Spring 3-2017

Improving Prosody Through Analysis by Synthesis

Kevin A. Lenzo
Carnegie Mellon University

IMPROVING PROSODY THROUGH ANALYSIS BY SYNTHESIS

KEVIN A. LENZO

Submitted in Partial Fulfillment of
Philosophy of Doctorate

Carnegie Mellon University
Robotics Institute

Comittee:

Alan W Black, CMU LTI
Julia Hirschberg, Columbia CS
Jack Mostow, CMU RI
Alex Rudnicky, CMU LTI

Date: March 28, 2017

version 1.0

Robotics Institute Tech Report
CMU-RI-TR-17-11

To my father and mother, wife and children, dogs and cats: thank you for all the support, love, and patience.

To Osamu Fujimura, who lit my path in ways I didn't always understand.

ABSTRACT

Prosody and prosodic modeling in trainable Speech Synthesis systems are often based on large corpora of automatically annotated training data; however, these annotations are often incorrect. In practice, this has been either addressed through labor intensive manual annotation or simply ignored. In order to overcome this problem and improve prosodic realization, an iterative model-based method is proposed for improving linguistic structure, segmentation, and prosodic annotations that correspond to the delivery of each utterance as regularized across the data. For each iteration, the training utterances are resynthesized according to the existing symbolic annotation. Values of various features and subgraph structures are "twiddled:" each is perturbed based on the features and constraints of the model. Twiddled utterances are evaluated using an objective function appropriate to the type of perturbation and compared with the unmodified, resynthesized utterance. The instance with least error is assigned as the current annotation, and the entire process is repeated. At each iteration, the model is re-estimated, and the distributions and annotations regularize across the corpus. As a result, the annotations have more accurate and effective distributions, which leads to improved control and expressiveness given the features of the model.

"After all our technology, the pseudo-intelligence algorithms, the vast exception matrices, the portent and content monitors, and everything else, we still can't come close to generating a human voice that sounds as good as a real, live ractor can give us."
– Hackworth, in Neal Stephenson's The Diamond Age

ACKNOWLEDGMENTS

A heartfelt thank-you to my committee members, Julia Hirschberg, Jack Mostow, Alex Rudnicky, and especially Alan W Black for support, encouragement, and collaboration through the years.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

1.1 RENDERING SYNTHETIC SPEECH

HERB's Sure Thing [1] is a rapid drama system for rehearsing and performing live robot theater. A robot was introduced into a live dramatic production, as a means to explore human-robot interaction. The setting was a dialog between two people – a man and a woman – with the robot performing as the man. A discussion takes place, which heads in one direction, then a "reset" happens to go back in time and deliver the lines in a different way, opening a different path. The "reset" schema is used many times through the script. Using synthetic speech for HERB leads to a number of interesting issues.

For one dramatic performance, it would be possible to record the speech in advance, and synchronize the robot's movements and speech. This approach leaves little flexibility beyond editing the audio to alter the performance.

Working with synthetic speech in this dramatic context depends on different readings of the same text, with different prosody, which opens a different path in the dialog after each reset. A single deterministic delivery of the text doesn't capture the expressive variation that a performer would provide from context.

A simple approach of modifying the audio by resampling and changing pitch to copy prosody was used. While the result had dramatic effect, the experience proved difficult: "To manage this gap in the level of abstraction versus level of control, either the audio operator needs to learn what low level changes map to high level effects, or the controls need to be raised to a more abstract level."

Using a mark-up language like W3C's Speech Synthesis Markup Language (SSML [2]) or SABLE [3] allows for a gross-level tag for emphasis (<emph>), which does not capture the changes in delivery; using fine-grained specification for the duration and frequency of each unit is difficult and tedious, short of directly copying the prosody from a human performance. Using this mark-up had similar issues to pre-recorded speech, in the absence of a larger framework.

In this case, the scripting of the performance could have been improved and made better, if there were high level controls over the

speech to adjust a synthetic performance with relatively few parameters. The synthesizer could deliver better prosody for the performance, and provide more flexibility for real-time dramatic reactions and effects. An interface tool could be built on top of it, to make the detailing process more accessible, without getting into the weeds to specify exactly how much to lengthen or change the pitch of each segment in detail.

Trainable speech synthesis systems are built from linguistically annotated corpora, and the nature and quality of the annotations have an impact on the models and spoken output. However, detailed manual annotation is labor intensive, and for some features, the details of how and what to annotate are not easily categorized. Furthermore, many systems use completely automatically labeled data which assumes a structure and labels using dead reckoning: the text analysis front end of the system is used to generate a guess, and it is modified only to account for segmental alignment, leaving the assumptions about stress, phrasing, syllable structure and other prosodic features unadapted. As a result, the models built from these alignments are degraded in comparison to perfectly annotated data.

We will introduce a formalism and methods for inducing regularized annotations across a speech corpus, using Expectation Maximization over model-based perturbations of the data. As a result, the models are more reliable and expressive when the features are modified as performance parameters.

1.2 AMBIGUITY OF TEXT

However, creating spoken output from text is an exercise in ambiguity. Any performance or utterance is informed by factors that may not be marked out in text. A word sequence can be spoken in a variety of ways by varying the prosody – changing the intonation, energy, timing, and other factors. When a text-to-speech synthesizer operates on "raw" text, it constructs a representation using a model. There is a good chance of missing or producing nuances that an author does not intend. It is safer to give an uninteresting rendering than one with marked prosody.

In working with synthetic speech, authors and language generation systems have two options: to rely on the system's text processing of the system, with few controls – perhaps an <emph> tag to add emphasis; or setting detailed phonetic duration, frequency, and energy on individual phonetic units. Some systems allow for entering ToBI labels directly. Each of these solutions has drawbacks – the text interpretation may be different than intended; specifying in terms of fine

detail is tedious; emphasis is too course of a tag to capture different accent types; ToBI labels require a certain level of linguistic training. On top of these issues, there is generally no way to finely adjust the performance of an utterance, other than copying values from another performance.

1.3 DATA SPARSITY

It is difficult to find enough naturally occurring speech from a single talker to have all combinations that need to be accounted for. In general, it is possible to model effects without having exhaustive examples, and synthesize effects from sparse examples. A specification for the linguistic units and features is built abstractly, and constructed by the "front end" in a data structure. This structure may be passed to a "back end" and used as input to a signal generation method. For example, unit selection synthesis produces spoken output even when many sequences of phonemes do not appear often.

In voice building, the voice talent is given a corpus of material to deliver, which covers the target space, and models are created. The texts are typically constructed, or selected "greedily" from larger corpora, in order to obtain samples of all the units and contexts needed for modeling. However, the prompts are typically presented with no motivation or mark-up to select, or constrain, the possible renderings. After many hours working on isolated, unconnected sentences, the talent settles into a sort of neutral style, with no unacceptable surprises.

1.4 VOICE TALENT DELIVERY

The CMU Arctic Database contains read sentences in isolation. Without performance instructions for each prompt, the collection led to a distribution of possible meanings over the word sequences. Each talker delivered comparable sentences differently. Each chooses one of the many acceptable deliveries for a given text.

A neutral or deadpan delivery gives few clues to the structure, implication, or emotional valence of elements in the speech. Neutral delivery may be helpful when the corpus is constructed without marked pragmatics or precise delivery instructions. The talker has no information of which possible delivery should be produced, and the corpus is not balanced for the variations: it is not designed to have examples of all the possible variations. Delivering in a deadpan or neutral style avoids prosodic mismatch by suppressing large pitch movements or

duration changes, leading to a more uniform corpus but with little or no expressive prosody.

Story or audiobook reading has a markedly different delivery (Black Beauty, Usborne) in comparison to read or spontaneous speech.

Radio and television news readers have another style, as demonstrated in the Boston University Radio corpus (BURNC).

A talker reading aloud may be using very little look-ahead, and know very little about what they are saying – or, at another extreme, may have read and studied the text deeply to get into the story or even a character for the delivery. Dialogs between talkers, where intentions, structures, and meanings are ambiguous, lend themselves to prosodic effects which depart from a neutral delivery.

1.5 MEANINGLESS PROSODY

Statistical methods which focus only on syntagmatic (structural and local neighborhood) factors are operating at the wrong level to account for pitch accent and break placement. By analogy, it is like predicting the meaning of a word from its phonemes: there will be some correlations, but the analysis is not generative at the right level. An alternative is to label prominence, or perhaps something more narrow such as "emphasis", which may lump all types of pitch accents and uses into a single marked category.

While systems have been created to model the prosody of individual talkers, these systems tend to focus on syntagmatic features such as position in the utterance, distance from boundaries, and part of speech sequences. However, when an utterance is created, there are choices that a talker makes that relate to their intentions in discourse: Any word in a sentence may receive a pitch accent, irrespective of the part of speech or position. The result of synthesis is a deadpan delivery, or some random variation overfit by the model, as the variation is averaged within the model if the variations are not properly annotated.

1.6 MEANING AND PROSODY

Models have been produced which relate meaning to intonation, at least for some operations. These models are mappings from discourse structure to tones, such as using a high tone to indicate new information, and using a high boundary tone to indicate an act which asks for completion (a fall to a low), perhaps for a "continuation" non-fall

or asking a question in English. Declination effects, and downstep, describe overall tonal changes. However, the labels are very coarse, and do not address how a specific talker relates units in terms of prominence.

On the one hand, the mark-up does not support authoring at the level of semantics; on the other, synthesizers typically do not offer easy controls over the performance parameters. There is no high-level interface or language which allows prosodic differentiation of meaning or intention, nor one which allows event-based controls where the author of synthetic speech can change a rendering at a high level.

1.7 ANNOTATING AND MODIFYING PROSODY

The realization of prosody is not fixed, and repeated deliveries of an utterance by human performers vary considerably in naturalistic contexts. One factor in variations of prosody seems to be a desire not to be monotonous or repetitive, so that repeated deliveries are varied as a performance choice to avoid sounding mechanical or boring. Prominence may be found in a high peak, a late peak, higher energy, or other phenomena which may be traded off in a given delivery.

One interesting implication of the variability is that a simple time or energy based metric, such as RMS error for intonation, will measure variance as error when it has little impact on the listener. That is, if there are many acceptable ways to render something, then not all measured differences are meaningful. This variety makes evaluation particularly challenging.[1]

1.8 BUILDING SYNTHETIC VOICES

Throughout this work, we build voices within the framework we set forth in FestVox [5] [6] for the Festival Speech Synthesizer [7] and Festival Lite (FLite) [8]. These tools are all available freely on-line at `www.festvox.org` along with documentation and example databases.

We use the Arctic speech databases [9] for most of the voice builds. In particular we use the awb, bdl, clb, jmk, ksp, rms, and slt data, which contain English as spoken by Scottish, US, Canadian, and Indian talkers.

1 In Firesign Theatre's "Don't Crush That Dwarf, Hand Me the Pliers," the characters in a skit show this dramatically by repeating the same sentence with emphasis moved around in the sentence: "What are going to DO, lieutenant?", "What ARE we going to do, lieutenant?", "What are WE going to do, lieutenant?" and so on.[4]

While many of the processes described here have been created with special tools and processing, everything runs inside of the existing systems when a voice is built. Anyone can learn how to make new voices in FestVox, and implement the techniques described here.

1.9 TOWARDS INCREASING ERROR RATES

In a paper discussing the state of the the art in Speech Recognition in 1996, a case was made for exploring techniques which may lead to a short-term increase in error rate, in return for progress toward the end goal of systems with greater utility (though worse in terms of conventional error rates) [10]:

> Permitting an initial increase in error rate can be useful, as long as three conditions are fulfilled: (1) solid theoretical or empirical motivations, (2) sound methodology (so that something can be learned from the results), and (3) deep understanding of state-of- the-art systems and of the specificity of the new approach.

This work can be seen in a similar light: several experiments show worsening metrics, at least at first, but end up creating new utility and descriptions that allow for different kinds of engagement. Optimizing for a combination of criteria may not lead to improvements in any one of them measured individually, as we search new spaces for meaningful distinctions.

1.10 OVERVIEW

In this work, we improve annotation of the prosody, and show how the annotation may be used to modify synthetic speech at a high level. We start with an overview of how the linguistic information in an utterance is organized, and critique the utterance representation as it used in Festival. This is followed by a motivated refactoring in a framework named Karnival to facilitate prosodic modeling, markup, and improvement. Experiments show application of the framework to improve annotation and synthesis. We close with a discussion of the work, and list some future directions.

2

PROSODY AND PROSODIC MODELING

Talker and style may be seen as parameter settings in the space of a generative prosodic model, and to consider transformations and differences between one parameterization and another. This perspective can gives us insight into the organization of speech; it allows us to imagine adapting one talker or style to another; and, if we can uncover general transformations, we can use data from many talkers to build the model and then modify it to fit a particular talker and style, even if we have little or no data from that combination a priori.

A number of studies point to the importance of segmental effects on prosodic realization; that is, the short-term changes in pitch, timing, energy, and voice quality that are either intrinsic to a particular phone, or resulting from transitions between them. These effects include pre- and post-consonantal perturbations of the fundamental frequency that may be as large as 30 Hz, as well as vowel-intrinsic pitch, which may contribute up to 20 Hz in [u] at the top of the pitch range down to about -15 in [a], as measured by observing the timing of pitch periods sample with an electroglottograph. There are some excellent instrumental studies, many notably by Lehiste [11, 12], that report particular conditions, but with the growth in computational power, storage, and relative ease of data collection, there is a case for systematically crafting corpora that cover the necessary inventory of units, and replicating them under each condition (intonation of vowels in context, embedded in syllabic structures). Local segmental effects on F_0 are on the same order as some pitch movements used to indicate prominence, and so can confound model estimation unless they are taken into account.

Ladd [13] notes the following points on terminology, contrasting with Lehiste's characterization of suprasegmentals as features of *Pitch, Stress, and Quantity* [12]:

> *Intensity* is physical and measurable. *Stress* is perceptual, and measurable through indirect means only. *Frequency* is physical, *Pitch* is perceptual. *Duration* is measurable, but *Quantity* is perceptual.

In the following discussion, we will focus as much as possible on measurable phenomena of the signal, but for evaluation, one is necessarily led to perceptual experiments, and perceptually mediated aspects of speech are ultimately what a listener hears. Ilse Lehiste's own

work shows her awareness of this distinction, and her instrumental experiments lay the basis for much of the modern view of intonation; however, I will attempt to follow Ladd in meaning.

Pitch-Synchronous OverLap-and-Add (PSOLA, [14]), Linear Predictive Coding (LPC), sinusoidal resynthesis, or any number of techniques allow one to alter pitch without modifying duration, and duration without modifying intonation. HMM synthesis also allows for separate control of acoustics, duration, and fundamental frequency [15]. This separate control is an important tool in resynthesizing speech for experimental conditions, as well as for altering segments from a unit inventory to reflect the predicted duration and time course of F_0 at synthesis time. These techniques help us focus on the duration and intonation components as distinct models. Other components of a complete prosody system include the modeling of the time course of energy, and voice quality or glottal source control.

The estimation of fundamental frequency in speech is by no means trivial, and pitch tracking errors are the bane of fine instrumental analyses of F_0. Problems in pitch estimation led to hand-marking all the data to analyze segmental influences on F_0 in [16].

Electroglottograph (EGG) data is helpful in the estimation of fundamental frequency. An EGG tracks electrical signals from muscle innervation near the larynx, rather than after the oral cavity; this helps reduce, but does not eliminate, pitch tracking error.

2.1 DURATION

Metrical structure is important in the prediction of unit duration, and encodes rhythm and phrasing through timing and breaks; Lieberman and Prince [17] set the stage for this approach in their manifesto on metrical structure. The intonational events are aligned in metrical time. In English, metrical time is expressed in syllable units, and appear as stress groups or metric feet. Their work prompted a number of studies in stress relations and how they may be represented, for instance, as trees or grids, and how focus and prominence rearrange the structures. Duration, and resultantly timing, has been modeled a number of ways for synthesis. Linear models are often used, including simply using mean duration in some systems.

Duration domain transformations, such as the root sinusoidal transformation [18] allow estimation of the combined factors using generalized additive models.

2.1.1 *Sums of Products*

Van Santen's sums-of-products model [19] is an example of generalized additive decomposition. Under this model, the duration of a segment is the (linear) sum of products, and each of these products represents some interactions, in the sense of interaction terms in multiple linear regression.

Rather than compute the product of all possible interactions (a daunting number) or reduce the model to a strictly additive one, the sums-of-products approach helps overcome the sparse data problem. Van Santen models classes of phonemes, rather than individual allophones, which is a further reduction of the sparseness. Thus, one simple example of the form of the model is

$$\mathrm{duration}(/\mathrm{i}/, \mathrm{voiced}, \mathrm{stressed}) = \mathrm{A}(\mathrm{i}) + \mathrm{B}(\mathrm{voiced}) \times \mathrm{C}(\mathrm{stressed})$$

Isotonic smoothing may be used to condition the matrix, so that interpolation makes sense even when there are no observations that exactly match the desired conditions. This is a natural result of functional analysis – assuming that there is an underlying function producing the observables we have, and trying to model the function.

Sums-of-products duration models and SVD-based duration transformation prediction [20] are one way of transforming one duration model into another. A matrix is generated for the language using SVD, and the talker parameters are then weights on the dimensions of each retained eigenvector. In most speech synthesis systems, the duration of the segmental material is used as input to the intonation prediction routine – segmental duration is estimated first, and then the intonation is draped over the segments, by either selecting the units from a database and using them as they are, or modifying them with a pitch- or time-scale modification such as PSOLA. This is clearly a simplification, as the duration of a presumed segment is completely dependent upon the timing of elemental gestures required for its realization, not the reverse. Changes in rate have different effects on different 'segments', that have varying compressibility; however, this duration-first assumption is quite powerful and useful, because it is so strongly correlated with gestures themselves.

2.1.2 *Decision Trees*

In CLUSTERGEN [21] statistical parametric synthesis, the duration of sub-phonetic states is modeled directly using clustering and decision trees [22] or random forests [23]. After the training utterances are

aligned using the FestVox `ehmm` tool, the units are clustered for the signal generation component, and then the state durations are modeled using questions over an utterance structure.

Linguistic features are enumerated and treated as independent variables, and trees are trained using a greedy tree-building strategy that optimizes information gain at each node [6][24].

The features for duration modeling are listed in a file called `dur.feats` for clunits or `statedur.feats`. Segmental identity, neighboring segment identity, phonetic features[1], syllabic stress of the parent and parent's neighboring syllables, position in syllable, word and phrase structures, and other attributes.

Decision trees have a lot of flexibility in partitioning the output function, and so are somewhat robust to nonlinear duration distributions. Log-duration, z-score, or pdf are possible representations for the modeling technique.

2.1.3 *Transformed Duration*

Each of the systems predicts duration, but may do so in a transformed domain. For example, Campbell [26] and, later, Barbossa [27] model log-normalized durations, while CLUSTERGEN operates on z-scores of unit distributions, and Zen [15, 28] uses probability density functions over HMM state durations and later Deep Neural Networks [29].

The duration transform and modeling method interact. In systems that are modeled with multiple linear regression, such as van Santeńs sum-of-products models, there are assumptions about smoothness and linearity; however, systems like Festival which use decision trees or random forests can split the measures in many places without much regard for these assumptions, and neural networks can manage nonlinear functions effectively as well.

The decision trees in Festival divide the search space into arbitrarily fine distributions, and so do not have the log- or linear-spacing assumptions built into multiple linear regression for duration modeling. However, the error function may be skewed if it assumes any linear symmetry. In the default Festival duration models, the z-scored durations are symmetric, while the durations, being bounded by zero on the lowest end, look more symmetric after log-normalizing them. While this is a source of bias in training, the state duration distributions are close to Gaussian in the time scale they operate within, as a

1 Basically the Sound Pattern of English features from Chomsky and Halle[25] based on phonetic identity. This allows tieing of mixtures based on common features.

consequence of the EHMM alignment process with Hidden Markov Models. Figure 28 has an example.

2.2 INTONATION

2.2.1 *Copy Synthesis*

Close-copy synthesis, which takes parameter values from a natural utterance and imposes it upon a synthetic contour, is useful to describe a warping with a distance metric, as well as for conducting experiments on the relative importance of components by altering them. However, while being a great demo, close-copy synthesis avoids the modeling problem entirely, as the parameter estimation is largely avoided, and does not decompose the prosody into its representative components. With this in mind, close-copy synthesis helps give a bound on how a perfect model might sound, and thus is a good "cheating experiment."

2.2.2 *General Text-to-Speech Synthesis*

For more general synthesis, either a latent model (with no distinct annotation) or an explicit model of intonation is needed. When Unit Selection synthesis became popular, it reduced the impact of explicit intonation models: The synthesis could make do with latent prosody, which came from the concatenation and smoothing of individual database examples in context. This led to something of a "winter" of intonational modeling in speech synthesis[2]. As parametric synthesis systems have become more practical, many of the systems have adopted similar techniques. As such, much of the active work on intonational description and modeling comes from before 2000.

2.2.3 *Tone and Break Theoretical Basis*

There are a number of theories of prosodic organization, but few works have had a greater influence than Janet Breckenridge Pierrehumbert's germinal thesis on the phonological structure of fundamental frequency contours [30]. Pierrehumbert posited a formal description of fundamental frequency contours in terms of a (tonally 'phonemic') binary opposition of High (H) and Low (L) tones, and

2 Alan W Black, personal communication

argued that the description was sufficient from a phonological perspective and did not overgenerate combinations.

The system consists of a set of pitch accents (described in ToBI as H*, L*); associated with syllables, phrasal tones (L-, H-), and boundary tones (L%, H%). Pitch accents may also be bitonal, represented by a plus, as in H*+L. By analysis of African tone languages, Pierrehumbert posited an inventory of bitonal accents and their concomitant effects – most notably 'downstep', where the following pitch accents are lowered in frequency following a H+L bitonal accent.

2.2.4 *Annotating Tones and Break Indices (ToBI)*

Pierrehumbert's formalism and the work of many other contributors underlies the annotation standard now known as ToBI [31], for Tones and Break Indices. The inventory and some of the positions have changed, but the basic theory remains: a pitch contour can be represented as an abstract sequence of symbols, each of which can be placed in minimal opposition with another and produce potentially meaningful differences. This allows one to take compositional approaches to a tune (intonational contour), and relate them, compositionally, to meaning, as in [32].

ToBI itself is not a strong theory of linguistic organization, but a picklist of tags. It is an annotation standard, forged through heated debate among interested linguists at the time it was created – before the World Wide Web, as an outgrowth of laboratory phonology and speech technology research. ToBI does not specify the phonetic realization; it is a set of symbols attached to existing contours. This annotation can be used as a common standard for databases, and there have been some efforts to predict the time course of F_0 from ToBI strings. The modeling then requires a theory of phonetic implementation to be realized in signal.

There has been some work on automatically detecting and determining prosodic annotations [33] [34]. The automatic detection rate for pitch accent type in AuToBi is about 75% when the accents consist of between 70-80% H*, and this result is the most successful (given a proper train/test division). This is better than for Maghbouleh [35] who reported 50% accuracy for automatic recognition of five pitch accents in the same Boston University Radio Corpus, compared with an inter-transcriber agreement reported in [36] of 73% overall, with him noting that 65% of the accents in the corpus are H* or uncertain, so one might achieve human performance by assigning H* to all accented syllables. Syrdal [37] found higher agreement – 90% – on tone presence, irrespective of type, and a 92% agreement on break place-

ment. This difference underscores issues of representation given only ToBI, without additional information such as an F_0 contour to which it might be attached: the impact of tone type is confuseable, and a single ToBI representation could be realized in many different ways.

Analor [38] is a tool for semi-automatic annotation of prosody in French which segments the data into prosodic units and accented syllables using a set of rules, reporting 84% agreement with human labelers on presence or absence of prominence in French. Tone type was not included.

Kim Silverman's thesis [31] contains a model of phonetic implementation from tone sequences. Silverman also demonstrated the importance of factoring segmental components into analysis – the effects of the phonetic segments upon the time course of F_0. These components include vowel-intrinsic pitch, which is presumed to be as much as 20 Hz at the high end of the pitch range; anticipatory perturbations, in which F_0 takes a sudden drop before constrictions in the oral tract; and perseveratory perturbations following the release of constriction. He also noted intrinsic effects in voiced obstruents and sonorants, but did not model them in great detail.

These segmental effects are overlaid on a fundamental frequency trajectory that is modeled in broad accordance with Pierrehumbert's model. Tonal movements are considered to be phonetic realizations of abstract tonal phonemes. Thus, all pitch contours can be described abstractly with a small number of labels that presumably represent all meaningful oppositions in intonation.

The segmental interactions with F_0 are not clearly agreed upon. Shih [39] reports the effect of consonants on F_0 in the following vowel for Mandarin level tones, showing somewhat different segmental effects than those reported by Silverman in [40] – there is a slight rise before F_0 drops in the post-obstruent case as opposed to a monotonic fall. However, this apparent difference can be explained by the difference in the granularity of their reports: Silverman used increments of five glottal periods to report conditions, while Shih shows the F_0 contour for each sample. This would indicate that the effects reported by Silverman could be even larger, but that his measurement points do not show segmental effect peak. Furthermore, van Santen and Möbius [41] find a single post-obstruent fall to be sufficient, and do not use any anticipatory or intrinsic effects in their model.

2.2.5 *Tone Sequence Modeling*

Any intonation contour can then be represented as a linear sequence of pitch accents (H*, L*, L+H*, etc), phrasal tones (H-, L-), and bound-

ary tones (H%, L%). Each pitch accent bears an abstract magnitude (T) in a transformed tonal space, specifically so that a single model can be used for all talkers. The transformed space, T, is also used to linearly combine the effects of vowel intrinsic pitch with the tone sequence contour. In Silverman's implementation, T is derived from a top line (Top), a reference line (Ref), and a baseline representing the lower limit of the talker's range (Floor), which are talker parameters.

$$F_0 = Floor + (Ref - Floor) \times (\frac{Top - Floor}{Ref - Floor})^T$$

$$T = \frac{\log(\frac{F_0 - Floor}{Ref - Floor})}{\log(\frac{Top - Floor}{Ref - Floor})}$$

The abstract magnitude determines the height of the pitch accent, and may contribute to the local shape of the tone. The Reference line is a constant proportion of the talker's pitch range, and the Floor is a constant. The implementation scales the pitch range throughout the utterance in response to tonal events, and events that lie on the top line have abstract magnitude of one. With this transformation, the components define a generalized additive model of segmental effects and the tone contour, where the speaking range of a particular talker is expressed in terms of Top, Ref, and Floor.

In between local shapes produced by each symbol, F_0 was linearly interpolated in [40]. Thus, without pitch accent or other tone, the intonational movement is locally unspecified, and only the salient points of oppositions are maintained.[3]

Silverman and Pierrehumbert [16] explored the timing of pitch accent peaks further in pre-nuclear high accents, and showed that peak timing varies with metrical embedding. As two syllables receiving pitch accents are separated by fewer and fewer intervening unaccented syllables, the local shapes of the pitch accents change, and the timing of the peak of high pitch accents moves. This is some support for using the type and time of the neighboring tones as parameters to predicting local shape.

Bruce [42] mentions the evocatively named Scandinavian accent orbit, attributed to Öhman[43]: Two accent types (Accent I and Accent II) in Scandinavian dialects differ systematically in the timing of the peak placement in an intonational movement. For two-syllable words in phrase-final position, Accent I is /H L* H L/, and Accent II is /H* L H L/, where the asterisk marks the associated strong syllable, and the timing of the pitch in the H moves smoothly between early and late in the second syllable. The relative timing of the peak has been related to the region of the talker's origin. That is, the tonal

3 This is also followed in some initial work on generating intonational contours below: the LSAFo model.

symbols are used differently to represent the same lexical material, and the only difference is dialect. Thus, one cannot simply say that a condition is a certain abstract set of symbols, but rather, the symbols are a result of the configuration, because the dialect influences the annotation.

