of the field. The systems were all lovingly hand-engineered with experts writing rules for every scenario that they could think of. It is likely that the leading experts at the forefront of the field of those times held the strong belief that theirs was the right way to solve the problem at hand. In the same way that statistical techniques put an end to such beliefs, it is *inevitable* that a future novel and completely unexpected idea will put an end to current paradigms of doing speech synthesis. When that happens, this and most other theses in the field will be rendered obsolete. So, rather than strongly argue that the ideas in this thesis will stand the test of time, I am willing to be satisfied if this work merely ends up as a stepping stone in the development of future ideas.

Bibliography

Alku, Paavo (1992). "Glottal wave analysis with pitch synchronous iterative adaptive inverse filtering". In: *Speech communication* 11.2, pp. 109–118 (cit. on p. 16).

Anumanchipalli, Gopala Krishna, Ying-Chang Cheng, Joseph Fernandez, et al. (2010). "KLATT-STAT: knowledge-based parametric speech synthesis." In: *SSW*, pp. 206–210 (cit. on p. 7).

Anumanchipalli, Gopala Krishna, Luís C Oliveira, and Alan W Black (2011). "A Statistical Phrase/Accent Model for Intonation Modeling". In: *Twelfth Annual Conference of the International Speech Communication Association* (cit. on p. 21).

Association, International Phonetic (1999). *Handbook of the International Phonetic Association: A guide to the use of the International Phonetic Alphabet*. Cambridge University Press (cit. on p. 36).

Birkholz, Peter, Dietmar Jackèl, and Bernd J Kröger (2006). "Construction and control of a three-dimensional vocal tract model". In: *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*. Vol. 1. IEEE, pp. I–I (cit. on p. 7).

Black, Alan W (2006). "CLUSTERGEN: a statistical parametric synthesizer using trajectory modeling." In: *INTERSPEECH* (cit. on pp. 8, 19, 39).

Black, Alan W and John Kominek (2009). "Optimizing segment label boundaries for statistical speech synthesis". In: *IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE (cit. on p. 22).

Black, Alan W, H Timothy Bunnell, Ying Dou, Daniel Perry, Tim Polzehl, et al. (2011). "New Parameterizations for Emotional Speech Synthesis". In: (cit. on p. 16).

Black, Alan W, H Timothy Bunnell, Ying Dou, et al. (2012). "Articulatory features for expressive speech synthesis". In: *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*. IEEE, pp. 4005–4008 (cit. on pp. 37, 38).

Breiman, Leo, Jerome Friedman, Charles J Stone, and Richard A Olshen (1984). *Classification and regression trees*. CRC press (cit. on pp. 20, 40).

Cabral, Joao P, Steve Renals, Korin Richmond, and Junichi Yamagishi (2007). "Towards an improved modeling of the glottal source in statistical parametric speech synthesis". In: *Proceedings of the 6th ISCA Workshop on Speech Synthesis* (cit. on p. 16).

Carreira-Perpiñán, Miguel A and Weiran Wang (2012). “Distributed optimization of deeply nested systems”. In: *arXiv preprint arXiv:1212.5921* (cit. on p. 31).

Chen, Ling-Hui, Tuomo Raitio, Cassia Valentini-Botinhao, Junichi Yamagishi, and Zhen-Hua Ling (2014). “DNN-based stochastic postfilter for HMM-based speech synthesis”. In: *Proc. Interspeech, Singapore, Singapore* (cit. on p. 26).

Chen, Ling-Hui, Tuomo Raitio, Cassia Valentini-Botinhao, Zhen-Hua Ling, and Junichi Yamagishi (2015). “A deep generative architecture for postfiltering in statistical parametric speech synthesis”. In: *Audio, Speech, and Language Processing, IEEE/ACM Transactions on* 23.11, pp. 2003–2014 (cit. on p. 26).

Coleman, John and Andrew Slater (2001). “Estimation of Parameters for the Klatt Synthesizer from a Speech Database”. In: *Data-Driven Techniques in Speech Synthesis*. Springer, pp. 215–238 (cit. on p. 6).

Collobert, Ronan, Koray Kavukcuoglu, and Clément Farabet (2011). “Torch7: A matlab-like environment for machine learning”. In: *BigLearn, NIPS Workshop*. EPFL-CONF-192376 (cit. on p. 28).

Davis, Steven B and Paul Mermelstein (1980). “Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences”. In: *Acoustics, Speech and Signal Processing, IEEE Transactions on* 28.4, pp. 357–366 (cit. on pp. 14, 23).

