Establishing the nature of context in speaker vowel space normalization

by
William F. Styler IV

A thesis submitted to the Faculty of the
Graduate School of the University of Colorado
in partial fulfillment of the requirement
for the degree of

Master of the Arts

Department of Linguistics

2008

This thesis entitled:
Establishing the nature of context in speaker vowel space normalization
written by William F. Styler IV
has been approved for the Department of Linguistics

____
Dr. Rebecca Scarborough

____
Dr. Bhuvana Narasimhan

Date_____________

The final copy of this thesis has been examined by the signatories, and we
Find that both the content and the form meet acceptable presentation standards
Of scholarly work in the above mentioned discipline.

HRC protocol # 0108.6

Styler, William F. IV (MA, Linguistics)
Establishing the nature of context in speaker vowel space normalization
Thesis directed by Dr. Rebecca Scarborough

This study was designed to gain insight into the process through which humans are able to adjust to and understand the speech of unfamiliar speakers, referred to a "speaker normalization". Prior research has suggested that, in order for normalization to occur, a listener has to have some speech data (or "context") to process. The goal of this work was to further elucidate the role of this context, by searching for any effects that frequency of token occurrence and ordinal primacy may have on normalization

Comparison of different stimuli in a forced-choice vowel identification task yielded no statistically significant differences, and even after an analysis of sources of error, neither a primacy effect nor a frequency effect was found to be supported by the data. This lack of support raises many interesting theoretical questions and suggests a variety of future avenues of exploration in the field of speaker normalization research.

Table of Contents

Appendices

List of tables

Even when two people speak the same language, the sounds that they produce in the process can vary greatly because every voice is unique. This uniqueness can stem from vocal tract shape and length differences, from differences in the way we produce sounds and other articulatory habits, from differences in the base pitch and pitch range of the voice, and from other physiological factors that may be congenital or acquired.

Not all of this variation comes from set characteristics, though. A speaker with any sort of upper respiratory illness or laryngitis will sound different than the same speaker would in good health, and any sort of stress or injury to any part of the vocal tract can affect speech drastically. Even throughout the course of the day, a person's speech will vary due to tiredness and wear from frequent talking.

When we add to these factors the sorts of changes that can be caused by missed articulations and speech errors, we see that there's no shortage of unconscious variation even among various utterances made by the same speaker. Once sociolinguistic, linguistic and dialectal variation factors are added in, our first interaction with an unfamiliar speaker is seen to be a formidable linguistic challenge. Although we never consciously notice the process, every time we're confronted with a new speaker, we're asked to very quickly recognize, interpret, and adjust to the fine distinctions between sounds of our language as produced by a unique set of articulations of an unfamiliar vocal tract, all in addition to the already complex process of comprehending speech and deriving meaning from an acoustic signal.

This process of adjustment to an unfamiliar voice, known as speaker normalization (or talker normalization), has been studied by linguists and psychologists for many years, from a variety of different angles. The difference in absolute vowel qualities among speakers was first noticed by Martin Joos as early as 1948 (Joos, 1948), but Ladefoged and Broadbent's landmark study (Ladefoged &

Broadbent, 1957) was one of the first to actually examine the perceptual effects caused by the variation of vowels between speakers. Through the presentation of an acoustically manipulated context, they first observed that context and prior knowledge of a speaker's voice affects our perception of words, and laid the groundwork for much of the continuing work in the field of speaker normalization.

In addition to providing the first concrete evidence that a normalization-like effect is present in speech processing, the fact that Ladefoged and Broadbent's methodology produced results also shows that context is very much relevant in speech processing. Knowing that context is important, the next logical question is to ask which sounds provide the 'best' information for coping with these variations. Early theories suggested that the point vowels (i, a, u) played a more significant role in speaker normalization (Joos, 1948); however, experiments by Verbrugge et. al. (Verbrugge, Strange, Shankweiler, & Edman, 1976) found that point vowels do not have a more significant role in speaker normalization than, for example, central vowels. Other hypotheses about the roles of particular sounds in context have been proposed and tested, ranging from the role of F3 in providing vocal tract length information (Nordström and Lindblom 1975) to a role for breath sounds in vowel normalization (Whalen & Sheffert, 1997), but no certain conclusions have been reached.

So, we see that great deal of work has been done to establish that we depend on context for speaker normalization, and that some have considered which specific sounds might aid in the normalization process. However, few have examined the role of the positioning and frequency of the calibrating sound in the context provided. In examining normalization to tone in Cantonese, Ciocca et al suggested that both preceding and following context had an effect on listener perception, but even more interestingly, suggested that listeners weight recent context more heavily than older context (Ciocca, Wong, Leung, & Chu, 2006). Beyond this study, though, I have found no information on the nature of the context required for speaker normalization, and the role of frequency and position seem to be largely unexplored.

This seems to be a significant oversight, because in typical interaction with an unfamiliar speaker, it's very uncommon to have only a single word or vowel on which to base one's normalization and form a model. More commonly, we are presented with a phrase or sentence containing a variety of different sounds and vowels. In many cases, a given sentence or utterance might contain multiple tokens of the same sounds, which given the variability of speech even in a single utterance, may not be acoustically identical.

The fact that we can still normalize to speech even in the presence of this variability is astounding, and the mechanism by which this occurs deserves consideration. In a situation where a person is presented with acoustically varying vowels in a sentence from an unfamiliar speaker, there are several different strategies that could conceivably be used to normalize to an unfamiliar vowel space (and then possibly refine any model that may be created).

If the data from a single vowel were sufficient to properly normalize to an unfamiliar speaker's voice (as Verbrugge et. al. suggest), then conceivably, vowel-space normalization could be based solely on the first vowel to which a listener is exposed, and further context might be unnecessary or ignored. This hypothesis (henceforth referred to as 'the primacy hypothesis') would assume that vowel-space normalization is a finite process, and that it takes place only for a short period of time when we first encounter an unfamiliar speaker. Under this account of normalization, a listener hears a single example of a given vowel, analyzes it, forms a model of the speaker's vowel space (although I'm here remaining neutral about which form that model may take), and is then equipped to understand the unfamiliar speaker's speech.

However, given the variations often present between different tokens of the same vowel produced by the same speaker, such a one-time normalization would seem somewhat unlikely. If a listener normalized, once and for all, based on the first token of a single vowel, and that vowel were somehow unusual either due to coarticulation, a mispronunciation, or just a slip of the tongue, the listener would be at an inherent disadvantage for the rest of their communication with the speaker.

In order to avoid this, one might expect a listener to normalize over a longer period of time, also paying attention to vowel tokens beyond the very first encountered. This way, the effects of an unusual first usage on the listener's model would be minimized by the tide of more or less consistent, more representative usages. This would assume that, as Ciocca et. al. (2006) have argued for tone, the listener keeps a sort of "running average" of the speaker's vowels, and every token influences that idealized, "canonical" vowel. Under such an account (henceforth 'the frequency hypothesis'), the frequency with which the different vowel qualities occur would play a large role in a listener's normalization to a given vowel phoneme.

In addition, it's possible that both primacy and frequency play a role in speaker normalization, and that both factors are weighed both alongside and against one another when a given vowel is being processed. This combined effect hypothesis posits a much more involved algorithm, a series of criteria and normalization methodologies applied to any given sound, which exchanges some simplicity for greater flexibility.

Historically, studies of normalization have considered the quality of a given vowel to be both unique to each speaker and consistent in their speech. Because vowel quality can vary greatly between multiple tokens in the speech of the same speaker, and as such, even a lengthy context will seldom be perfectly consistent, the process by which a listener normalizes when faced with inconsistency is important to the study of speaker normalization. By studying the normalization process when the vowels in the context given are not of consistent quality, I hope to gain insight into the process by which we adjust to inconsistent speech, as well as into the normalization process in general.

I – Experimental Design background

This experiment is designed to investigate the previously discussed mechanisms of speaker normalization by examining the effects, if any, that the frequency and position of a given acoustic realization of a vowel will have on speaker normalization. My goal is to do this by manipulating the formant pattern of one or more instances of the vowel /i/ in a context sentence, and then measuring listeners' reaction times in a simple word identification task using target words which include the manipulated vowel.

During the study, listeners will be asked to listen to a series of 40 strategically altered context sentences, each followed directly by either "beet" or "bit". After each sentence, the listener will indicate (using a button box) which of the two words that they heard, and their reaction time will be measured. There will be twenty test sentences, followed by a "beet", interspersed with twenty filler sentences (with "bit") to keep listeners interested in the task. The reaction times from the twenty test sentences will be then compared to gauge difficulty in normalization.

One underlying assumption of this methodology is that processing speech from an unfamiliar speaker requires extra processing time, and that speakers will require more time to identify a vowel that does not fit the expectations created by normalization than they would for a vowel that does. For example, if a listener is presented with five consistent /i/ phonemes of a given quality and then asked to identify an /i/ phoneme of differing quality, we would expect that they would take longer to do so than had they been given five /i/ phonemes of consistent quality and asked to identify an /i/ of the same quality. Prior research by Haggard & Summerfield, 1977 supports these assumptions:

> *"For pairs of voices having different average formant frequencies, and hence involving a perceptual adjustment to a different vocal tract size, there was a substantial increase in RT [reaction time] on trials when the voice changed but the response did not. Such increments did not occur for voice differences such as pitch, even though these were perceptually salient."*
>
> (pp. 261)

Based on their data, reaction time does seem to be a valid measure of the presence (and extent) of normalization happening in a listener's mind. Therefore, we should be able to assess, based on a comparison of listeners' reaction times in identifying a vowel in a series of altered contexts, which elements of a context are most important to the normalization process.

If the primacy hypothesis is accurate and normalization occurs based on the first token of a vowel that a listener is exposed to, listeners will more quickly identify a target vowel if the first instance of the vowel in the context is the same in quality.

However, if the frequency hypothesis is accurate, and the method of normalization relies on the frequency of occurrence of a given acoustic realization of a vowel in the context sentence, the listener should more quickly identify a vowel with a given acoustic quality if there were more vowels with that same quality than differing vowels in the sentence. In addition, the listener should be quicker to respond as the frequency of vowels acoustically similar to the target vowel goes up.

Any combined effect would manifest as an increased speed of identification when a listener is presented with a context where the altered vowels are both more frequent than unaltered vowels *and* are the first tokens encountered. With both methods working in tandem and in agreement, one might expect reaction speed to be even faster in such a situation than in situations where frequency or primacy alone are present.

II – Methodology

2.1 – Recording speech samples for processing

Rather than attempting to sort through a large spoken corpus to find examples of vowel inconsistency, I chose to create inconsistency in recorded sentences using the source-filter resynthesis techniques found in the Praat phonetics software suite.

Twelve speakers were recruited from within the University of Colorado Linguistics department. These participants were all native speakers of American English, and the participants were not screened or selected for dialectal variations.

Once the procedure had been explained, the speaker was recorded in a sound-attenuated area. All recordings were captured directly to a hard drive, using a Shure head-worn microphone. Speakers were asked to recite the following list of sentences twice:

Table 2.1: Stimulus Sentences

1. Steve saw the keys on the table, but the bee can't see me.
2. Rita said that these slopes are easy for skiiers.
3. Please don't upset the peace, we have plenty of crazies here.
4. Peas are free next week, but John already has plenty.
5. Geese slowly waddled between the neon lamps, resplendent for all to see.
6. Lisa sees twenty people a week.
7. Deep seas host many funny creatures.
8. Steve can't see the movie over the lady's antique hat.
9. Skee-ball leads to many beachfront tragedies.

10. "Seeds can be very yummy", mused Ashley.

The word is beet.

The word is bit.

I hear the beat.

He's chomping at the bit.

These sentences were chosen because they (in canonical, GA English) each contain exactly five examples of our target vowel, the /i/ phoneme, and no examples of /ɪ/ (outside of diphthongs). In addition, the first words were chosen because they were stressed and because there was no confusable English word with /ɪ/ in place of /i/ (e.g. 'Jean' and 'gin'). Sentences were broken into individual .wav files for future processing.

Once all the speakers were recorded, two speakers were selected out, and their speech was marked for use only in filler sentences.

2.2 – Stimulus Preparation

2.2.1 – Creating inconsistency using source-filter resynthesis

In order to gain useful information about our normalization process, very specific patterns of inconsistency must be created. In this experiment, these patterns were created by altering the formant patterns of some tokens of the /i/ phoneme in the context sentences. There are five patterns in which the /i/ tokens in the context sentence were altered, and these patterns are explained in the next section.

In order to create this controlled inconsistency in pronunciation (above and beyond any natural inconsistency), the source-filter resynthesis (SFR) features built into the Praat phonetics software were used to alter the formant structure of certain vowels. To do this, first, the sentence was down-sampled to 11000hz, and the vowel was removed from its sentence context. Once it was alone, Praat's LPC function was used to generate an LPC object for the vowel, which was then inverse filtered with the original to isolate the pure voicing, and a filter (formant object) was created. Vowel tokens that were not altered were reanalyzed without changing their formant structures. In this case, the source and filter were then combined, yielding a resynthesized vowel nearly identical in quality to the original. If the vowel was to be altered, the frequencies of the isolated formants were modified as described below, using Praat's built in tool, and then the source and modified filter were combined, yielding a modified version of the original vowel. Finally, the vowel was spliced back into the sentence in the exact location from whence it was removed. For a precise description of the steps taken, please examine the annotated copy of the script used, which is included here as Appendix I.