Symbol choice, and the annotation of the data with respect to it, remains a central theoretical problem, in that it is language dependent. A computational model was given by Pierrehumbert and Beckman [44], which gave flesh to the skeletonized annotations set forth in Janet Pierrehumbert's thesis, for Japanese. True to the nature of tone sequences, the model predicts the time course of fundamental frequency from the tone sequence and moraic time alone. The moraic timing in Japanese allowed them to reduce some of the issues of stress-timed languages such as English, in which a stressed syllable has a higher correlation with longer duration. The implementation is similar to that of Silverman [40], and also used interpolation between the locally specified portions of the model.

Since the work on Japanese tone structure, there have been a number of models that predict the fundamental frequency contour from ToBI labels, including Dusterhoff and Black [45], who used ToBI label as a predictive feature. However, this is not done from ToBI alone, as ToBI does not specify abstract magnitude, or F_0, or timing.

Nolan and Grabe [46] assert that ToBI is a weak representation of variations in dialects of British English, and point to a number of issues, noting that ToBI expresses phonetic oppositions that sometimes do not manifest, leading to multiple possible interpretations (in terms of which pitch accents) of some intonational contours, and that ToBI is an uneasy mix of "phonetic specification and linguistic generalization."

2.2.6 *Superpositional Approaches*

While tone sequences are one approach, they are not the only one. Fujisaki (e.g. [47, 48]) posited a superpositional model of fundamental frequency (after Öhman [43]) which consists of two components: an accent component, represented by the response to a step function, riding on a longer phrasal curve, which is an impulse response. Segmental effects on the time course of F_0 may also be added in. This is the canonical superpositional model – all of the components are additive in the $\log F_0$ domain. Fujisaki interprets the shape of the accent (step) and phrase (impulse) responses as talker-independent, and a base value is set for each talker.

> In other words, by positing a speaker-specific base value, the phrase and accent commands become speaker-independent to a first-order approximation.

[48]

Ladd [13] notes problems when attempting to apply the Fujisaki model to English (or other non-Japanese languages):

> The principal difficulty is in modeling low or low-rising accentual contours (as in a common pronunciation of English "Good morning") - a feature that is completely absent from Japanese. The quantitative details of Fujisaki's model are such that negative accent commands yield contours of the wrong shape. It is possible to approximate the low-rising contours by negative phrase components, but this is inconsistent with the intended function of the phrase command.

(Ch. 1, note 8)

The rise does actually occur in Japanese casual speech, for example in the long rise on "Eeeeee?," a questioning-with-surprise-and-interest that all manifests on a vocalic monosyllable, but it probably did not occur in the laboratory data. Still, there are low movements in English, that cause the pitch to drop to nearly the bottom of the range. If the pitch accents are positive, and the phrase curve is positive, then the nadir of these low regions must lie on or above the phrase curve, making the phrasal contribution very small, and requiring that the pitch accents be very high in order to produce the same contour – if it is even possible.

Nevertheless, in Möbius' work [49, 50], linguistic constraints are applied to limit the solution space for automatically deriving the parameters for German using a variation of the Fujisaki model. All phrase and accent commands are positive. He specifically does not handle segmental effects on F_0 until later papers; the early work assumes they are irrelevant, which is nearly accurate for all-sonorant material, and if the data has a good mix of vowel-intrinsic pitches that cancel each other out. In [49], the decomposition of the model parameters is described, and notably:

> Based on the principle of superposition, the step of determining the phrase command parameters and the basic value F_{min}, which is the first step in the algorithm, can be separated from the subsequent determination of the accent command parameters. The contour resulting from F_{min} and the phrase parameters is approximated to the measured F_0 curve. Once the parameters of the

> phrase component have been optimized, the resulting difference signal is interpreted by the accent component of the model.

[49]

One damping factor was used for the phrase component, and one for the accent component, computed as the arithmetic mean of the fitted damping factors for all the talkers. The value for the accent command was found by a process similar to simulated annealing, as the range of the parameter estimation was reduced on successive iterations until it became fixed for all talkers. Interestingly enough, they report that constant damping factors were

> perceptually as similar to the original as were the versions with varying values of the damping factors.

Furthermore,

> no dependency of phrase command amplitude on utterance duration or speech tempo was found.

Style variation using the Fujisaki model is reported in [51] for Japanese, in which they model style differences using fixed differentials for the bottom line, the phrase amplitude, and the accent amplitude. This approach allows for mixing styles together – there is a vector difference between the parameter settings, and so they can be combined as weighted averages. Thus, one could in principle estimate the "sad" parameters and apply them to a voice by the vector difference from the normative model.

Van Santen and colleagues [41, 52, 19, 53] generalize the superpositional model with convex functions, in which the time course of F_0 is composed of phrase and accent curves. The response of the filters in the Fujisaki model are of a restricted class, but if we were to allow them to be arbitrary positive curves that rise to a peak and return to zero, it approaches the same order and power as the van Santen model. The sum-of-products model is more general than using a damped exponential response, because the shape of the response may be time-warped.

The contour for an accent group has several points estimated as percentages of the peak height, and can be seen as a time-warping of an underlying curve that is dependent upon the sub-durations of the accented syllable and the unstressed following syllables in the stress group. This approach can capture the data with higher fidelity than the Fujisaki model, which is also superpositional, but with the additional complexity of more parameters. Van Santen also takes into account, somewhat, the segmental material, but uses only a single perturbation in post-obstruent sonorants. While the model has only

phrase and accent curves, a tantalizing possibility is suggested near the end of the paper:

> We could generalize the concept of accent group, which is based on syllables being dichotomized into stressed and unstressed syllables. For example, we could trichotomize syllables into Strong, Medium, and Weak, and posit that there are two types of accent groups, Strong and Medium, that might overlap (share syllables). Strong accent groups would start with strong syllables and be terminated by strong, but not be medium or weak, syllables; medium accent groups would start with medium syllables and be terminated by either strong or medium, but not by weak, syllables.

[41]

This may address some problematic issues in phrase and accent shape in utterances with more than one phrase [54]. Venditti and Maeda [53] showed phonetic variation of fundamental frequency of Japanese boundary tones that depended upon illocutionary type, which clearly shows that a single magnitude on an abstract J-ToBI pitch accent is not sufficient to represent the surface variation, even within-talker. J-ToBI is a version of ToBI adapted for Japanese intonation.

Fujimura's Converter-Distributor model of phonetic implementation [55] models the organization and information flow in order to realize physical signals from abstract representations, but does not specify what the important phonetic results for F_0 are – it focuses on the realization of phonemes, but there is no reason that F_0 could not also be modeled in a similar framework.

2.2.7 *The Rise/Fall/Connection (RFC) Model*

The Rise/Fall/Connection (RFC) model [56] and Tilt [57] are phonetically descriptive of the time course of F_0, but do not model a theory of phonological implementation. This is a phonetically descriptive model, as is INTSINT [58]; however, neither model segmental effects explicitly. Rather, they describe the contour directly, and so are prone to confounds from segmental effects.

Template-prosody [59] is an interesting approach to prosodic generalization based on the syntagmatic structure. In much the way speech segments are stored in databases and referenced by contextual effects, whole intonational contours in phrases are coded symbolically, and looked up at run-time. This approach is well-suited to micro-domain synthesis, and could benefit substantially by modeling the segmental

effects separately. Factoring out the segmental contributions to the time-course of F_0 will make for higher-fidelity models as well as improve generalization.

Vector-quantized shapes of F_0 movements [60] may also benefit significantly from segmental models, as peak timing has been shown to be strongly related to the segmental material. Segmental effects may be considered codeword-dependent models.

All these approaches are susceptible to the confounds of segmental perturbations in the analysis phase, excepting by hand-analysis, and none have been worked into a framework of talker and style variation to a great degree. Smoothing is usually used, which blurs the segmental effects with the tonal motion, and interpolation is often used between voiced regions.

2.2.8 *Intonation From Text*

The prediction of intonation from text remains a difficult problem. Some systems use systems of rules over symbols (e.g., [40]), while others try to directly predict the phonetic form from feature vectors (with Tilt or ToBI [45]). Naturally, the assignment of prosody to express meaning by manipulating prominence and scope depends on a reasonable model of text understanding, but simple effects such as given/new effects (using a queue for a paragraph), word class, position in the phrase, and other relatively easily measured quantities are still in wide use.

The general form of intonation prediction is

$$F_0(\mathit{unit}[i]) = f(\mathit{UtteranceTree}, i)$$

where the utterance tree is the decomposition of the utterance in terms of its feature geometry. One model is that an Utterance contains Phrases, which are made up of Words, which contain Syllables, which join into Stress Groups. Syllable quantities can then be predicted as functions of their syntagmatic and paradigmatic components: the syntagmatic ones describe the context and containment structure (such as position in syllable, word or phrase), whereas the paradigmatic components are those that name the type itself (stress, segment name, part of speech, and so on). The components that the F_0 is determined to depend upon in the tree can be turned into an extended vector that describes the unit's identity and embedding, and each of these vectors can then be used for training. Generalized additive models may be used, as in the sum-of-products model [41], linear regression models [61], Hidden Markov Models, neural networks, decision trees, or other methods.

2.2.9 *Inducing Local Parameters from Constrained Tone Sequences*

A corpus was designed that systematically varied three things: Metrical structure, tonal structure, and phonetic structure. These are described at length in two papers [62, 61]. This work was built using sequences of ToBI labels as in [31]. For the phonetic structure, the demisyllabic assumption was made: onsets are relatively independent of codas given the vowel. The material was limited to two pitch accents per utterance, over two words of varying syllabic length. Each prompt was constructed with these variation, with an initial carrier phrase ("Like a", "As a").

The metrical structure was varied, as in this example with two monosyllabic words:

* : .] "BAD man."

. : *] "bad MAN."

* : *] "BAD MAN."

* | *] "BAD, MAN."

The process was continued, adding more syllables to each word incrementally. The approach is similar to enumerating state machines by diagonalization.

* . : *]

* . | *]

* : . *]

* | . *]

...

....* | ....*]

where

- * is a pitch accent and a lexical primary stress (L*, H*, L+H*, L*+H)
- . is an unaccented syllable (also unstressed, but may have secondary)
- : is a word boundary
- | is a minor phrase boundary (L-)
-] is an phrasal end tone (L%, H%)

Pitch accents were systematically varied over the * accents, and delivered by talent with detailed knowledge of ToBi. The same metrical material appeared in balanced contrast over the pitch accents on same phonetic material. Thus, the metrical corpus was expanded and recorded, in order to try to capture all the necessary variations.

Words were selected from the CMU dictionary, with cross-word phonetic constraints, using a greedy algorithm, and utterances were formed from common words in order to create a balanced corpus crossing demisyllable contents and metrical embedding in carrier phrases[4]. Words were used in each prompt with a carrier phrase, in variety of contexts[5] There are parallel phonetic structures for different phrasal contrasts. The data was collected using natural and reiterant speech, substituting the syllable "ma" for each syllable, as spoken by Victoria Anderson, a trained linguist. Each tone adds a number of degrees of freedom to the overall tune. The result is smoothed with a rectangular window of fixed duration, making a smooth contour. The reiterant speech was used for modeling, and segmental effects presumed negligible[6].

For modeling, we constructed local shapes of two or three points joined by straight lines for each pitch accent, phrasal, and boundary tone based on observation of the data and used linear interpolation between local tone realizations for each utterance in the training corpus. An initial set of annotations was made using the specifications that went into to corpus design, with an initial approximation of each tone type aligned with the F_0. The piecewise intonation contour was smoothed using a Hamming window.

The annotations were then modeled using the data, to produce parametrized local templates. These were fit to the data, and, similar to the EM approach with epsilon scattering in grapheme-to-phoneme modeling, iteratively improved by gradient descent to minimize the error the local templates with respect to the entire corpus. The new templates were then fit again to each of the training examples, and the process repeated to refine the template shapes. The control points of the templates were predicted using multiple linear regression, with the metrical structure and syllable structure as factors.

The LSAFo model of interpolative model constraints was used to find a set of parameters for F_0 contours using simulated annealing. The model consisted of a set of local templates representing control points placed on a metrical grid and smoothed. The local templates consisted of:

4 An example of the text is "Like an ARCHduke LAUNdress," with and without a break between the final two words.

5 "rife undertaking, anyhow fanatic, grab disruptive?"

6 Even though they could clearly be seen in F_0 tracks.

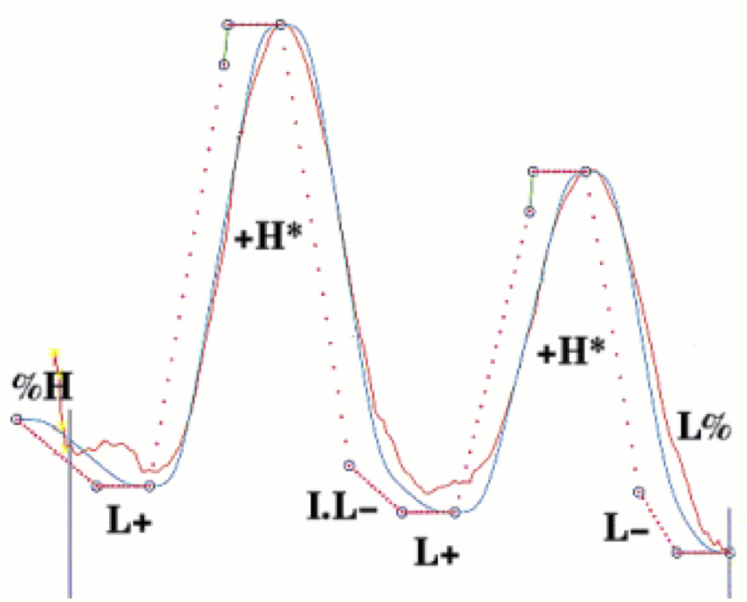

Figure 1: LSAFo Model Fit to an Utterance

- Pitch Accents * (H*, L*). A Starred Accent adds 3 control points: One anchored at the peak, one a window's length before the peak, and one free in time and F_0, but within 400 milliseconds (ms).
- Bitonal Components + (L+H*, H*+L). The second tone in bitonal adds 2 points: the first point placed at the peak or nadir of the minor tone, within 500ms of the temporally first control point in the starred accent; the second point a window's length preceding it, at the same F_0.
- Phrasal Tones - (L-, H-). A Phrase adds 1 point at the inflection point
- Boundary Tones % (%H, %L, L%, H%). A Boundary adds 3 points at the final boundary. One may be sufficient at the initial boundary.

- Smoothing window size: One parameter for the whole model (rectangular)

With these constraints, and the annotated data, a contour was fit to each utterance from the ToBI labels and the time and F_0 of the associated syllables. We then began alternating minimization using gradient descent: first the global shape was optimized, then local models estimated. The constraints of these local models were then propagated back to the global configuration for each utterance, and fit again using gradient descent. This process was repeated to convergence. Once the best-fit annotations of the training data were found, context-dependent linear models for the local movements of control points as a function of accent type, abstract magnitude, metrical distance, and prosodic structure were produced. The result was about 2.7 Hz average distortion with respect to the training set.

Evaluation of models has been fraught with the usual problems that perceptual studies with many confounding factors bring, but this has been aggravated by a lack of systematic pairwise evaluations of models in minimal contrast conditions.

2.2.10 F_0 *Modeling in CLUSTERGEN*

In CLUSTERGEN, frame-based durations are predicted using a single decision tree for each state type[7]. The current implementation first predicts F_0 for each frame, and then spline-smoothes through a beginning, middle, and end of each syllable over a three-frame window. Voiced-unvoiced prediction is part of the frame prediction. The independent features include accented/non-accented but does not use ToBI label itself[8]. Variance has always been a problem, leading to the introduced global variance (for F_0 and duration and all mel-generalized cepstral coefficients), but this is not in the baseline FestVox).

For evaluation, Anumanchipalli [63] preferred ABX evaluation by human subjects over objective measures of RMSE and correlation[9]. The objective measures are difficult to correlate to improvements in subjective assements.

7 Typically, there are three states in a segmental model in CLUSTERGEN. The phoneme `/s/` would have three associated state names, one for each state (e.g., `s_1`).

8 A runtime feature function checks if there is a pitch accent or tone associated with the syllable.

9 "As RMSE and correlation are not ideal metrics for evaluating perceptual goodness of synthetic intonation, subjective ABX listening tests on each pairs of the above models were carried out." p. 40

2.3 ENERGY

Models of time course of energy of the source may also be estimated from data, and are important to perceptions of relative magnitude [40]. These estimations are similar to the those of the time course of F_0. There have been several characterizations of overall energy or intensity using decision trees [64] and neural networks in one form or another. Recently, global variance modification and [65] the modulation spectrum have been used for it (and nearly all parameters in HMM synthesis [66] [67]). These methods focus on overall averages across a corpus, rather than performance variations, so even when the variance is increased to approach that of the training corpus it is a statistical average and not a reflection of accent placement or explicit prosodic modeling.

2.3.1 *Energy Modeling in CLUSTERGEN*

Energy modeling in CLUSTERGEN is a by-product of the mel-generalized cepstral estimation [68]. Decision trees or random forests estimate the parameters on a per-frame basis, includes the frame number and distance (in abstract units) from the state margins. While energy is not independently modeled from the short-term acoustic frame generation, the trees consider prosodic features, including stress and phrasing in finding the best leaf node containing parameters.

2.4 PRONUNCIATION

Each word to be synthesized is placed into the utterance with some internal structure. It contains, at a minimum, a sequence of segments which correspond to each word, but typically also includes grouped information such as syllables, which contain segments and have an associated stress feature.

In the initial stages of text processing for English in Festival, the input text is tokenized, and parts of speech are associated with the textual tokens. The 'Token' relation itself could be called 'Token_Word', because it encodes not only the tokens, but also the Word representation as used by the rest of the system. The parts of speech of the relevant parent tokens are used to determine the tags on the related Words when they are expanded by the Token-to-Word module.

2.4.1 *Known Words*

Each word to be spoken must be assigned a pronunciation. Text-to-speech synthesis systems such as Festival rely to one degree or another on a simple dictionary lookup. In the Festival version of the CMU Dictionary (version 0.4 [10]), there are 105,376 entries. Most of these use the part of speech 'nil', which indicates that pronunciation does not differ based on part of speech. When the word pronunciation is looked up, the dictionary is checked to see if there is a separately defined entry for the corresponding part of speech, and if not, the system falls back to the 'nil' pronunciation.

2.4.2 *Grapheme-to-Phoneme Conversion*

When there is no static, predefined pronunciation in the dictionary, there are several paths a Text-to-Speech system may take to produce a pronunciation. There may be some transforms, such as decompounding, which may generalize the existing dictionary based on some model. After delegating this process, if there is no pronunciation, a grapheme-to-phoneme converter does the best job it can using the input features to infer a pronunciation.

One set of approaches to out-of-vocabulary pronunciation generation involves creating a set of likely alignments between symbols in the grapheme and phoneme sequences [69] [70], and then training a machine learning technique to produce mappings in a way that generalizes to produce acceptable pronunciations given the alignment.

The technique takes a training set – the CMU dictionary – and generates an initial set of letter-phoneme alignments, where there are never more phonemes than letters. For cases where a letter often spans two phones, a cover symbol of those phones together is created (such as 'x' -> /k s/). Because there are often fewer phonemes than letters, some alignments will include null strings, or epsilons, on the phonetic side, which changes the alignments from being L:P to being L:L; that is, one-to-one and onto. A model is trained from these alignments, and typically a single best pronunciation is used for synthesis. As an example, one reasonable alignment between the letters and phonemes of the word "algebraically" is

```
a  l  g  e  b  r  a  i  c a  l  l  y
AE L  JH AH B  R  EY IH K _  L  _  IY
```

10 This is the version that is still the most current one in Festival as of this writing. While there have been updates since, they have not been converted into Festival format, which would take some additional effort to enumerate the parts of speech of homographs

The underscores represent the "epsilon" phonemes, where the letter is associated with an empty output. For this example, the two epsilon outputs could be associated with any of the letters, so long as the ordering is unchanged. The number of possibilities for this example is $C(n, k)$, or $\frac{13!}{11!2!} = 78$. With over 100,000 entries, the total search space of possible alignments is intractably large to approach by brute force.

2.4.3 *Epsilon Scattering to Improve Alignments*

Given an initial alignment, each letter is expanded into an extended vector, which contains the letter context and possibly a part of speech or other tag, and the "observed" phonemic output for an alignment. However, for each letter/phoneme sequence, there are many possible epsilon assignments. Without prior probabilities, these are estimated through *epsilon scattering*, which checks all possible combinations, ranks them with respect to the model, and then assigns the best alignment to replace each instance in the training set. The model is built again, and after several iterations of Expectation Maximization, the alignments converge. The final models, made from the regularized alignment data, are then used to create pronunciations for out-of-vocabulary words at run-time.

Epsilon scattering here was the process of enumerating the possible combinations of alignments or states which should be evaluated with respect to the model. It is less important to note that the combinatorial function $C(n, k)$ was used than to recognize the enumeration process.

While improvements in generalizing N:M sequence mappings have come since, such as Phonetisaurus [71] using Weighted Finite State Transducers, the technique opened the possibility of iteratively improving the annotation by using the model to find likely interpretations. The alignments are regularized over time, based on what is likely given a model made from the aggregate of alignments.

2.4.4 *Pronunciation Variation*

Pronunciation of words is one of the many ways in which talkers differ. While the vanilla system produces a deterministic predicted pronunciation, human beings are quite flexible and varied in their delivery. You say tomato, I say... tomato. Reduction in vowels is one common kind of post-lexical variation. While some of these are due to dialect and idiolect distinctions, allophonic variation may come about as a results of a variety of factors in production [72], [73], [55]. Neural networks have been used to model postlexical variation in a talker-

dependent way [74], and abstract lexicons that contain a structure to represent dialect variations are modeled in [75].

The Festival front-end generates the same pronunciation for each voice database in Arctic by default, and does not take into account individual variation. While it is possible to include or change the pronunciations of words, this is not commonly done. It is normative in this sense, whereas the talkers do not play by the same rules when they speak: the actual delivery often diverges from the predicted pronunciation, phrasing, and stress.

One attempt to get around this dead-reckoning approach to labeling the corpus introduced alternate pronunciations into a CMU Sphinx pronunciation dictionary [76]. While this does mitigate the issues of multiple pronunciations in the training data, the variation was over a small set limited number of function words and the speech synthesizer only a single form from the front-end processing irrespective of the voice used. That work also extended to making predictors for the various forms of the function words using decision trees, in order to improve the generation of pronunciations from Festival's front-end processing.

3

HETEROGENEOUS RELATION GRAPHS IN FESTIVAL

Festival [7] is a widely used platform for speech synthesis. FestVox [6] is a set of tools, data and documentation for creating voices and languages in Festival. Many researchers have built speech synthesizers using these platforms.

Heterogeneous Relation Graphs (HRGs) [77, 78] are used to represent and encode the linguistic information in Festival and FestVox. The structures and values for each element necessary for synthesis are elaborated into a graph for each utterance in the training material, and is used as a specification of what should be rendered for output. The HRG acts as an interchange format between functions or modules that operate on the linguistic structure.

Taylor et al. [78] argue that the HRG, as an abstract concept, is quite general and "theory neutral" with respect to the structures that can be encoded. However, the only available public implementation is in Festival, with much of that coming from the Edinburgh Speech Tools [24]. In reviewing the HRGs, we refer to the Festival implementation and the Festival/FestVox conventions for them, rather than the abstract HRG, except where the contrast is called out.

The organization of the utterance structure represents a model of speech for synthesis that has been derived from applying theories and observations of engineering results over time. Within Festival, the utterance contains a set of nodes with feature values, and a set of relations, which describe structure and type. As the utterance structure is elaborated in order to pass to the back-end for synthesis, relations are added to the structure. The text is added in a `Token` relation; the tokens are assigned part of speech by a textual part of speech tagger. The `Word` relation is added, using information in the `Token` relation, and so on, with each new relation providing data to the next stage of the processing pipeline.

The singly-rooted tree structures of systems like `MaryTTS` [79] and `idlak` [80] may be converted into HRGs, but the converse is not always possible. The techniques and discussion which apply to HRGs may also be applied to simple trees converted to HRGs.

3.1 STRUCTURE OF AN HRG

An HRG can be serialized to a file format; in FestVox, these files typically end with `.utt`. The structure of the HRG, and all accompanying data, may be placed into these files.

While the file format is not, strictly speaking, the theoretical description of the HRG, we use it here as an existence case, being the only common implementation that we have available. It is worth separating the concept and generality of the formal HRG from the implementation, especially as we consider improving on the serialization or expose new functionality in libraries. The theoretical HRG encodes a multigraph, with a common set of nodes and several graphs defining relations between the nodes. Nodes may have features, and participate in relations, but with constraints.

The outline of the utterance file consists of an `EST_File` header, a list of `Stream_Items` (nodes), and a `Relations` section, which contains a list of `Relation` definitions.

The header contains basic annotations of the file format and contents, including format version and magic number so that utterances may be identified easily.

`Stream_Items` enumerate the objects represented by the nodes in the graph. The first number is the node id, and it is followed by a list of feature names and their values for that node. The features are delimited by semicolons and spaces; within the feature, the name and value are space-separated, and the value is a quoted string when necessary.[1]

```
18 id _9 ; name Author ; pos_index 8 ; pos_index_score 0 ;
   pos nn ; phr_pos n ; phrase_score -7.42703 ; pbreak_index 1 ;
   pbreak_index_score 0 ; pbreak NB ;
```

Figure 2: A `Stream_Items` Entry

Here is an example for the first word of the first prompt in the Arctic [9] database US English talker SLT, from a basic build using FestVox (figure 2).

The `Relations` section follows. Each `Relation` must have a unique name. Within each relation, the edges and edge types are listed in a particular format. A short example of the edge lists for the `Word` relation of an utterance is shown in figure 3.

1 Note that there is a bookkeeping feature named `id` (here, `_9`), and the actual identifier using within the stream, which is `18` (the very first number). This feature is, confusingly, only used internally to keep track of things, but not as an identifier for computation.

```
Relations
...
Relation Word ; "(" ")" ;
1 2 0 0 2 0
2 3 0 0 3 1
3 4 0 0 0 2
End_of_Relation
...
End_of_Relations
```

Figure 3: Serialized HRG Relations

Each row in the relation has six numbers. The first two are identifiers: A relation-internal, relative `id`; and a `Stream_Item id`. The relation-internal `id`s then appear in the four numbers relating to the edge types: up, down, next, previous. A `0` indicates no edge of that type from the node; otherwise, it contains the relative `id` of the target within the relation.

```
relative_id stream_id up down next previous
```

These edges encode doubly-linked lists in two dimensions, capable of representing either a tree or a flat structure (a degenerate tree). Each `id` may only have one row associated with it (the first two columns). A node may only have zero or one exit edge of each type. That is, within a relation instance, any node may have only up to one down, up, next, or previous node.

The file format also includes with some closing material for each section to ensure data integrity. For each `Relation` there is an `EndRelation`, and similarly for the header, stream items, and the file as a whole.

3.2 CONVENTIONAL RELATIONS IN FESTIVAL

An HRG that contains a representation of an utterance is referred to as an utterance structure. It contains the linguistic objects (`Stream_Items`), and a conventional set of `Relations`. In FestVox voice builds, the base utterances are in the `festival/utts` directory.

The root of the tree is implicit with a depth of zero. The conventional relations each have a tree depth of between one and three, with one being a list-like structure which may be considered a "type", though there is no explicit notion of type outside of participation in a relation. For each relation, we can view them in terms of the degree one types which they connect. The depth of each relation is fixed, and are listed in table 1.

Relation	Depth	Types
Token	2	Token, Word
Word	1	Word
Syllable	1	Syllable
Segment	1	Segment
SylStructure	3	Word, Syllable, Segment
IntEvent	1	IntEvent
Intonation	2	Syllable, IntEvent
Target	2	Segment, Target
Phrase	2	Phrase, Word

Table 1: Conventional Relations in Festival

While this is the conventional baseline set of relations, the HRG structure allows for any number of relations and shapes along these lines. Other relations may be added, depending on the synthesis method. The parametric CLUSTERGEN and unit selection `clunits` methods each use some additional information internally as well, some of which is not typical serialized into a file – and so is not generally observable.