Dines, John, Junichi Yamagishi, and Simon King (2010). “Measuring the gap between HMM-based ASR and TTS”. In: *Selected Topics in Signal Processing, IEEE Journal of* 4.6, pp. 1046–1058 (cit. on p. 45).

Duchi, John, Elad Hazan, and Yoram Singer (2011). “Adaptive subgradient methods for online learning and stochastic optimization”. In: *The Journal of Machine Learning Research* 12, pp. 2121–2159 (cit. on p. 28).

Elman, Jeffrey L (1990). “Finding structure in time”. In: *Cognitive science* 14.2, pp. 179–211 (cit. on p. 27).

Falk, Tiago H and Sebastian Möller (2008). “Towards signal-based instrumental quality diagnosis for text-to-speech systems”. In: *Signal Processing Letters, IEEE* 15, pp. 781–784 (cit. on pp. 48, 49).

Fan, Yuchen, Yao Qian, Fenglong Xie, and Frank K Soong (2014). “TTS synthesis with bidirectional LSTM based recurrent neural networks”. In: *Proc. Interspeech*, pp. 1964–1968 (cit. on p. 34).

Fant, G and Q Lin (1988). “Frequency domain interpretation and derivation of glottal flow parameters”. In: *STL-QPSR* 29.2-3, pp. 1–21 (cit. on p. 21).

Fant, Gunnar (1971). *Acoustic theory of speech production: with calculations based on X-ray studies of Russian articulations*. Vol. 2. Walter de Gruyter (cit. on pp. 7, 14).

Fant, Gunnar, Johan Liljencrants, and Qi-guang Lin (1985). “A four-parameter model of glottal flow”. In: *STL-QPSR* 4.1985, pp. 1–13 (cit. on p. 14).

Fujisaki, Hiroya and Mats Ljungqvist (1986). “Proposal and evaluation of models for the glottal source waveform”. In: *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP’86*. Vol. 11. IEEE, pp. 1605–1608 (cit. on p. 14).

Garofalo, John, David Graff, Doug Paul, and David Pallett (1993). “Continous Speech Recognition (CSR-I) Wall Street Journal (WSJ0) news, complete”. In: *Linguistic data consortium, philadelphia* (cit. on p. 38).

Graves, Alex and Navdeep Jaitly (2014). “Towards end-to-end speech recognition with recurrent neural networks”. In: *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pp. 1764–1772 (cit. on p. 11).

Gray, Henry (1918). *Anatomy of the human body*. Lea & Febiger (cit. on pp. 52–54).

Hermansky, Hynek (1990). “Perceptual linear predictive (PLP) analysis of speech”. In: *the Journal of the Acoustical Society of America* 87.4, pp. 1738–1752 (cit. on p. 14).

Hinterleitner, Florian, Sebastian Möller, Tiago H Falk, and Tim Polzehl (2010). “Comparison of approaches for instrumentally predicting the quality of text-to-speech systems: Data from Blizzard Challenges 2008 and 2009”. In: *Blizzard Challenge Workshop 2010* (cit. on pp. 48, 49, 60).

Hochreiter, Sepp and Jürgen Schmidhuber (1997). “Long short-term memory”. In: *Neural computation* 9.8, pp. 1735–1780 (cit. on p. 62).

Holdsworth, I, D McKeown, C Zhang, and M Allerhand (1992). “Complex sounds and auditory images”. In: *Auditory Physiology and Perception: Proceedings of the 9th International Symposium on Hearing Held in Carcens, France on 9-14 June 1991*. 83. Pergamon, p. 429 (cit. on p. 55).

Huggins-Daines, David, Mohit Kumar, Arthur Chan, et al. (2006). “Pocketsphinx: A free, real-time continuous speech recognition system for hand-held devices”. In: *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*. Vol. 1. IEEE, pp. I–I (cit. on p. 39).

Hunt, Andrew J and Alan W Black (1996). “Unit selection in a concatenative speech synthesis system using a large speech database”. In: *Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. Conference Proceedings., 1996 IEEE International Conference on*. Vol. 1. IEEE, pp. 373–376 (cit. on p. 6).

Imai, Satoshi (1983). “Cepstral analysis synthesis on the mel frequency scale”. In: *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP’83*. Vol. 8. IEEE, pp. 93–96 (cit. on p. 8).

Itakura, Fumitada (1975). “Line spectrum representation of linear predictor coefficients of speech signals”. In: *The Journal of the Acoustical Society of America* 57.S1, S35–S35 (cit. on pp. 7, 16).

Kawahara, Hideki (2006). “STRAIGHT, exploitation of the other aspect of VOCODER: Perceptually isomorphic decomposition of speech sounds”. In: *Acoustical science and technology* 27.6, pp. 349–353 (cit. on pp. 7, 8).