For the purposes of this experiment, the vowels were altered by lowering both F2 and F3 by 300hz each. This specific amount was chosen for several reasons. First, 300hz seems to be the largest alteration which can be performed without significantly affecting the perceived voice quality of the resynthesized version. Beyond this point, the alteration becomes very obvious, and results in an intensely artificial sounding vowel. Also, because the goal is to introduce some ambiguity between the /i/ and /ɪ/ phonemes, this change creates an /i/ that is far closer to the /ɪ/ phoneme for most English speakers. Finally, this amount of variation could conceivably result from articulatory variation. It's not unrealistic to think that periodically, a speaker might produce an /i/ with significantly lower F2 and F3 values due to mis-speech or due to an otherwise unusual articulation. However, for a speaker to suddenly raise the F2 and F3 of the /i/ phoneme, a point vowel, to this high of a degree would require the speaker to burst through the edges of their vowel space (if not their vocal tract), and would be a very unlikely sort of inconsistency.

To make the finished stimuli sound more "natural" and eliminate some of the high frequency "popping" associated with source-filter resynthesis, I implemented a pass-merge step in the stimulus preparation. During this step, the bottom 6000hz (0-6000hz) of the reanalyzed stimulus sentence are merged with the top 16,050hz (6000hz-22050hz) of the original sentence, resulting in a single, hybrid file which is then used as the final stimulus. Because all of the relevant formants (F1, F2 and F3) are well below 6000hz, this doesn't affect the quality of the altered vowels, and very neatly removes or minimizes many of the high frequency pops and artifacts which could easily distract the listener. For more detail, please see the attached annotated Praat script in Appendix I.

Post-hoc analysis revealed that because of the imperfect nature of Praat's formant-finding and source-filter resynthesis process, in practice, the average formant heights are not moved precisely 300hz in every vowel. In the final stimulus set, F2 was lowered by 264hz, on average, and F3 was lowered by 277hz, on average. Although there were notable exceptions (see section 4.1.1), in this study, the resynthesis process fairly consistently produced inconsistency.

It is worth noting that vowel length was not altered in any way. Although there is a vowel length contrast in American speech between /i/ and /ɪ/ (Rositzke, 1939), eliminating this contrast would only serve to introduce another variable and another barrier to perception. Even though this length contrast may provide some information to speakers in the actual identification task, the nature of the experiment (in which identification accuracy is secondary to reaction time) makes this extra available information irrelevant to the study of the hypotheses.

2.2.2 – The nature of this inconsistency

As expected, even within the individual source sentences, speakers exhibited variation in formant structure from token to token of /i/. Although F1, F2 and F3 values for individual vowels generally stayed within a range of around 300hz from the mean /i/ formant values in each sentence, there were some examples of variation up to 700hz from the mean. This is not unexpected (and in fact, if such variation didn't occur, this study would be without purpose).

It's important, though, to point out that this variation will likely not interfere with or "cancel out" the alterations created above. This study is not trying to contrast two specific acoustic vowels or formant patterns, but instead, to contrast the acoustic effects of two different vocal tracts or means of speaking. By consistently changing the formant patterns of certain vowels, regardless of their starting values, this should set up a contrast between vowels with F2 and F3 at the speaker's natural baseline, and vowels with an F2/F3 baseline around 300hz lower. All of the variations present in normal articulation will still be there, but it will be, in many ways, as if the speaker is switching back and forth between two contrasting vocal tracts with slightly different patterns of resonance.

In this way, natural variations in pronunciation will fade somewhat into the background, and the variation created artificially should be consistently contrastive, regardless of the specific nature of the vowels being modified.

2.2.3 – Patterns of Inconsistency

In order to test for specific patterns of normalization, the context sentences were altered in five different patterns of altered and unaltered (but reanalyzed) vowels in the context sentence. All tokens of the /i/ vowel were resynthesized in some form, either with or without a vowel quality change. Because the reanalysis alone does change the nature of the vowel slightly, and because the goal is to make vowel quality the only contrast, reanalysis of all tokens eliminated any secondary contrast between reanalyzed and untouched tokens. Throughout all of these patterns, only tokens of /i/ were modified or reanalyzed, with all other vowels left as the speaker pronounced them. A total of 20 test stimuli were prepared, including four examples of all five alteration patterns.

(Note that in the following pattern descriptions, "A" refers to an /i/ token where the formant structure has been altered, and "U" refers to an unaltered token. The final vowel, always altered, is the target.)

Condition A: First token altered (AUUUU A)
Condition B: Second token Altered (UAUUU A)

Conditions A and B are designed to test the hypothesis that speakers normalize based on the first token that they hear (the primacy hypothesis). Stimuli prepared to Condition A have only the first vowel and target vowels altered, and all others simply reanalyzed. Condition B is designed to contrast with A solely based on primacy (as the frequency of occurrence of altered tokens is identical), and features an altered second vowel and target vowel.

In analysis, if Condition A is faster than Condition B, the primacy hypothesis will be supported.

Condition C: Three altered (UAAAU A)
Condition D: Two altered (UAUAU A)

If frequency of occurrence is a key component of normalization, listeners should find it easier to identify a given acoustic vowel which matches the more commonly presented vowel. Conditions C and D are designed to test whether this is, the case. C and D contrast solely based on frequency of altered token occurrence, where C as 3 out of 5 tokens altered, and D has only 2 out of 5. The first token has been left unaltered in both of these conditions to avoid interference from any primacy effect.

In analysis, if Condition C is faster than Condition D, the frequency hypothesis will be supported, and comparison with other conditions will provide supplementary data.

Condition E: First and Frequent (AUUAA A)

Because Condition E has both a high frequency of occurrence (3/5) and an altered first token, it should be sensitive to both primacy effects and frequency of occurrence effects, and could serve as a secondary example of either, in case either effect is demonstrated. However, more importantly, if both hypotheses show merit, Condition E will help to show whether or not there's any combined effect.

In analysis, Condition E contrasts with both C and A. If Condition E is faster than both A and C, it will strongly support a combined effect.

2.2.4 – Implementing these patterns

As an illustration of the implementation of each pattern, we'll examine the following stimulus sentence (transcribed broadly below):

Steve can't see the movie over the lady's antique hat. Beet.

/ stiv kænt si ðə muvi oʊvɹ̩ ðə lɛɪdiz æntik hæt. bit /

If the sentence were used as a stimulus and prepared according to the conditions above, both F2 and F3 would be lowered by 300hz in the bold-italic vowels:

Condition A:

/ st***i***v kænt si ðə muvi oʊvɹ̩ ðə lɛɪdiz æntik hæt. b***i***t /

Condition B:

/ stiv kænt s***i*** ðə muvi oʊvɹ̩ ðə lɛɪdiz æntik hæt. b***i***t /

Condition C:

/ st***i***v kænt si ðə muv**i** oʊvɹ̩ ðə lɛɪdiz ænt**i**k hæt. b***i***t /

Condition D:

/ stiv kænt s***i*** ðə muvi oʊvɹ̩ ðə lɛɪd***i***z æntik hæt. b***i***t /

Condition E:

/ st***i***v kænt si ðə muvi oʊvɹ̩ ðə lɛɪd***i***z ænt***i***k hæt. b***i***t /

2.2.5 – The Stimulus Set

To increase the number of observations for each condition, each condition was tested a total of 4 times per listener, resulting in 20 test stimuli. Because of the nature of the experiment, the only target word is an altered 'beet', so, in the interest of maintaining listener attention and keeping the premise of this being a 'vowel identification' exercise, 20 'filler' stimuli were mixed in with the overall stimulus set. These filler sentences were prepared using the same conditions as the test stimuli, but instead of being followed by an altered 'beet', were followed by either an altered or unaltered 'bit' (ten altered, ten unaltered, distributed randomly). Additionally, the presence of the fillers made the listeners feel that they had to attend to the stimuli in order to properly identify the vowels. When filler and test stimuli are combined, the stimulus set is comprised of 40 sentences total.

2.2.6 – Ordering the stimuli

Because this experiment relies heavily on the novelty of a speaker's voice and vowel space, special considerations were taken to ensure that no biasing context was developed. Three rules were observed when creating a psuedo-random order for the stimuli:

1) No one speaker may be used more than twice to create test stimuli

2) Two stimuli featuring the same speaker may not occur next to one another

3) No speaker may be used for filler before having been used for test stimuli

The first two rules were designed to prevent a listener from developing a strong memory of the voice or vowel space of a given speaker. Given some degree of separation between the speaker's two stimuli (as well as the various other speakers in the interim), it seems unlikely that any listener would remember the specifics of a speaker's voice across the two test instances.

After a speaker has been used twice for test stimuli, that speaker's recordings can continue to provide filler stimuli without affecting the experiment. For filler stimuli, it's not relevant whether or not the listener remembers the speaker's voice, as the data was not analyzed, regardless.

Throughout this experiment, steps were taken to ensure that a listener didn't become familiar with the speaker's voice. As such, no exposure to a given speaker's voice was squandered. The stimulus ordering was designed to avoid using a still-unfamiliar speaker as filler, establishing a possible context without good reason to do so. The only exceptions to this were the two filler speakers, whose sentences were used only as filler.

With a pool of 10 test speakers and two filler speakers, an ordered list of speakers and stimuli was created in such a way that all of these rules were met. In practice, the data was arranged in such a way that at least three stimuli featuring other speakers occurred between test stimuli featuring the same speaker. See Appendix II for the final arrangement of speakers, alternations, and sentences in the 40 stimuli.

2.3 – Procedure

2.3.1 – Experimental Design

The experiment itself was designed using PsyScope X, and was designed to use an ioLab Response Box for user interaction and accurate response time measurement.

The experiment script was made up of two sections. First, in the practice and orientation section, the user was presented with a series of instructional screens (see Appendix III), explaining the task and orienting them to the use of the button box.

Once the orientation was complete, the listener then went through three practice trials (identical to the test trials described below).

Stimuli for the practice trials came from consented speakers not included in the main stimulus set, and were selected to provide a good exposure to the approaching trials. Conditions C, A, and B were chosen to give a good overview of the different sorts of alterations being made, and there were both "bit" and "beet" trials. In addition, sentences 3, 4 and 9 were chosen because, in recording, speakers remarked that they found these sentences humorous, and hopefully, this will allow participants to get any laughing or giggling out of the way in the practice trials before reaction times were being measured.

Once the practice trials ended, a ready screen was called up, giving the listener a chance to pause before the test trials began, instructing them to press either button to begin the formal trials. Once the listener opted to continue, a screen came up with "bit" and "beet", color coded and labeled according to their color and orientation on the button box, and the stimulus played through the listener's headphones.

Reaction timing was configured to begin at the start of the sound file, such that a response could be measured as soon as the listener felt that they could accurately respond (rather than forcing them to wait for a new screen). In order to obtain accurate reaction times, the total time between the start of each file and the start of the target vowel (defined here as the first pulse after the release of the /b/ closure) was measured. Then, these numbers are simply subtracted from the total reaction times recorded by PsyScope to find the precise reaction time from the start of the vowel.

Once a response was registered, the listener was presented with another ready screen, as above, and the process was repeated for all 40 stimuli.

2.3.2 – Listeners

Listeners for the experiment were paid undergraduate volunteers from the University of Colorado community, recruited through word of mouth as well as through posters around the CU Campus. A total of 22 listeners participated in the experiment, over the course of three weeks.

During the scheduling and correspondence process, potential participants were questioned to make sure that they were native English speakers, had not had any formal training in audiology, phonetics or speech science, and had not been diagnosed with any sort of hearing disorder. These three criteria are relevant to eliminate people who aren't familiar enough with the language to note the distinction, those who might have a honed sense of vowel perception due to past training, and those who might be physically unable to hear the contrast.

2.3.3 – Carrying out the experiment

Once a listener was in the lab and had signed the relevant consent forms they were seated in front of the computer. The usage of the button box was demonstrated, and they were asked to choose a single method of button pushing (using thumbs, index fingers, etc) and then use it consistently throughout the experiment.

Once they had been briefed, the listener ran through the experiment, as described above, with the stimuli played over Audio Technica ATH-M40fs headphones. Once the listener finished and it was verified that data had indeed been collected, the listener was compensated, any questions were answered, and they were dismissed.

III – Analysis

3.1 – Analytical methodology and expected effects

In the 40 total trials, four examples of each condition were tested with each listener. In order to allow a by-subject analysis, the reaction times for each condition were averaged for each listener. This helps to compensate for individual aberrations within conditions, and allows the capability to conduct a series of paired t-Test analyses in order to gauge the significance of any findings.

This series of t-Tests comparing the different conditions was the principal means of statistical analysis, and their results were used to support or reject the three hypotheses presented, as follows:

3.1.1 – Supporting the Primacy hypothesis

The primacy hypothesis states that we normalize solely based on the first token of a given vowel that we hear.

In order to support this hypothesis in its strongest form, the average reaction times between Condition A and Condition B will need to be compared. If the average reaction times for A (AUUUU A) are confirmed to be significantly greater than those of B (UAUUU A), then the first token hypothesis will be supported.

In addition, for full support of the strongest form of this hypothesis, reaction times for Condition E (AUAAA A) must be significantly faster than Condition C (UAAAU A), because those two also contrast solely based on primacy.