3.3 INDUCING AN UTTERANCE STRUCTURE

Synthesis in Festival is a set of operations that induce new graphs from existing ones. We begin with the empty graph and call a function on it with a few arguments, such as the input text in text-to-speech synthesis. The output is a copy of the original graph, with the new relation added.

We will not discuss each of the typical functions in a Festival voice; these are described in the Festival manual [] and FestVox []. In this discussion, we look at the process of building an HRG using composition of functions.

When the system is called, it looks at the type of request received in order to call a composition of functions which elaborate their respective relations into the utterance structure (an HRG). The output of each function is passed on to the input of the next, and the HRG is populated with relations and feature values as the functions are applied. Typically, information is not modified, but a function may, in principle, modify part of the HRG when it produces output.[2]

2 Code within Festival had to be changed in order to really be in the functional style. The function which iterated over the elaboration functions held a reference to the

```
(defUttType Text
  (Initialize utt)
  (Text utt)
  (Token_POS utt)
  (Token utt)
  (POS utt)
  (Phrasify utt)
  (Word utt)
  (Pauses utt)
  (Intonation utt)
  (PostLex utt)
  (Duration utt)
  (Int_Targets utt)
  (Wave_Synth utt)
  )
```

Figure 4: Function Application Order Definition for Text

The function `defUttType festival/lib/synthesis.scm` shown in figure 4 defines the order of function application for a synthesis method. The method "Text" begins with text segmentation, followed by part of speech imputation over the token sequence onto the tokens. This function initializes the token relation from the text, adding the tokens as a list of nodes into the output graph. The result is input to a Phrasify function which adds an initial Phrase relation. Then, to Word, which duplicates some items and moves them into two levels in Token (token to word), and two levels in Phrase (phrase to word). The next phase introduces pauses, given the latest utterance structure. The process continues with each of the enumerated functions.

A fully elaborated utterance may contain several thousand nodes. For the Arctic prompt `arctic_a0407`, the graph contains 1695 nodes and 7306 edges in 13 relations with cardinality shown in table 2. The text of this utterance is

> Mercedes screamed, cried, laughed, and manifested the chaotic abandonment of hysteria.

from Jack London's *The Call of the Wild*.

3.4 VISUALIZING THE FESTIVAL UTTERANCE STRUCTURE

An HRG for CLUSTERGEN synthesis is shown as a force-directed graph using the `dot` utility of `GraphViz` [81] in figure 5. This rendering

original utterance, which prevented removing or changing some things with dependencies. This was clearly a bug, and not as the authors intended it.

	Relation	Nodes	Edges
1	Token	26	50
2	Word	11	20
3	Syllable	24	46
4	Segment	72	142
5	SylStructure	102	202
6	IntEvent	8	14
7	Intonation	15	28
8	Target	96	190
9	Phrase	12	22
10	HMMstate	202	402
11	segstate	274	546
12	mcep	1311	2620
13	mcep_link	1513	3024
	Total	1695	7306

Table 2: Node and Edge Counts by Relation for CLUSTERGEN Synthesis of `arctic_a0407`

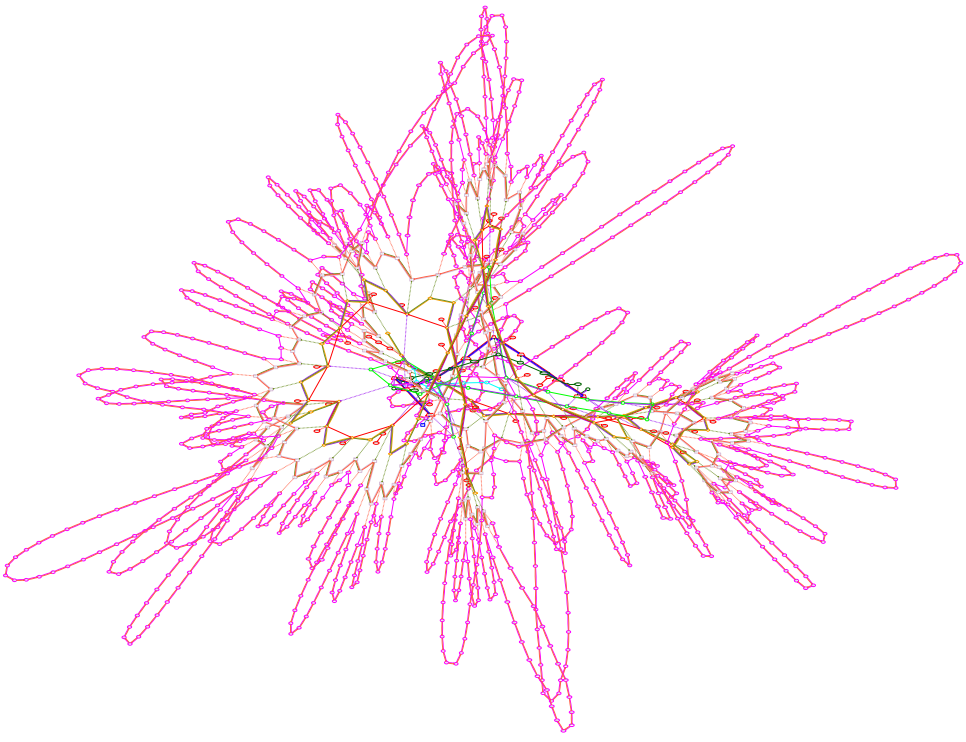

Figure 5: Force Directed Graph of a Large HRG

is the same Arctic prompt `arctic_0407` as in table 2. To create these graphs, a program was written to express the relations in Festival utterances and HRGs in the graph specification language of `GraphViz`. The figure shows the relative complexity of the graph structure. For large graphs, it is unwieldy to show the whole graph when inspecting local structures.

Visualization of the utterance exposes structures that are otherwise rather abstract. To get a view of the structures, tools were created in the `Karnival` framework, discussed further in Chapter 6. A utility named `hrg.py` was made which reads and writes HRGs, and optionally converts the graph structure to `dot` format for plotting with `GraphViz` [81]. The tool can produce Scalable Vector Graphics (SVG) output with tooltips revealing the features of each node and optional hyperlinks.

A simple specification format for the graph conversion program allows customization of which nodes to show, and induces type names from the relations of depth 1. To avoid clutter, only the `down` and `next` edges are retained without any significant loss of information in the visualization – next and previous are always paired, as are up and down.

The conventional relations for the text "Ah, indeed" (`arctic_a0329`) are shown in figure 6, down to the level of `Segment`. With the tool, we can look at a few of the relations together and see their characteristic structure. While the HRG definition is simple, and the types and relations appear straightforward, unexpected quirks are revealed in the graph. This will be illustrated in some detail in the next section.

3.5 FEATURES, PATHS, AND FEATURE FUNCTIONS

In Festival, features are attached to nodes in HRGs. Some of these nodes are placed into the utterance structure, while others are computed dynamically and may not be inserted. These dynamic features are referred to as feature functions, and are not available when inspecting the serialized utterance file. Features are used in all aspects of the system, and provide a flexible way to configure what data is available during training, as well as what is produced by the front end for synthesis.

Feature paths and functions are a convenient way to enumerate and describe what information from the general graph may be made available to machine learning algorithms as sequences of extended observation vectors. The provide an interface between the utterance structure and the serializations used for training models.

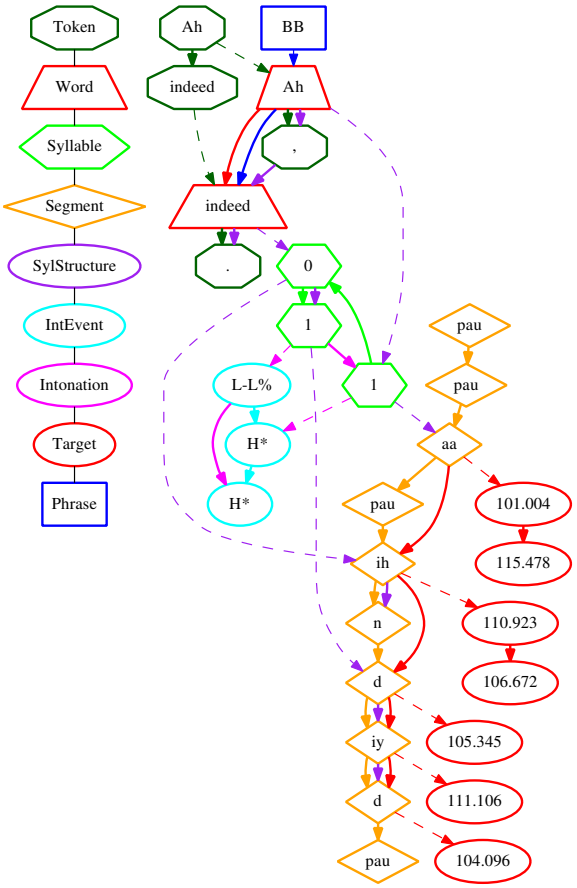

Figure 6: Visualization of a Small HRG

One type of feature function expands linguistic features for a phonetic segment, which is essentially a table lookup based on the name of the segment. Rather than encode these deterministic expansions in the tree, there are functions which compute the values given the existing paradigmatic values. These include `ph_vc`, which encodes whether the segment is vocalic or not; ph_ctype which encodes manner, ph_vlng long vowel, and so on. For phonetic features, these are defined in the phoneset definition for the language. Some examples are shown from the `radio_phones.scm` definition in table 3.

Some convenience functions operate over the structure to compute a result, such as looking across relations to determine whether a syllable bears a pitch accent. Typically, if the result can be deterministically derived from the HRG, it is not written into a file when it is serialized. For example, the `accented` feature function takes an utterance and a `Syllable`, walks from there to the `Intonation` relation which connects `Syllables` and `IntEvents`, and checks whether there is a pitch accent connected. If this feature is called at run-time, it will produce a value derived from whether a pitch accent is associated with the syllable, and is usually not written into the file when it is serialized.

Segment	vc	vlng	vheight	vfront	vrnd	ctype	cplace	cvox
aa	+	l	3	3	-	0	0	0
ae	+	s	3	1	-	0	0	0
ah	+	s	2	2	-	0	0	0
ao	+	l	3	3	+	0	0	0
aw	+	d	3	2	-	0	0	0
ax	+	a	2	2	-	0	0	0
axr	+	a	2	2	-	r	a	+
ay	+	d	3	2	-	0	0	0
b	-	0	0	0	0	s	l	+
ch	-	0	0	0	0	a	p	-
d	-	0	0	0	0	s	a	+
dh	-	0	0	0	0	f	d	+

Table 3: Sample Phonetic Features in `radio` phoneset

Features may also have a path associated with them, in a unique syntax. Steps in the path are separated by a dot, with the last step being a feature name as attached to the node or a feature function. The moves (steps) are listed in table 4.

In addition to path moves, there are relation selectors, which act as both a predicate and a coordinate shift. These start with the prefix `R:`, followed by a relation name; for example, `R:SylStructure` selects the `SylStructure` relation. Any moves on the path until the next `R:` are within the named relation.

Features with paths are used, importantly, in enumerating extended observation vectors for training of duration, intonation, and signal generation models. A configuration for each of these is simply a list of features (with paths), starting from the node of interest.

In figure 7 we see the first few features in the

```
festival/dur/statedur.feats
```

file, which enumerates the features for state duration in `clustergen`. The first item, lisp_zscore_dur, is the feature to be predicted (duration z-score computed in a feature function) from those that follow. When dumping out the features, the `HMMstate` relation is traversed, which each node in turn used as the origin of the path. Each path is walked, and the resulting value is placed into an observation vector for the unit. These vectors are then used in both training and run-time processing to produce synthetic output.

Notation	Move
n.	next
p.	previous
nn.	next next
pp.	previous
parent.	most previous, then up
daughter1.	first daughter
daughter2.	second daughter
daughtern.	last daughter
first.	most previous item
last.	most next item

Table 4: Feature Path Moves

```
lisp_zscore_dur
name
p.name
n.name
R:segstate.parent.name
R:segstate.parent.p.name
R:segstate.parent.n.name
R:segstate.parent.R:SylStructure.parent.syl_onsetsize
R:segstate.parent.R:SylStructure.parent.syl_codasize
R:segstate.parent.R:SylStructure.parent.R:Syllable.n.syl_onsetsize
R:segstate.parent.R:SylStructure.parent.R:Syllable.p.syl_codasize
R:segstate.parent.R:SylStructure.parent.position_type
R:segstate.parent.R:SylStructure.parent.parent.word_numsyls
R:segstate.parent.pos_in_syl
R:segstate.parent.syl_initial
R:segstate.parent.syl_final
R:segstate.parent.R:SylStructure.parent.pos_in_word
R:segstate.parent.p.seg_onsetcoda
R:segstate.parent.seg_onsetcoda
R:segstate.parent.n.seg_onsetcoda
R:segstate.parent.pp.ph_vc
R:segstate.parent.p.ph_vc
R:segstate.parent.ph_vc
```

Figure 7: Example Features in `statedur.feats`

`segstate` is a relation which connects a segment to its sub-segment states. `R:segstate.parent.name` is a path from the `HMMstate`, switching to the `segstate` relation, then going to its parent `Segment` and

getting the feature `name`. This may be continued with more steps and crossing over to other relations:

```
R:segstate.parent.R:SylStructure.parent.R:Syllable.n.syl_onsetsize
```

switches from the parent segment to the `SylStructure` relation which connects words to syllables and syllables to their segments, to the parent syllable, then switches to the `Syllable` relation and steps `n.` to the next one and evaluates the feature function `syl_onsetsize` which dynamically calculates the size of the onset. Feature functions and paths provide a flexible means of addressing information in the graph without writing any code, given the utterance structure and code definitions.

4

AUTOMATIC FEATURE DISCOVERY

In statistical speech synthesis, spoken audio data is annotated in order to make models for rendering synthetic output. Utterances are typically represented as a graph with nodes that correspond to linguistic items, with a set of features for the items. These may be divided into two main types: paradigmatic features, related to what an element is; and syntagmatic ones, describing structure.

We automatically collect paradigmatic features and generate syntagmatic features from arbitrary relations by walking neighborhoods in a collection of HRGs[1]. This removes the need to hand-engineer feature lists and paths, and allows for new relations and features to be introduced without manually enumerating the features. The resulting feature set is evaluated, and promising features retained. Automatic structural feature discovery makes feature exploration easier, and reduces the effort of implementing linguistic theories and new languages.

The approach applies to statistical parametric speech synthesis as well as trainable unit selection synthesis. The graphs are traversed to produce extended observation vectors for training and run-time decoding. The results for using automatic derived structural features are comparable to the best hand-written sets for mature languages in Festival [7].

Statistical speech synthesis methods ([82], [21], [28], [80] and others) train voice models from spoken corpora. While a corpus may initially consist only of audio, or a set of sentences to be spoken, more annotations may be added, until the corpus has been fully decorated with features. Given a list of files and a set of waveforms, we may pass the text to the front end of a Text-To-Speech system and obtain a representation of features and utterance structure as have seen in Chapter 3.

An example is shown in Figure 8 of an HRG down to the segment level for the sentence "How much was it?", which is the smallest utterance structure in the Arctic database ([9], prompt a0207).[2]

1 The Heterogeneous Relation Graphs described in Chapter 3

2 The HMMstate, mcep_link and mcep relations are omitted to show the general form, which consists of multiple relations over a common node set.

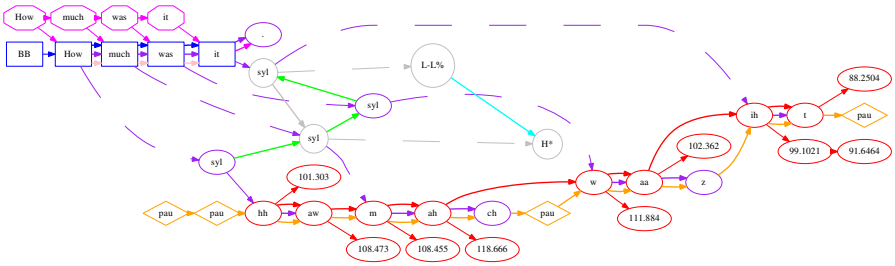

Figure 8: *HRG for "How much was it?"*

Once the utterance information is fully elaborated, models are trained using the features that are in the structure. In Festival, these are encoded as paths from one node to another in a graph, where paths may traverse between relations. These paths end in a feature, which may be either a static value or a function which lazily computes the value.

4.1 PARADIGMATIC AND SYNTAGMATIC FEATURES

The features may be divided into two categories: paradigmatic features, deriving from identity (or substitutability); and syntagmatic ones which describe relationships and structure. In the Festival feature specification, a path is an edge traversal and a feature function applied to the relative node. The traverse part is syntagmatic, and the feature function typically applies to elaborate the paradigmatic information, such as expanding phone names into phonetic features. However, the split is not entirely clean in this case, because there are feature functions which encode some structural information – position in syllable, for example.

These paths are produced manually, and depend on knowledge the topology input HRGs. When a new paradigmatic feature is added, there are several places where changes need to be made manually, including new paths to be added to the feature path specification, and redefining the data definitions. We believe these relational features are rarely inspected or modified when new voices and languages are built with FestVox [6] and Festival. Furthermore, there has been no easy way to add or modify features from outside the base processes in FestVox and Festival, which limits experimentation.

As trees are a subset of relational graphs, it is easy to create an HRG from a tree-based structure as described by the XML structures used in MaryTTS [79] and Idlak [80]. However, HRGs are more powerful, in that there is not complete containment as in XML trees: while pauses are segments, they are not in syllables or words – and by extension, not in phrases. A pause segment may not have any parents at all,

whereas in a total tree ordering, each element must be attached. Total trees can be converted to HRGs by defining a new relation which visits every node in the graph, adding a sequential relation for each element type, and then running processes on the graph.

We propose a method for automatically capturing features, enumerating their values, and deriving structural features from a voice corpus by traversing paths over neighborhoods in HRGs.

4.2 EXPLORING STRUCTURAL FEATURES IN A PATCH

For each transcription in a voice database, the front-end is run to create an initial structure, and aligned to the audio. The HRGs are stored externally, and other processes may add relations or decorate them further. Once the utterances have been fully elaborated, the structures are cataloged. Input structures may be converted to HRGs for processing.

In order to find feature paths, we traverse one of the relations and explore the neighborhood. For CLUSTERGEN [21] voices, we use the `mcep`[3] relation, which describes a sequence of fixed-duration frames. A neighborhood is described which is to be explored around each node in the relation, which may be either a list of paths or a policy for exploration. In order to approximate a neighborhood used previously, we limited the paths to up to two hops back, one hop forward, and any number of hops up, where a hop is traversing a single edge in any relation.

3 mel-generalized cepstra

Relation	Direction	Max Exits
combined	all	6
combined	up	5
combined	down	0
combined	prev_next	2
Token	all	0
Phrase	all	0
Word	all	1
Syllable	all	1
Segment	prev	2
Segment	next	1
HMMstate	all	1
mcep	all	0
SylStructure	prev_next	0
segstate	prev_next	0
mcep_link	prev_next	0

To save processing time, the paths are enumerated before evaluating the feature functions, and these paths may be re-used as long as no new relations are added. The list of paths present is counted for the corpus, and any with trivially low counts may be dropped. In order to save some time, the paths may be approximated using a subset of the corpus for rapid development – the longest 10 utterances are very likely to contain all the paths, so that path determination may be done quickly.

To automatically generate syntagmatic features, we visit each node in the relation of interest and apply each path in the path list to find relative nodes. For each relative node, we determine all the relations in which it participates, and describe it in terms of only three generic positional features: integer indices from the left and right in each relation (`num_prev`, `num_next`), and a floating point proportion within parent (`position`). These are decorated into the node with a unique name prefixed by the relation type, as `Segment_num_next`, `SylStruct_position`, etc. A node may participate in many relations, and the structural information is named and determined for each. In Festival, each child gives different position results for each parent. A node in `Segment` and `SylStructure` will have different values for the structural features in each relation: `Segment` is a stream and `SylStructure` is a tree which intersects a subset of nodes that participate in `Segment`.

For each node in the relation of interest, we visit every node in the path list and inspect all the features in the relative node. Feature values are cataloged by final relation in the path: all the values for `Syllable.stress` are enumerated, and their type is inferred by looking at the number and shape of unique values (checking if there are any non-numbers, non-ints, all floating point, and whether there is a very large number of values[4]). The union of the values is accumulated across the corpus, and their distribution is used to make a paradigmatic feature description for any features present – including those added outside of the Festival process, in our example case. The features now also include the structural information in relative paths, such as the number of children in the parent.

4.3 GENERATING SYSTEM CONFIGURATION

The feature description file is generated. In CLUSTERGEN, this is the `mcep.desc` file. Features may be selected using another control file (`mcep.feats`) and this is generated automatically. The importance of each is determined in order to remove features with little impact. For this work, we built decision trees or random forests [83] with all the features present. The features were then ranked by total average information gain summed over 20 trees, and results evaluated for different cut-off values. For this comparison, the hand-enumerated structural features are retained in the baseline, but removed from the generated set so that only automatic structural features are used in the new decision trees.

The resulting feature list contains 597 items, covering both the original paradigmatic features extended to nodes by class and the structural features.

For a series of cutoff points in the ranked feature list, we build new single decision trees and calculate the Mel-Cepstral Distortion (MCD) for each held-out test utterance. The baseline system comprises 64 features, created manually, and yields a Mel-Cepstral Distortion of 4.882. The best system, with 120 features, scores at 4.895, giving a difference of 0.013. In [84], it is reported that absolute differences below 0.08 are not statistically significant for MCD. The results are shown in the table below.

4 More than 100.

Num features	MCD
Baseline 64	4.882
50	4.914
60	4.915
70	4.922
80	4.913
90	4.913
100	4.911
110	4.906
120	4.895
130	4.896
140	4.898
150	4.905
160	4.907
180	4.918
200	4.923
220	4.921
250	4.923
all 597	4.919

The lowest value produces a result with a MCD difference from the baseline that is lower than the reported significance level for MCD differences.

4.4 COMPARISON TO BASELINE

The narrow range of MCD over the cut-off levels is a testament to the power of random forests when the information is present somewhere in the features. While there are only 3 basic types of primitive structural features, applying them over all relations within even a small neighborhood produces a lot of potential features: the cross-product of paths and feature functions.

When using features derived automatically from a relation graph, the structure of the relations have an impact on how discoverable a feature may be. A feature may be attached to an existing node in a relation, or a new relation may be added. Using structural feature discovery, items along a relation path may be evaluated in terms of their previous and next elements without explicitly coding them. For example, while stress may be added at a syllable level, it is interesting

to know how far away the next stressed syllable is – which may be added as a relation itself, which will make it more discoverable and generate contextual information which encodes foot structure.

There are two main advantages to automatic feature discovery: allowing arbitrary new features from outside the otherwise insular process, and removing the need for manually changing other parts of the system to accommodate new relations or features. These advantages may be useful in rapid development for new languages, and for experimenting with feature representation in order to improve results. For example, choices may be made as to whether demisyllables are included as in [85], either by encoding information on segments, or by creating a new relation, which may be traversed and used to create contextual features.

We have not used these structures to introduce new relations, but focused so far on existing ones. Future work will involve working with new information and creating new relations, in order to evaluate which features improve the results and how to encode them in ways that favor discovery. Another application is in automatic corpus design [86], where structural features are included in a selection algorithm to make a training set balanced over a wider array of features. We also intend to explore new structural features, such as cross-relation distance, which can capture quantities such as how many segments there are to the next accented syllable in an automated way for all pairs of relations.

We have demonstrated an automatic method for discovering structural features by walking neighborhoods of a relational graph for speech synthesis, which has comparable performance to manually engineered features in mature languages in Festival. The method has general application across statistical speech synthesis systems, and reduces the feature engineering and specification burden to defining paradigmatic features and relations.

While the results do not improve over a mature hand-engineered structural feature set, they are comparable. The approach provides a new and flexible way to approach the automatic enumeration of features which removes some of the development burden for introducing new features and relationships.

4.5 ISSUES IN USING STOCK FESTIVAL HRGS

While walking the graph in patches around nodes in the Festival HRGs, it became clear that items in some of the relations were not as assumed.

The first issue we found was that many of the features we wanted to examine were dynamically created in feature functions, which were not encoded in the utterance structure but computed on-the-fly. This led to issues in expanding those features and using them, as well as having the system honor changes to those values in later processing steps.

Some relations, such as SylStructure, contained unexpected empty nodes which weakened the predictive value of adjacency. When traversing the first (depth 1) level of the relation and looking forward or backward N steps, some of those were not words but empty nodes left over from the initial graph induction of the front end.

In other cases, the order within the relations, such as Intonation and IntEvent, were not in the expected order. This reduced the effectiveness of searching adjacency elements, due to a mismatch between the temporal and structural order.

Another problem arose in the item sequences for the `Word` relation: During tokenization, hyphenated words are split, and some morphemes[5] were both split from the main portion of the word and contained no syllables in the `SylStructure` relation. There are surprising structural differences between what one would expect based on sequential ordering, and how these are encoded into the Festival HRGs.

In summary, there were issues in the structure produced in the conventional relations which hindered automatic search and discovery. These will be described more thoroughly in the next chapter.

One observation does stick out, however: the features of identity, syllable stress, and phrase breaking have a significant role in the model, both in terms of the segment or parent, and of neighboring node feature values.

5 such as the possessive morph s

5

CRITIQUE OF HRG CONVENTIONS IN FESTIVAL

Between the definition and implementation of HRGs, there are a number of issues that can lead to unexpected or poor results. In this chapter, we offer a critique of the formalism and implementation, in anticipation of reformulating them for further experiments in prosody improvement.

5.1 VISUALIZING QUIRKS

In the following sections, we convert HRGs to GraphViz format and create visualizations with `dot`. Bold arrows represent 'next' edges; dashed lines signify a 'down' edge. The redundant 'previous' and 'up' edges are not shown. The shapes of the nodes represent their primary "type," when it can be inferred[1] To aid in decoding the graphs, a legend is provided, linked by edges with no arrowheads.

5.1.1 *Tokens, Phrases and Words*

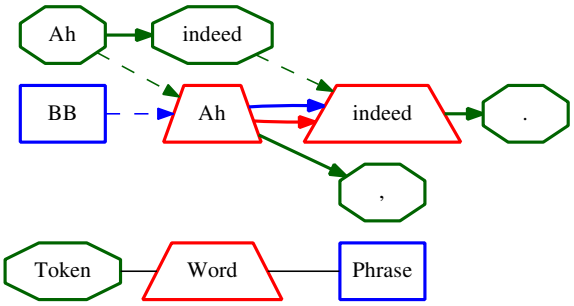

Figure 9: Token, Phrase, and Word
"Ah, indeed."

The `Phrase`, `Token`, and `Word` relations for the text "Ah, indeed" (`slt_arctic_a0329`) are shown in figure 9. There are four tokens: two tokens have corresponding words, and the other two are punctuation. There is a single phrase, that ends in a "Big Boundary" (`BB`), which

1 While nodes are not typed in Festival, it is possible to use the 1-level relations such as Phrase, Word, Syllable, and Segment to infer a node type.

points down[2] to the first word. The first word has a next pointer to the next word in the phrase. In a quirk of how Festival creates the Word relation, the punctuation tokens are children of the related words. This does not immediately follow from the process of elaborating tokens into words. Visualizing the relations of a few utterances reveals how our expectations may be violated by proceeding from reasonable assumptions such as temporal or sequential ordering.