Kim, Chanwoo and Richard M Stern (2012). “Power-normalized cepstral coefficients (PNCC) for robust speech recognition”. In: *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*. IEEE, pp. 4101–4104 (cit. on p. 14).

Kindermann, Ross, James Laurie Snell, et al. (1980). *Markov random fields and their applications*. Vol. 1. American Mathematical Society Providence (cit. on p. 67).

Kirchhoff, Katrin (1999). “Robust speech recognition using articulatory information”. PhD thesis. Bielefeld University (cit. on p. 37).

Kirchhoff, Katrin, Gernot A Fink, and Gerhard Sagerer (2002). “Combining acoustic and articulatory feature information for robust speech recognition”. In: *Speech Communication* 37.3, pp. 303–319 (cit. on p. 37).

Klatt, Dennis H (1987). “Review of text-to-speech conversion for English”. In: *The Journal of the Acoustical Society of America* 82.3, pp. 737–793 (cit. on p. 6).

Kominek, John and Alan W Black (2004). “The CMU Arctic speech databases”. In: *Fifth ISCA Workshop on Speech Synthesis* (cit. on pp. 19, 28, 38, 49).

Kominek, John, Tanja Schultz, and Alan W Black (2008). “Synthesizer Voice Quality of New Languages Calibrated with Mean Mel Cepstral Distortion”. In: *Spoken Languages Technologies for Under-Resourced Languages* (cit. on pp. 30, 46).

Kubichek, Robert F (1993). “Mel-cepstral distance measure for objective speech quality assessment”. In: *Communications, Computers and Signal Processing, 1993., IEEE Pacific Rim Conference on*. Vol. 1. IEEE, pp. 125–128 (cit. on pp. 28, 46).

Langner, Brian and Alan W Black (2005). “Improving the Understandability of Speech Synthesis by Modeling Speech in Noise.” In: *ICASSP (1)*, pp. 265–268 (cit. on p. 8).

Lombard, Etienne (1911). “Le signe de l’elevation de la voix”. In: *Ann. Maladies Oreille, Larynx, Nez, Pharynx* 37.101-119, p. 25 (cit. on p. 8).

Lyon, Richard F (1982). “A computational model of filtering, detection, and compression in the cochlea”. In: *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP’82.* Vol. 7. IEEE, pp. 1282–1285 (cit. on p. 55).

Lyon, Richard F and Niels Lauritzen (1985). “Processing speech with the multi-serial signal processor”. In: *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP’85.* Vol. 10. IEEE, pp. 981–984 (cit. on p. 55).

Makhoul, John (1975). “Linear prediction: A tutorial review”. In: *Proceedings of the IEEE* 63.4, pp. 561–580 (cit. on pp. 8, 14).

Mariniak, Arnd (1993). “A global framework for the assessment of synthetic speech without subjects”. In: *Third European Conference on Speech Communication and Technology* (cit. on p. 48).

Meddis, Ray (1986). “Simulation of mechanical to neural transduction in the auditory receptor”. In: *The Journal of the Acoustical Society of America* 79.3, pp. 702–711 (cit. on pp. 55, 57).

Metze, Florian (2005). “Articulatory features for conversational speech recognition.” PhD thesis. Karlsruhe Institute of Technology (cit. on p. 37).

Muthukumar, Prasanna Kumar and Alan W Black (2014). “Automatic discovery of a phonetic inventory for unwritten languages for statistical speech synthesis”. In: *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, pp. 2594–2598 (cit. on p. 44).

Muthukumar, Prasanna Kumar, Alan W Black, and H Timothy Bunnell (2013). “Optimizations and fitting procedures for the liljencrants-fant model for statistical parametric speech synthesis.” In: *INTERSPEECH*, pp. 397–401 (cit. on p. 24).

Nesterov, Yuri (2012). “Efficiency of coordinate descent methods on huge-scale optimization problems”. In: *SIAM Journal on Optimization* 22.2, pp. 341–362 (cit. on p. 18).

Prahallad, Kishore, Alan W Black, and Ravishankhar Mosur (2006). “Sub-phonetic modeling for capturing pronunciation variations for conversational speech synthesis”. In: *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*. Vol. 1. IEEE, pp. I–I (cit. on pp. 22, 38, 50).

Prahallad, Kishore, Naresh Kumar, Venkatesh Keri, S Rajendran, and Alan W Black (2012). “The IIIT-H Indic Speech Databases.” In: *INTERSPEECH* (cit. on p. 39).