3.1.2 – Supporting the Frequency hypothesis

The frequency hypothesis states that we normalize based on frequency of occurrence of differing vowel qualities, and therefore, we will recognize a more frequently occurring form more easily than one with lower frequency of occurrence.

This hypothesis was examined by comparing the difference in reaction times between C and D. If reaction times for Condition C are found to be significantly faster than those for Condition D, the Frequency Hypothesis was considered to be supported.

In addition, if frequency is the sole relevant effect, all of the Conditions should fall nicely into the following hierarchy of reaction speed, from fastest to slowest, based on the frequency of occurrence of altered vowels:

Fastest
Conditions C/E: 3 altered vowels
Condition D: 2 altered vowels
Condition A/B: 1 altered vowel
Slowest

3.1.3 – Supporting a combined effect

If both a frequency effect as well as a primacy effect manifest themselves in the data, both may simultaneously contribute to our understanding, and play a combined role in speaker normalization.

If any sort of combined effect is present, then Condition E (which has both primacy and a high frequency of occurrence) should be the easiest to process of any of the conditions. If this is the case, t-Test analyses should show Condition E to be significantly faster than all other Conditions.

This outcome would suggest that these means of normalization are working in parallel, combining their outputs into our final capability to understand the speaker's voice.

3.2 – Data processing

3.2.1 – Initial Data processing

Because response times were measured from the start of the sound file, actual response times had to be determined by subtracting the length of the sentence before the start of the target vowel from the total measured reaction time.   After this step, all filler sentences were separated out and excluded from further analysis.

Then, discretion was used to remove any data which seemed obviously unreliable. The sole example of this was the exclusion of all data from participant six. Upon analysis, she displayed exceptionally fast reaction times, most < 300ms, compared to an overall mean of 610ms. In addition, she also made responses with some negative reaction times (where her response came before the target word).   She also had very low accuracy (85%) compared to all other speakers (who had more 99% accuracy overall). Given that this participant had remarked at the time of the study that she had "found the pattern" and claimed to be able to predict the target vowel based on the sentence, it was fairly clear that for many (if not all) of the trials, she was not actually performing a vowel identification task. Based on all these factors, her data was deemed unreliable and excluded from the final analysis.

Then, in order to minimize the effects of outliers on my final data, the average of each participant's response times was taken, and any reaction times that were faster or slower than two standard deviations from that average were thrown out.   This had the effect of getting rid of obvious outliers (e.g, 2500ms response times when all others were in the 600ms-1000ms range), but generally removed only one or two data points per speaker.

Finally, in keeping with standard practice in reaction time data analysis, reaction time data from incorrect responses was thrown out.

A copy of the processed and unprocessed data used here is attached as Appendix

IV.

3.2.2 – Per-Subject Analysis

Once the data had been processed and prepared, the data was separated by listener and condition. Because some data had been tossed in the above steps, paired t-Tests were no longer possible comparing all four trials of each condition, so listener's responses to each condition were averaged together. The end result was a table of average reaction times, arranged by listener:

Table 3.1: Sample per-listener-per-condition averages (in ms)

	1	**2**	**3**	**4**	**...**	**21**	**22**
A	589.67	752.33	640.00	623.75	...	519.50	797.25
B	663.75	735.67	730.50	610.25	...	519.25	882.50
C	646.25	834.50	697.00	496.50	...	445.25	772.67
D	593.75	854.75	625.25	565.00	...	399.75	642.75
E	604.75	702.75	700.00	595.00	...	651.00	748.25

Once this table was compiled, a series of paired t-Tests was run on the data using the R Statistical Computing Environment. t-Tests were performed comparing all averages of Condition A to all averages of Condition B, all averages of C to D, all of A to E, and all of C to E, and their results were compared.

3.3 – Findings

3.3.1 – Accuracy and overall means

Overall, response accuracy was extremely high. In test stimuli, only four incorrect answers were given (out of 420 total test stimuli administered), yielding an accuracy rate higher than 99%. Inaccurate responses did not occur more often with any particular condition, sentence, or speaker.

The mean response time to all test stimuli was 631.1ms.

3.3.2 – Per-Subject means and t-Tests

The means of all per-listener-per-condition averages are shown in Table 3.2, below:

Table 3.2: Average response times for each condition (in ms)

A	B	C	D	E
630.5	640.6	622.1	613.3	647.1

Table 3.3: Paired t-Test results (df = 20 for all pairings)

Pairing	A and B	C and D	A and E	C and E	D and E
Mean Difference (in ms)	-10.05	8.86	-16.61	-24.98	-33.84
p-value (t-stat)	0.6245 (-0.4971)	0.6929 (0.4006)	0.5453 (-0.6153)	0.2503 (-1.184)	0.1747 (-1.402)

Although the means did obviously display some variation, t-Tests indicated that none of these variations were statistically significant.

3.4 – Hypothesis discussion

3.4.1 – The Primacy Hypothesis

As stated above, in order for the primacy hypothesis to be supported, Condition A had to be faster than Condition B, and Condition E had to be faster than Condition C.

The mean response time for A was, in fact, faster than B, by a 10ms margin. However, this varied greatly from speaker to speaker (see Appendix IV), and even more importantly, a paired t-Test comparing the two yielded a p-value of 0.62. In addition, means for Condition C were markedly *faster* than those for Condition E, with a p-value of 0.25.

Finally, D is faster than E by a 33ms margin, with a p-value of 0.17. This casts strong doubt on the strongest form of the Primacy hypothesis, because E (with an initial altered) should be faster than D if such a hypothesis holds.

Given that neither of the outlined criteria was statistically satisfied, the Primacy hypothesis is not supported in this data.

3.4.2 – The Frequency Hypothesis

Earlier, two criteria were stated which must be satisfied to support the Frequency Hypothesis. The first stated that Condition C had to be faster than Condition D, and the second predicted the following distribution:

Fastest

Conditions C/E: 3 altered vowels

Condition D: 2 altered vowels

Condition A/B: 1 altered vowel

Slowest

In the data, however, neither of these criteria were met. Condition D was actually slightly faster (~9ms) than Condition C (albeit without statistical significance), and once again, individual listeners varied greatly in their mean response times. In addition, the per-condition reaction times did not follow the predicted distribution, and instead were arranged as follows:

Fastest

Condition D: 2 altered vowels (613ms)

Condition C: 3 altered vowels (622ms)

Condition A: 1 altered vowel (630ms)

Condition B: 1 altered vowel (640ms)

Condition E: 3 altered vowels (647ms)

Slowest

Finally, the fact that D is faster than E by a 33ms margin (with $p = 0.17$) casts strong doubt on the idea that frequency is the sole criterion for normalization.

Once again, neither of the criteria needed to support the Frequency hypothesis was satisfied, and therefore, we have no choice but to consider the frequency hypothesis unsupported in this data.

3.4.3 – The Combined Hypothesis

Given that neither of the two component hypotheses were supported in this data, the combined hypothesis logically cannot be supported. In addition, the fact that Condition E was the slowest measured condition means that all criteria put forth for supporting it were not met.

IV – Potential sources of Error

As stated above, none of the initially proposed hypotheses were supported in the data collected, and even more importantly, the data was frustratingly inconsistent, leading to high p-values in all analyses. I'd like to spend some time more closely examining the methodology and stimuli in an attempt to account for this variability, and to search out any sources of bias or confusion that may be masking any underlying results.

4.1 – Stimulus variability

4.1.1 – Difficulties of Source-Filter Resynthesis in Experimental Methodology

As mentioned in section 2.2.1, because of the imperfect nature of the formant-finding algorithms used, Source-Filter Resynthesis ('SFR') inherently produces varied results. Depending on factors such as speaker, context, and background noise, different vowels will be modified with more and less success. This variation was one of the primary suspects which arose when the data returned was fairly inconsistent. In an attempt to examine this variability, average formant measurements were taken for each vowel which was touched by the reanalysis script, both in the unmodified, raw test stimuli and in the resynthesized, prepared test stimuli.

As previously mentioned, the average lowering of F2 and F3 in resynthesized, altered vowels was 268hz and 279hz, respectively. This amount of change is very perceptible, and represents an acceptable amount of overall change. Interestingly, though, simple reanalysis (SFR without specified formant change) occasionally caused a raise in formants, generally less than 200hz. Although it should be taken as a cautionary sign about the variability of SFR, in this study, it is not a complicating factor as it would have only served to heighten the contrast between unaltered and altered vowels.

In general, most tokens were altered acceptably. As an example, the average F2 and F3 measurements before and after SFR for stimulus 16 are shown in Table 4.1 below:

(See next page)

Table 4.1: An example of an acceptable SFR (Stimulus 16, Condition D, in hz)

Vowel	F2 before	F3 before	F2 after	F3 after	F2 drop	F3 drop
/i/ 1	2223	2660	2228	2693	-5	-32
/i/ 2 *Altered*	2176	2859	1913	2601	264	258
/i/ 3	2166	2724	2230	2704	-65	19
/i/ 4 *Altered*	2407	2760	2150	2451	257	308
/i/ 5	2290	2768	2266	2747	24	21
Target /i/ *Altered*	2527	2833	2227	2512	300	321

However, there were several individual tokens whose formant patterns were badly altered (or, in some cases, altered when they shouldn't have been). For instance, take the F2 and F3 measurements of stimulus 34, shown below in Table 4.2:

Table 4.2: An example of unacceptable SFR variability (Stimulus 34, Condition D, in hz)

Vowel	F2 before	F3 before	F2 after	F3 after	F2 drop	F3 drop
/i/ 1	2241	2911	2436	2991	-195	-80
/i/ 2 *Altered*	2026	2772	1826	2484	199	288
/i/ 3	2668	3198	2543	2950	125	248
/i/ 4 *Altered*	2399	2964	2419	3080	-20	-116
/i/ 5	2586	3074	2643	3187	-57	-113
Target /i/ *Altered*	2262	2999	2107	2715	155	284

This particular stimulus caused a fair amount of difficulty for Praat. The second vowel was altered normally, but the fourth vowel was not lowered at all, and instead, was raised by a small amount. Interestingly, the third vowel (whose formant height should not have been changed at all) seems to have experienced a major formant drop. In this particular case, the end result still has two out of five vowels altered.

However, there were other stimuli whose formant patterns were so mangled during the SFR process that they became questionable exemplars of their condition (Stimuli 12, 16, 18, and 37), either due to extremely weak alteration, or some vowels not altered where they should be. Although the majority of stimuli prepared were acceptable, in less than 10% of stimuli, alterations may have lead to confusion.

This post-hoc analysis of the stimulus set shows that although Source-Filter Resynthesis can be used to prepare stimuli effectively, those stimuli will necessarily vary, both in terms of degree of formant change and quality of formant change.

4.1.2 – Compensating for Source-Filter Resynthesis problems

As stated above, Stimuli 18 (E), 26 (D), 34 (D) and 37 (E) were all badly processed by Praat, and thus, may not have been good examples of their respective conditions.

Interestingly, though, the mean reaction times for these stimuli did not seem to follow the hypotheses as we might expect. For instance, Stimulus 37 (Condition E) was intended to have three altered vowels, but instead, only had two. If a frequency effect were present, one might expect it to be the slowest of the Condition E stimuli, but instead, it was actually faster than another which had a well-altered formant structure.

However, to avoid simple one-to-one comparisons and to better see whether data from these stimuli may be inappropriately skewing the data and masking results, the data for these stimuli was removed from the sample set (leaving only two examples each of Condition D and E) and the statistical tests were run again:

Table 4.3: Average response times for each condition without faulty stimuli (in ms)

A	B	C	D	E
630.5	640.6	622.1	617	666.6

Table 4.4: Paired t-Test results without faulty stimuli (df = 20 for all pairings)

Pairing	A and B	C and D	A and E	C and E	D and E
Mean Difference (in ms)	-10.05	5.15	-36.07	-44.44	-49.5
p-value (t-stat)	0.6245 (-0.4971)	0.8844 (0.1473)	0.4038 (-0.8529)	0.2369 (-1.2194)	0.059 (-2.0023)

When these (potentially) faulty stimuli are removed, p-values do drop, especially when comparing D and E. However, the amount of total data (and thus, the significance) is lessened here because the per-listener-per-condition averages for both D and E are based here on half of the data that they previously were.

4.2 – Per-item effects

4.2.1 – Problems arising from the use of a single stimulus set/order

Because of the complexity of arranging and creating stimuli (and due to the restricted timeframe in which work took place), a conscious decision was made early in this experiment to use only one set of stimuli ordered in one specific manner for all trials.

Although this was perhaps a necessary evil, this single order and single stimulus set could easily have compounded any variability between individual tokens. Although steps were taken to compensate for this through the careful arrangement of stimuli, the use of multiple stimulus sets could have made the data collected more resistant to error and per-item bias.

4.2.2 – Examining the possibility of per-item effects

In order to check for per-item effects a per-item analysis was performed on the test stimuli, examining the reaction times of each individual trial (sentence-target pair). Although not necessarily significant to the analysis of the hypotheses, if there were to be a single trial that stood out either for an exceptionally fast or slow average reaction time, this would require further investigation and might have a bearing on the final analysis and interpretation of the data.