5.1.2 *Words to Segments*

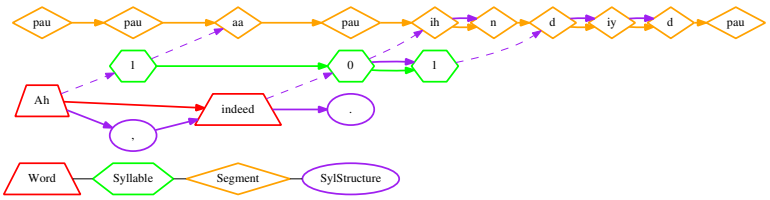

Figure 10: Word, Syllable, Segment, and SylStructure
"Ah, indeed."

In figure 10, the simple `Word`, `Syllable`, `Segment`, and the depth 3 `SylStructure` relations are shown. From this view we can see that the punctuation appears in the `Word`-level (depth 1) of `SyllStructure`, but not in `Word`. If one were to traverse the `SylStructure` assuming that depth 1 contained only `Syllables`, it would produce unexpected results.

A key feature of HRGs in general is also revealed: the relations are not a singly-rooted in a global tree. In figure 9, `Word` is dominated by `Phrase` and `Token` in a manner that defies total hierarchy. In figure 10, there are `Segments` that have no other parent: the `pau` (silence) segments have no linguistic parents. A corollary to this is that the silence (or pause) segments are not under any `Phrase`. Traversing the utterance by `Phrases` and traversing the children across relations to the `Segment` would not include `pau` units, and a `Phrase` comprises noncontiguous `Segments`. The lack of a single rooted global tree is a design feature, but it does have implications for adjacency, counting, and position of nodes.

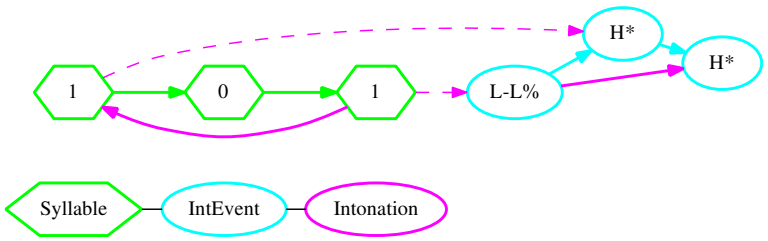

Figure 11: Syllable, Intonation, and IntEvent
"Ah, indeed."

5.1.3 *Syllables and Intonation*

In order to represent intonation in US English ToBI [31] tone labels, Festival relates `IntEvents` to `Syllables` though the `Intonation` relation. Figure 11 shows the annotations for the same utterance as before. Here, we see another unexpected sequence: the order of the `IntEvents` and Intonation is not the order in which they appear in the `Syllable` sequence. The first item in `IntEvent` is the final phrasal and endtone encoded as `L-L%` attached to the final syllable. The `next` edge points to the high pitch accent `H*`, which is attached to the first syllable. Traversing either of these may be a trap. And the `Intonation` has syllables at depth 1, with the same ordering, but with another twist: the second `H*` is the `next` after the endtone, which doesn't fit the ToBI standard annotation order.

5.2 HRG AND OTHER GRAPH FORMALISMS

The original HRG definition came about in the early 1990ś, when XML was in the cradle. There was yet not GraphML [], which now could also encode the formalism; JSON [87], YAML [88], and RDF [89] weren't invented yet. The HRG structure was a practical definition of a flat ASCII file which encoded everything needed to instantiate theories of computation about speech synthesis into a graph. We could consider any of these widely known and supported formats to be the basis for the information, and bring along with it the communities, source libraries and documentation. There is an isomorphism between them, and Festival utterances as HRGs are convenient for speech synthesis implementations.

2 Down in edge definition, even when it appears to the right visually. In keeping the graphs planar, the packing algorithm does not always put things that are edgewise "down" beneath the nodes.

5.3 EXPRESSIVE POWER VS. IMPLICIT CONSTRAINTS

An HRG multigraph has more expressive power than a single global tree by virtue of being a more general graph, but many of the issues raised so far, about ordering and constituency, are removed in a singly-rooted representation. In an XML representation as in `idlak` or `MaryTTS`, all nodes are ordered and all the children are ordered, so that it is always clear what is contained within what, and in which order. The HRG invites many different traversals, each with different results, whereas a singly-rooted representation induces a total, deterministic order in depth-first traversal from the root. In a global tree, each child has only one position in its immediate parent, but in an HRG, a node can have several different immediate parents and each parent may have a different ordering of children.

XML documents may refer to other files and elements, and define graph structures within a document using standards such as XLink [] or XPointer []. However, the approach of systems that use XML is not to use references, but rather to enforce total order and child containment.

5.4 UNCERTAINTY OF STRUCTURAL DEPENDENCY

If one thing is changed, how many relations need to be recomputed? When trying to edit any part of the utterance structure, there is a great deal of bookkeeping and maintenance of relationships. For example, adding a phrase break involves modifying not only the Phrase relation, but also the Intonation and IntEvent relations, which are distal from the the phrase: we must go from the top of the Phrase level (depth 1) to a child word, then to its final syllable, to the Intonation relation, and modify each in turn.

5.5 OPAQUE UNSERIALIZED FEATURE FUNCTIONS

Many of the features used come from feature functions, which are not injected into the HRG. While the computed features are available within Festival, they are completely opaque to any process which treats the utterance as an interchange format or API outside of the runtime system. The system was designed so that things are available at the Scheme layer, but even so they must be recomputed, possibly differently, when another phase of the synthesis occurs. There is no easy way to export the utterance structure to an outside tool and maintain all the information for analysis or modification.

5.6 REDUNDANCY OF PAIRED EDGES

The explicit encoding of paired, doubly linked next-previous and up-down edges in the relations is unnecessary[3]. It is scaffolding that may be provided by the implementation library, without putting it into the file format. While it mirrors how Festival works internally, these double edges introduce complexity in examining the files, and incur unnecessary maintenance overhead for external tools to keep it consistent. It also bloats the format.

5.7 UNNECESSARY INDIRECTION IN RELATIVE IDS

In each Relation edge row, there is a stream ID and a relative ID within the relation. The edge targets are coded in terms of the relative ID. The stream nodes are related by id through all relations, and may appear as the head of each row only once in each Relation.

The relative ID is unnecessary and incurs an unnecessary dereference, as well as bloating the format. When examining the utterance structure, it becomes tedious to move back and forth between stream ID and relative ID. The relative, within-relation IDs may be done away with entirely, and the stream IDs used throughout.

5.8 RELATION NAMES AS IDENTIFIERS

While the HRG definition paper points to allowing instances of Relations, rather than only one, in Festival it is required that a Relation appear only once. Without relation instances, there is no way to aggregate alternatives within a single utterance. For example, there is only one instance of the `SylStructure` relation, and there cannot be any alternative `SylStructures`. Furthermore, all the nodes in a relation must be connected; by contrast, there could be one syllable structure relation for each `Word` without connecting the nodes[4].

5.9 FIXED DEPTH RELATIONS

Each conventional relation in the Festival HRG has a fixed depth, with implicit types at each level. This precludes homogeneous tree structures such as parse trees.

3 These are visible in the example in figure 3.

4 Though the top levels would be connected in the `Word` relation, they are redundantly connected in `SylStructure` as well.

5.10 NO LOGICAL OPERATOR RELATION

There are no logical operation relations, as are present in And/Or graphs or ontologies. While there is no immediate use of these in Festival, they represent a common formalism in logic programming and weighted finite state automata that are missing from the HRGs in their basic form.

5.11 LIMITATIONS OF HRG FEATURES

Every feature on a node is a scalar, with no structure. Lists or dictionaries must be introduced as additional relations, even if they are logically within a node.

5.12 LACK OF EXPLICIT TYPE

Depth 1 relations stand for effective object type, and the fixed nature of each relation depends on it. Furthermore, the nodes are flat feature lists rather than hierarchic object serializations. This means that the flat structure must be marshalled onto objects through a very close and careful understanding of each feature. Within the Festival system, this is not much of a problem, because the nodes all correspond to shallow S-Expressions in the object graph. However, it prevents introspection of the utterance without a good deal of knowledge about the Festival conventions.

5.13 NO NAMESPACE OR INTROSPECTION

Related to the lack of explicit type, it not possible to start from a node and determine type without examining each of the relations in which it participates and inferring relations. A corollary is that it is not knowable which features belong to which relation, so that when a node is removed from a relation, the features are left behind. There is no namespacing over the features or relations, which is a common feature of modern mark-up formalisms.

5.14 INCONSISTENT TEMPORAL ORDERING

Within some of the conventional relations, the order of the constituents does not always reflect temporal order, or align with the order of

other relations. This is a potential source of error and confusion if a user is not deeply away of the uses and limitations of each relation. These quirks are apparent in the visualizations in figures 9, 10, and 11.

5.15 NO RELATION STRUCTURE DECLARATION

The depths and meanings of each level in a Relation is a matter of convention. There is no declaration of category for a relation – the user must either know the conventions for a Relation, or try to walk the items to infer a topology.

5.16 INCOMPLETE SET OF SIMPLE RELATIONS

While the `Word`, `Syllable`, and `Segment` levels exist as simple (depth 1) relations, they also appear in the cross-type relation SylStructure, which binds them together. However, the Phrase relation consists of a top level of phrases which do not have a corresponding simple relation, while the words on the lower (depth 2) level are all in the simple Word relation. This is also the case in the `Token` relation, which contains tokens and words, and then tokens as children of words as well.

6

KARNIVAL AND THE KRG

For this work, a set of tools and conventions was created to address the issues of Festival HRGs as they relate to automatic discovery and modification, called Karnival. Karnival is a module which interoperates with Festival, which incorporates a number of design improvements based upon the Heterogeneous Resource Graph described in Chapter 3, the lessons learning in Chapter 4, and the critique of Chapter 5.

6.1 MONKEY IN THE MIDDLE

With a few small changes to the Scheme file `festival/lib/synthesis.scm`, we create a path out of Festival and the Karnival environment. Inside Karnival, we convert to an internal format and do most everything we need to do, then pass the result back in as a well-formed HRG for complete the downstream processing.

6.2 THE KARNIVAL RESOURCE GRAPH

The Karnival Resource Graph, or KRG, is a modified form of HRG which is used in the Karnival system.

6.2.1 *Typing*

During conversion, type is inferred from membership in a simple (depth 1) relation such as `Word`. In the conventional HRG, nodes only appear in one of these. The type name is stamped into the node as a feature, `k_type`, which makes identification of nodes easier, and allows enumeration of features within type for generalization to paths.

6.3 OBJECTS AS STREAM ITEMS

Rather than using flat feature nodes, KRG embeds the data structure and serializes objects into YAML. YAML, like JSON, can encode ar-

bitrary hierarchic object structure and references within a document. The object id is synonymous with the stream id.

6.4 REMOVING INDIRECTION AND DOUBLY-LINKED LISTS

In the internal representation, the unique stream item id is used directly rather than adding a with-relation relative id. The next-prev and up-down pairs are implemented as scaffolding over the structure, so that only one of the edges is needed (and reversed as needed).

6.5 REGULARIZING RELATIONS

The KRG of the Token, Phrase and Word relations of `slt_arctic_a0329` are shown in figure 12. This is converted from the same utterance shown as a stock HRG in figure 9, with some notable differences.

- There are five relations rather than the three in the baseline utterance.
- The new relations all begin with the namespace prefix k_. This allows them to coexist with the original HRG relations.
- There are simple (depth 1) relations for k_Token, k_Phrase, and k_Word which correspond to types for their constituents.
- There are depth 2 relations for k_Token_Word and k_Phrase_Word, with a clear typed naming convention.
- Punctuation tokens have been isolated into the simple relation k_Token, rather than be the token-next of Word.
- Phrase boundary tone has been moved to the `k_Phrase`, and out the `k_IntEvent` relation which consists of pitch accents. The phrase level still contains the break strength feature as a `k_`-prefix feature as well.

Showing both together, we have figure 13. Since the new relations do not conflict with the original we can have both in the same utterance. There are two new relations here: k_Morph and k_Word_Morph. These connect the old HRG Word nodes to the KRG structure, but operate within the k_ namespace.

The Morph level is introduced to regularize words. It combines the word chunks from the Festival HRG into full words, including hyphenations. Since some of the nodes overlap, only one type determines the shape and color of each node, but the edges are all color-coded by the relation name as given in the legend below the utterance.

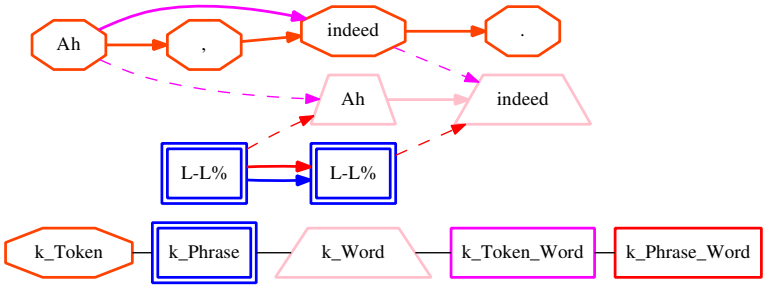

Figure 12: KRG of Token, Phrase, and Word
"Ah, indeed."

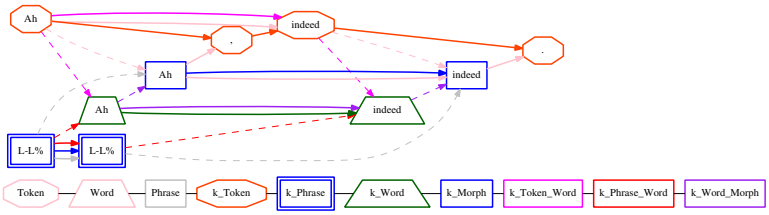

Figure 13: KRG of Token, Phrase, and Word
"Ah, indeed."

Figure 14 shows `slt_arctic_b0262`, which demonstrates this. The `'s` of `Saxon's` is an HRG Word and a KRG k_Morph. The k_Word combines the two morph chunks into a single parent word.

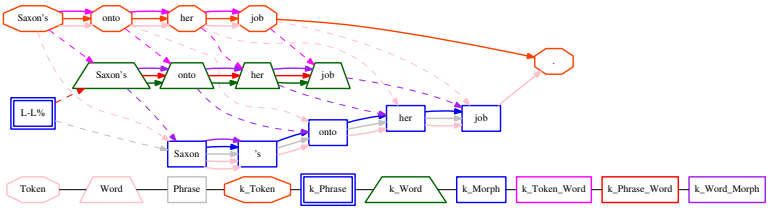

Figure 14: HRG+KRG of Token, Phrase, and Word
"Saxon's onto her job."

6.5.1 *Removing Vestigial Nodes*

Figure 10 is the sister to figure 10 as a KRG, but including the k_Token relation as well to clarify that the syllable-free words do not appear.

Another key feature is that the depth 3 relation, `SylStructure`, is factored into two two-level relations with consistent nomenclature by type at the upper and lower levels.

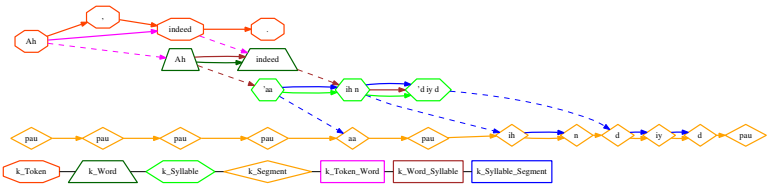

Figure 15: KRG of Word, Syllable, Segment
"Ah, indeed."

6.5.2 *Fixing Temporal Order*

We correct the unexpected ordering and constituency of the conventional relations. Figure 16 is the KRG pair of 11. The nodes in each relation correspond to temporal order, which removes some of the ambiguity as well. A similar treatment is applied to the Targets, which are not used in CLUSTERGEN.

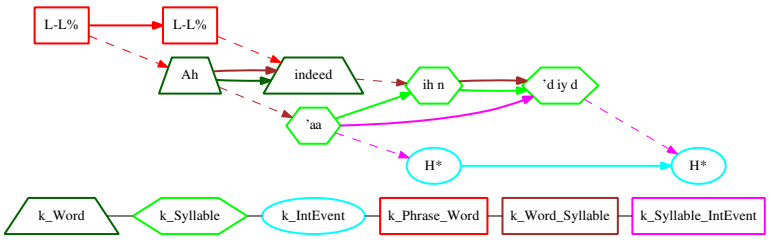

Figure 16: Syllable, Intonation, and IntEvent
"Ah, indeed."

6.6 GENERALIZING HRG RELATIONS

6.6.1 *The k_ Namespace*

In order to maintain the HRG and KRG relations in the same utterance for comparison, we introduce a name-spacing prefix k_ on all KRG relations and features. This has an added benefit that it is possible to strip all the KRG data from the whole utterance, or from one relation.

6.6.2 *Generalized Containment*

Node rows become lists of typed children, which may by of mixed type. Fixed types at depths are not necessary.

6.6.3 *Allowing Multiple Parentage*

Single parentage is no longer enforced. For practical reasons of returning to the Festival HRG, however, we remain within a singly rooted structure.

6.6.4 *Adding a Dependency Parse*

Figure 17 shows the addition of a dependency parse, which may be arbitrarily deep.

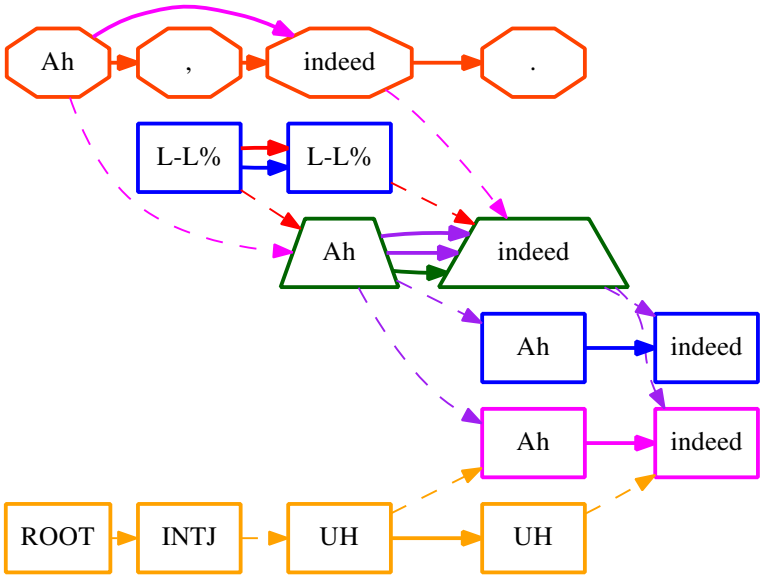

Figure 17: KRG of Word, Syllable, Segment "Ah, indeed."

6.7 RPATH: AN XPATH-LIKE QUERY LANGUAGE FOR RELATIONS

The feature path selector which is used to enumerate features for modeling can be extended analogously to XPath [90] expressions. This is convenient for examining and operating on nodes in a utterance.

In an HRG, the feature paths are used to traverse the graph and get the values relative to one node. However, XPath expressions can both be applied to and return node sets, and they may contain additional predicates. An interpreter of this sort is straightforward to implement and creates an expressive path query language over relational graphs.

For example, this feature path:

R:Segment.R:SylStructure.parent.stress

can be decoded as "for this segment, get the parent in the SylStructure, and then return the value of the stress feature".

In an xpath-like expression, we can do things like:

R:Syllable[@stress=1].R:SylStructure.children.@name

which selects segments with a parent with a stressed syllable. The at symbol is used to identify feature names in contrast to a node, by analogy to XML attributes. For awb_arctic_a0001, this expression yields "ao d ey n t r ey l f ih s t iy l z eh t s eh" for the initial utterance alignment.

Here is the sequence of syllable stresses in the same utterance. For consistency with XPath, we also allow slashes instead of dots on the path names.

```
$ hrg.py -e'R:Syllable/@stress' awb_arctic_a0001.utt

1 0 0 0 1 0 1 1 0 1 1 1 0 0
```

By this extension, we can select nodes that fit some criteria, and then operate on the node set. When we have a node at the end of the path, we can return the node or nodes that correspond. The test may also be applied to a node set. The results are a union of the nodes that evaluate to true.

We assume the path on the left starts with all nodes in the utterance. Each step in the path is then either a predicate, which reduces the input nodes set for the nodes; or a feature function, which returns the value of the feature for that node. The bracketed predicates in an expression serve as a predicate which reduces the current node set (before the bracket) without moving the node pointer to the leaf.

```
$ hrg.py -e'R:Syllable[@stress=1]/R:SylStructure/children'
      awb_arctic_a0001.utt

{'end': '0.775', 'name': 'ao'}
{'end': '1.3', 'name': 'd'}
{'end': '1.42', 'name': 'ey'}
{'end': '1.48', 'name': 'n'}
...
```

Throughout this work, these sorts of expressions are used to collect statistics and to modify utterances. Looking ahead, we can use these

selectors to find nodes to operate on, and then systematically vary them to search alternatives.

6.8 PARADIGMATIC AND SYNTAGMATIC FEATURES

In Festival synthesis, there are a number of features that are not explicitly in the HRG, but are computed using feature functions. Included in these are all the phonetic features for a given phone or segment – the place, manner, and type features that are often used. When using KRGs, we explicitly expand these into the features of the graph, so they can be inspected and changed.

```
$ hrg.py -e'R:k_Segment[@name=ao]' awb_arctic_a0001.utt

{'end': '0.775', 'name': 'ao', 'k_type': 'Segment',
 'k_end': '0.775', 'k_name': 'ao',
 'k_cplace': '0', 'k_ctype': '0', 'k_cvox': '0',
 'k_vc': '+', 'k_vfront': '3', 'k_vheight': '3',
 'k_vlng': 'l', 'k_vrnd': '+'}
```

Paradigmatic features like these are related to the features of the node type or class – they relate the the identity of what the node is, rather than the embedding or structure the node lives in. We can draw the distinction between these and features which relate to what neighbors are, or how many items are in a parent, which are syntagmatic or structurally defined.

Looking at the syllable level, we have a different kind of paradigmatic feature. This is an aggregation function, which indicates whether the parent has any children matching a feature. Here, it captures place and manner of consonantal segments, but the syllable has null onset and no coda beyond the nucleus (vowel).

```
hrg.py -e'R:k_Syllable[1]' awb_arctic_a0001.utt
{'name': 'syl', 'stress': '1', 'k_type': 'Syllable'
 'k_name': "'ao", 'k_stress': '1'}
 'k_coda_cplace_a': '-', 'k_coda_cplace_b': '-',
 'k_coda_cplace_d': '-', 'k_coda_cplace_g': '-',
 'k_coda_cplace_l': '-', 'k_coda_cplace_p': '-',
 'k_coda_cplace_v': '-',
 'k_coda_ctype_a': '-', 'k_coda_ctype_f': '-',
 'k_coda_ctype_l': '-', 'k_coda_ctype_n': '-',
 'k_coda_ctype_r': '-', 'k_coda_ctype_s': '-',
 'k_onset_cplace_a': '-', 'k_onset_cplace_b': '-',
 'k_onset_cplace_d': '-', 'k_onset_cplace_g': '-',
```

```
'k_onset_cplace_l': '-', 'k_onset_cplace_p': '-',
'k_onset_cplace_v': '-',
'k_onset_ctype_a': '-', 'k_onset_ctype_f': '-',
'k_onset_ctype_l': '-', 'k_onset_ctype_n': '-',
'k_onset_ctype_r': '-', 'k_onset_ctype_s': '-',
```

6.9 AUTOMATIC ENUMERATION OF FEATURES

6.10 SYNTAGMATIC FEATURE DISCOVERY

In the KRG conventions, each Relation is either a Simple list of typed objects, such as Phrase, Word, Syllable or Segment; or a Container list of objects such as Word_Syllable, which consists of a "top" sequence of one type (Word) and an ordered list of children (Syllable). In contrast to the conventional Festival HRG, there are no three-level relations – no SylStructure.

With these conventions, we can automatically define the size of each Container parent, and the position of each child within the parent from both the left and right boundaries. This gives an implicit structural feature, which we make explicit by iterating over the Container relations and writing the structural features for each node.

With these structural features in place, the same query as above includes the syntagmatic features. Here, the segment is participating in two parent relations, Syllable_Segment and ONC_Segment. _L and _R represent counting from the Left and Right margin of the parent container, respectively.

```
'k_in_Syllable_Segment_L': '1',
'k_in_Syllable_Segment_R': '1',
'k_in_ONC_Segment_L': '1',
'k_in_ONC_Segment_R': '1',
```

Syllable_Segment captures the segmental position in the syllable. ONC stands for Onset/Nucleus/Coda, which is an additional structural level introduced corresponding to conventional index functions. There is also a Syllable_ONC relation which maps each syllable to a sequence of onset, nucleus and coda slots. This way we can automatically derive both position in Syllable and position inside the subsyllabic structure.

Looking at the parent Syllable, we see the corresponding syntagmatic features. However, this time the Syllable is both a parent (in the Syllable_Segment and Syllable_ONC) – showing num_ChildType –

and a child (Word_Syllable, Phrase_Syllable, Utterance_Syllable, Foot_Syllable), appearing as _in_ParentType_ChildType.

```
'k_num_ONC': '3',
'k_num_Segment': '1',
'k_in_Word_Syllable_L': '1',
'k_in_Word_Syllable_R': '2',
'k_in_Phrase_Syllable_L': '1',
'k_in_Phrase_Syllable_R': '7',
'k_in_Utterance_Syllable_L': '1',
'k_in_Utterance_Syllable_R': '14',
'k_in_Foot_Syllable_L': '1',
'k_in_Foot_Syllable_R': '4'
```

Here we see a pair of tree structures where one cannot contain the other, and so a strict tree structure would have different implications: the Foot_Syllable and Word_Syllable relations. Neither one can dominate the other, and the relations coexist. This contrasts with the Syllable_ONC and Syllable_Segment hierarchies, where we put a strict hierarchy on them (Syllable/ONC/Segment), but having both representations available makes the index more direct.

The structural indexing is injected into the nodes in the utterance for all Container type objects, and so comes automatically from creating the structure itself. In some cases a relation is introduced in order to create a metric space for containment and make features more proximal to the nodes which depend on them.

6.11 THE PROSODIC SIGNATURE

A *signature* in this context is a traverse of the utterance graph, with the features of interest linearized into a string. For this work, the features include the segmental material, syllabification and stress, word and phrase boundaries.

For the arctic_b0228, the prompt "You were engaged." has a default prosodic signature with three words, four syllables, and one phrase. Each syllable is one character long, being either S or w (1 or 0, strong or weak). The upper and lower case contrast redundantly codes the stress value. Word boundaries constrain where a phrase break can occur – no phrase boundary can occur within a word. In order to keep the strings of comparable length irrespective of phrasing changes, each word boundary is marked with a single character depicting the break status. A colon (:) indicates no break; slash (/), a minor break; and a major break is characterized with a closing bracket (]).

```
    S:w:wS]
```

Table 18 shows some more examples of prosodic signatures in the AWB arctic data.

```
awb_arctic_a0001      "AUTHOR OF THE DANGER TRAIL, PHILIP
                                              STEELS, ETC"
                      Sw:S:w:Sw:S|Sw:S:SSww]

awb_arctic_a0002      "Not at this particular case, Tom,
                                 apologized Whittemore."
                      S:S:S:wSww:S|S:wSwS:Sw]

awb_arctic_a0003      "For the twentieth time that evening
                               the two men shook hands."
                      S:w:Sww:S:S:Sw|w:S:S:S:S]

awb_arctic_a0004      "Lord, but I'm glad to see you again,
                                                 Phil."
                      S|S:S:S:w:S:S:wS:S]
```

Figure 18: Example Initial Prosodic Signatures

If we expect that utterances to be compared differ in segmental material, the segmental material can also be encoded, so that the signatures can be directly compared to determine whether two utterances differ. Here, a syllable is prefixed with a stress value, and a dash (-) is introduced as a syllable separator. This gives the segmental signature, which also contains the prosodic structure. The equals sign (=) is a special boundary after pauses.

```
    pau=1yuw:0wer:0ehn-1geyjhd]pau=
```

Finally, we can make this more readable by stripping the initial and final pauses, spreading it out with some white space, and using UPPER and lower case to indicate stress condition.

```
    YUW : wer : ehn-GEYJHD ]
```

Signatures can be used to show or check differences between utterances, and also may act as an edit over an existing utterance. The new signature can be applied to the existing structure, modifying it. These are a major feature of the resynthesis approach outlined here.