Quackenbush, Schuyler R, Thomas Pinkney Barnwell, and Mark A Clements (1988). *Objective measures of speech quality*. Prentice Hall (cit. on p. 47).

Rabiner, Lawrence R (1989). “A tutorial on hidden Markov models and selected applications in speech recognition”. In: *Proceedings of the IEEE* 77.2, pp. 257–286 (cit. on p. 49).

Raitio, Tuomo, Antti Suni, Junichi Yamagishi, et al. (2011). “HMM-Based Speech Synthesis Utilizing Glottal Inverse Filtering”. In: *IEEE Transactions on Audio, Speech and Language Processing* 19.153-165, pp. 459–476 (cit. on p. 16).

Rix, Antony W, John G Beerends, Michael P Hollier, and Andries P Hekstra (2001). “Perceptual evaluation of speech quality (PESQ)-a new method for speech quality assessment of telephone networks and codecs”. In: *Acoustics, Speech, and Signal Processing, 2001. Proceedings.(ICASSP’01). 2001 IEEE International Conference on*. Vol. 2. IEEE, pp. 749–752 (cit. on p. 47).

Salakhutdinov, Ruslan (2009). “Learning deep generative models”. PhD thesis. University of Toronto (cit. on p. 67).

Seneff, Stephanie (1990). “A joint synchrony/mean-rate model of auditory speech processing”. In: *Readings in speech recognition*. Morgan Kaufmann Publishers Inc., pp. 101–111 (cit. on pp. 55, 56).

Sitaram, Sunayana, Gopala Krishna Anumanchipalli, Justin Chiu, Alok Parlikar, and Alan W Black (2013). “Text to Speech in New Languages without a Standardized Orthography”. In: *Proceedings of 8th Speech Synthesis Workshop, Barcelona* (cit. on p. 39).

Slaney, Malcolm (1998). “Auditory toolbox”. In: *Interval Research Corporation, Tech. Rep* 10 (cit. on p. 55).

Stern, Richard and Nelson Morgan (2012). “Hearing is believing: Biologically-inspired feature extraction for robust automatic speech recognition”. In: *IEEE signal processing magazine* 29.34-43, p. 170 (cit. on p. 52).

Stüker, Sebastian, Tanja Schultz, Florian Metze, and Alex Waibel (2003). “Multilingual articulatory features”. In: *Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP’03). 2003 IEEE International Conference on*. Vol. 1. IEEE, pp. I–144 (cit. on p. 37).

Stylianou, Ioannis (1996). “Harmonic plus noise models for speech, combined with statistical methods, for speech and speaker modification”. PhD thesis. Ecole Nationale Supérieure des Télécommunications (cit. on p. 19).

Sutskever, Ilya (2013). “Training recurrent neural networks”. PhD thesis. University of Toronto (cit. on p. 27).

Toda, Tomoki and Keiichi Tokuda (2007). “A speech parameter generation algorithm considering global variance for HMM-based speech synthesis”. In: *IEICE TRANSACTIONS on Information and Systems* 90.5, pp. 816–824 (cit. on p. 26).

Tokuda, Keiichi, Takao Kobayashi, Takashi Masuko, and Satoshi Imai (1994). “Mel-generalized cepstral analysis-a unified approach to speech spectral estimation.” In: *ICSLP*. Citeseer (cit. on pp. 7, 14, 23, 51).

Tokuda, Keiichi, Takayoshi Yoshimura, Takashi Masuko, Takao Kobayashi, and Tadashi Kitamura (2000). “Speech parameter generation algorithms for HMM-based speech synthesis”. In: *Acoustics, Speech, and Signal Processing, 2000. ICASSP’00. Proceedings. 2000 IEEE International Conference on*. Vol. 3. IEEE, pp. 1315–1318 (cit. on p. 26).

Vincent, Damien, Olivier Rosec, and Thierry Chonavel (2005). “Estimation of LF glottal source parameters based on an ARX model.” In: *INTERSPEECH*, pp. 333–336 (cit. on p. 19).

– (2007). “A new method for speech synthesis and transformation based on an ARX-LF source-filter decomposition and HNM modeling”. In: *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*. Vol. 4. IEEE, pp. IV–525 (cit. on p. 19).

Weide, Robert (2005). “The Carnegie mellon pronouncing dictionary [cmudict. 0.6]”. In: *Carnegie Mellon University: http://www.speech.cs.cmu.edu/cgi-bin/cmudict* 9 (cit. on p. 43).

Werbos, Paul J (1990). “Backpropagation through time: what it does and how to do it”. In: *Proceedings of the IEEE* 78.10, pp. 1550–1560 (cit. on p. 28).