Examining the mean reaction times for the 20 test stimuli, they were found to be fairly closely clustered around the mean (631ms), with a standard deviation of only 74ms. (See Appendix IV for a complete listing of the item averages)

Only one stimulus fell outside of 2 standard deviations, Stimulus 2 (Condition B, with an average RT of 791ms). Given that the formant values seem to have been properly processed by the SFR script, and the other stimuli with similar characteristics (speaker two, sentence two, condition B) did not exhibit similar slow RTs, this seems to be an anomaly rather than a representative of a more widespread pattern of frequency or primacy.

Overall, individual items did not seem to vary predictably, and this does not seem to be a likely source of confusion or bias.

4.3 – Re-examination of the hypotheses

As discussed in the prior section, when viewed in hindsight, there are several factors which could have been better controlled in the initial setup and implementation of this experiment, and as discussed above, could all have in some way led to a bias or confusion of the results. Although some of these factors (like stimulus ordering or listener misunderstanding) cannot be compensated for, others can, like the most salient source of error, the varied output of the Source-Filter Resynthesis process.

Fascinatingly, even once these factors are compensated for by removing questionable stimuli and performing a per-item analysis, there is still no support for either the frequency or primacy hypotheses.

V – Discussion

The fact is that even when the data is reconsidered in light of potentially confusing and biasing factors, no patterns emerge which might seem to support these hypotheses. Response accuracy suggests that despite the measures taken here to confuse normalization (far beyond those which might occur in everyday language situations), listeners had little difficulty with the task. Although the statistical power is not sufficient to completely rule out any effect of frequency of use and primacy, based on both initial and post-hoc analysis of the data, the evidence seems to be against either primacy nor frequency of vowels in context sentences having a strong effect.

Based on the results of this study, the sole reasonable choice is to reject the both the Frequency and Primacy hypotheses, and by association, the combined hypothesis. In this section, the implications of this result will be discussed, in terms of differing theories of speaker normalization, as well as in terms of their suggestions for future studies.

5.1 – Theoretical Implications for Algorithmic Models

5.1.1 – Inherent assumptions of this methodology

The assumption throughout this experiment (and indeed, throughout much research in speaker normalization) is that our normalization mechanism consists of an algorithm (generally coupled with a model that it creates). When a listener first meets a speaker, the assumption has been that the listener sends an initial portion of the speaker's speech through some sort of algorithm. This algorithm takes this initial speech sample ('context') and uses it to create a model of the speaker's voice, which can later be referred to in order to interpret variations in speech.

The goal of this experiment has been to make the listener create an inaccurate, altered model by providing altered vowels as a part of this context. These varying altered models would lead to an increased (or decreased) difficulty when the listener is asked to identify a vowel which is ambiguous in context. This difficulty would then map to increased reaction time, which could be used to measure the effects of varying contexts.

None of the theoretical assumptions made thus far are particularly controversial. Algorithm-and-model theories of normalization are far from uncommon, and Ladefoged and Broadbent (1957) have neatly shown that we do judge vowels from an unfamiliar speaker using information from prior context. In addition, measuring normalization difficulty using reaction time is a well supported methodology (see Haggard and Summerfield 1977). All of these assumptions have stood the test of time, and therefore, seem unlikely to be the source of this study's lack of results.

5.1.2 – Normalization independence of phonemes

However, there is one assumption made in this methodology that is somewhat more controversial. Because, in this study, all alterations in context are occurring in only one vowel, /i/, we must assume that we normalize to each vowel phoneme independently, and that each vowel only provides a listener with information about itself. In practice, this means that if a listener hears the only sentence "he sees three cats" from an unfamiliar speaker, the listener will be able to get a very good idea of the nature of the speaker's /i/ phoneme (having heard it three times), and will have some idea of the nature of the speaker's /æ/ phoneme. However, the listener will have no information about the remainder of the speaker's vowel space, and, for instance, would still have to develop a model for the speaker's /u/ and /ɑ/ phonemes later in the conversation.

If the algorithm does process each vowel independently, then one would expect that altered /i/ phonemes in a context might make recognition of later altered /i/ phonemes easier, no matter what other vowels were present in the remainder of the context. However, in this study, difficulty (as measured by relative RTs) did not change significantly with any alterations of context, no matter their nature or severity.

If we continue to assume that we normalize to vowels independently of one another, this result is somewhat baffling. However, if we're willing to acknowledge the possibility that acoustic information from one vowel phoneme can provide information which helps us normalize to another vowel phoneme (interdependent vowel normalization), these results make a great deal more sense.

In sentence eight ("Steve can't see the movie over the lady's antique hat."), there are a total of thirteen vowels, five of which are /i/. In this study, when testing the frequency hypothesis, three of those five /i/ tokens were altered. That was considered to be a high frequency of occurrence, which it was, if only /i/ vowels are relevant to normalization. However, if all vowels in the context sentence were used to refine our model of a speaker's /i/ vowel, then the true frequency of altered vowels was only 3 out of 13, less than 25% of all vowels. This effect is even greater in sentence five, which has eighteen total vowels. There, the "high frequency of occurrence" of altered vowels condition only alters around 16% of all vowels.

Above and beyond vowel independence, if we allow for the possibility that non-vocalic sounds provide information used in normalization (see Whalen and Sheffert 1997), the number of unmodified segments useful for normalization goes up significantly, and the numbers of alterations made here are seen to be pitifully small compared to the useful, unaltered phonemes in the larger context.

5.1.3 – Implications

What do these results really mean for an algorithm-based theory of normalization? The lack of statistically significant results could tell us several fundamentally different stories.

The extremely high accuracy of identification despite large degrees of change tells us that if vowels normalize independently, whatever normalization algorithm is used must be extremely resilient and capable of dealing with large variations in vowel quality even with unreliable data. At the same time, this lack of results still seems to strongly indicate that neither frequency nor primacy play strong role in this process, and that some unknown method of normalization is used which was not affected by the alterations made here.

If we maintain the existence of an independent model of vowel normalization, in order to explain the lack of results, we have to posit an extraordinarily resilient model which is able to effortlessly overcome the alterations in the /i/ vowels. However, an interdependent model of vowel normalization would seem to explain the findings in a far more intuitive way.

As mentioned above, if other vowels are actually useful and relevant for the normalization of individual vowel phonemes, then the Condition C and E (high frequency of occurrence) stimuli actually demonstrated a very low ratio of altered vowels to unaltered ones. When viewed in the light of interdependent vowel normalization, the results were exactly as would be expected. Listeners paid to the most common (unaltered) vowels, and threw out the infrequent (altered) ones. Also, because the total ratio of altered to unaltered vowels is so low, it's unlikely that the small changes in normalization difficulty would show up in the data strongly enough to be meaningful. Because there were no stimuli where altered vowels outnumbered the unaltered vowels, this stimulus set is inadequate for making any determinations about the role of frequency in an algorithm which allows interdependent vowel normalization. Therefore, this data does not completely rule out a role for frequency of occurrence in an algorithmic model of speaker normalization.

However, even in the case of interdependent vowel normalization, the primacy hypothesis is unsupported by this data. Because no other vowels occurred before the initial modified /i/ tokens in the Condition A and E stimuli, the modified initial /i/ should have still skewed the listener's model, if it were based solely on the first vowel observed and disregarded future information. Unlike the frequency hypothesis, if the data in this study is, in fact, accurate, then the primacy hypothesis is unsupported, no matter which particular characteristics one attributes to the algorithm used for normalization.

Although, due to the nature of the study and stimulus set, there was no strong evidence of variable reaction time delay due to increased or decreased difficulty in speaker normalization, this theoretical ambiguity means that the lack of significant results should not be interpreted as a strike against algorithm-based normalization in general. Similarly, although the data here did not support the frequency hypothesis, the uncertainty regarding the independent and interdependent nature of vowel normalization prevents us from dismissing it outright.

5.2 – Implications for an exemplar-based normalization process

Although this study was designed to test hypotheses stemming from an algorithmic model of normalization, these results (or lack thereof) should also be examined in light of the strongest competing category of theories, the exemplar-based models of speaker normalization.

In an exemplar-based model of normalization, instead of context being used only to build an abstract model (and then being discarded), we are assumed to keep information about specific instances of sounds and words in our memory. Once this database (sometimes referred to as an "exemplar cloud" or "episodic lexicon" (Goldinger, 1997)) has been established, new sounds are compared to previously recorded tokens, and through this comparison, phoneme (or word) identification is made.

5.2.1– Validity of the hypotheses in an exemplar-based model

Perhaps the most salient question is whether these hypotheses could possibly be valid with or applied to an exemplar-based model of normalization.

In order for even the barest semblance of an exemplar based theory to apply, multiple tokens need to be stored and considered in future normalization tasks. The primacy hypothesis states explicitly that only the first token is considered in normalization, and therefore, could not co-exist with an exemplar-based model.

However, the frequency hypothesis does seem like it could be relevant for exemplar-based model as well. In an algorithmic model, frequency of occurrence effects might well take the form of a "running average" sort of calculation, where individual tokens are processed, the model is adjusted to account for the variation, and the actual token is forgotten. In an exemplar-based model, a frequency effect would instead come from the relative frequency of a given vowel quality in the episodic lexicon of the listener.

For instance, if a listener is presented with a vowel in an ambiguous context, he or she would immediately begin searching through prior tokens to find a match. If a matching (or very similar) vowel quality occurred five times in an /i/ context and once in an /ɪ/ context, one might expect the /i/ match to be weighted more heavily, and thus, be the more natural choice for interpreting the vowel.

So, although the functional nature of a frequency of occurrence effect could easily vary between an algorithmic model and an exemplar-based model, we can see that the frequency hypothesis could theoretically be valid with either model.

5.2.2– Interpretation of results in an exemplar-based model

Because of the inherent contradiction between the primacy hypothesis and an exemplar-based model, the lack of support for the primacy hypothesis would be considered obvious (and in fact, support for it would cast doubt on an exemplar theory in general).

Interestingly, although frequency of occurrence effects are likely a factor in any weighting scheme, the lack of significant difference based on frequency is easily explicable by even a very basic exemplar-based theory of normalization.

One basic implementation of exemplar-based normalization would include only simple matching. If a vowel quality has occurred in a known context, it is tagged as an exemplar of the phoneme it represented, and future matching instances of that vowel quality are instantly associated with that phoneme. In this situation, multiple acoustic variations which occurred in the same context might still be tagged as being exemplars of the same phoneme. So long as a single variation didn't show up in the context of two different phonemes, the question of which variation is more frequent doesn't even arise when it's presented in an ambiguous context. In this situation, a match is a match, and frequency is irrelevant. So, in reality, matching the varied forms would be no more difficult than matching the canonical ones.

Because the context sentences in this study were purposefully designed to not include words that were lexically ambiguous between /i/ and /ɪ/, there are no situations where altered /i/ tokens occur in a possible /ɪ/ context. Therefore, the situation would be as described above, and identifying the target vowel would be a simple question of matching the acoustic properties of the target vowel to those of past tokens.

In every stimulus, regardless of the sentence or condition applied, both altered and unaltered /i/'s occurred in /i/ contexts (and only in /i/ contexts). Therefore, normalization (if done using an exemplar-based method) would be a simple question of matching. Thus, the identification process would be no more or less difficult for any given stimulus.

5.2.3– Implications of results on an exemplar-based model of speaker normalization

So, even though an exemplar-based model might well display some degree of frequency effect, because of the lack of contextual ambiguity discussed above, the stimuli presented here wouldn't be expected to trigger it. Therefore, in light of an exemplar based model, the lack of demonstrable effect produced by the conditions in this study are not only explicable, but to be expected. Although the lack of results could potentially be seen as reflecting our existing understanding of exemplar-based speaker normalization, there is, unfortunately, little here to expand our understanding of such a model.

VI – Future Studies

Although this particular study yielded no significant results, it did produce a wealth of practical data and insight into what questions should be addressed in future studies, and what methodologies should be used (and avoided) to address them.

6.1 – Future methodological changes

In this study, the biggest source of methodological problems was the use of Source-Filter Resynthesis in stimulus preparation. Although its use should not be avoided completely, in the future, its output would have to be far more closely monitored, and more precise means of controlling Praat's formant height adjustments would have to be found.

However, there may be advantages to performing future experiments using computer voice and vowel synthesis to create stimuli. In this way, the stimuli could be controlled with far greater precision, and there would be little stimulus variability which was not designed. Of course, further trials and research would be necessary to evaluate the feasibility of the process, to see how listeners respond to non-human voices, and to see if there may be other unintended effects.

Finally, in an effort to counteract per-item effects, trials of future experiments would almost certainly use more speakers, and therefore, more than one distinct stimulus set and ordering. This would filter out per-stimulus effects far more effectively than post-hoc analysis, and ideally, provide more reliable data for future analyses.

6.2 – Future questions and studies

6.2.1 – Testing the effects of context quantity

To confirm the basic assumption that more context increases ease of normalization, a study should likely be performed to measure the effects of context of different lengths on normalization speed. Such a study would likely take the form of a similar context/identification task pairing, where each context is unaltered, but has a different number of total vowels, ranging from one or two vowels to as many as twenty.

In this way, the difference in reaction times could be compared between trails preceded by more or less context, and the underlying assumption that more context improves ease of normalization could be tested. If reaction times are fastest following the most context, and slowest when little context is present, this assumption is well supported.