7

ITERATIVELY IMPROVING PROSODY

The Arctic voice databases [9] are typical of many speech synthesis systems, in that each talker was given isolated sentences of text as prompts for the utterance recordings, with no mark-up or context. The delivery is left to the performer, and is generally in a relatively neutral style – no large prosodic movements. This works well for avoiding unexpected excursions, but leads to an overall neutral delivery in the resulting voice. Large excursions, without causal annotations in the features of the data, can degrade the models overall quality. Even so, any sentence in a stress-timed language may be produced in a number of different ways, by rearranging the strong and weak patterns or inserting breaks, even without producing a high degree of emphasis.

Revisiting the signature for `arctic_b0228`, the prompt is "You were engaged.". Even this short sentence can have several different deliveries without extreme prosody; some examples are show in table 5. `S` signifies a strong syllable, `w` weak; colon signifies word boundary, vertical bar minor break, and bracket major break.

We use two levels of stress, rather than trichotomizing as [41] suggests, and three levels of break rather than the 4 levels of ToBI.

Pros. Sig.	Segmental Signature	Delivery
`S:w:wS]`	`YUW : wer : ehn-GEYJHD ]`	YOU were en-GAGED.
`w:S:ww]`	`yuw : WER : ehn-geyjhd ]`	You WERE en-gaged.
`w:w:wS]`	`yuw : wer : ehn-GEYJHD ]`	You were en-GAGED.

Table 5: Example Alternative Signatures of "You were engaged."

When the talker delivers the prompt, they give one such pattern – but the automatic labeling and alignment process may not have the same signature. In Festival, each prompt is annotated with an utterance structure made largely from dead reckoning: it makes its best guess and runs with it without feedback.

The one exception to this in the Festival build process is break insertion around pauses. During the phonetic alignment process (`ehmm`), short silences may be introduced, which may be promoted to phrase

breaks if they are over a specified length when utterance files are build (with `build_utts`).

In order to overcome this, we introduce a method of incrementally modifying the annotations of the training data using Expectation Maximization over synthetic perturbations of the utterance in order to improve the duration model. The same method can be applied to regularizing different kinds of features, such as pronunciation differences, stress and phrasing.

7.1 TWIDDLING

In order to evaluate whether a change in annotation will be useful, we "twiddle" each utterance to produce a set of variations over the signature. The twiddle may be over any level of representation in an HRG or KRG – segmental content, syllable structure or features, phrase break and type, or anything else we want to evaluate.

The name of each twiddle is set to the original file identifier with an edit signature appended to it. The edit signature contains a short, lossless description of the changes applied to the original. When two twiddle names are the same, the annotations are identical. Likewise, if the signatures are identical, so are the annotations.

The space of possible annotations for each utterance is quite large, even with the limited number of alternatives for each item. If we twiddle stress between 0 and 1 for n syllables, there are 2^n possible labellings; with m words and 3 possible break conditions, there are 3^{m-1} variations if we assume the final boundary is always a break. If both are varied, this is $2^n \times 3^{m-1}$ possible combinations.

For a 7-syllable, 5-word sentence like "Will we ever forget it", there are 10,368 possible annotations. This number makes it intractable to evaluate every possible alternative. Rather than try them all by brute force, we instead make a smaller set of single "moves" which mutate the original signature, and allow only one move per twiddle.

Each twiddle, including an unmodified copy of the original, is resynthesized, compared to the original, and ranked by an error metric. The twiddle with lowest error when compared with the recordings or state level alignments is considered the best twiddle. For each utterance, the best twiddle will become the annotation for the next build iteration. If the best twiddle is the copy of the previous, the annotation is unchanged.

Table 6 shows some example twiddles for `awb_arctic_a0005`. The meanings of the edit signature will be discussed under the sections

Twiddle	Signature
a0005	`WIHL : WIY : EH-ver : fer-GEHT : IHT ]`
a0005_w01s010	`wihl : WIY : EH-ver : fer-GEHT : IHT ]`
a0005_w01s010p1	`wihl / WIY : EH-ver : fer-GEHT : IHT ]`
a0005_w01s011p1	`WIHL / WIY : EH-ver : fer-GEHT : IHT ]`
a0005_w02s010	`WIHL : wiy : EH-ver : fer-GEHT : IHT ]`
a0005_w02s010p1	`WIHL : wiy / EH-ver : fer-GEHT : IHT ]`
a0005_w02s011p1	`WIHL : WIY / EH-ver : fer-GEHT : IHT ]`
a0005_w03s010	`WIHL : WIY : eh-ver : fer-GEHT : IHT ]`
a0005_w03s020p1	`WIHL : WIY : EH-ver / fer-GEHT : IHT ]`
a0005_w03s021	`WIHL : WIY : EH-VER : fer-GEHT : IHT ]`
a0005_w03s021p1	`WIHL : WIY : EH-VER / fer-GEHT : IHT ]`
a0005_w03x021	`WIHL : WIY : eh-VER : fer-GEHT : IHT ]`
a0005_w04s011	`WIHL : WIY : EH-ver : FER-GEHT : IHT ]`
a0005_w04s020	`WIHL : WIY : EH-ver : fer-geht : IHT ]`
a0005_w04s020p1	`WIHL : WIY : EH-ver : fer-geht / IHT ]`
a0005_w04s021p1	`WIHL : WIY : EH-ver : fer-GEHT / IHT ]`
a0005_w04x020	`WIHL : WIY : EH-ver : FER-geht : IHT ]`
a0005_w05s010	`WIHL : WIY : EH-ver : fer-GEHT : iht ]`

Table 6: Example Prosodic Twiddle Signatures for "Will we ever forget it."

pertaining to each type of twiddle, along with a rationale for each allowable move.

7.2 CREATING A BASELINE

Because we will be working with durations, we want to have a good starting point for measuring them. Getting good phonetic alignments is a problem unto itself, but we can establish a baseline carefully with the existing tools and small modifications.

We use mixed excitation CLUSTERGEN [21] voices, because they offer the best overall voice quality in parametric synthesis in FestVox. The dependencies for voice builds in FestVox are shown in figure 19. All voices perform the steps in the figure for the unit selection voice (clunits); CLUSTERGEN also involves some additional steps for acoustic parameters, and the mixed excitation build deviates from the default build slightly. The main packages used are FestVox [5] [6], Festival [7], Edinburgh Speech Tools (EST) [24], and Speech Processing ToolKit (SPTK) [91].

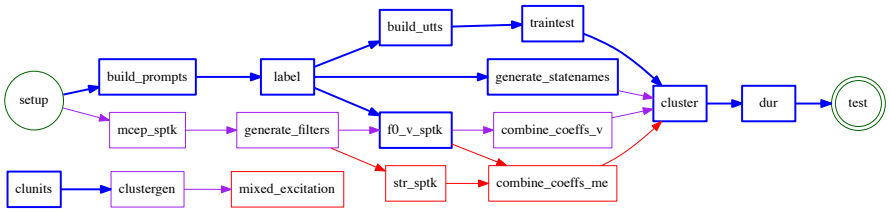

Figure 19: Build Dependencies for CLUSTERGEN Mixed Excitation Voices in FestVox

The voice is built initially using the FestVox process, with a stock distribution and features, and short silence insertion with phrase break promotion. These short silences tend to be overgenerated, and sometimes are confusable with closure events. Very short silences are often false positives in the initial segmental alignment system (EHMM [92]), so we remove all silences shorter than 80ms, before building a mixed excitation parametric synthesis voice. The voice is built normally from there.

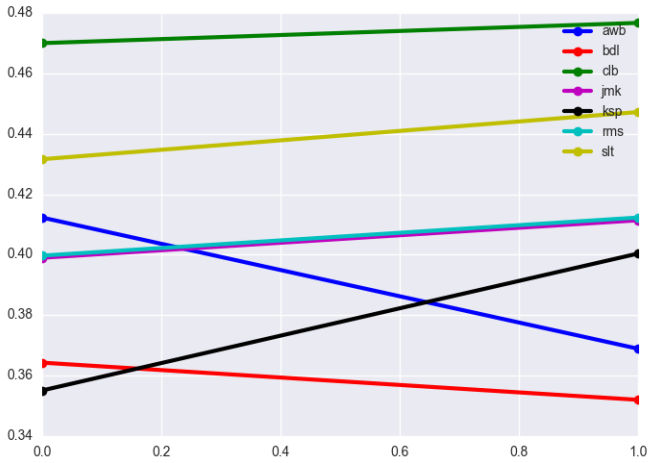

Figure 20: Correlation Changes for Baseline after Initial Resynthesis

For the initial alignments (build "zero"), we pool the data and build a voice from all of Arctic combined in order to get the best initial phonetic alignments that are consistent across the talkers. These are then split into individual voices (build 1).

In the build process, there is a mismatch between the initial prompt alignment and the utterance as produced by the run-time synthesizer. This comes in part from the method of short silence deletion, which simply deletes a boundary and absorbs the silence into another seg-

0	Combined Data Build (for alignment)
1	Split Standard FestVox HRG Build
2	Collapse Silences and Remove Short (< 80ms)
3	Build as Karnival KRG

Table 7: Steps to Create a Baseline

ment; there are also other differences that come from the utterance building process, and how pauses are created. To overcome this mismatch, we re-synthesize the prompts with the newly built voice and use the results as the new prompts, but without allowing short pause insertion during alignment. We systematically built voices with differing cutoff rates for the short silences to settle on the 80ms range.

Some of the remaining pauses shrink during the EM, and we then allow phrase insertion for any silence over 20 ms (after we have already deleted anything under 80ms in the first build, removing the closure confusion). The voice is then converted to the Karnival KRG format, and the baseline voice is built. The result is improved state-level alignments and initial pauses (or silences). For five of the seven Arctic voices, the correlation improves as shown in (figure 20). The stock system is on the left (at 0), and the new baseline system is on the right (at 1.0). For the talkers `awb` and `bdl`, it does not, though we still believe this is an improved state-level alignment. Silences and pauses are not included in the error metric.

7.3 REDUCTION AND PRONUNCIATION VARIATION

Many function words have more than one realization for a given talker. Here we consider two components: removal of lexical stress, and phonetic reduction.

The front-end (FE) of the synthesizer generates on form or another in various contexts. In the default US English Festival front-end, there is a model which assigns pronunciations for words, and it reduces some function words.

However, the talker often performs the utterances differently from the output of the front end. If the results are used directly as an utterance structure for aligning and annotating the training data, there may be a large number of mismatches. While these may be hand-corrected, we use analysis by synthesis to correct for the variation.

Table 8 compares the pronunciations of one the Arctic talkers, SLT, with the output of the default US English front end. There are on aver-

age more than two occurrences of words in this set in each utterance of the data.

Incorrect stress marking makes the stress feature less powerful for modeling, and when there are many that are different from the talker's delivery, the effect becomes more pronounced. Several function words are given 1 stress by the front end, which are never stressed by SLT: an, and, are, as, at, be, but, can, had, have, it's, its, of, or, should, some, the, to, were, will, with, would, and one form of the word "a". In many other cases, a 1-stress appears in relatively few cases compared with a 0-stress: am, could, for, has, in, is, on, up, was, what.

Phonetic reduction occurs in some cases where there is no (or "zero") stress: ae and ah will often reduce to ax, and iy may reduced to ih. It is fair to point out here that ix is not used in this phone set, otherwise we would see it in comparable contexts.

7.3.1 *Twiddling Phonetic Variation in Reducing Words*

To help overcome the discrepancy between the default output of the front-end and a particular performance, we will start by improving the annotation of function words which are often reduced. Once we have the re-annotated content, we can modify the pronunciations the front-end produces to better match the annotated training data for each voice.

First, we change the syllabification and segments of the reducible words to remove stress and to use unreduced segments (no ax). Without a distribution of the actual pronunciations, this places the model mass of these instances into their full-form vowels as a starting point. Once this is build (build 4), we twiddle the pronunciations of the reducible words to find which segmental sequences were given.

In principle, the approach is similar to getting better initial alignments by adding pronunciation variants to a speech decoder [76], but there are some differences. For one thing, we are using the synthesizer rather than the recognizer in the process. Beyond that, though, we can also manipulate the stress annotation which may be absent from recognizer output, and we can define general reduction processes which can generate pronunciation as an alternative to preset lists or pronunciations.

In the current implementation, we need to re-do the EHMM alignment step whenever the segment sequence changes, and recalculate acoustic parameters that depend on segmental identity, such as the mixed excitation strength coefficients. However, if we get all the seg-

word	count	pron from FE	full form	-stress	ax/er	alt
a	2	1 ey	1 ey	0 ey		
a	213	0 ax		0 ah	0 ax	
am	6	1 ae m	1 ae m	0 ae m	0 ax m	
an	35	1 ae n		0 ae n	0 ax n	
and	258	1 ae n d		0 ae n d	0 ax n d	
are	37	1 aa r		0 aa r	0 er	
as	37	1 ae z		0 ih z	0 ax z	
at	51	1 ae t		0 ae t	0 ax t	
be	29	1 b iy		0 b iy		
because	1	0 b ih . 1 k ao z	0 b iy . 1 k ao z			
before	13	0 b iy . 1 f ao r	0 b iy . 1 f ao r	0 b ih . 1 f ao r		
between	7	0 b ih . 1 t w iy n	0 b iy . 1 t w iy n	0 b ih . 1 t w iy n		
but	47	1 b ah t		0 b ah t	0 b ax t	
can	11	1 k ae n		0 k ae n	0 k ax n	
could	19	1 k uh d	1 k uh d	0 k uh d	0 k ax d	
every	5	1 eh . 0 v er . 0 iy	1 eh v . r iy			
for	67	1 f ao r	1 f ao r	0 f ao r	0 f er	
from	37	1 f r ah m			0 f r ax m	
had	93	1 hh ae d		0 h ae d	0 h ax d	
has	9	1 hh ae z	1 h ae z	0 h ae z	0 h ax z	
have	36	1 hh ae v		0 h ae v	0 h ax v	
in	164	0 ih n	1 ih n	0 ih n		
into	20	0 ih n . 1 t uw	1 ih n . 0 t uw		0 ih n . 0 t ax	1 ih n . 0 t ax
is	68	1 ih z	1 ih z	0 ih z		
it's	8	1 ih t s		0 ih t s		
its	16	1 ih t s		0 ih t s		
must	7	1 m ah s t	1 m ah s t	0 m ah s t		
neither	1	1 n iy . 0 dh er	1 n ay . 0 dh er			
of	215	1 ah v		0 ah v (?)	0 ax v	
on	45	1 aa n	1 aa n	0 aa n		
or	19	1 ao r		0 oa r	0 er	
our	18	1 aw . 0 er	1 aw . er	0 aa r		
should	7	1 sh uh d			0 sh ax d	
some	12	1 s ah m		0 s ah m	0 s ax m	
that	82	1 dh ae t	1 dh ae t	0 dh ae t	0 dh ax t	
the	442	0 dh ax		0 dh iy	0 dh ax	
this	50	1 dh ih s	1 dh ih s	0 dh ih s		
to	178	0 t ax		0 t uw	0 t ax	
up	17	1 ah p	1 ah p	0 ah p		
was	225	1 w aa z	1 w aa z	0 w aa z	0 w ax z	
were	58	0 w er			0 w er	
what	25	1 w ah t	1 w ah t	0 w ah t	0 w ax t	
will	15	1 w ih l		0 w ih l		
with	93	1 w ih dh		0 w ih th		
would	21	1 w uh d		0 w uh d		
	2819					

Table 8: Function Word Pronunciations in SLT Arctic

mental changes out of the way, we can work in the prosodic twiddle space without realigning. This seems reasonable – that the prosodic differences have less fine-grained acoustic impact than they do on duration, intensity, and fundamental frequency. Furthermore, the duration of each segment is dependent on its phonemic name.

When the segmental material or syllable structure changes, we can no longer compare segment features pairwise, which precludes the use of segmental duration or error as metric. It is possible to go "up" a level and measure the durations of the next higher stable level of structure, such as the syllable or word, but this sacrifices precision in relative timing. In early tests, state-level durations were more sensitive to syllable stress and phrase boundary than aggregate durations.

Rather than using duration as the metric, we can use measures which compare the entire acoustics. The measure used here is to DTW align the synthetic waveform to the original, and calculate the summed mel-scale cepstral distortion over the best alignment path. In order to account for duration differences, we use the summed absolute error rather than normalizing by the path length. This has the added benefit of making the measures comparable in magnitude between utterances.

7.3.2 *Twiddling Word Sense and Alternate Pronunciations*

Analysis by synthesis can be used for overcoming other pronunciation errors as well: when there are more than one acceptable pronunciation of a word, and to correct parsing errors where different word senses can have different pronunciations.

For an example of variant pronunciations, the word 'neither' may be pronounced as '1 n iy . 0 dh er' or '1 n ay . 0 dh er' by different talkers, or even the same talker in different utterances.

Word sense differences may be realized as lexical stress differences, phonetic differences, or both. Table 9 shows the counts where the predicted pronunciation differed from the final annotated by talker in Arctic. The counts were taken after iteratively adjusting the pronunciations, which will be described in the next section.

There are very few of them overall. Of the 221 words with differing pronunciations by word sense in the CMU dictionary[1], 26 appear in the Arctic prompts, and of these, only 6 words (aged, articulate, close, fragments, object, use) are produced as errors. There are about 10,000 words in each of the Arctic talker databases.

However, it should be recognized that the same method used to find model-based reduction differences can be extended to other differences, including pronunciation and stress variation, in a unified manner.

7.3.3 *Iteratively Refining Pronunciations*

The two sets of words, reducible and word sense variants, are combined into a list of words that differ in segment sequence. Stress is left as 0 on the reducible words with full-form vowels, and segmentally contrastive word sense or pronunciation variants are included.

When we twiddle a pronunciation, the allowed moves are the pronunciations of each word[2]. Each twiddle contains a pronunciation change in exactly one word. For combination moves, where more

1 CMU Dictionary version 0.4 used in Festival US English. While this is a relatively old version at the time of this writing, there is no more current one that is publicly available for Festival.

2 For each word that has alternative pronunciations, a pronunciation change is considered one move.

word	instances	forms in corpus	awb	bdl	clb	jmk	ksp	rms	slt	
aged	1	2	0	1	1	1	1	0	0	4
ally	2	1	0	0	0	0	0	0	0	0
articulate	2	2	0	1	1	1	0	1	1	5
close	2	2	1	1	1	1	1	1	1	7
compound	2	1	0	0	0	0	0	0	0	0
conduct	2	1	0	0	0	0	0	0	0	0
defect	1	2	0	0	0	0	0	0	0	0
desert	1	2	0	0	0	0	0	0	0	0
does	4	1	0	0	0	0	0	0	0	0
excuse	2	2	0	0	0	0	0	0	0	0
fragment	1	2	0	0	0	0	0	0	0	0
fragments	1	2	1	1	1	1	1	1	1	7
house	2	1	0	0	0	0	0	0	0	0
import	1	2	0	0	0	0	0	0	0	0
insult	1	1	0	0	0	0	0	0	0	0
live	1	1	0	0	0	0	0	0	0	0
minute	5	1	0	0	0	0	0	0	0	0
mouth	3	1	0	0	0	0	0	0	0	0
object	1	1	1	1	1	1	1	1	1	7
produce	1	1	0	0	0	0	0	0	0	0
produces	1	1	0	0	0	0	0	0	0	0
progress	1	1	0	0	0	0	0	0	0	0
subject	1	1	0	0	0	0	0	0	0	0
tear	1	1	0	0	0	0	0	0	0	0
use	2	2	1	1	1	1	1	1	1	7
wind	1	1	0	0	0	0	0	0	0	0
	43	36	4	6	6	6	5	5	5	37

Table 9: Word Sense Errors in Arctic by Talker

Label	Signature
a0317	HHIY : waaz : ah : WAYZ : hhay-IY-nax]
a0317_w02v2	HHIY : waxz : ah : WAYZ : hhay-IY-nax]
a0317_w03v1	HHIY : waaz : ey : WAYZ : hhay-IY-nax]
a0317_w03v3	HHIY : waaz : ax : WAYZ : hhay-IY-nax]

Table 10: Example Edit Signatures for Three Iterations of Pronunciation Twiddling

than one thing can change within a twiddle instance[3], the cross-product of variations are generated.

The computational complexity is linear in the number of words, as each independently twiddled word is considered as a move candidate. Operating on one item in the sequence for evaluation avoids exploring the entire space, which would otherwise be represented by the product of the possible variations.

The edit signature which is appended to the original symbol name consists of a word index and a value index. Table 10 shows an example for `slt_arctic_a0307`, "He was a wise hyena", with phonetic changes for two words ("was" and "a"). The original has no appended edit signature.

We perform 10 iterations of pronunciation twiddling to refine the annotations. In an iteration, each utterance is twiddled with respect to the enumerated pronunciations, and the best is selected. The aggregate of utterances then becomes the new prompts for alignment by the EHMM process, and the voice is built again. In the next iteration, the models reflect the impact of the annotation changes.

The changes in the objective function score by iteration is shown in figure 21, and the proportion of utterances changed over the process is shown in figure 22. Both of these go in the right direction or, at worst, do not go much in the wrong direction.

However, for the metrics that Festival uses itself, the results are not as good – because the objective function differs, and because the models change during every iteration. The overall duration model correlation and RMS error in terms of the internal metrics are shown in figure 23 and figure 24.

It appears that the process at least causes no harm. The range of the error changes is narrow.

3 The final syllable of a word may change in stress value, and the break value at the end of the word may change within a single twiddle. This does not happen when performing pronunciation improvement, but appears later in iteratively improving stress and break annotations.

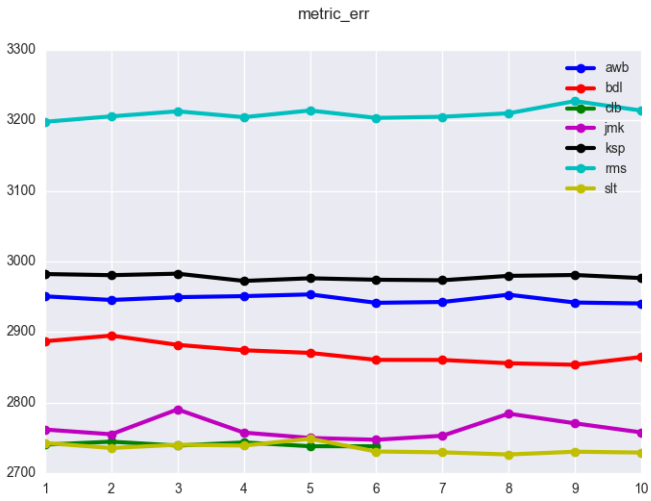

Figure 21: Metric Error for 10 Iterations of mcep-Based EM Perturbation of Function Word Pronunciation

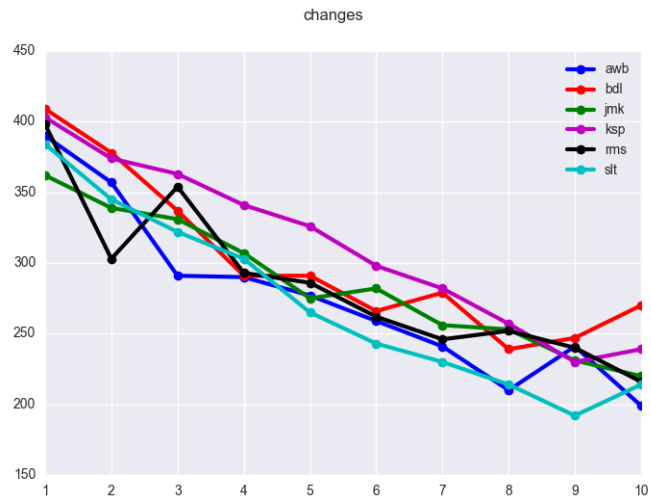

Figure 22: Number of Changes over 10 Iterations of mcep-Based EM Perturbation of Function Word Pronunciation

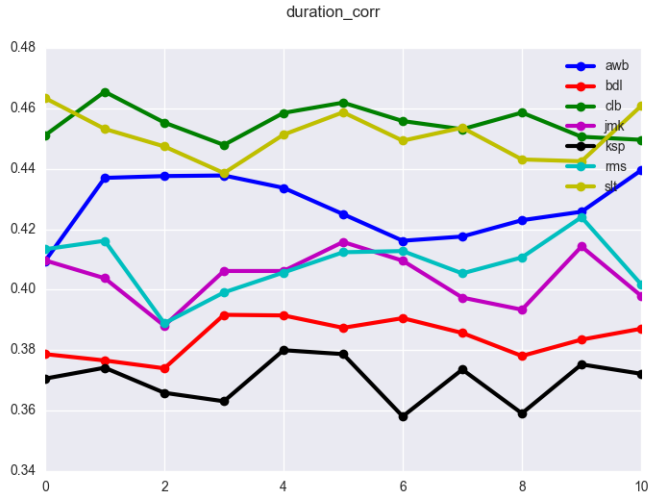

Figure 23: Duration Correlation for 10 Iterations of mcep-Based EM Perturbation of Function Word Pronunciation

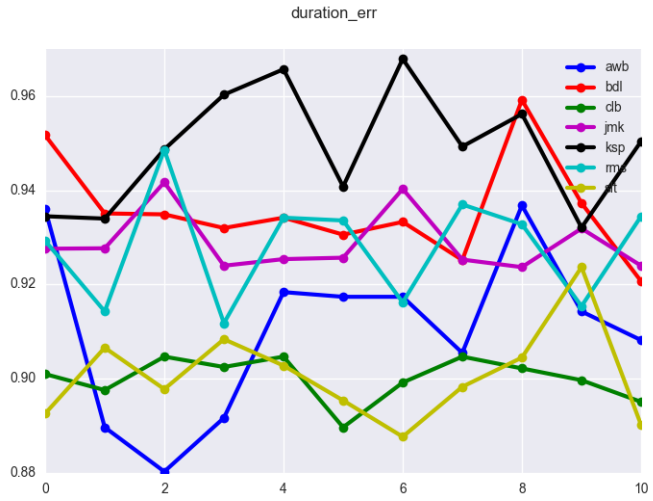

Figure 24: Duration Error for 10 Iterations of mcep-Based EM Perturbation of Function Word Pronunciation

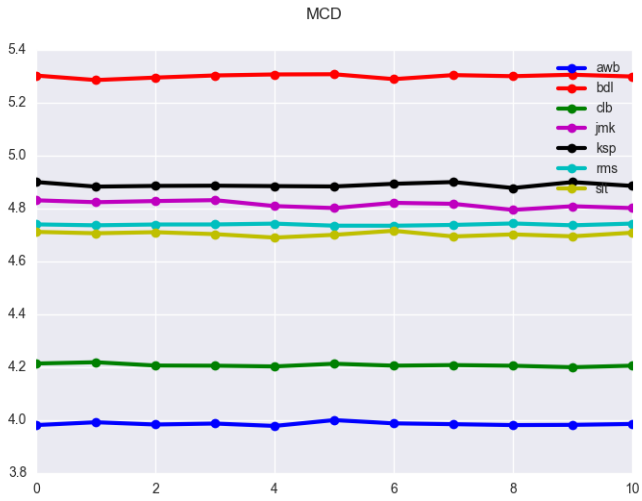

Figure 25: Mel-Cepstral Distortion for 10 Iterations of mcep-Based EM Perturbation of Function Word Pronunciation

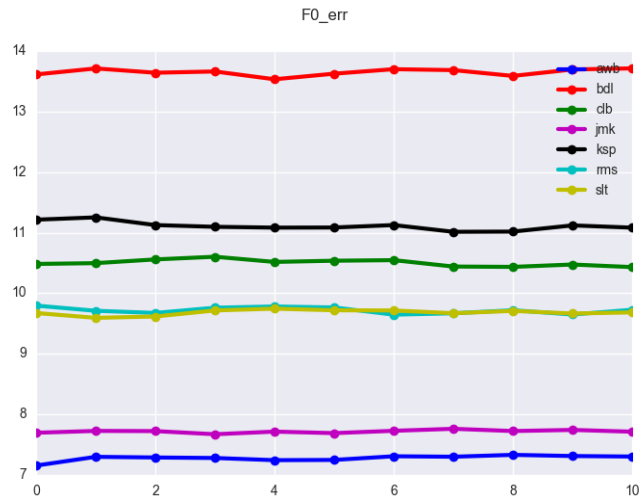

Figure 26: F_0 Error for 10 Iterations of mcep-Based EM Perturbation of Function Word Pronunciation

In figure 27, we can see the changes made to one utterance of the AWB voice over the ten iterations. The word "But" loses its stress and changes /ah/ to the phonetically reduced /ax/; a similar effect occurs on "am" and "at". For "of", /ah/ becomes /ax/. And for the word "resources", the stress is moved due to a word sense or pronunciation variant.

```
( awb_arctic_a0415 "But I am at the end of my resources." )
BAHT : AY : AEM : AET : dhax : EHND : AHV : MAY : RIY-saor-sihz ]
baxt : AY : aem : aet : dhiy : EHND : ahv : MAY : riy-SAOR-sihz ]
baxt : AY : aem : axt : dhiy : EHND : ahv : MAY : riy-SAOR-sihz ]
baht : AY : aem : axt : dhiy : EHND : ahv : MAY : riy-SAOR-sihz ]
baht : AY : aem : axt : dhiy : EHND : ahv : MAY : RIY-saor-sihz ]
baht : AY : aem : axt : dhiy : EHND : axv : MAY : RIY-saor-sihz ]
baht : AY : aem : aet : dhiy : EHND : axv : MAY : RIY-saor-sihz ]
baht : AY : aem : axt : dhiy : EHND : axv : MAY : RIY-saor-sihz ]
baht : AY : axm : axt : dhiy : EHND : axv : MAY : RIY-saor-sihz ]
baxt : AY : axm : axt : dhiy : EHND : axv : MAY : RIY-saor-sihz ]

baxt : AY : axm : axt : dhiy : EHND : axv : MAY : RIY-saor-sihz ]
```

Figure 27: Function Word Pronunciation Twiddle Changes over 10 Iterations for `awb_arctic_a0415`. Colored changes are textual differences in the strings and may not phonemes, as in /ax/ vs /ah/.