Wrench, Alan (1999). “The MOCHA-TIMIT articulatory database”. In: *Univ. of Edinburgh. Edinburgh, UK, Nov* (cit. on p. 37).

Yoshimura, Takayoshi, Keiichi Tokuda, Takashi Masuko, Takao Kobayashi, and Tadashi Kitamura (2001). “Mixed excitation for HMM-based speech synthesis.” In: *EUROSPEECH 2001 Scandinavia, 7th European Conference on Speech Communication and Technology, 2nd INTERSPEECH Event, Aalborg, Denmark, September 3-7, 2001*, pp. 2263–2266 (cit. on pp. 19, 46).

Zen, Heiga and Andrew Senior (2014). “Deep mixture density networks for acoustic modeling in statistical parametric speech synthesis”. In: *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, pp. 3844–3848 (cit. on p. 34).

Zen, Heiga, Takashi Nose, Junichi Yamagishi, et al. (2007). “The HMM-based speech synthesis system (HTS) version 2.0”. In: (cit. on p. 19).

Zen, Heiga, Keiichi Tokuda, and Alan W Black (2009). “Statistical parametric speech synthesis”. In: *Speech Communication* 51.11, pp. 1039–1064 (cit. on pp. 6, 8).

Zen, Heiga, Andrew Senior, and Mike Schuster (2013). “Statistical parametric speech synthesis using deep neural networks”. In: *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, pp. 7962–7966 (cit. on p. 34).

Software

Johnson, David, Dan Ellis, Chris Oei, Chuck Wooters, and Philipp Faerber (2004). *ICSI QuickNet Software Package* (cit. on p. 38).

Parlikar, Alok (2012). *TestVox: Web-based Framework for Subjective Evaluation of Speech Synthesis* (cit. on p. 19).

Acknowledgement

To this day, I don’t know why my advisor decided to take me on as his PhD student. I still believe that he probably accidentally sent me the acceptance email originally intended for someone else. This one folly aside, he has been the very best of advisors that anyone could have hoped for. He has always had his door open for me and his other students to walk in to talk to him, and the discussions we have had have never ceased to delight me. I have thoroughly enjoyed every minute of research that I have done under him, and so I owe him a great deal of gratitude. I shall certainly miss having an office just one or two doors away from his.

Math has never been my strongest subject, and so I must thank Prof. Bhiksha Raj for always being ready to help out whenever I needed help. He has always amazed me by his knowledge of even the most obscure of research ideas. Publishing papers is not my forte either, and for this I must thank Prof. Tim Bunnell who helped me write my first paper, and has been nice enough to let me pick his brain whenever I had questions. My first serious dive into speech signal processing was in Prof. Rich Stern’s class many years ago. It was the very first class that I took at CMU, and as my advisor will acknowledge, I have not stopped taking classes since.

I have also been lucky enough to have some amazing friends throughout this PhD journey. A special note of thanks to Subhodeep Moitra who helped shape from an electrical engineer who couldn’t code to save his life to an electrical engineer who can write somewhat reasonable code. His ability to never mince his words when talking to me has also been a great resource. Meghana Kshirsagar and Waleed Ammar also deserve special mention in always being there to help out whether it was to cheer me up when I’m feeling down, or help me finish off those bananas in my office before they went bad.

I have also had great colleagues in Pallavi Baljekar, Sunayana Sitaram, Kevin Lenzo, Alok Parlikar, and Brian Langner. In addition to being a great colleague, Nathan Schneider helped expose me to a lot of culture through opera, musicals, plays, and the amazing Pittsburgh Symphony Orchestra.

CMU has been exceptionally good in attracting some of the smartest people on the planet. This was particularly obvious in the case of my three current officemates: Shoou-I Yu, Xinlei Chen, and Chenyan Xiong. Despite being junior to me, I have always been blown away by how adept each seemed at his research. I have especially learned important lessons by the amount of hard work each put into implementing his ideas.

The last few years of my PhD would not have been possible without the support of my wife, Shruthi. She was particularly nice in being patient with me when my mind drifted off to my thesis in the middle of our Skype conversations.

Finally, the most important people responsible for any success I've had are my parents. They have always put my needs far above their own, and have always been supportive of whatever I decision I've made. I sincerely owe a lot to them.

Colophon

This thesis was typeset with $\LaTeX\,2_\varepsilon$. It uses the *Clean Thesis* style developed by Ricardo Langner. The design of the *Clean Thesis* style is inspired by user guide documents from Apple Inc.

Download the *Clean Thesis* style at `http://cleanthesis.der-ric.de/`.