6.2.2 – Testing the frequency hypothesis assuming interdependent vowel normalization

Because the frequency hypothesis wasn't adequately tested in this study if vowel normalization does not occur for each phoneme independently, a followup study should be performed to further investigate and clarify its role. This study would likely use a similar methodology, but include modification to all vowels in the context, rather than just to tokens of /i/. Ideally, each of these alterations would cause a slight distortion to the listener's model of the speaker's vowel space, which would make it more difficult for them to identify an accurate, unaltered vowel in a simple identification task.

In such a study, a similar series of trials could be run testing the effect of different frequencies of occurrence of altered vowels (one vowel, 1/4 of all vowels, 1/2 of all, 3/4 of all, and all vowels), when coupled with an unaltered target vowel. If, as the overall frequency of alteration goes up, listeners are slower to identify an unaltered vowel, the frequency hypothesis would be unambiguously supported, and our understanding of the role of all vowels in single vowel normalization would be greatly clarified.

6.2.3 – Testing for a frequency of occurrence effect assuming an exemplar-based model of normalization

The stimuli prepared for this study were, by their very nature, completely incapable of triggering any frequency effect that might be at work in an exemplar-based model of speaker normalization. In order to effectively test for such an effect, a different sentence set and stimulus preparation method could be used in conjunction with the existing methodology used here.

The very nature of an exemplar-based model suggests that we normalize to each vowel based on prior instances of that vowel (and not based on other vowels). Therefore, we could safely set aside the question of independent versus interdependent normalization and work only with tokens of a particular vowels.

However, unlike in this study, each context sentence would have to contain a measured number of unambiguous instances of /i/ (likely more than just five, as exemplar theories thrive on data), and in addition, an equal number of contexts where only /ɪ/ fits. Then, in the modification process for each stimulus, a vowel of intermediate, set quality could be substituted for a certain number of the /i/ phonemes, for a lesser or greater number of the /ɪ/ vowels, and for the target word of the identification task. This way, an ambiguity is created, as the target vowel will match a prior examples of both /i/ and /ɪ/ phonemes.

Ideally, by adjusting the ratio of intermediate vowels in /i/ contexts versus /ɪ/ contexts, the listener's set of exemplar data could be weighted in such a way that the intermediate target vowel would seem more likely to be /i/, or more likely to be /ɪ/. One might expect that a context sentence with a high number of intermediate /i/ vowels and low number of intermediate /ɪ/'s would trigger the listener to identify the contextually ambiguous target vowel as /i/, and vice versa.

If such an effect is present and statistically significant, then it very strongly suggests that a frequency/weighting effect is present in exemplar-based models of normalization, neatly supporting the frequency hypothesis proposed here, albeit in a different theoretical context.

VII – Summary and Conclusions

This study was designed to gain insight into the process by which humans are able to adjust to and understand the speech of unfamiliar speakers, referred to a "speaker normalization". Prior research has suggested that, in order for normalization to occur, a listener has to have some speech data ("context") to process. The goal of this study was to further elucidate the role of this context, by examining precisely what parts of the context are necessary for vowel space normalization, and how that context is processed to allow listeners to accurately interpret an unfamiliar speaker's speech.

During the course of this study, two hypotheses were tested. The first hypothesis (the primacy hypothesis) stated that listeners will normalize based on the first vowel token to which they're exposed, and the second (the frequency hypothesis) stated that in the presence of variability, listeners will consider the more frequently occurring vowel quality to be the default, canonical quality, and will identify future tokens accordingly. In addition, data was collected to check for the presence of any combined effect from both hypotheses together.

These hypotheses were tested by altering (using source-filter resynthesis) a series of recorded stimuli in five specific patterns and presenting them to listeners in a timed, forced-choice vowel identification task. Two of these patterns contrasted to test for primacy, the first with an altered first vowel, the other featuring an altered second vowel. Two contrasted to test for frequency, featuring three vowels altered out of five total, and and two vowels altered, respectively. Finally, one was designed to test for a combined effect, as it included both three altered vowels and an altered first token.

Per-listener average reaction times for each different alteration pattern were then compared to test the validity of the hypotheses. This comparison yielded no statistically significant differences between the different preparations which were designed to elicit frequency and primacy effects, and even after extensive post-hoc analysis of sources of error, neither hypothesis was found to be supported by the data.

Although a role for primacy and frequency of occurrence in speaker normalization was unsupported by the data, the statistical power was insufficient to completely rule out either. However, the lack of strong primacy and frequency effects in this study raised several interesting theoretical questions, both for algorithmic and for exemplar-based models of speaker normalization, and suggests a variety of future avenues of exploration in the field of speaker normalization research.

These theoretical questions, combined with the lessons learned in the process of carrying out this experiment, pave the way for a variety of future research into the role of context in speaker normalization, and hopefully, for a better understanding of this fascinating process.

VIII – Acknowledgements

Many thanks, first and foremost, to Dr. Rebecca Scarborough, my thesis advisor and thesis committee chair at the University of Colorado at Boulder. Her advice, encouragement, statistical and technical knowledge has made this work possible, and her guidance has helped turn casual scholarly interest into a workable experiment.

In addition, I'd like to thank Dr. Lise Menn and Dr. Bhuvana Narasimhan for their guidance, advice, sympathy, and willingness to sit on my committee.

I'd also like to acknowledge Carolyn Buck-Gengler at the University of Colorado for her advice on structuring the experiment and finding participants, and Paul De Decker at NYU for his advice suggesting the addition of a Bandpass Merge step in reanalysis. I'd like to also acknowledge the University of Colorado Linguistics department for their contribution of funding for participant payment, and for the support given to our phonetics lab.

Finally, I'd like to warmly thank the participants, both speakers and listeners, who made this experiment possible with their participation, patience, and cooperation.

IX – References

Ciocca, V., Wong, N. K. Y., Leung, W. H. Y., & Chu, P. C. Y. (2006). Extrinsic context affects perceptual normalization of lexical tone. *The Journal of the Acoustical Society of America*, *Vol. 119, No. 3*, 1712-1726.

Gardener, M. Using R for Statistical Analyses. Retrieved March 25, 2008, from http://www.gardenersown.co.uk/Education/Lectures/R/basics.htm

Goldinger, S. D. (1997). Words and Voices: Perception and Production in an Episodic Lexicon. In K. Johnson & J. W. Mullenix (Eds.), *Talker Variability in Speech Processing*. (pp. 33-68). San Diego: Academic Press.

Haggard, M. & Summerfield, Q. (1977). *Perceptual Calibration for Parameters of Speaker Differences - Measures from Sequential Reaction Time Increment Studies*. Paper presented at the Sixth International Symposium on Attention and Performance, Hillsdale, NJ.

Joos, M. (1948). *Acoustic Phonetics - Supplement to Language*. Baltimore: Linguistic Society of America.

Ladefoged, P. & Broadbent, D. E. (1957). Information Conveyed by Vowels. *The Journal of the Acoustical Society of America*, *Volume 29, Number 1*, 98-104.

Nordstrom, P. & Lindblom, B. 1975 A Normalization Procedure for Vowel Formant Data. in *Proceedings of the International Congress of Phonetic Sciences*, Leeds, England.

Rositzke, H. A. (1939). Vowel-Length in General American Speech. *Language*, *Vol. 15, No. 2*, 99-109.

Verbrugge, R. R., Strange, W., Shankweiler, D. P., & Edman, T. R. (1976). What information enables a listener to map a talker's vowel space? *Journal of the Acoustical Society of America*, *Vol. 60, No. 1*, 198-212.

Whalen, D. H. & Sheffert, S. M. (1997). Normalization of Vowels by Breath Sounds. In K. Johnson & J. W. Mullenix (Eds.), *Talker Variability in Speech Processing*. (pp. 133-143). San Diego, CA: Academic Press Ltd.

Appendices

Included appendices:

Appendix I: Stimulus Alteration Script

```
################################################
##           Will Styler's Vowel Skewing script
##
       ## This script is designed to work with a sound/textgrid pair
## which contains five marked vowels, each marked with "i".
## However, as long as all vowels that you'd like skewed are
## marked with "i" on the Textgrid, Conditions A and B will
## work wonderfully, as will "All Altered" and "All Unaltered".  The target
vowel must be marked with 'T'.  Select the sound, then run.
##

## Written in Fall 2007 based on scripts by (and with
## assistance from) Rebecca Scarborough.  Cannibalize
## this script and the methods herein as you wish.
################################################

# Present the user with a form to choose which stimulus preparation to use.
form Select your Stimulus Alteration
   comment Select your Alteration
        choice Cond: 1
        button Alteration A: First (AUUUU A)
             button Alteration B: Second (UAUUU A)
        button Alteration C: High Freq (UAAAU A)
        button Alteration D: Low Freq (UAUAU A)
        button Alteration E: First freq (AUUAA A)
             button All Altered (AAAAA A)
             button All Unaltered (UUUUU A)
# This simply allows the user to choose which of the alteration patterns
(conditions) to apply to the stimulus
   comment Skew the Target?
        choice tar: 1
        button Yes
             button No
# This allows the user to choose whether to skew the target
   comment Bandpass and remerge?
        choice pass: 1
        button Yes
        button No
# This allows a choice as to whether or not to bandpass and remerge.
   comment Pass Boundary?
       integer passnum 6000
# This sets the boundary between the pure and the skewed if a bandpass step
is included
    comment Find how many formants?
       integer formnum 5
# Having Praat find more formants can be good for certain speakers, but isn't
often necessary
endform

if tar = 1
       tar$ = "Y"
endif
if tar = 2
       tar$ = "N"
endif

# Get the selected file
sn$ = selected$ ("Sound")

# Here, I've turned the vowel-skewing per se into a procedure to keep the
code cleaner later
procedure Skew_Vowels
       # Resampling really improves the LPC quality
```

```
	Resample... 11025 50
	Rename...  'sn$'_origdown
	# First, we create the LPC (for voicing isolation) and the formant
(for tweaking)
	To LPC (burg)...     12 0.025 0.005 50
	select LPC 'sn$'_origdown
	Rename...  'sn$'_LPC
	select Sound 'sn$'_origdown
	To Formant (burg)... 0 formnum 5500 0.025 50
	# Then, we isolate the voicing
	select LPC 'sn$'_LPC
	select Sound 'sn$'_origdown
	plus LPC 'sn$'_LPC
	Filter (inverse)
	Rename...  'sn$'_Voicing
	# The LPC object isn't needed anymore
	select LPC 'sn$'_LPC
	Remove
	# Rename the formant, then tweak it such that F2 and F3 are lowered
by 300hz
	select Formant 'sn$'_origdown
	Copy...  'sn$'_distorted300
	Formula (frequencies)...  if row = 2 then self - 50 else self fi
	Formula (frequencies)...  if row = 2 then self - 50 else self fi
	Formula (frequencies)...  if row = 2 then self - 50 else self fi
	Formula (frequencies)...  if row = 2 then self - 50 else self fi
	Formula (frequencies)...  if row = 2 then self - 50 else self fi
	Formula (frequencies)...  if row = 2 then self - 50 else self fi
	Formula (frequencies)...  if row = 3 then self - 50 else self fi
	Formula (frequencies)...  if row = 3 then self - 50 else self fi
	Formula (frequencies)...  if row = 3 then self - 50 else self fi
	Formula (frequencies)...  if row = 3 then self - 50 else self fi
	Formula (frequencies)...  if row = 3 then self - 50 else self fi
	Formula (frequencies)...  if row = 3 then self - 50 else self fi
	# Create the skewed version
	select Formant 'sn$'_distorted300
	plus Sound 'sn$'_Voicing
	Filter
	Rename...  'sn$'_Skewed-300
	# Match the intensity of the original by getting the intensity
(called 'dudeintense' here) and scaling the skewed to the same level
	select Sound 'sn$'
	dudeintense = Get intensity (dB)
	select Sound 'sn$'_Skewed-300
	Copy...  'sn$'_skewedvowel
	Scale intensity... dudeintense
	# Resample the skewed vowel to 44100 such that it can be copypasted
into the original file
	Resample... 44100 50
	select Sound 'sn$'_skewedvowel
	Remove
	# Rename the 44100 version as sound_skewedvowel and clean up the files
used in the process
	select Sound 'sn$'_skewedvowel_44100
	Rename... 'sn$'_skewedvowel
	select Formant 'sn$'_origdown
	Remove
	select Formant 'sn$'_distorted300
	Remove
	select Sound 'sn$'_Skewed-300
	Remove
	select Sound 'sn$'_origdown
	Remove
	select Sound 'sn$'_Voicing
	Remove
	select Sound 'sn$'_temp
	Remove
```

```
endproc

# This procedure is identical to Skew_Vowels, except that the formant isn't
actually tweaked.  This is used to produce a pure-yet-reanalyzed vowel
procedure Reanalyze
	# Resampling really improves the LPC quality
	Resample... 11025 50
	Rename...  'sn$'_origdown
	# First, we create the LPC (for voicing isolation) and the formant
(for tweaking)
	To LPC (burg)...	12 0.025 0.005 50
	select LPC 'sn$'_origdown
	Rename...  'sn$'_LPC
	select Sound 'sn$'_origdown
	To Formant (burg)... 0 formnum 5500 0.025 50
	# Then, we isolate the voicing
	select LPC 'sn$'_LPC
	select Sound 'sn$'_origdown
	plus LPC 'sn$'_LPC
	Filter (inverse)
	Rename...  'sn$'_Voicing
	# The LPC object isn't needed anymore
	select LPC 'sn$'_LPC
	Remove
	# Rename the formant
	select Formant 'sn$'_origdown
	Copy...  'sn$'_distorted300
	select Formant 'sn$'_distorted300
	plus Sound 'sn$'_Voicing
	Filter
	Rename...  'sn$'_Skewed-300
	# Match the intensity of the original by getting the intensity
(called 'dudeintense' here) and scaling the skewed to the same level
	select Sound 'sn$'
	dudeintense = Get intensity (dB)
	select Sound 'sn$'_Skewed-300
	Copy...  'sn$'_skewedvowel
	Scale intensity... dudeintense
	# Resample the skewed vowel to 44100 such that it can be copypasted
into the original file
	Resample... 44100 50
	select Sound 'sn$'_skewedvowel
	Remove
	# Rename the 44100 version as sound_skewedvowel and clean up the files
used in the process (Yes, it's not skewed here, but it's easier to leave
everything
	# named as skewed for later procedures)
	select Sound 'sn$'_skewedvowel_44100
	Rename... 'sn$'_skewedvowel
	select Formant 'sn$'_origdown
	Remove
	select Formant 'sn$'_distorted300
	Remove
	select Sound 'sn$'_Skewed-300
	Remove
	select Sound 'sn$'_origdown
	Remove
	select Sound 'sn$'_Voicing
	Remove
	select Sound 'sn$'_temp
	Remove
endproc

# This procedure does the dirty work of actually cutting the pure, copying
the skewed and pasting it in the pure's place
procedure Replace_With_Skewed
```

```
	# Select the pure sound (which has already been renamed _tweaked by
this time in the for-loop)
	select Sound 'sn$'_tweaked
		Edit
		# Use the editor window to select between the start and end
times of the interval, then cut it out (removing it)
		editor Sound 'sn$'_tweaked
			Select... vstart vend
			Cut
			Close
		endeditor
		# Open the previously created skewed vowel file, then copy the
exact interval to the clipboard
		select Sound 'sn$'_skewedvowel
		Edit
		editor Sound 'sn$'_skewedvowel
			Select... vstart vend
			Copy selection to Sound clipboard
			Close
		endeditor
		# Reopen the pure sentence, move the cursor to the interval
start, and paste in the skewed vowel
		select Sound 'sn$'_tweaked
		Edit
		editor Sound 'sn$'_tweaked
			Move cursor to... vstart
			Paste after selection
			Close
		endeditor
endproc

# This  Procedure merges the bottom 5000hz of the altered file with the 5000+
of the unaltered file.  It will work even if the skewing results in minor
changes in duration.
procedure Passcombine
	# First, we measure the duration of both sounds
	select Sound 'sn$'_tweaked
	tweakdur = Get total duration
	select Sound 'sn$'
	Copy... 'sn$'_pure
	puredur = Get total duration
	if tweakdur > puredur
		durdiff = tweakdur - puredur
		select Sound 'sn$'_pure
		Edit
		editor Sound 'sn$'_pure
			Select... 0 durdiff
			Copy selection to Sound clipboard
			Move cursor to... 0
			Paste after selection
		endeditor
	endif
	if tweakdur < puredur
		durdifft = puredur - tweakdur
		select Sound 'sn$'_tweaked
		Edit
		editor Sound 'sn$'_tweaked
			Select... 0 durdifft
			Copy selection to Sound clipboard
			Move cursor to... 0
			Paste after selection
		endeditor
	endif
	# Select the tweaked sound and filter out everything above 5000hz
(although here, it's controlled by the variable 'passnum', set to 5000
elsewhere
	select Sound 'sn$'_tweaked
```

```
        Filter (pass Hann band)... 0 passnum 1
        Rename... 'sn$'_lo
        # Filter out everything below 5000hz in the pure file
        select Sound 'sn$'_pure
        Filter (stop Hann band)... 0 passnum 1
        Rename... 'sn$'_hi
        # clean up the old "tweaked" file to avoid ambiguity
        select Sound 'sn$'_tweaked
        Remove
        # Select the highs from the pure and the tweaked lows and combine
them to stereo, in effect, merging the files
        select Sound 'sn$'_hi
        plus Sound 'sn$'_lo
        Combine to stereo
        # Put it back to Mono so the two combined are just one mono sound
        Convert to mono
        # Renamed the merged file, and do some cleanup
        Rename... 'sn$'_tweaked
        select Sound 'sn$'_hi
        plus Sound 'sn$'_lo
        plus Sound 'sn$'_lo_'sn$'_hi
        plus Sound 'sn$'_pure
        Remove
endproc

# Now, we actually start the work of the script using the above procedures

# Start Textgrid work
select Sound 'sn$'
# Copy the sound to _tweaked as to not overwrite the original
Copy... 'sn$'_tweaked
select TextGrid 'sn$'
numint = Get number of intervals... 1

# Check to see which condition is requested, and then carry it out


if cond = 1
                select TextGrid 'sn$'
                label$ = Get label of interval... 1 2
                # Check to see if the interval has a vowel
                if label$ = "i"
                        vstart = Get starting point... 1 2
                        vend = Get end point... 1 2
                        select Sound 'sn$'
                        Extract part... vstart-0.25 vend+0.25 Hanning 1 yes
                        Rename... 'sn$'_temp
                        call Skew_Vowels
                        call Replace_With_Skewed
                        # Remove the Skewedvowel
                        select Sound 'sn$'_skewedvowel
                Remove
                endif
        for i from 1 to numint
                select TextGrid 'sn$'
                label$ = Get label of interval... 1 'i'
                # Check to see if the interval has a vowel
                if label$ = "i"
                        if i <> 2
                                vstart = Get starting point... 1 'i'
                                vend = Get end point... 1 'i'
                                select Sound 'sn$'
                                Extract part... vstart-0.25 vend+0.25 Hanning 1
yes
                                Rename... 'sn$'_temp
                                call Reanalyze
                                call Replace_With_Skewed
```

```
                        # Remove the Skewedvowel
                        select Sound 'sn$'_skewedvowel
                        Remove
                endif
        endif
        if label$ = "T"
                if tar = 1
                        vstart = Get starting point... 1 'i'
                        vend = Get end point... 1 'i'
                        select Sound 'sn$'
                        Extract part... vstart-0.25 vend+0.25 Hanning 1
yes
                        Rename... 'sn$'_temp
                        call Skew_Vowels
                        call Replace_With_Skewed
                        # Remove the Skewedvowel
                        select Sound 'sn$'_skewedvowel
                        Remove
                endif
                if tar = 2
                        vstart = Get starting point... 1 'i'
                        vend = Get end point... 1 'i'
                        select Sound 'sn$'
                        Extract part... vstart-0.25 vend+0.25 Hanning 1
yes
                        Rename... 'sn$'_temp
                        call Reanalyze
                        call Replace_With_Skewed
                        # Remove the Skewedvowel
                        select Sound 'sn$'_skewedvowel
                        Remove
                endif
        endif
    endfor
    if pass = 1
        call Passcombine
    endif
    select Sound 'sn$'_tweaked
    Rename... 'sn$'_CondA_'tar$'
endif

if cond = 2
        select TextGrid 'sn$'
        label$ = Get label of interval... 1 4
        # Check to see if the interval has a vowel
        if label$ = "i"
                vstart = Get starting point... 1 4
                vend = Get end point... 1 4
                select Sound 'sn$'
                Extract part... vstart-0.25 vend+0.25 Hanning 1 yes
                Rename... 'sn$'_temp
                call Skew_Vowels
                call Replace_With_Skewed
                # Remove the Skewedvowel
                select Sound 'sn$'_skewedvowel
        Remove
        endif
    for i from 1 to numint
        select TextGrid 'sn$'
        label$ = Get label of interval... 1 'i'
        # Check to see if the interval has a vowel
        if label$ = "i"
                if i <> 4
                        vstart = Get starting point... 1 'i'
                        vend = Get end point... 1 'i'
                        select Sound 'sn$'
```

```
                    Extract part... vstart-0.25 vend+0.25 Hanning 1
yes
                    Rename... 'sn$'_temp
                    call Reanalyze
                    call Replace_With_Skewed
                    # Remove the Skewedvowel
                    select Sound 'sn$'_skewedvowel
                    Remove
                endif
            endif
            if label$ = "T"
                if tar = 1
                    vstart = Get starting point... 1 'i'
                    vend = Get end point... 1 'i'
                    select Sound 'sn$'
                    Extract part... vstart-0.25 vend+0.25 Hanning 1
yes
                    Rename... 'sn$'_temp
                    call Skew_Vowels
                    call Replace_With_Skewed
                    # Remove the Skewedvowel
                    select Sound 'sn$'_skewedvowel
                    Remove
                endif
                if tar = 2
                    vstart = Get starting point... 1 'i'
                    vend = Get end point... 1 'i'
                    select Sound 'sn$'
                    Extract part... vstart-0.25 vend+0.25 Hanning 1
yes
                    Rename... 'sn$'_temp
                    call Reanalyze
                    call Replace_With_Skewed
                    # Remove the Skewedvowel
                    select Sound 'sn$'_skewedvowel
                    Remove
                endif
            endif
        endfor
        if pass = 1
            call Passcombine
        endif
        select Sound 'sn$'_tweaked
        Rename... 'sn$'_CondB_'tar$'
endif


# These final conditions are hackish.  Sadly, I have to just specify which
intervals are to be skewed.
# So, rather than saying "the 1st, 3rd, and 5th", I have to say "interval 2,
6 and 10"
# However, when provided with a 5-skewed-vowel input, this hackish method
works like a charm

if cond = 3
            select TextGrid 'sn$'
            label$ = Get label of interval... 1 4
            # Check to see if the interval has a vowel
            if label$ = "i"
                vstart = Get starting point... 1 4
                vend = Get end point... 1 4
                select Sound 'sn$'
                Extract part... vstart-0.25 vend+0.25 Hanning 1 yes
                Rename... 'sn$'_temp
                call Skew_Vowels
                call Replace_With_Skewed
                # Remove the Skewedvowel
```

```
	select Sound 'sn$'_skewedvowel
Remove
endif
select TextGrid 'sn$'
label$ = Get label of interval... 1 6
# Check to see if the interval has a vowel
if label$ = "i"
	vstart = Get starting point... 1 6
	vend = Get end point... 1 6
	select Sound 'sn$'
	Extract part... vstart-0.25 vend+0.25 Hanning 1 yes
	Rename... 'sn$'_temp
	call Skew_Vowels
	call Replace_With_Skewed
	# Remove the Skewedvowel
	select Sound 'sn$'_skewedvowel
Remove
endif
select TextGrid 'sn$'
label$ = Get label of interval... 1 8
# Check to see if the interval has a vowel
if label$ = "i"
	vstart = Get starting point... 1 8
	vend = Get end point... 1 8
	select Sound 'sn$'
	Extract part... vstart-0.25 vend+0.25 Hanning 1 yes
	Rename... 'sn$'_temp
	call Skew_Vowels
	call Replace_With_Skewed
	# Remove the Skewedvowel
	select Sound 'sn$'_skewedvowel
Remove
endif
select TextGrid 'sn$'
label$ = Get label of interval... 1 2
# Check to see if the interval has a vowel
if label$ = "i"
	vstart = Get starting point... 1 2
	vend = Get end point... 1 2
	select Sound 'sn$'
	Extract part... vstart-0.25 vend+0.25 Hanning 1 yes
	Rename... 'sn$'_temp
	call Reanalyze
	call Replace_With_Skewed
	# Remove the Skewedvowel
	select Sound 'sn$'_skewedvowel
Remove
endif
select TextGrid 'sn$'
label$ = Get label of interval... 1 10
# Check to see if the interval has a vowel
if label$ = "i"
	vstart = Get starting point... 1 10
	vend = Get end point... 1 10
	select Sound 'sn$'
	Extract part... vstart-0.25 vend+0.25 Hanning 1 yes
	Rename... 'sn$'_temp
	call Reanalyze
	call Replace_With_Skewed
	# Remove the Skewedvowel
	select Sound 'sn$'_skewedvowel
Remove
endif
select TextGrid 'sn$'
label$ = Get label of interval... 1 12
if label$ = "T"
	if tar = 1
```

```
                        vstart = Get starting point... 1 12
                        vend = Get end point... 1 12
                        select Sound 'sn$'
                        Extract part... vstart-0.25 vend+0.25 Hanning 1
yes
                        Rename... 'sn$'_temp
                        call Skew_Vowels
                        call Replace_With_Skewed
                        # Remove the Skewedvowel
                        select Sound 'sn$'_skewedvowel
                        Remove
                  endif
                  if tar = 2
                        vstart = Get starting point... 1 12
                        vend = Get end point... 1 12
                        select Sound 'sn$'
                        Extract part... vstart-0.25 vend+0.25 Hanning 1
yes
                        Rename... 'sn$'_temp
                        call Reanalyze
                        call Replace_With_Skewed
                        # Remove the Skewedvowel
                        select Sound 'sn$'_skewedvowel
                        Remove
                  endif
            endif
      if pass = 1
            call Passcombine
      endif
      select Sound 'sn$'_tweaked
      Rename... 'sn$'_CondC_'tar$'
endif

if cond = 4
            select TextGrid 'sn$'
            label$ = Get label of interval... 1 2
            # Check to see if the interval has a vowel
            if label$ = "i"
                  vstart = Get starting point... 1 2
                  vend = Get end point... 1 2
                  select Sound 'sn$'
                  Extract part... vstart-0.25 vend+0.25 Hanning 1 yes
                  Rename... 'sn$'_temp
                  call Reanalyze
                  call Replace_With_Skewed
                  # Remove the Skewedvowel
                  select Sound 'sn$'_skewedvowel
            Remove
            endif
            select TextGrid 'sn$'
            label$ = Get label of interval... 1 6
            # Check to see if the interval has a vowel
            if label$ = "i"
                  vstart = Get starting point... 1 6
                  vend = Get end point... 1 6
                  select Sound 'sn$'
                  Extract part... vstart-0.25 vend+0.25 Hanning 1 yes
                  Rename... 'sn$'_temp
                  call Reanalyze
                  call Replace_With_Skewed
                  # Remove the Skewedvowel
                  select Sound 'sn$'_skewedvowel
            Remove
            endif
            select TextGrid 'sn$'
            label$ = Get label of interval... 1 10
            # Check to see if the interval has a vowel
```

```
		if label$ = "i"
			vstart = Get starting point... 1 10
			vend = Get end point... 1 10
			select Sound 'sn$'
			Extract part... vstart-0.25 vend+0.25 Hanning 1 yes
			Rename... 'sn$'_temp
			call Reanalyze
			call Replace_With_Skewed
			# Remove the Skewedvowel
			select Sound 'sn$'_skewedvowel
		Remove
		endif
		select TextGrid 'sn$'
		label$ = Get label of interval... 1 4
		# Check to see if the interval has a vowel
		if label$ = "i"
			vstart = Get starting point... 1 4
			vend = Get end point... 1 4
			select Sound 'sn$'
			Extract part... vstart-0.25 vend+0.25 Hanning 1 yes
			Rename... 'sn$'_temp
			call Skew_Vowels
			call Replace_With_Skewed
			# Remove the Skewedvowel
			select Sound 'sn$'_skewedvowel
		Remove
		endif
		select TextGrid 'sn$'
		label$ = Get label of interval... 1 8
		# Check to see if the interval has a vowel
		if label$ = "i"
			vstart = Get starting point... 1 8
			vend = Get end point... 1 8
			select Sound 'sn$'
			Extract part... vstart-0.25 vend+0.25 Hanning 1 yes
			Rename... 'sn$'_temp
			call Skew_Vowels
			call Replace_With_Skewed
			# Remove the Skewedvowel
			select Sound 'sn$'_skewedvowel
		Remove
		endif
		select TextGrid 'sn$'
		label$ = Get label of interval... 1 12
		if label$ = "T"
			if tar = 1
				vstart = Get starting point... 1 12
				vend = Get end point... 1 12
				select Sound 'sn$'
				Extract part... vstart-0.25 vend+0.25 Hanning 1
yes
				Rename... 'sn$'_temp
				call Skew_Vowels
				call Replace_With_Skewed
				# Remove the Skewedvowel
				select Sound 'sn$'_skewedvowel
				Remove
			endif
			if tar = 2
				vstart = Get starting point... 1 12
				vend = Get end point... 1 12
				select Sound 'sn$'
				Extract part... vstart-0.25 vend+0.25 Hanning 1
yes
				Rename... 'sn$'_temp
				call Reanalyze
				call Replace_With_Skewed
```

```
                        # Remove the Skewedvowel
                        select Sound 'sn$'_skewedvowel
                        Remove
                endif
            endif
        if pass = 1
            call Passcombine
        endif
        select Sound 'sn$'_tweaked
        Rename... 'sn$'_CondD_'tar$'
endif

if cond = 5
            select TextGrid 'sn$'
            label$ = Get label of interval... 1 2
            # Check to see if the interval has a vowel
            if label$ = "i"
                vstart = Get starting point... 1 2
                vend = Get end point... 1 2
                select Sound 'sn$'
                Extract part... vstart-0.25 vend+0.25 Hanning 1 yes
                Rename... 'sn$'_temp
                call Skew_Vowels
                call Replace_With_Skewed
                # Remove the Skewedvowel
                select Sound 'sn$'_skewedvowel
            Remove
            endif
            select TextGrid 'sn$'
            label$ = Get label of interval... 1 8
            # Check to see if the interval has a vowel
            if label$ = "i"
                vstart = Get starting point... 1 8
                vend = Get end point... 1 8
                select Sound 'sn$'
                Extract part... vstart-0.25 vend+0.25 Hanning 1 yes
                Rename... 'sn$'_temp
                call Skew_Vowels
                call Replace_With_Skewed
                # Remove the Skewedvowel
                select Sound 'sn$'_skewedvowel
            Remove
            endif
            select TextGrid 'sn$'
            label$ = Get label of interval... 1 10
            # Check to see if the interval has a vowel
            if label$ = "i"
                vstart = Get starting point... 1 10
                vend = Get end point... 1 10
                select Sound 'sn$'
                Extract part... vstart-0.25 vend+0.25 Hanning 1 yes
                Rename... 'sn$'_temp
                call Skew_Vowels
                call Replace_With_Skewed
                # Remove the Skewedvowel
                select Sound 'sn$'_skewedvowel
            Remove
            endif
            select TextGrid 'sn$'
            label$ = Get label of interval... 1 4
            # Check to see if the interval has a vowel
            if label$ = "i"
                vstart = Get starting point... 1 4
                vend = Get end point... 1 4
                select Sound 'sn$'
                Extract part... vstart-0.25 vend+0.25 Hanning 1 yes
                Rename... 'sn$'_temp
```

```
                call Reanalyze
                call Replace_With_Skewed
                # Remove the Skewedvowel
                select Sound 'sn$'_skewedvowel
            Remove
            endif
            select TextGrid 'sn$'
            label$ = Get label of interval... 1 6
            # Check to see if the interval has a vowel
            if label$ = "i"
                vstart = Get starting point... 1 6
                vend = Get end point... 1 6
                select Sound 'sn$'
                Extract part... vstart-0.25 vend+0.25 Hanning 1 yes
                Rename... 'sn$'_temp
                call Reanalyze
                call Replace_With_Skewed
                # Remove the Skewedvowel
                select Sound 'sn$'_skewedvowel
            Remove
            endif
        select TextGrid 'sn$'
            label$ = Get label of interval... 1 12
            if label$ = "T"
                if tar = 1
                    vstart = Get starting point... 1 12
                    vend = Get end point... 1 12
                    select Sound 'sn$'
                    Extract part... vstart-0.25 vend+0.25 Hanning 1
yes
                    Rename... 'sn$'_temp
                    call Skew_Vowels
                    call Replace_With_Skewed
                    # Remove the Skewedvowel
                    select Sound 'sn$'_skewedvowel
                    Remove
                endif
                if tar = 2
                    vstart = Get starting point... 1 12
                    vend = Get end point... 1 12
                    select Sound 'sn$'
                    Extract part... vstart-0.25 vend+0.25 Hanning 1
yes
                    Rename... 'sn$'_temp
                    call Reanalyze
                    call Replace_With_Skewed
                    # Remove the Skewedvowel
                    select Sound 'sn$'_skewedvowel
                    Remove
                endif
            endif
        if pass = 1
            call Passcombine
        endif
        select Sound 'sn$'_tweaked
        Rename... 'sn$'_CondE_'tar$'
        select Sound 'sn$'_CondE_'tar$'
endif

if cond = 6
        for i from 1 to numint
            select TextGrid 'sn$'
            label$ = Get label of interval... 1 'i'
            # Check to see if the interval has a vowel
            if label$ = "i"
                vstart = Get starting point... 1 'i'
                vend = Get end point... 1 'i'
```

```
                    select Sound 'sn$'
                    Extract part... vstart-0.25 vend+0.25 Hanning 1 yes
                    Rename... 'sn$'_temp
                    call Skew_Vowels
                    call Replace_With_Skewed
                    # Remove the Skewedvowel
                    select Sound 'sn$'_skewedvowel
              Remove
              endif
              if label$ = "T"
                    if tar = 1
                          vstart = Get starting point... 1 'i'
                          vend = Get end point... 1 'i'
                          select Sound 'sn$'
                          Extract part... vstart-0.25 vend+0.25 Hanning 1
yes
                          Rename... 'sn$'_temp
                          call Skew_Vowels
                          call Replace_With_Skewed
                          # Remove the Skewedvowel
                          select Sound 'sn$'_skewedvowel
                          Remove
                    endif
                    if tar = 2
                          vstart = Get starting point... 1 'i'
                          vend = Get end point... 1 'i'
                          select Sound 'sn$'
                          Extract part... vstart-0.25 vend+0.25 Hanning 1
yes
                          Rename... 'sn$'_temp
                          call Reanalyze
                          call Replace_With_Skewed
                          # Remove the Skewedvowel
                          select Sound 'sn$'_skewedvowel
                          Remove
                    endif
              endif

       endfor
       if pass = 1
              call Passcombine
       endif
       select Sound 'sn$'_tweaked
       Rename... 'sn$'_AllAlt_'tar$'
endif

if cond = 7
       for i from 1 to numint
              select TextGrid 'sn$'
              label$ = Get label of interval... 1 'i'
              # Check to see if the interval has a vowel
              if label$ = "i"
                    vstart = Get starting point... 1 'i'
                    vend = Get end point... 1 'i'
                    select Sound 'sn$'
                    Extract part... vstart-0.25 vend+0.25 Hanning 1 yes
                    Rename... 'sn$'_temp
                    call Reanalyze
                    call Replace_With_Skewed
                    # Remove the Skewedvowel
                    select Sound 'sn$'_skewedvowel
                    Remove
              endif
              if label$ = "T"
                    if tar = 1
                          vstart = Get starting point... 1 'i'
                          vend = Get end point... 1 'i'
```