7.3.4 *Improving Reduction Prediction from Twiddled Pronunciations*

The procedure ran for ten iterations, we count how many times each pronunciation appears and make a set of default pronunciations by voice using the most likely pronunciation. This could be improved by building word-level decision trees, one for each word [76], but this first approximation is already an improvement over the stock front-end.

A new default reduction pronunciation set is then produced by voting across the seven arctic voices, which is used in the absence of a voice-specific list.

7.4 TWIDDLING PROSODY

7.4.1 *Exploring the Space of Signatures*

The initial signature is produced given the contents of baseline utterances, by walking the KRG relations using feature paths and functions.

If we assume that some parts of an utterance are fixed (immutable), then the signature can be used to compare two renditions of an utterance, and string comparisons can illuminate the differences.

```
YUW : wer : ehn-GEYJHD ]
yuw : wer : ehn-GEYJHD ]
YUW / wer : ehn-GEYJHD ]
yuw : WER / ehn-geyjhd ]
```

The last signature in the example above shows a "phrase" which has no primary stress. The signature does not impose constraints on the structure; it reflects the annotations in the utterance structure. While this may be ill-formed with respect to a prescribed definition of what is possible in a phrase, the signature is agnostic about it. We experimented with adding constraints in processing which disallow variations with model violations, but found that it was best to impose these after searching the space, as sometimes annotations will assume an "invalid" configuration while iterating towards a better marking.

For Arctic `awb`, there are around 1138 prompts in the baseline, with 10,010 words and 14,156 syllables. Without introducing constraints, the size of the space of possible prosodic signatures is $2^{10010} \times 3^{14156}$ – each syllable may be strong or weak, and each word may have 3 boundary types. This number is incomprehensibly large. It would take far too long to evaluate every possible configuration for the whole database.

To constrain the search, we assume that small, incremental changes may be made to the signature which result in a lower error score than the original, in EM style. We assume that there is enough data and accuracy in the initial set of signatures that will induce some regularity in the prosodic models for the features we are looking at, which may not be the case.

Given an initial signature for an utterance, we can perturb the signature with small changes and resynthesize the utterance. This opportunistic, gradient approach greatly reduces the search space: rather than $2^{|\mathrm{Syllables}|} \times 3^{|\mathrm{Words}|}$, we have something that is effectively linear in the number of words and/or syllables at each iteration of EM.

We can place some constraints on the signatures, to help convergence and to avoid wandering or using symbols in unexpected ways.

- Every phrase must have at least one strong syllable in it.
- Phrase boundaries followed by silence may not be completely deleted.
- The final boundary remains a major break boundary.
- Words and Syllables may not change number or constituency (signature is the same length).
- No phonetically reduced vowel pronunciation such as /ax/ may receive a 1-stress.

A move may involve flipping the strength of a syllable or changing the word boundary status. However, we found that leaving these as independent moves, where the system may take one or the other, led to a local maximum that made it hard to move over. Instead, the possible moves are:

- Any non-word-final syllable may be flipped ($2 \times |\texttt{Syllable}|$)
- Word-final syllable moves are all possible stress $\times$ boundary (2×3)

In addition to these basic moves, we added the ability to move stresses and phrases. While the general case would allow moving more than one unit away, we restrict these to a single swap and rely on the insertion and deletion moves to account for more distal swaps.

- Adjacent 0 / 1 or 1 / 0 stresses may be swapped
- Adjacent nonbreak / break or break / nonbreak may be swapped

In order to keep the complexity down, we perform alternating mimimization over a sequence of conditions. In some configurations, only the function word stress is considered; or only stress in content words; or in both; or in both with boundary. It is also optional whether or not to use validation constraints for each step, or apply them at the end, allowing for ill-formed phrases in the intermediate search. For this work, we performed function word pronunciation improvement first, then improve the stress values of the function words. Finally, we improve stress and phrase breaking jointly over the corpus.

The edit signature for prosodic twiddles consists initially of a word index which is being affected. The next symbol is either a P, s or x, indicating whether it is a bare phrase move (P) or involves a syllable move (s or x). If it is a P, it is a swap move, and it is followed by a two-digit phrase index and then a single digit encoding 0 or 1 for

Label	Signature
a0317	w|w:w:w:SSw]
a0317_w01P010px	w]w:w:w:SSw]
a0317_w01P011px	S]w:w:w:SSw]
a0317_w01s010p0	w:w:w:w:SSw]
a0317_w01s011	S|w:w:w:SSw]
a0317_w01s011p0	S:w:w:w:SSw]
a0317_w02s010p1	w|w|w:w:SSw]
a0317_w02s011	w|S:w:w:SSw]
a0317_w02s011p1	w|S|w:w:SSw]
a0317_w03s010p1	w|w:w|w:SSw]
a0317_w03s011	w|w:S:w:SSw]
a0317_w03s011p1	w|w:S|w:SSw]
a0317_w04s010p1	w|w:w:w|SSw]
a0317_w04s011	w|w:w:S:SSw]
a0317_w04s011p1	w|w:w:S|SSw]
a0317_w05s010	w|w:w:w:wSw]
a0317_w05s020	w|w:w:w:Sww]

Table 11: Example Prosodic Twiddle Edit and Signatures

left and right respectively, followed by the letters 'px' indicating the swap. If the character after the word index is an s, it is followed by a two-digit syllable index and then a digit for the new stress value of the symbol; if it is an x, then two syllables are swapped, and it is followed by the syllable index and the new stress value at the index (the other syllable takes the opposite label). Finally, when the final syllable is twiddled, it may also have a phrase change, indicated by a p and the new break level.

An example of edit and prosodic signatures for twiddles of `rms_arctic_a0317` is shown in figure 11. This is the same text as the pronunciation prompt ("He was a wise hyena").

7.4.2 *State Duration Objective Function for Prosodic Twiddles*

We use RMS state z-scored duration error as once objective function to evaluate twiddles. Festival/FestVox uses z-score normalized state durations in prediction and when reporting correlation and error, but other authors have suggested that log duration may be appropriate because they look more like Poisson than Gaussian distributions, bounded from below. However, state durations are very short and highly quantized into 5ms frames in FestVox. There is an over-

abundance of 1-frame states, but the rest looks pretty Gaussian, as figure 28 shows for distribution of state 2 durations (excluding states 1 and 3 of a three-state model, and excluding silence or pause segments). This is likely due to assumptions of state-level models in the EHMM alignment step.

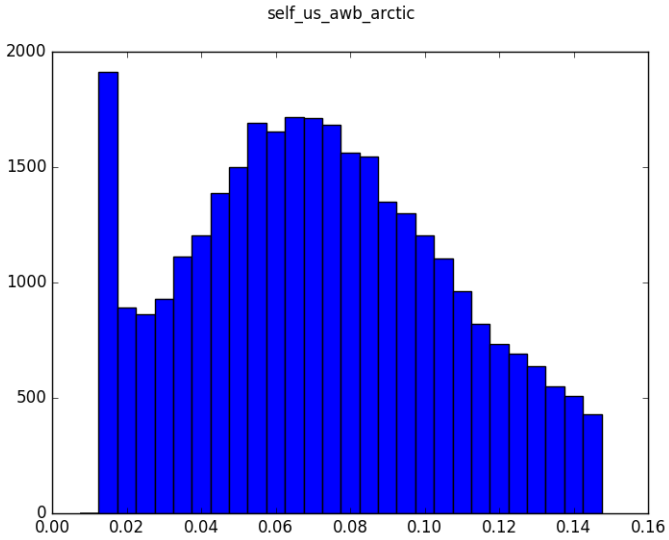

Figure 28: Distribution of 2nd State Durations for AWB Arctic

Figure 29 shows correlation improvements for all Arctic voices over 33 iterations of perturbation of function word stress alone using only state-based duration as the objective function. While the first few iterations of EM may not improve the resulting model, subsequent passes result in an upward trend overall. Working with only function words reduces the size of the search space, so builds may go more quickly. It also seems reasonable insofar as function words are often mislabeled based on reduction models in the initial front end output.

In figure 30, we have 11 iterations of stress perturbation over all words, function and content. The curves are bumpier, but the best improvements in 11 iterations are already better then 33 of the function words alone.

When we perturb everything at once, things move up quickly, but the first few iterations are volatile, as in figure 31.

Table 12 shows an example of changes in the prosodic signature over 8 iterations of EM over analysis by synthesis. The utterance is `awb_arctic_a0293`, "The weeks had gone by, and no overt acts had been attempted." Note that the prompt identifiers do not always correspond across data sets in Arctic; awb is particularly different from the rest. Each line represents the best "move" for that iteration. Sometimes nothing is changed.

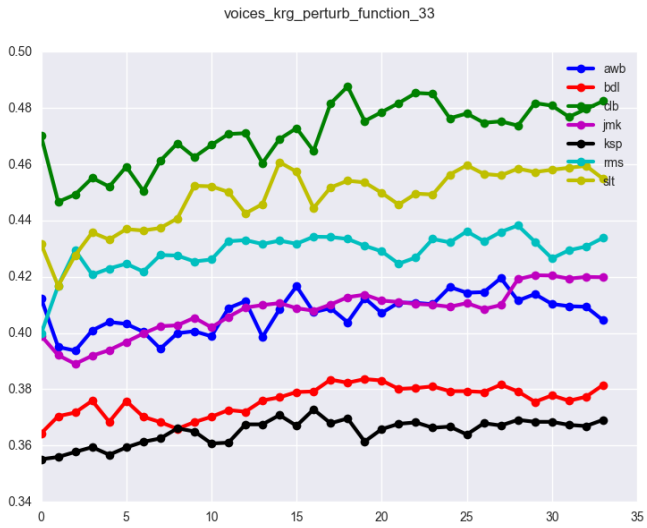

Figure 29: Correlation Changes over Arctic Voices for 33 Iterations of Duration-Based EM Perturbation of Function Word Stress

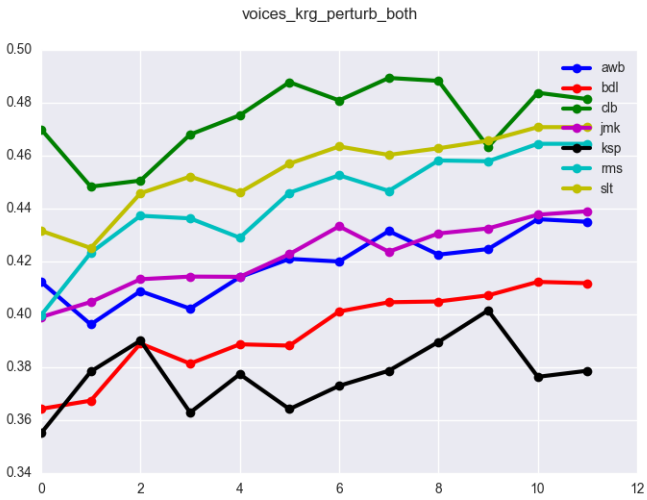

Figure 30: Correlation Changes for 11 Iterations of Duration-Based All Word Stress Perturbation

Raw numbers for RMSE and correlation changes per iteration for each voice are given in Appendix A for the state-duration based objective function.

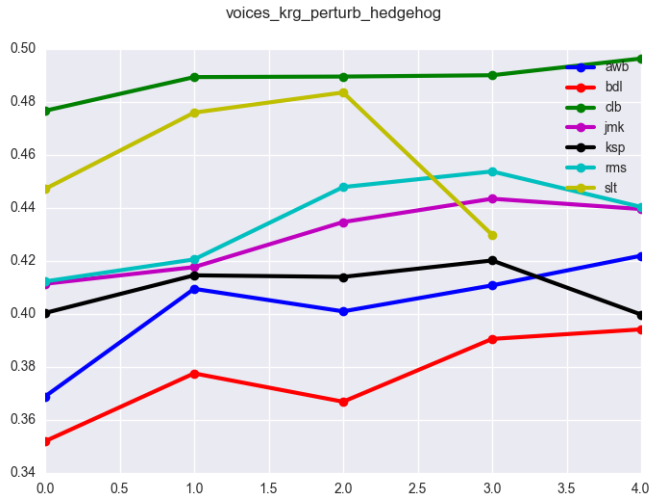

Figure 31: Correlation Changes for 4 Iterations of Duration-Based All Word Stress and Boundary

N	Signature
0	w:S:S:S:S:S:S:wS:S:S:S:wSw]
1	w:S:S:S:S|S:S:wS:S|S:S:wSw]
2	w:S:S:S:S|S:S:wS:S|S:S:wSw]
3	w:S:S:S:S|S:S:wS:S|w:S:wSw]
4	w:S:S:S:S|S:w:wS:S|w:S:wSw]
5	w:S:w:S:S|S:w:wS:S|w:S:wSw]
6	w:S:w:S:S|S:w:wS:S|w:S:wSw]
7	w:S:w:S:S|S:S:wS:S|w:S:wSw]
8	w:S:w:S:S|w:S:wS:S|w:S:wSw]

Table 12: Example of Iterative Signature Changes

7.4.3 *State Duration Evaluation on Hand-Corrected Data*

There are many alignment errors in the initial segment labels. For this experiment, we do the same 3 builds as the initial set up, but then we hand-correct the most errorful 5%, in terms of duration error. Then we rebuild, and evaluate these again, removing the very bad alignments. There were only a few (< 5) really horrible alignments for each voices. As part of the annotation, the /q/ symbol is introduced for glottalization between segments.

The error rate improves more smoothly over time with respect to the same state-based duration evaluation function when the segmen-

tal identity is more correct overall, and the few worst alignments are removed. This shows that prosodic twiddling is improved when the segment annotation quality is improved.

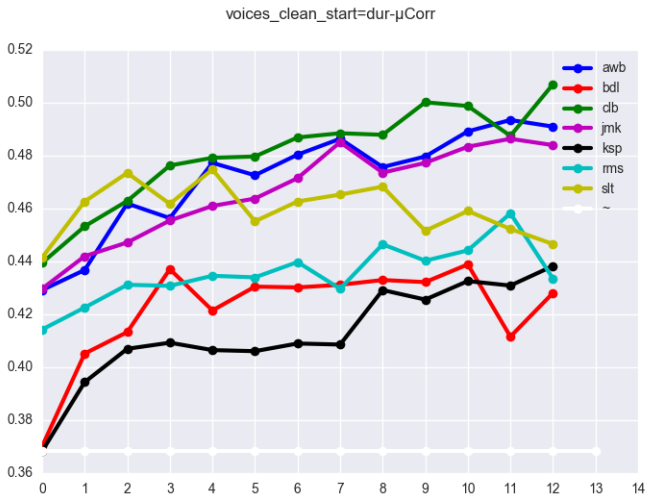

Figure 32: Duration Correlation Changes for 12 Iterations of Duration-Based All Word Stress Twiddling on Partially Hand-Corrected Data

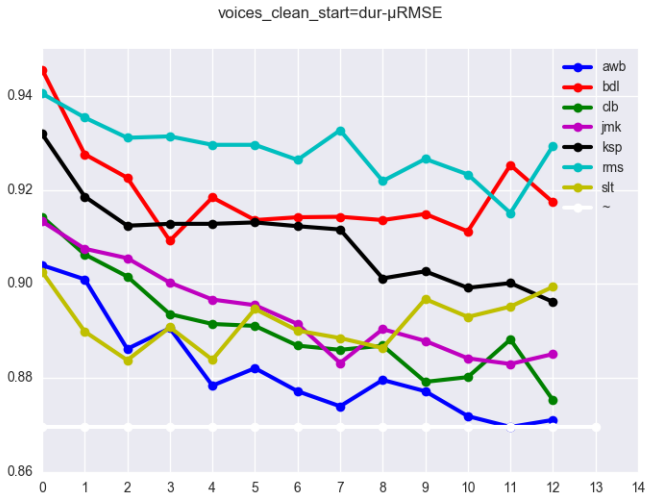

Figure 33: Duration RMSE Changes for 12 Iterations of Duration-Based All Word Stress Twiddling on Partially Hand-Corrected Data

7.4.4 *Combined* F_0, C_0/C_1, *and Duration Evaluation Function*

The last form of objective function for evaluation is the most complicated. Rather than using a single measure, such as path-DTW MCD or state-based duration, we construct a function which incorporates

F_0, C_0/C_1, and duration, which correspond to the classically defined correlates of stress [12] [11]. Because duration is one of the measures, this can only be applied after the segments and syllable structure have been fixed. The changes are to the prosodic signature.

F_0 error is calculated using RMS error over DTW of the log fundamental frequency, subtracting off the baseline as Silverman [40] proposed in order to keep the dynamic range of errors similar across talkers and to normalize across speaking range. We truncate the mel-cepstral coefficients and keep only C_0 and C_1, corresponding to log energy and spectral tilt, respectively. Distortion for these is modeled in the same manner as the mel-cepstral DTW path score before. Duration error is calculated as before.

We twiddle the entire corpus for each talker, and then collect the error statistics for each feature. The error distributions are used to normalize each measure into a z score, which is equivalent to a unit variance Gaussian with mean 0. The measures are then combined using euclidean distance to create a single measure. In the absence of other information, we treat the combination as a diagonal covariance matrix.

The motivation for this measure is to avoid favoring any single dimension during optimization, and instead to try to simultaneously satisfy each of the classical correlates of stress simultaneously.

We also removed phrase well-formedness constraints, because we found that it was useful for the algorithm to move through ill-formed states on the way to a final improvement, which we could force to well-formedness at the end.

7.4.4.1 *Twiddling All Utterances with Combined Score*

Initially, we twiddled content word stress and phrasing as before, using the combined evaluation metric to chose the best twiddles for each utterance. The process takes on the order of 3 hours per build despite parallelization. For comparison, a stock build takes less than an hour on the base machine.[4]

Duration correlation using Festival's usual metrics are shown in figure 36, and RMS duration error in figure 35. Correlation moves in the right direction again.

For the `bdl` voice, the mel-cepstral distortion as measured by the usual FestVox metric improves, but the F_0 bounces around, not convincingly improving. The improvement on the MCD is 0.03, which is a perceptible improvement according to [84]; small, but this is only 5

4 3.5 GHz 6-Core Intel Xeon E5 running Mac OS X 10.x

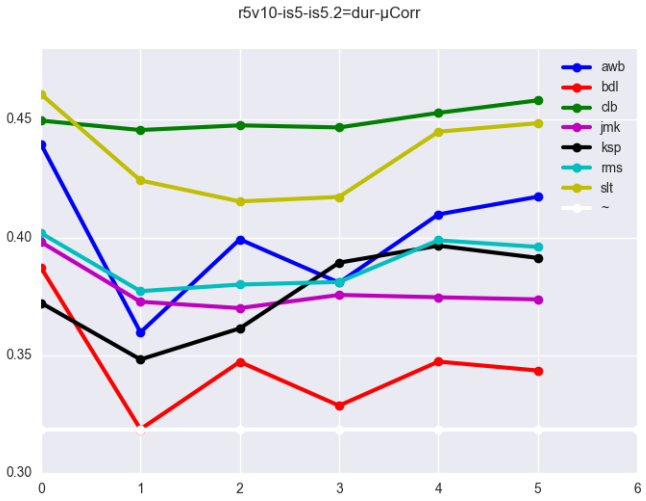

Figure 34: Duration Correlation Changes for 5 Iterations of Combined Content Word Twiddling

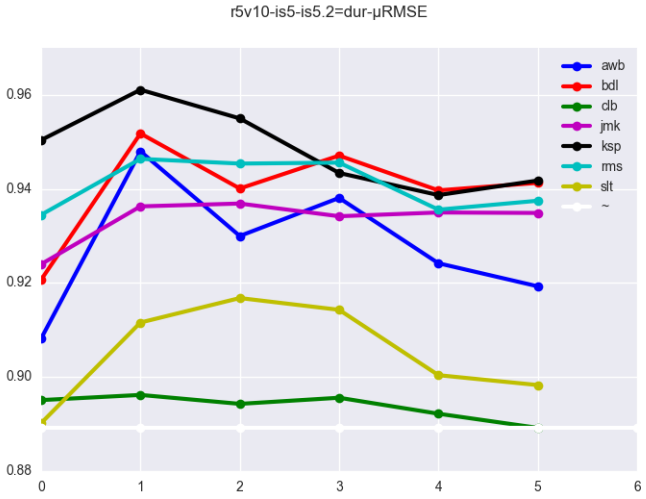

Figure 35: Duration RMS Error Changes for 5 Iterations of Combined Content Word Twiddling

iterations. Not all voices follow this pattern over the 5 iterations, but the overall trend is toward improvement for all measures.

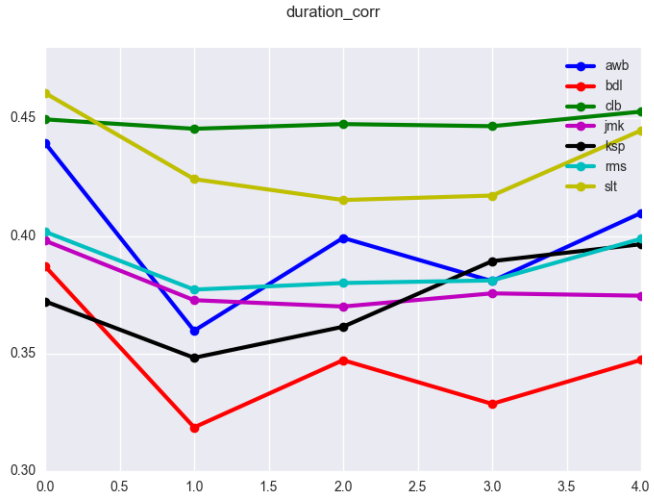

Figure 36: Duration Correlation for 5 Iterations of Combined Content Word Twiddling

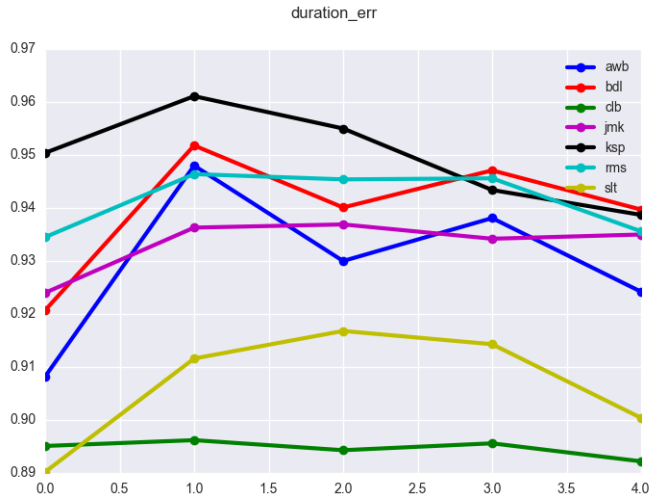

Figure 37: Duration Error for 5 Iterations of Combined Content Word Twiddling

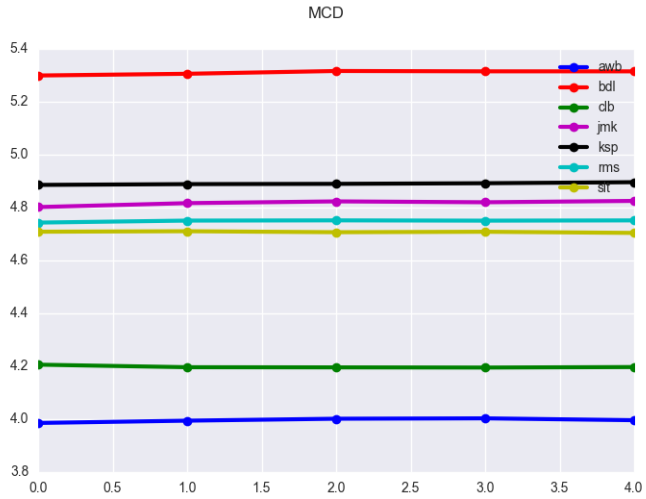

Figure 38: Mel-Cepstral Distortion for 5 Iterations of Combined Content Word Twiddling

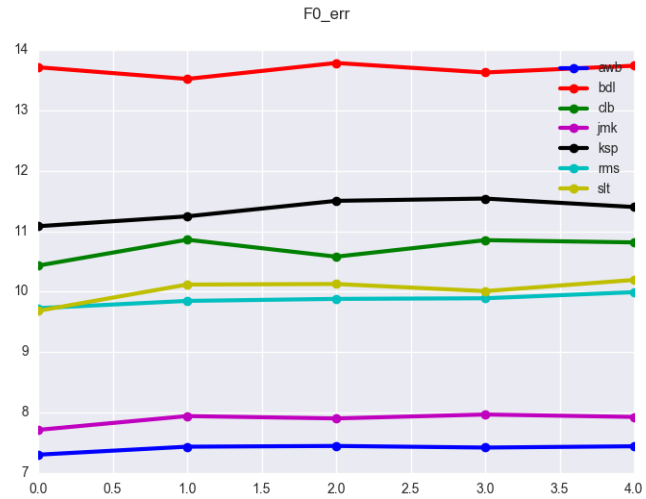

Figure 39: F_0 Error for 5 Iterations of Combined Content Word Twiddling

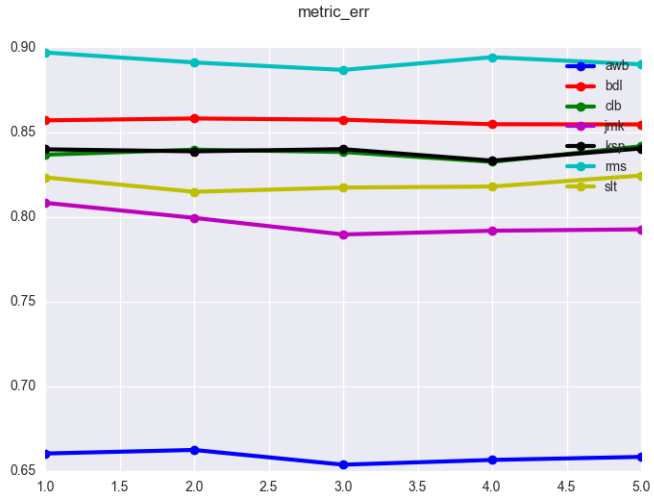

Figure 40: Metric Error for 5 Iterations of Combined Content Word Twiddling

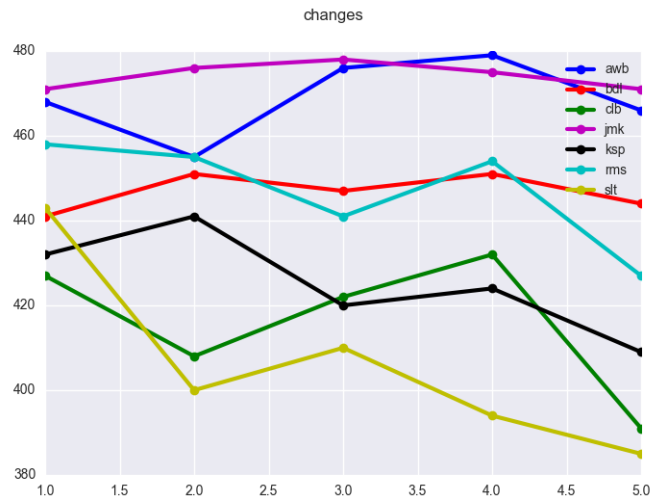

Figure 41: Number of Changes over 5 Iterations of Combined Content Word Twiddling

7.4.4.2 *Ranked Largest Error Subset Twiddling with Combined Score*

In order make progress more quickly, we implemented a strategy of rank-ordering the utterances by the resynthesis error of the untwiddled original, and taking the 50% with largest error and twiddling them. This cut the build time nearly in half. The strategy arose when building another corpus, and audio book rendition of Black Beauty with nearly 6,000 utterances. With some 30 twiddles per utterance, this was taking too long to produce and evaluate, and so we adopted this utilitarian approach.

Since the overall number of twiddles was reduced, it was easy to twiddle all words, stress and phrasing.

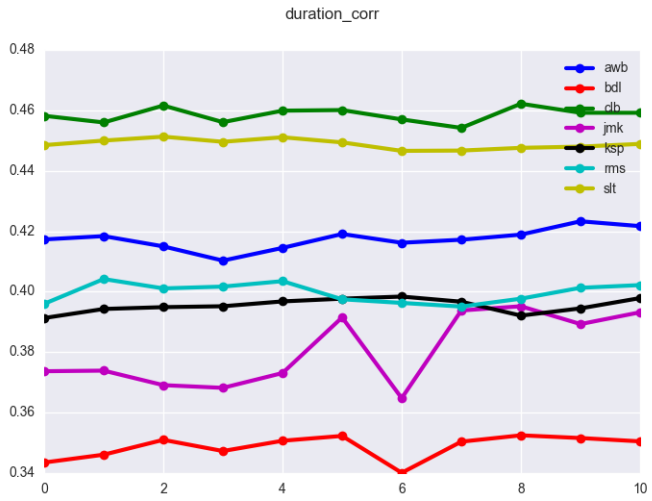

Figure 42: Duration Correlation for 10 Iterations for 7 Voices

7.4.4.3 *Comparing Results from Metrics*

In this experiment, we use the SLT Arctic data and twiddle it for 15 iterations, using four different metrics: F_0, C_0/C_1, duration ("dur"), and the combined metric ("combi").

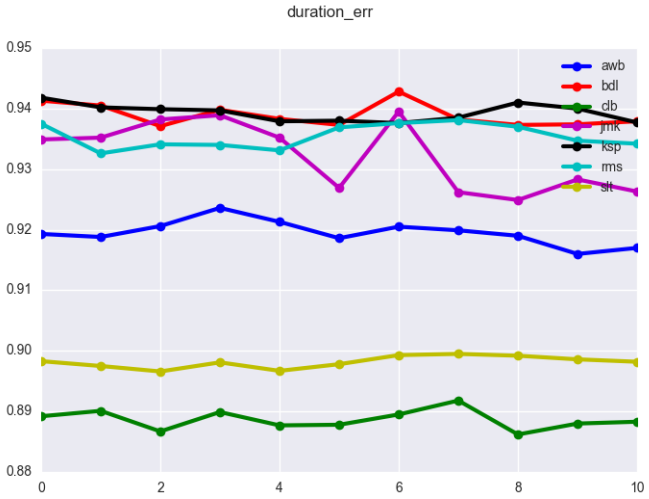

Figure 43: Duration Error for 10 Iterations for 7 Voices

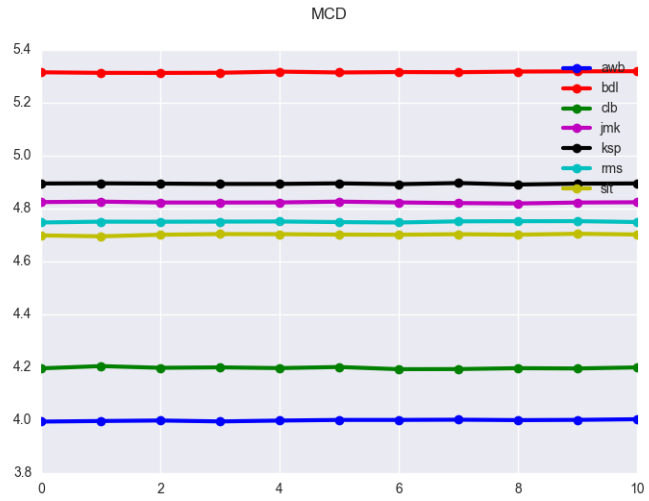

Figure 44: Mel-Cepstral Distortion for 10 Iterations for 7 Voices

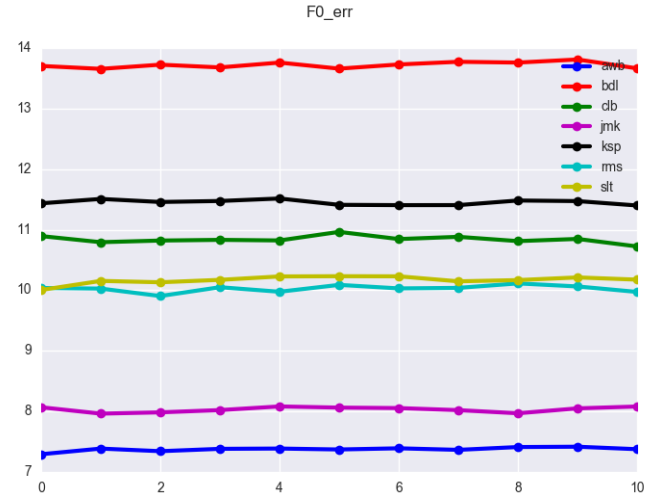

Figure 45: F_0 Error for 10 Iterations for 7 Voices

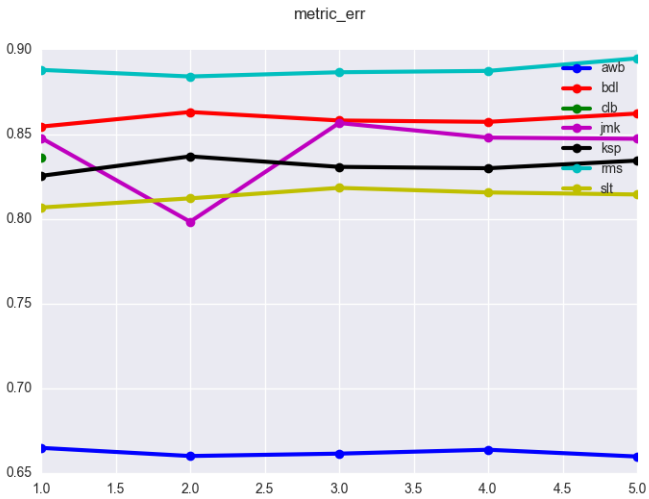

Figure 46: Metric Error for 10 Iterations for 7 Voices

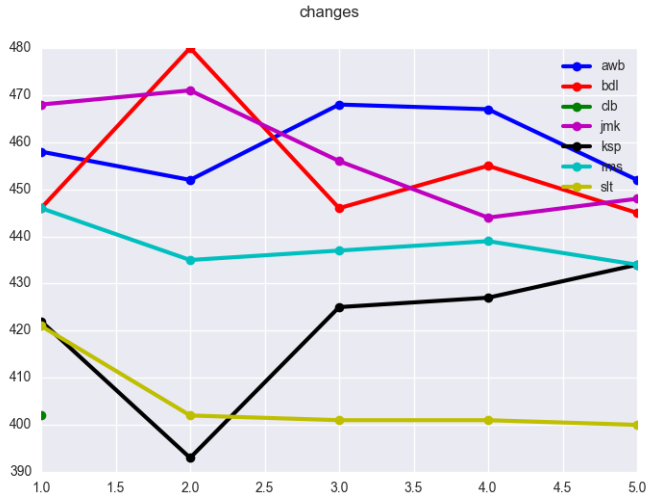

Figure 47: Number of Changes over 10 Iterations for 7 Voices

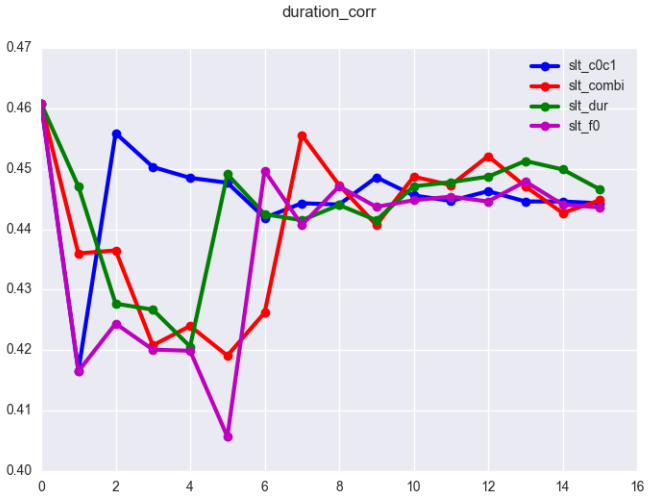

Figure 48: Duration Correlation for 15 Iterations of SLT With Four Metrics

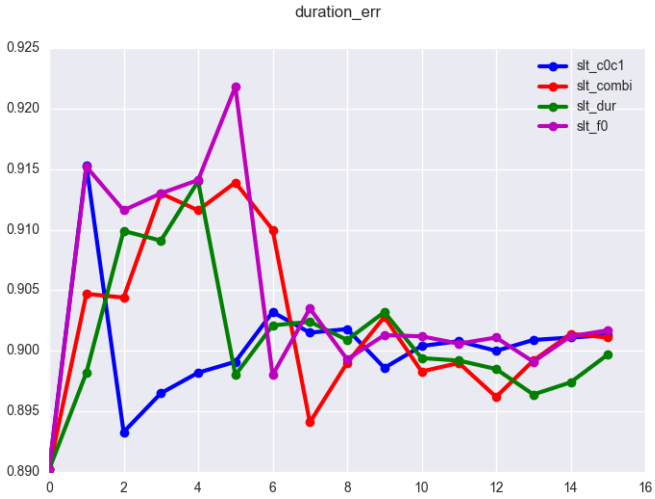

Figure 49: Duration Error for 15 Iterations of SLT With Four Metrics

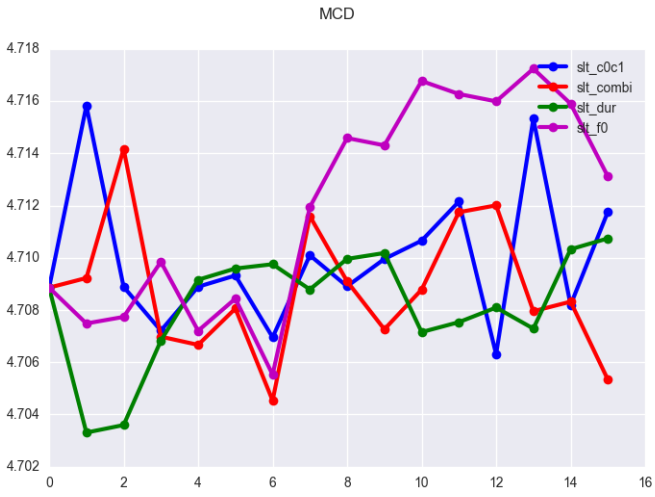

Figure 50: Mel-Cepstral Distortion for 15 Iterations of SLT With Four Metrics

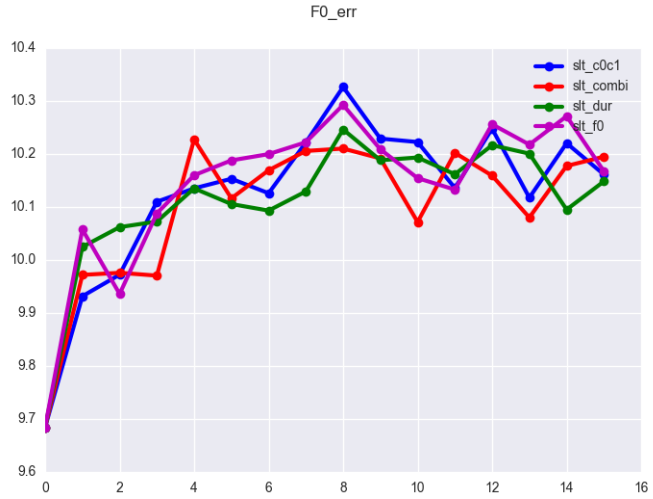

Figure 51: F_0 Error for 15 Iterations of SLT With Four Metrics

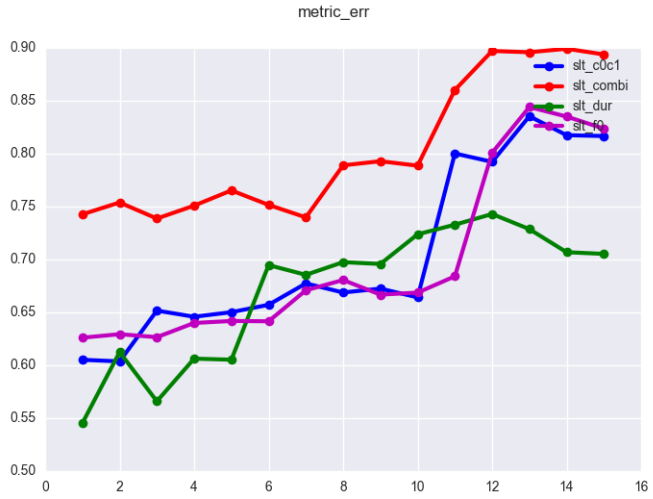

Figure 52: Metric Error for 15 Iterations of SLT With Four Metrics

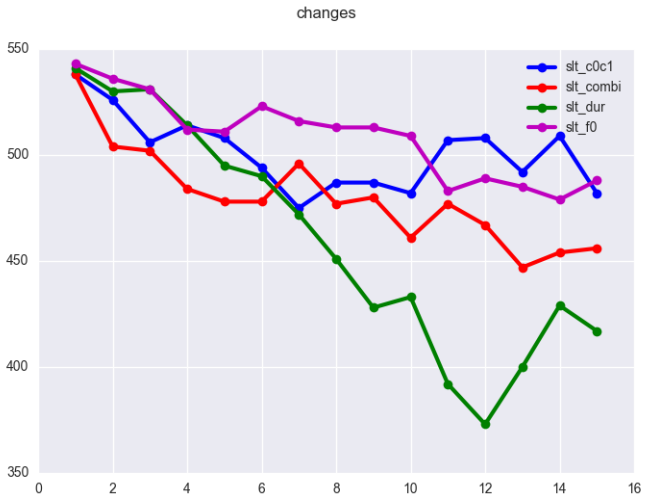

Figure 53: Number of Changes over 15 Iterations of SLT With Four Metrics

7.5 RECAP OF EM ANALYSIS BY SYNTHESIS PROCEDURE

The process is similar, differing generally in choosing the objective function to apply, and what to twiddle in the utterances.

7.5.1 *Stock Voice Setup*

- Set up stock voice build
- Build prompts and prompt labels
- Align using EHMM with short silence insertion
- Remove short silences under 80 ms
- Build utterances from initial prompts and prompt labels
- Introduce phrase breaks for silences over 20 ms
- Extract base signal features
- Build clustergen and duration models
- Collect metrics for HRG baseline

7.5.2 *Baseline Voice Build*

- Resynthesize utterances (clustergen)
- Use resynthesized for new prompts, prompt labels
- Align segments using EM without short silence insertion
- Build utterances with new representation

- Convert to KRG build (decorate, change clustergen and duration features)
- Extract dependent signal features
- Build clustergen and duration models
- Collect metrics for KRG baseline

7.5.3 *EM Iterations (Epsilon Scattering)*

Iterate:

- Resynthesize utterances
- Order decreasing by objective function
- Take top N or proportion (perhaps all)
- Twiddle the selected utterances
- Take best twiddles for each selected utterance
- If segmental: Copy into prompts and prompt labels and realign, remake build utts
- If prosodic: Copy into build utterances
- Extract dependent signal features
- Build clustergen and duration models
- Collect metrics

8

SUBJECTIVE EVALUATION

While the objective measures of F_0 error, Mel-Cepstral Distortion, and duration error have been reported in Chapter 7, subjective listening tests were also carried out.

TestVox [93] was used in each experiment and deployed to Amazon Mechanical Turk (MTurk) as Human Intelligence tasks.

8.1 REDUCIBLE WORDS

In this evaluation, we compared user preference for voices after the reducible words update, with those from the baseline without it. We take a stock voice, and a voice after phonetic iterative updates, and present pairs of stimuli to listers as an A/B test to determine which sounded better, with an option to choose neither.

The function words were iteratively updated using the analysis by synthesis Expectation Maximization procedure for pronunciations as in Section 7.3.3 over 10 epochs. The most common pronunciation was then used as the pronunciation for each voice.

Data was selected from additional "nice" Arctic data that was not included in the voice databases. The SLT voice was used in this test, with 16 unique workers and 300 total observations.

The results are presented in Table 13. The updated reduced pronunciations were mildly preferred, 44% to 40%, with 16% declared undecideable.

Preference	Count	Percentage
Baseline	120	40%
Updated	133	44%
Neither	47	16%

Table 13: Baseline vs. Iterated Function Word Reduction

Preference	Count	Percentage
Baseline	177	55%
Updated	106	33%
Neither	116	37%

Table 14: Baseline vs. Iterated Prosodic Update

8.2 ITERATED PROSODIC IMPROVEMENT

For this evaluation, we perform an A/B test again (with option for "neither") using held-out Arctic sentences using MTurk. The baseline voice is compared against a voice iteratively updated over 5 iterations using the combined objective function as in section 7.4.

The baseline prediction was used for both sets of samples; that is, the stock Festival symbolic prosody placement as trained on the Boston University Radio data (talker f2b) for break, stress and accent placement. No changes in prediction based on the updated, iterated corpus.

For this test, there were 24 unique workers, with 320 observations on the SLT voice.

The results are shown in Table 14. The listeners favor the baseline 55% to updated 33%, with 37% judged equal.

While this was not the expected results, the train/test mismatch in break, stress and accent prediction seems to have a had an effect on preference.

8.3 LIMERICKS WITH ESTABLISHED SIGNATURES

For this test, we use limericks from Lear's "The Book of Nonsense" [94].

The default Festival front end for US English text processing was used to generate the initial signature, and then the Limerick signature was edited by hand and imposed up the utterance. The signature-imposed utterances were then re-synthesized by the baseline synthesis and the updated voices, using the same voice builds as in the previous experiment.

Each Limerick was annotated with the traditional Limerick stress phrase pattern of

Preference	Count	Percentage
Baseline	120	39%
Updated	147	47%
Neither	43	14%

Table 15: Results for Limerics, Baseline vs. Updated

```
1   w* S w* S w* S w* ]
2   w* S w* S w* S w* ]
3   w* S w* S w* /
5   w* S w* S w* /
4   w* S w* S w* S w* ]
```

An example:

```
There was a Young Lady of Ryde,
Whose shoe-strings were seldom untied;
She purchased some clogs,
And some small spotty dogs,
And frequently walked about Ryde.
```

The signatures, before and after:

```
before S:w:w:S:Sw:w:S|S:Sw:w:Sw:wS|S:Sw:w:S|w:w:S:Sw:S|w:Sww:S:wS:S]
after  w:S:w:w:Sw:w:S]w:Sw:w:Sw:wS]w:Sw:w:S|w:w:S:ww:S|w:Sww:S:ww:S]
```

Utterances were again compared in an A/B test, with 22 workers and 310 total observations. The results are in Table 15.

Here, the preference shifts back to the updated voices. The prosodic annotation is equally mismatched to the baseline and updated voices, in that it is not tailored to either, but the updated voice is mildly preferred for Limericks with pre-defined break, stress, and accent placement.

8.4 COMPARING RESYNTHESIS OF BASELINE AND UPDATED VOICES

In order to work around the train/test mismatch in the prosodic annotations between the stock voices and the prosodically updated voices, we take the annotations of each and resynthesize them.

In this evaluation, we compared user preference for resynthesized marked-up utterances, with those from the baseline without it. We

Preference	Count	Percentage
Baseline	218	44%
Updated	245	49%
Neither	37	7%

Table 16: Comparing Baseline and Updated Resynthesis to Natural Speech

take a stock voice, and a voice after iterative prosodic updates, and present pairs of stimuli to listers as an A/B test to determine which sounded better, with an option to choose neither.

The results are in Table 16.

Again we see a slight preference for the updated voices. This appears to support the hypothesis of a train-test mismatch between the front-end and the results of the process, and implies that if the new format were produced or predicted, the synthesis would be improved.

9 CONCLUSION

9.1 DISCUSSION

We started by searching the graph neighborhoods to discover and enumerate features automatically. This led to a reformulation of the utterance structure to overcome some of the shortcomings of the traditional FestVox/Festival structures. Finally, we used the new, smoother representations to automatically improve annotation and, by extension, improve prosodic models and the realization of prosodic features for authoring.

The approach we have taken here opens the door to much more exploration of the synthesis model itself, while lowering the complexity of use. With the KRG-style representation, many of the opaque and unexpected issues of the HRG fall away: we can visualize the graphs, inspect them by type, have more general relations, and automatically perform some of the tasks.

Feature enumeration, syntagmatic structure extraction, and automatic exploration of the graph structure reduce the complexity of creating new relations. This should make it easier to explore new structures, and build new languages quickly.

9.1.1 *KRG: Isomorphically Reformulating the HRG*

We found that the conventions and structures used in FestVox voice builds and Festival are quite powerful, but have limitations for automated discovery and iteration improvement. In particular, there are many surprises in temporal ordering, irregularities of type coding, and no mechanism for multiple parentage.

With some reformulation, the KRG introduces nodes in the expected order (good for neighborhood graph search), object type encoding, reduced relations to two basic types (simple and bipartite), and introduced syntagmatic relations (Foot, ONC) and syntagmatic features (position in parent, number of children) to replace some of the otherwise one-off feature functions (ssyl_in, ssyl_out, pos_in_syll, is_final, etc.).

These changes made it easier to perform some of the automated processes outlined, and also make it much easier to add and remove relationships and features in the overall modeling, which simplifies experimentation.

9.1.2 *Value of the Inductive Graph*

The graph of an HRG or KRG leads to a structure which can be decorated and searched as an entire graph, in contrast to a list of features alone. With the inductive graph approach, a new graph is always well-formed and built from another graph (initially, the empty graph) using well-defined operations. Because of this, it can be examined, queried and displayed consistently.

By incrementally adding relations, we preserve all the linguistic information at each step if we want to – it can be stripped to the necessary final parts as in FLite, but we have a complete structure that is isomorphic to the run-time objects in the synthesis system.

As a well-formed graph, it also lends itself to graph-oriented tools like `GraphViz`, which was very helpful for visualization.

Finally, any fully-rooted tree structure, such as MaryXML, can be trivially and losslessly converted to a graph, but the converse is not true. Thus, the XML-based formats are a topological subset of the HRG-like graph formalism, and the techniques outlined here can be applied to them easily.

9.1.3 *rpath and Feature Paths*

Feature paths have always been one of the great strengths of Festival: they provide a method of accessing relative nodes and features in the graph without any re-coding of the underlying system. With `rpath`, we extend the original and results of paths to node sets, similar to `XPath`, which makes it very easy to query and subset the data.

Feature paths in Festival are used to get to features, but from a programmatic perspective, it is often useful to work on sets. It is convenient, for instance, to select all words by using a call (in python)

```
utt.find('R:Word')
```

When gathering statistics, we can count them and find distributions of their feature values. When we are exploring the data, we can easily query for a particular word and see its features. And when

twiddling, we can use the same path mechanism to find the nodes we are twiddling and operate on them.

While a path traversal expression language is syntactic sugar, it made life better for this developer.

9.1.4 *HRG as API*

One important aspect of the HRG style is that it can be used as an Application Programmer Interface as much as an interchange format. The utterance structure is passed from module to module, and each reads the graph and decorates it (as an inductive graph).

However, since Festival was really only made to work with itself, many of the feature values are not written into the utterance itself, but maintained in memory or computed dynamically. This makes the operations opaque and resistant to modification of the types we have done here.

If all the features that are used are serialized into the HRG, it becomes easier to allow other operations to observe, act on, and modify the contents. Another great benefit is that systems can be compared directly based on their inputs and outputs, deterministically, which allows for improved modularization and testing. As an HRG can always be turned into a KRG and back, there is no loss of fidelity or generality in the representation, and internal processing of a KRG is more efficient due to reduced indirection and relation simplification.

With the HRG as an API, we can compare how two signal generation systems perform, given precisely the same input. This has not been possible in the Blizzard challenge, but could be if a challenge included using the HRG as an API.

9.1.5 *Importance of Segmental Alignment*

The iterative improvement model depends strongly on the quality of segmental alignments. All duration measures are derived from the segment boundaries, and the acoustic measures are created and predicted with respect to them. For this work, we relied on the EHMM utility in FestVox for alignments; this was enough to prove the concept, but we should have the best labeling we can in order to do iterative EM-based modeling of the prosody and segments.

When we hand corrected the words of 5% of the utterances, the improvements were more consistent and moved more smoothly in the right directions (getting better). Without the hand correction, we

get more oscillation in the modeling: the overall scores go the right way, then the wrong way, and back and forth, though generally going in the right direction.

We could also do away with the segmental pronunciation iterations if we used a better aligner, which allowed for pronunciation variations in the alignments, which EHMM does not.

While all this did get working, there were several quirks in the free software versions which had to be overcome. The FestVox builds have surprising hidden dependencies and small errors in the code which compound to trap for the unwary developer. One that stands out is the coupling between the alignment process and the creation of the utterances, which had to be re-created nearly in entirety in order to modify the data CLUSTERGEN operates upon. This could be streamlined quite a bit.

9.1.6 *Glottalization*

In the hand-correction phase, it became clear that there was a lot of glottalization going on which was not being modeled well. A /q/ unit was introduced for these, but since we only updated the 5% of the data, it wasn't extended through the whole corpus. But the effects were quite large and consistent across talkers: introducing glottalization especially before strong vowels with null onset, such as "The Eighteenth Amendment" /dh iyo : q ey1 t iyo n th : ax0 . m eh1 n d . m ax nt/. Furthermore, the front-end does not predict glottalized forms or a glottal segment like /q/. It is not generated at synthesis time, and the units with and without glottalization are all lumped together, degrading the statistics.