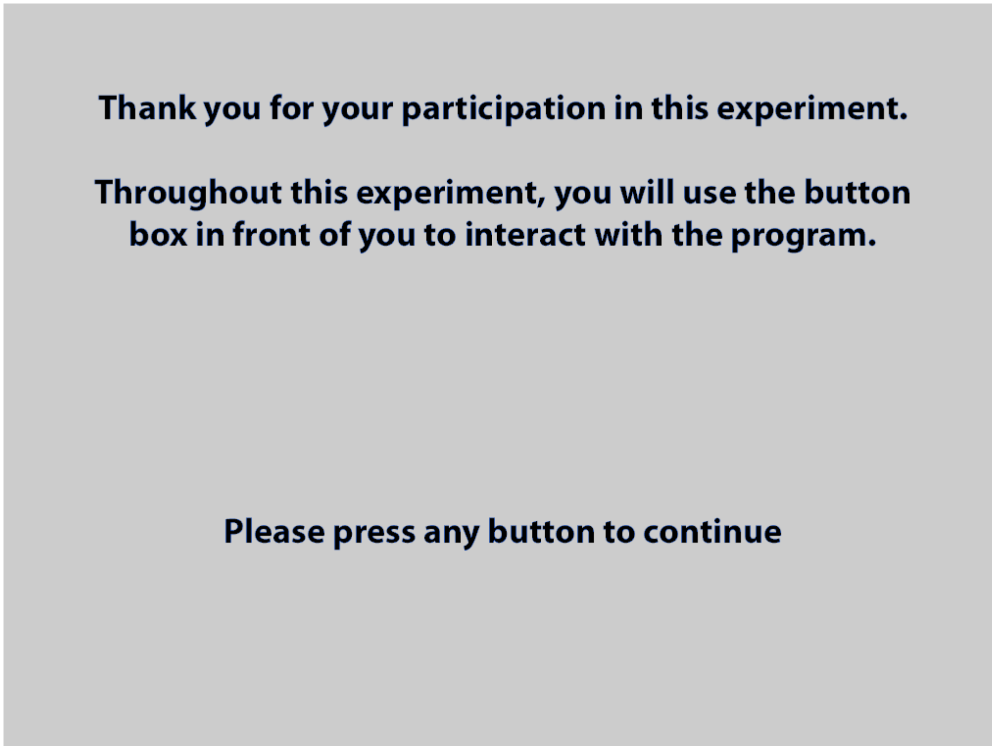
```
                    select Sound 'sn$'
                    Extract part... vstart-0.25 vend+0.25 Hanning 1
yes
                    Rename... 'sn$'_temp
                    call Skew_Vowels
                    call Replace_With_Skewed
                    # Remove the Skewedvowel
                    select Sound 'sn$'_skewedvowel
                    Remove
                endif
                if tar = 2
                    vstart = Get starting point... 1 'i'
                    vend = Get end point... 1 'i'
                    select Sound 'sn$'
                    Extract part... vstart-0.25 vend+0.25 Hanning 1
yes
                    Rename... 'sn$'_temp
                    call Reanalyze
                    call Replace_With_Skewed
                    # Remove the Skewedvowel
                    select Sound 'sn$'_skewedvowel
                    Remove
                endif
            endif
    endfor
    if pass = 1
        call Passcombine
    endif
    select Sound 'sn$'_tweaked
    Rename... 'sn$'_AllPure_'tar$'
endif
```