9.1.7 *Accent Group vs. Phrase*

Especially as we iterated further with the combined F_0, C_0/C_1, and duration metric, more and more minor breaks were introduced, to the point where things start looking more like accent groups than phrases. It seems as if the three levels of break were not sufficient: we should go for four, as ToBI suggests, in order to capture these levels.

While it may be useful to represent break level is a continuous rather than discrete value, it is useful to quantize it and treat it as symbolic for iteration operations.

9.1.8 *Nuclear or Very Heavy Prominence*

With the two values of stress here, we are also missing a notion of extra heavy or nuclear phrase accent. Because of this, there is a tendency for the system to compensate by using phrase breaks to increase duration. For example, if we were to have

```
I'm THE Easter Bunny
```

the system would begin with the signature

```
AYM dhiy IYster BAHniy ]
```

and would quickly move the stresses:

```
aym DHIY iyster bahniy ]
```

But, simply changing the word "the" to a strong syllable doesn't fully account for the extra duration and intensity of the emphatic stress, and so it will add a break after:

```
aym DHIY / iyster bahniy ]
```

This is also not really enough, so it raises break value after it to a large break, and may even place a boundary before the emphatic stress which acts like pre-lengthening:

```
aym / DHIY ] iyster bahniy ]
```

And the algorithm says "I did what you told me, boss!"

These marks are not technically wrong. The marks were annotated this way, and a model trained across the aggregate. It is likely to improve the synthesis, but it no longer corresponds to our notions of phrasing. This compensation related to having only three levels of break as well, but in a more oblique manner.

9.1.9 *Unsupervised vs. Supervised Learning*

One way to work around the issues of poor alignments, phrase level issues and stress level issues is to use the approach for supervised

incremental improvement, rather than operating completely automatically. This would allow the system to improve more in line with our intuitions about the mark-up.

This could work by allowing the system to propose changes to a person who would decide whether or not to accept the correction, rather than just taking them all. The work can be prioritized in terms of resynthesis error, similar to the automatic fashion – the ranked results can be given the same way, It could be possible then to supervise correction of the worst prosodic errors semi-automatically.

9.1.10 *Ranking and Tractability*

Speaking of ranking, one of the apparent results was that ranking the utterances by resynthesis errors in order to choose those with the largest error did not help smooth out the metrics or speed convergence. We had expected ranking and taking the worst to provide a sort of smoothing effect, to prevent oscillations in the measured objective function, but it did not. This may have been due to segmentation or other errors, but it was a bit of a surprise. Better results were achieved by twiddling everything and take the best for every utterance. However, ranking did speed up the overall build process for each iteration.

9.1.11 *Subsetting*

Subsetting the twiddle effects, however *was* effective. It was very useful to operate on either function or content words, and work on either stress or phrasing for some iterations.

For example, it was effective to twiddle stress in function words alone, irrespective of potential phrase boundary changes. This helped regularize the data safely, because function words are likely to be reduced and unlikely to be followed by a break in English.

In later iterations, we can operate over everything with less risk if the easy cases have been corrected and the model improved first. While this ordering seemed sensible for English, we did not compare the result in a factorial design to permute the experimental design[1]. In future work in other languages, subsetting and ordering should depend on what operations can be done to improve the model safely before opening up unbounded gradient descent.

1 For example, compare twiddling all of them for all steps vs. using this order.

9.1.12 *Inducing Tone Type*

We did not look at different tone types, despite extensive discussion of ToBI in Chapter 2. We could have tried to induce these, but instead assumed that the default ToBI labels produced by Festival were not particularly accurate. We did start some work on the Boston University Radio Corpus (BURNC), which has annotated ToBI labels, but there are some structural problems in converting the annotation data into Festival (most noticeably, different pronunciations and reductions in the lexicon, stop closures v. no stop closures, differing numbers of break types, and differing tone coding). That said, it should be possible to perform iterative improvement from a sensible baseline.

For the case of Arctic, there are few questions and almost no low pitch accents. The utterances are relatively neutral, without a lot of tonal variation. Perhaps all of them are something like /L+H* H* (H* H* ...) L- L%/. Due to this regularity, there wasn't much tonal difference to capture, except perhaps downstep.

Tone type be more interesting with the "Black Beauty" audio book experiment, but we haven't explored this fully.

9.1.13 *Authoring TTS Prompts for Recording*

One of the underlying issues with the Arctic databases is that the talkers delivered them with no mark-up, setup or "back story" to each prompt. That is both a strength and a weakness: on the one hand, they are delivered naturalistically, without too much contrivance; on the other, the data is not annotated to reflect this, and the talkers delivered quite varied versions of the same prompts (emphasizing different words, different phrasing, differing pronunciations).

In the LSAFo work, we carefully made the prompt list and annotations to systematically explore the space, but at the expense of naturalness.

It may be worthwhile to give guidance somewhere in between these, such as the text and the segmental signatures as we have here. A corpus is made, the default signatures produced and hand-edited to make sensible deliveries, and then the prompts spoken by the voice talent for recording. The annotation is taken as a baseline for modeling, and the iterative method used to check if there were notable deviations from the script.

9.1.14 *Authoring With Synthesis*

As a result of this work, after improving the annotation of the database, an author crafting synthetic output from mark-up should be able to get more predictable and reliable expressive control from stress and break placement. Another use for the text-to-speech output author is to generate a palette of possibilities from the twiddles, which can each be listened to – and, similarly to the iterative system, the one closest to the author's intent can be chosen.

The signature also lends itself to graphical tools, which could lead to future synthetic authoring environments where the values can easily be altered. When the signature is changed, there are topological changes to the graph behind it, but they are factored away in the presentation and editing.

9.2 FUTURE WORK

There are many possibilities for future work.

9.2.1 *Breaks and Stress from Annotated Results*

While we got quite far in automated improvement and discovery, we did not complete the process to convergence for a very large corpus and create break and stress prediction from that. It is a natural extension of this work to create new, trained models that have a close match to the contents of the database and annotation system.

While not every annotation produced by the system fits with intuition, the aggregate is regularized across the data, and so predicting the same sort and placement of features should result in improvement.

We also did not look at the impact on meaning and discourse on the placement of breaks and stress, but this is made possible with an iterative prosodic annotation improvement system. After a large corpus is regularized, it may be used to train prosodic realization models from pragmatic and discourse relations.

9.2.2 *More Stress and Phrase Values*

Since we suffered from not having a stress level to indicate emphasis, we can go to three or more stress values to explore that space; the

same can be said for breaks. With three breaks, the combined evaluation metric favored accent groups over phrases. We will explore enhancing the annotation scheme to include more levels of each, to see if we can recover better overall structures.

9.2.3 *Optimize Independent Modal Features for Fo, Co/C1, Duration*

Conflating the combined measures into one does produce a simple annotation set, but it may be useful to model the separate pieces independently, and create an annotation for each – intonation events, intensity events, duration events. To do this, we can use the same method but using each error function to iteratively produce its own annotation.

There are cases where intensity is increased without a pitch accent, and vice versa. We can try to annotate and predict them, comparing the separated results with the combined. This may also lead to more expressive control for authoring synthetic speech in flexible environment[2].

9.2.4 *Other Languages, Different Styles*

It will be interesting to see how things work in languages that have markedly different prosodic organization from English, such as Spanish, French, and Mandarin Chinese. This may help toward creating a new theory of organization, perhaps with some universal underpinnings.

9.2.5 *Apply to Other Synthesizers*

Since Festival is only one synthesizer and we have proposed a general method, we can try the same techniques by converting between some existing XML-style systems and the HRG to run the same experiments, and re-construct the training data after improvements. This will show that the method applies universally, and may lead to a synthesizer-independent Twiddling Toolkit.

2 Rather than narrating the whole thing. If only one rendering is ever needed, it may be better to record a performance. In practical use of text-to-speech, there are often cases that are produced badly by default, and a method of sculpting the prosody without getting into the parametric weeds is helpful.

9.2.6 *Supervised Improvement*

As pointed out above, the method lends itself to an iterative supervised learning context. Rather than allow the system to work on its own, we may produce an ordered set of possible "edits" to be approved by a human operator. We can explore how effective the ordering is (how much correction leads to how much improvement, when ranked by error), to see often and when a person agrees with the system, and to compare the final annotations and predictions between the supervised and unsupervised conditions.

9.2.7 *Tone Type Annotation*

While we did not explore tone type annotations due to the corpora and time limitations, this is another area we want to go into. However, the default tone annotations from Festival do not give us a very good starting point. Instead, we will try to start from a system like AuToBi, and iteratively improve upon the results.

9.2.8 *Annotating and Authoring Environment*

A graphical tool to edit and compare twiddles will be extremely useful. We can show these in a terminal, and using GraphViz, but the process is not easy to directly manipulate. An editing tool which operates over the features we have described could help both in annotating data, and in producing synthetic output with increased expressiveness.

9.3 CONCLUSION

We have shown a method for automatic exploration of linguistic structures, refactored a conventional representation to aid discovery, improving the automatic annotation of prosody, and improving the resulting synthesis quality. The techniques used in exploring the space may also be used to modify the rendering of synthetic speech to better match a desired performance and improve corpus annotation broadly.

BIBLIOGRAPHY

[1] G. Zeglin, A. Walsman, L. Herlant, Z. Zheng, Y. Guo, M. Koval, K. Lenzo, H. J. Tay, P. Velagapudi, K. Correll, and S. Srinivasa, "Herb's sure thing: a rapid drama system for rehearsing and performing live robot theater," in *IEEE Workshop on Advanced Robotics and its Social Impacts*, September 2014. (Cited on page 1.)

[2] M. Walker, A. Hunt, and D. Burnett, "Speech synthesis markup language (SSML) version 1.0," W3C, W3C Recommendation, Sep. 2004, http://www.w3.org/TR/2004/REC-speech-synthesis-20040907/. (Cited on page 1.)

[3] R. Sproat, A. Hunt, M. Ostendorf, P. Taylor, A. Black, K. Lenzo, and M. Edgington, "Sable: A standard for tts markup," in *The Third ESCA/COCOSDA Workshop (ETRW) on Speech Synthesis*, 1998. (Cited on page 1.)

[4] F. Theatre, *Don't Crush that Dwarf, Hand Me the Pliers*. Columbia, 1970. (Cited on page 5.)

[5] A. Black and K. Lenzo, "Building voices in the festival speech synthesis system," 2000. (Cited on pages 5 and 69.)

[6] ——, "Festvox: Building synthetic voices," *Language Technologies Institute, Carnegie Mellon University, PA, USA, Tech. Rep*, 2002. (Cited on pages 5, 10, 29, 42, and 69.)

[7] A. Black, P. Taylor, R. Caley, R. Clark, K. Richmond, S. King, V. Strom, and H. Zen, "The festival speech synthesis system, version 1.4. 2," *Unpublished document available via http://www. cstr. ed. ac. uk/projects/festival. html*, 2001. (Cited on pages 5, 29, 41, and 69.)

[8] A. W. Black and K. A. Lenzo, "Flite: a small fast run-time synthesis engine," in *4th ISCA Tutorial and Research Workshop (ITRW) on Speech Synthesis*, 2001. (Cited on page 5.)

[9] J. Kominek and A. W. Black, "The cmu arctic speech databases," Tech. Rep., 2004. (Cited on pages 5, 30, 41, and 67.)

[10] H. Bourlard, H. Hermansky, and N. Morgan, "Towards increasing speech recognition error rates," *Speech communication*, vol. 18, no. 3, pp. 205–231, 1996. (Cited on page 6.)

[11] I. Lehiste and G. E. Peterson, "Some basic considerations in the analysis of intonation," *The Journal of the Acoustical Society of America*, vol. 33, p. 419, 1961. (Cited on pages 7 and 89.)

[12] I. Lehiste, "Suprasegmentals." 1970. (Cited on pages 7 and 89.)

[13] D. R. Ladd, *Intonational phonology*. Cambridge University Press, 2008. (Cited on pages 7 and 16.)

[14] E. Moulines and F. Charpentier, "Pitch-synchronous waveform processing techniques for text-to-speech synthesis using diphones," *Speech communication*, vol. 9, no. 5, pp. 453–467, 1990. (Cited on page 8.)

[15] H. Zen, T. Nose, T. Masuko, A. W. Black, and K. Tokuda, "The hmm-based speech synthesis system (hts) version 2.0." (Cited on pages 8 and 10.)

[16] K. E. A. Silverman and J. Pierrehumbert, "The timing of prenuclear high accents in english," in *Laboratory Phonology I: Between the Grammar and Physics of Speech*, ser. Papers in Laboratory Phonology. Cambridge University Press, 1990, pp. 72–106. (Cited on pages 8 and 14.)

[17] M. Liberman and A. Prince, "On stress and linguistic rhythm," *Linguistic inquiry*, vol. 8, no. 2, pp. 249–336, 1977. (Cited on page 8.)

[18] J. R. Bellegarda and K. E. Silverman, "Improved duration modeling of english phonemes using a root sinusoidal transformation." in *ICSLP*, vol. 98, 1998, pp. 21–24. (Cited on page 8.)

[19] J. P. Van Santen, "Assignment of segmental duration in text-to-speech synthesis," *Computer Speech & Language*, vol. 8, no. 2, pp. 95–128, 1994. (Cited on pages 9 and 17.)

[20] C. Shih, W. Gu, and J. P. v. Santen, "Efficient adaptation of tts duration model to new speakers," in *The Third ESCA/COCOSDA Workshop (ETRW) on Speech Synthesis*, 1998. (Cited on page 9.)

[21] A. W. Black, "ClusterGen: a statistical parametric synthesizer using trajectory modeling." in *INTERSPEECH*, 2006. (Cited on pages 9, 41, 43, and 69.)

[22] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, "Classification and regression trees," 1998. (Cited on page 9.)

[23] A. W. Black and P. K. Muthukumar, "Random forests for statistical speech synthesis," in *INTERSPEECH 2015, 16th Annual Conference of the International Speech Communication Association, Dresden, Germany, September 6-10, 2015*, 2015, pp.

1211–1215. [Online]. Available: http://www.isca-speech.org/archive/interspeech_2015/i15_1211.html (Cited on page 9.)

[24] S. King, A. Black, P. Taylor, R. Caley, and R. Clark, "Edinburgh speech tools library," 2003. (Cited on pages 10, 29, and 69.)

[25] M. Halle and N. Chomsky, *The sound pattern of English*. Harper & Row, 1968. (Cited on page 10.)

[26] W. N. Campbell and S. D. Isard, "Segment durations in a syllable frame," *Journal of Phonetics*, vol. 19, no. 1, pp. 37–47, 1991. (Cited on page 10.)

[27] G. Bailly and B. Holm, "Sfc: a trainable prosodic model," *Speech Communication*, vol. 46, no. 3, pp. 348–364, 2005. (Cited on page 10.)

[28] H. Zen, K. Tokuda, and A. W. Black, "Statistical parametric speech synthesis," *Speech Communication*, vol. 51, no. 11, pp. 1039–1064, 2009. (Cited on pages 10 and 41.)

[29] H. Zen, A. Senior, and M. Schuster, "Statistical parametric speech synthesis using deep neural networks." (Cited on page 10.)

[30] J. B. Pierrehumbert, "The phonology and phonetics of english intonation," Ph.D. dissertation, Massachusetts Institute of Technology, 1980. (Cited on page 11.)

[31] K. E. Silverman, M. E. Beckman, J. F. Pitrelli, M. Ostendorf, C. W. Wightman, P. Price, J. B. Pierrehumbert, and J. Hirschberg, "ToBI: A standard for labeling English prosody." in *ICSLP*, vol. 2. Banff: International Conf. on Spoken Language Processing, 1992, pp. 867–870. (Cited on pages 12, 13, 20, and 51.)

[32] J. Pierrehumbert, "The meaning of intonational contours in the interpretation of discourse janet pierrehumbert and julia hirschberg," *Intentions in communications*, p. 271, 1990. (Cited on page 12.)

[33] A. Rosenberg, "Automatic detection and classification of prosodic events," Ph.D. dissertation, COLUMBIA UNIVERSITY, 2009. (Cited on page 12.)

[34] ——, "Autobi-a tool for automatic tobi annotation." in *INTERSPEECH*, 2010, pp. 146–149. (Cited on page 12.)

[35] A. Maghbouleh, "Tobi accent type recognition," *ISSUES*, vol. 3, p. 1, 1998. (Cited on page 12.)

[36] J. F. Pitrelli, M. E. Beckman, and J. Hirschberg, "Evaluation of prosodic transcription labeling reliability in the tobi framework." in *ICSLP*, vol. 1, 1994, pp. 123–6. (Cited on page 12.)

[37] A. K. Syrdal and J. T. McGory, "Inter-transcriber reliability of tobi prosodic labeling." in *INTERSPEECH*, vol. 2000, 2000, pp. 235–238. (Cited on page 12.)

[38] M. Avanzi, A. Lacheret-Dujour, and B. Victorri, "A corpus-based learning method for prominence detection in spontaneous speech," in *Proceedings of Prosodic Prominence, Speech Prosody 2010 Satellite Workshop, Chicago, May 10th*, 2010. (Cited on page 13.)

[39] C. Shih, "A declination model of mandarin chinese," in *Intonation*. Springer, 2000, pp. 243–268. (Cited on page 13.)

[40] K. E. A. Silverman, "The structure and processing of fundamental frequency contours," Ph.D. dissertation, Cambridge University, 1987. (Cited on pages 13, 14, 15, 19, 24, and 89.)

[41] J. v. Santen and B. Möbius, "Modeling pitch accent curves," in *Intonation: Theory, Models and Applications*, 1997. (Cited on pages 13, 17, 18, 19, and 67.)

[42] G. Bruce, "Models of intonation-from the lund horizon," in *Intonation: Theory, Models and Applications*, 1997. (Cited on page 14.)

[43] S. Öhman, *Word and sentence intonation: A quantitative model*. Speech Transmission Laboratory, Department of Speech Communication, Royal Institute of Technology, 1967. (Cited on pages 14 and 15.)

[44] J. Pierrehumbert and M. Beckman, "Japanese tone structure," *Linguistic Inquiry Monographs*, no. 15, pp. 1–282, 1988. (Cited on page 15.)

[45] K. Dusterhoff and A. W. Black, "Generating fo contours for speech synthesis using the tilt intonation theory," in *Intonation: Theory, Models and Applications*, 1997. (Cited on pages 15 and 19.)

[46] F. Nolan and E. Grabe, "Can'tobi'transcribe intonational variation in british english?" in *Intonation: Theory, Models and Applications*, 1997. (Cited on page 15.)

[47] H. Fujisaki, "Dynamic characteristics of voice fundamental frequency in speech and singing," in *The production of speech*. Springer, 1983, pp. 39–55. (Cited on page 15.)

[48] H. Fujisaki and S. Ohno, "Comparison and assessment of models in the study of fundamental frequency contours of speech," in *Intonation: Theory, Models and Applications*, 1997. (Cited on pages 15 and 16.)

[49] B. Möbius, M. Pätzold, and W. Hess, "Analysis and synthesis of german< i> f</i>< sub> o</sub> contours by means of fujisaki's model," *Speech Communication*, vol. 13, no. 1, pp. 53–61, 1993. (Cited on pages 16 and 17.)

[50] B. Möbius, "Synthesizing german intonation contours," *Progress in Speech Synthesis, Springer*, pp. 401–415, 1997. (Cited on page 16.)

[51] N. Higuchi, T. Hirai, and Y. Sagisaka, "Effect of speaking style on parameters of fundamental frequency contour," *Progress in speech synthesis*, pp. 417–428, 1997. (Cited on page 17.)

[52] J. P. van Santen, B. Möbius, J. Venditti, and C. Shih, "Description of the bell labs intonation system," in *Proceedings of Third ESCA Workshop on Speech Synthesis*, vol. 98, 1998, p. 18. (Cited on page 17.)

[53] J. J. Venditti, K. Maeda, and J. P. v. Santen, "Modeling japanese boundary pitch movements for speech synthesis," in *The Third ESCA/COCOSDA Workshop (ETRW) on Speech Synthesis*, 1998. (Cited on pages 17 and 18.)

[54] K. Lenzo, C.-l. Shih, and J. van Santen, "Clefting of one/two companies," *Unpublished internship work at Bell Labs*, 1997. (Cited on page 18.)

[55] O. Fujimura, "C/d model: A computational model of phonetic implementation," *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, vol. 17, pp. 1–20, 1994. (Cited on pages 18 and 26.)

[56] P. Taylor, "The rise/fall/connection model of intonation," *Speech Communication*, vol. 15, no. 1, pp. 169–186, 1994. (Cited on page 18.)

[57] ——, "The tilt intonation model," 1998. (Cited on page 18.)

[58] J. Louw and E. Barnard, "Automatic intonation modeling with intsint," *Proceedings of the Pattern Recognition Association of South Africa*, pp. 107–111, 2004. (Cited on page 18.)

[59] F. Malfrère, T. Dutoit, and P. Mertens, "Automatic prosody generation using suprasegmental unit selection," in *The Third ESCA/-COCOSDA Workshop (ETRW) on Speech Synthesis*, 1998. (Cited on page 18.)

[60] G. Möhler and A. Conkie, "Parametric modeling of intonation using vector quantization," in *The Third ESCA/COCOSDA Workshop (ETRW) on Speech Synthesis*, 1998. (Cited on page 19.)

[61] J. R. Bellegarda, K. E. Silverman, K. Lenzo, and V. Anderson, "Statistical prosodic modeling: from corpus design to parameter estimation," *Speech and Audio Processing, IEEE Transactions on*, vol. 9, no. 1, pp. 52–66, 2001. (Cited on pages 19 and 20.)

[62] K. E. Silverman, J. R. Bellegarda, and K. A. Lenzo, "Smooth contour estimation in data-driven pitch modelling." in *INTERSPEECH*, 2001, pp. 1167–1170. (Cited on page 20.)

[63] G. K. Anumanchipalli, "Intra-lingual and cross-lingual prosody modelling," Ph.D. dissertation, Carnegie Mellon University, 2013. (Cited on page 23.)

[64] P. C. Bagshaw, "Unsupervised training of phone duration and energy models for text-to-speech synthesis." in *ICSLP*, vol. 98, 1998, pp. 17–20. (Cited on page 24.)

[65] T. Tomoki and K. Tokuda, "A speech parameter generation algorithm considering global variance for hmm-based speech synthesis," *IEICE TRANSACTIONS on Information and Systems*, vol. 90, no. 5, pp. 816–824, 2007. (Cited on page 24.)

[66] S. Takamichi, T. Toda, A. W. Black, and S. Nakamura, "Parameter generation algorithm considering modulation spectrum for hmm-based speech synthesis," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 4210–4214. (Cited on page 24.)

[67] S. Takamichi, T. Toda, G. Neubig, S. Sakti, and S. Nakamura, "A postfilter to modify the modulation spectrum in hmm-based speech synthesis," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 290–294. (Cited on page 24.)

[68] K. Tokuda, T. Kobayashi, T. Masuko, and S. Imai, "Mel-generalized cepstral analysis-a unified approach to speech spectral estimation." in *ICSLP*, vol. 94, 1994, pp. 18–22. (Cited on page 24.)

[69] A. W. Black, K. Lenzo, and V. Pagel, "Issues in building general letter to sound rules," 1998. (Cited on page 25.)

[70] V. Pagel, K. Lenzo, and A. Black, "Letter to sound rules for accented lexicon compression," *arXiv preprint cmp-lg/9808010*, 1998. (Cited on page 25.)

[71] J. R. Novak, D. Yang, N. Minematsu, and K. Hirose, "Phonetisaurus: A wfst-driven phoneticizer," *The University of Tokyo, Tokyo Institute of Technology*, pp. 221–222, 2011. (Cited on page 26.)

[72] C. P. Browman and L. Goldstein, "Some notes on syllable structure in articulatory phonology," *Phonetica*, vol. 45, no. 2-4, pp. 140–155, 1988. (Cited on page 26.)

[73] E. L. Saltzman and K. G. Munhall, "A dynamical approach to gestural patterning in speech production," *Ecological psychology*, vol. 1, no. 4, pp. 333–382, 1989. (Cited on page 26.)

[74] C. Miller, "Individuation of postlexical phonology for speech synthesis," in *The Third ESCA/COCOSDA Workshop (ETRW) on Speech Synthesis*, 1998. (Cited on page 27.)

[75] S. Fitt and S. Isard, "Representing the environments for phonological processes in an accent-independent lexicon for synthesis of english," 1998. (Cited on page 27.)

[76] C. L. Bennett and A. W. Black, "Prediction of pronunciation variations for speech synthesis: A data-driven approach." in *ICASSP (1)*, 2005, pp. 297–300. (Cited on pages 27, 72, and 81.)

[77] P. Taylor, A. W. Black, and R. Caley, "The architecture of the festival speech synthesis system," 1998. (Cited on page 29.)

[78] ——, "Heterogeneous relation graphs as a formalism for representing linguistic information," *Speech Communication*, vol. 33, no. 1, pp. 153–174, 2001. (Cited on page 29.)

[79] M. Schröder, M. Charfuelan, S. Pammi, and I. Steiner, "Open source voice creation toolkit for the mary tts platform," in *12th Annual Conference of the International Speech Communication Association-Interspeech 2011*. ISCA, 2011, pp. 3253–3256. (Cited on pages 29 and 42.)

[80] M. P. Aylett, R. Dall, A. Ghoshal, G. E. Henter, and T. Merritt, "A flexible front-end for hts," in *INTERSPEECH*, 2014, pp. 1283–1287. (Cited on pages 29, 41, and 42.)

[81] J. Ellson, E. Gansner, L. Koutsofios, S. North, G. Woodhull, S. Description, and L. Technologies, "Graphviz – open source graph drawing tools," in *Lecture Notes in Computer Science*. Springer-Verlag, 2001, pp. 483–484. (Cited on pages 33 and 35.)

[82] A. J. Hunt and A. W. Black, "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, 1996, pp. 373–376. (Cited on page 41.)

[83] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001. (Cited on page 45.)

[84] J. Kominek, T. Schultz, and A. W. Black, "Synthesizer voice quality of new languages calibrated with mean mel cepstral distortion." in *SLTU*, 2008, pp. 63–68. (Cited on pages 45 and 89.)

[85] O. Fujimura, "Phonology and phonetics. a syllable-based model of articulatory organization." *Journal of the Acoustical Society of Japan (E)*, vol. 13, no. 1, pp. 39–48, 1992. (Cited on page 47.)

[86] A. W. Black and K. A. Lenzo, "Optimal data selection for unit selection synthesis," in *4th ISCA Tutorial and Research Workshop (ITRW) on Speech Synthesis*, 2001. (Cited on page 47.)

[87] D. Crockford, "The application/json media type for javascript object notation (json)," 2006. (Cited on page 51.)

[88] O. Ben-Kiki, C. Evans, and B. Ingerson, "Yaml ain't markup language (yaml) version 1.1," *yaml. org, Tech. Rep*, 2005. (Cited on page 51.)

[89] W. W. W. Consortium *et al.*, "Rdf 1.1 concepts and abstract syntax," 2014. (Cited on page 51.)

[90] J. Clark, S. DeRose *et al.*, "Xml path language (xpath) version 1.0," 1999. (Cited on page 61.)

[91] S. Imai, T. Kobayashi, K. Tokuda, T. Masuko, K. Koishida, S. Sako, and H. Zen, "Speech signal processing toolkit (sptk), version 3.3," 2009. (Cited on page 69.)

[92] G. K. Anumanchipalli, K. Prahallad, and A. W. Black, "Festvox: Tools for creation and analyses of large speech corpora," in *Workshop on Very Large Scale Phonetics Research, UPenn, Philadelphia*, 2011. (Cited on page 70.)

[93] A. Parlikar, "Testvox: Web-based framework for subjective evaluation of speech synthesis," *Opensource Software*, p. 13, 2012. (Cited on page 103.)

[94] E. Lear, *The complete nonsense book*. Dodd, Mead, 1921. (Cited on page 104.)