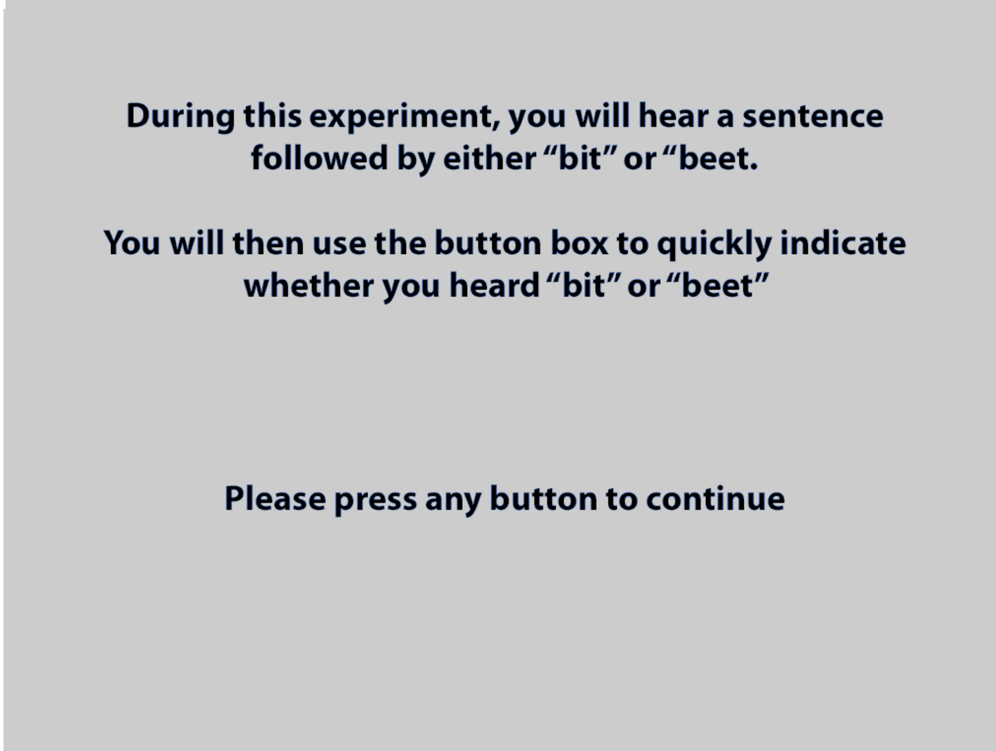
Appendix II: Final Stimulus Ordering

Subject #	Sentence	Condition?	Skewed Target?
1a	1	A	
2a	2	B	
F1A	10	A	y
3a	3	C	
F2A	9	B	n
4a	4	D	
1b	5	E	
F1B	8	C	y
F2B	7	D	n
1c	6	E	y
2b	6	A	
6a	8	B	
2c	5	A	n
3b	7	C	
2d	10	B	y
4b	9	D	
3c	3	C	n
5b	10	E	
1d	2	D	y
7a	1	A	
3d	4	E	n
4d	9	A	y
8a	2	B	
4c	8	B	n
6b	3	C	
10b	4	D	
6c	7	C	y
8b	5	E	
5a	6	A	
9a	7	B	
5c	6	D	n
6d	5	E	y
7b	8	C	
9b	9	D	
7c	4	A	n
5d	3	B	y
10a	10	E	
8c	2	C	n
9c	1	D	y
10c	10	E	n
	Green indicates a test trial, red indicates a filler trial		

(All filler trials have either a 'y' or 'n' in the "Skewed Target" column, whereas the target is assumed to be skewed in test trials)

Appendix III: Orientation Screens





Once the sentence has played, the speaker will say either "bit" or "beet".

If you hear "Beet", press the left-most button (Red)
If you hear "Bit", press the right-most button (Purple)

If you're unsure, give your best guess.

Press the BEET button to continue...

Once you have answered, you may pause for as long as you'd like, and then move to the next trial by pressing either button.

You will hear the same sentences more than once, spoken by different speakers and with different answers.

Press the BIT button to continue...

Full Name*	#	Cond.	1	2	3	4	5	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
1One1A	1	A1	500	776	658	668	570	651	695	633	603	862	543	984	814	1137	828	592	519	456	373	517	1297
11Two6A	11	A2	735	790	625	579	486	723	518	524	607	496	875	976	569	732	1269	578	405	441	418	545	563
20Seven1A	20	A3	567	1456	662	562	409	669	422	508	599	420	508	747	544	650	678	565	616	283	1132	418	655
29Five6A	29	A4	467	691	615	686	563	821	546	670	705	748	699	999	503	1033	752	689	737	337	459	598	674
2Two2B	2	B1	847	816	780	724	529	1504	706	945	736	590	699	895	781	943	935	604	482	299	784	590	1423
12Six8B	12	B2	879	1603	812	793	546	778	2410	402	647	1192	587	2066	536	837	807	961	367	263	602	373	909
23Eight2B	23	B3	464	674	705	455	294	626	888	604	550	500	585	784	631	574	836	589	359	223	590	493	638
30Nine7B	30	B4	465	717	625	469	347	579	523	473	590	653	475	780	607	680	658	544	505	254	454	621	560
4Three3C	4	C1	583	718	754	587	491	2876	600	657	770	745	862	926	643	812	1051	668	410	379	456	417	1657
14Three7C	14	C2	666	1354	712	650	424	890	681	656	677	500	517	650	518	614	712	536	432	315	561	333	688
25Six3C	25	C3	868	741	794	338	317	750	1001	861	473	836	602	861	432	822	708	549	461	431	1302	635	976
33Seven8C	33	C4	468	525	528	411	315	665	669	503	503	523	482	656	531	581	821	496	554	294	554	396	654
6Four4D	6	D1	769	605	658	708	606	2197	577	804	663	708	846	819	567	761	910	643	218	281	521	417	767
16Four9D	16	D2	656	1323	630	606	471	697	547	685	583	755	591	964	510	959	730	680	586	495	505	427	625
26Ten4D	26	D3	529	836	623	626	397	861	603	103	504	578	533	606	470	686	744	534	388	361	570	410	617
34Nine9D	34	D4	421	655	590	320	246	531	628	478	440	909	370	736	395	885	671	459	386	358	486	345	562
7One5E	7	E1	774	682	738	962	550	2029	686	919	555	623	760	1207	634	774	835	610	219	287	641	469	749
18Five10E	18	E2	457	633	612	481	318	634	574	849	702	405	550	823	512	535	696	579	330	434	674	813	579
28Eight5E	28	E3	679	765	658	568	372	706	665	931	618	536	542	792	436	577	852	432	419	450	513	671	708
37Ten10E	37	E4	509	731	792	736	462	700	703	644	632	933	707	600	875	935	903	686	432	321	478	1061	957
Mean - 2sd			308.761	234.596	521.579	282.897	221.638	-287.03	-98.462	224.82	432.306	275.423	338.394	263.979	321.521	439.131	525.429	377.714	191.38	189.675	140.59	173.958	192.833
Mean + 2sd			921.539	1474.5	835.521	910.003	649.662	2275.73	1562.66	1060.08	783.394	1075.78	894.906	1523.12	829.279	1113.57	1114.17	821.686	691.12	506.525	1066.71	880.942	1432.97
1sd			153.195	309.977	78.4853	156.777	107.006	640.692	415.281	208.815	87.772	200.089	139.128	314.786	126.939	168.609	147.185	110.993	124.935	79.2125	231.53	176.746	310.033
Mean			615.15	854.55	678.55	596.45	435.65	994.35	732.1	642.45	607.85	675.6	616.65	893.55	575.4	776.35	819.8	599.7	441.25	348.1	603.65	527.45	812.9

***Stimulus full names are of the form [Stimulus#][Speaker Number][Sentence#][Condition]**

Ex. 14Three7C is stimulus 14, consisting of sentence seven, spoken by speaker three, prepared to Condition C

Per Listener Condition Averages

	1	2	3	4	5	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
A	567.25	928.25	640	623.75	507	716	545.25	583.75	628.5	631.5	656.25	926.5	607.5	888	881.75	606	569.25	379.25	595.5	519.5	797.25
B	663.75	952.5	730.5	610.25	429	871.75	1131.75	606	630.75	733.75	586.5	1131.25	638.75	758.5	809	674.5	428.25	259.75	607.5	519.25	882.5
C	646.25	834.5	697	496.5	386.75	1295.25	737.75	669.25	605.75	651	615.75	773.25	531	707.25	823	562.25	464.25	354.75	718.25	445.25	993.75
D	593.75	854.75	625.25	565	430	1071.5	588.75	517.5	547.5	737.5	585	781.25	485.5	822.75	763.75	579	394.5	373.75	520.5	399.75	642.75
E	604.75	702.75	700	686.75	425.5	1017.25	657	835.75	626.75	624.25	639.75	855.5	614.25	705.25	821.5	576.75	350	373	576.5	753.5	748.25

A Listener Average	657.048
B Listener Average	697.893
C Listener Average	667.083
D Listener Average	613.333
E Listener Average	661.667

Reaction time data (in ms), before full processing

Full Name*	#	Cond.	1	2	3	4	5	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	Per item Mean
1One1A	1	A1		776	658	668	570	651	695	633	603	862	543	984	814	1137	828	592	519	456	373	517	1297	708.80
11Two6A	11	A2	735	790	625	579	486	723	518	524	607	496	875	976	569	732		578	405	441	418	545	563	609.25
20Seven1A	20	A3	567		662	562	409	669	422	508	599	420	508	747	544	650	678	565	616	283		418	655	551.68
29Five6A	29	A4	467	691	615	686	563	821	546	670	705		699	999	503	1033	752	689		337	459	598	674	658.26
2Two2B	2	B1	847	816	780	724	529	1504	706	945	736	590	699	895	781	943	935	604	482	299	784	590	1423	791.05
12Six8B	12	B2	879		812	793	546	778		402	647		587		536	837	807		367	263	602	373	909	633.63
23Eight2B	23	B3	464	674	705	455	294	626	888	604	550	500	585	784	631	574	836	589	359	223	590	493	638	574.38
30Nine7B	30	B4	465	717	625	469	347	579	523	473	590	653	475	780	607	680	658	544	505	254	454	621	560	551.38
4Three3C	4	C1	583	718	754	587	491		600	657	770	745	862	926	643	812	1051	668	410	379	456	417		659.42
14Three7C	14	C2	666	1354	712	650	424	890	681	656	677	500	517	650	518	614	712	536	432	315	561	333	688	623.14
25Six3C	25	C3	868	741	794	338	317	750	1001	861	473	836	602	861	432	822	708	549	461	431		635	976	672.80
33Seven8C	33	C4	468	525	528	411	315	665	669	503	503	523	482	656	531	581	821	496	554	294	554	396	654	529.95
6Four4D	6	D1	769	605	658	708	606	2197	577	804	663	708	846	819	567	761	910	643	218	281	521	417	767	716.43
16Four9D	16	D2	656	1323	630	606	471	697	547	685	583	755	591	964	510	959	730	680	586	495	505	427	625	667.86
26Ten4D	26	D3	529	836	623	626	397	861	603	103	504	578	533	606	470	686	744	534	388	361	570	410	617	551.38
34Nine9D	34	D4	421	655	590	320	246	531	628	478	440	909	370	736	395	885	671	459	386	358	486	345	562	517.67
7One5E	7	E1	774	682	738		550	2029	686	919	555	623	760	1207	634	774	835	610	219	287	641	469	749	737.05
18Five10E	18	E2	457	633	612	481	318	634	574	849	702	405	550	823	512	535	696	579	330	434	674	813	579	580.48
28Eight5E	28	E3	679	765	658	568	372	706	665	931	618	536	542	792	436	577	852	432		450	513	671	708	623.55
37Ten10E	37	E4	509	731	792	736	462	700	703	644	632	933	707	600		935	903	686	432	321	478		957	676.89
		Mean	621.21	779.56	678.55	577.21	435.65	895.32	643.79	642.45	607.85	642.89	616.65	831.84	559.63	776.35	796.16	580.68	426.06	348.10	535.50	499.37	768.47	631.75

***Stimulus full names are of the form [Stimulus#][Speaker Number][Sentence#][Condition]**

Ex. 14Three7C is stimulus 14, consisting of sentence seven, spoken by speaker three, prepared to Condition C

Per Listener Condition Averages

	1	2	3	4	5	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
A	589.67	752.33	640.00	623.75	507.00	716.00	545.25	583.75	628.50	592.67	656.25	926.50	607.50	888.00	752.67	606.00	513.33	379.25	416.67	519.50	797.25
B	663.75	735.67	730.50	610.25	429.00	871.75	705.67	606.00	630.75	581.00	586.50	819.67	638.75	758.50	809.00	579.00	428.25	259.75	607.50	519.25	882.50
C	646.25	834.50	697.00	496.50	386.75	768.33	737.75	669.25	605.75	651.00	615.75	773.25	531.00	707.25	823.00	562.25	464.25	354.75	523.67	445.25	772.67
D	593.75	854.75	625.25	565.00	430.00	1,071.50	588.75	517.50	547.50	737.50	585.00	781.25	485.50	822.75	763.75	579.00	394.50	373.75	520.50	399.75	642.75
E	604.75	702.75	700.00	595.00	425.50	1,017.25	657.00	835.75	626.75	624.25	639.75	855.50	527.33	705.25	821.50	576.75	327.00	373.00	576.50	651.00	748.25

A Listener Average	630.56
B Listener Average	640.62
C Listener Average	622.20
D Listener Average	613.33
E Listener Average	647.18
Average of all test stimuli	631.1542

Reaction time data (in ms) with inaccurate responses and outliers removed