A Brief Introduction to Vowel Resynthesis for Speech Perception Research

Daniel Lawrence, University of Edinburgh
dlawrenc@staffmail.ed.ac.uk
www.danielplawrence.com

Introduction

When preparing speech perception experiments, we usually have to consider a trade-off between the level of control we have over the phonetic parameters of speech stimuli and their naturalness. While commercially-available speech synthesis packages allow very fine-grained control over the output of vowel synthesis, the resulting stimuli often have a very robotic quality. The reason for this is that while it is relatively straightforward to computationally model the vocal tract resonances, irregularities in the voicing source are very hard to imitate.

A potential solution to this problem is to attempt to estimate the characteristics of the voicing source from natural speech, then pass this signal through a set of digital filters representing the desired vocal tract resonances. This enables full control and manipulation of formant structure, whilst retaining a fairly natural quality.

In this document I will explain how this is possible using *Praat*. First, I will give a brief overview of the intuition behind inverse filtering, which is at technique for estimating the contribution of the voicing source from a natural vowel token. Next, I'll briefly discuss how continuum steps can be calculated, since generating vowel continua is a typical goal for perception studies. The final section of this document explains how the accompanying scripts can be used to implement these methods.

Inverse Filtering

A key concept underpinning most speech synthesis methods is the source-filter model of speech production, first introduced in Fant (1960). This model allows us to describe speech signals very precisely, by splitting the speech signal into:

- The voicing source, representing the contribution of the vocal folds vibrating as air passes through them from the lungs.
- The vocal tract transfer function, representing the contribution of the vocal tract setting in boosting and dampening different frequencies in the signal.
- The lip radiation effect, representing the effect of speech leaving the lips and escaping out into the world.

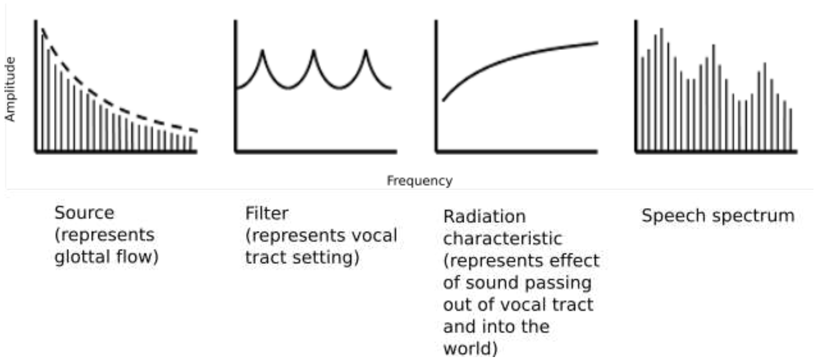

Amplitude
Frequency
Source (represents glottal flow)
Filter (represents vocal tract setting)
Radiation characteristic (represents effect of sound passing out of vocal tract and into the world)
Speech spectrum

Figure 1: Fant (1960) – the source-filter model of speech production

Digital speech synthesizers simulate each of these processes. As mentioned in the introduction, the problem with most synthesis methods is that simulating the voicing source (left-hand pane above) is very difficult. This is because natural glottal vibrations contain lots of irregularities, which are hard to generate synthetically.

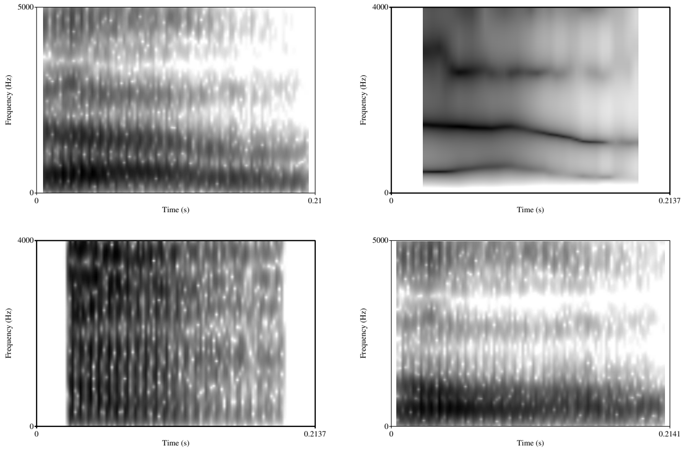


Figure 2: Top left: Original vowel token; Top right: Smoothed spectrogram estimated by linear prediction; Bottom left: Estimated glottal source; Bottom right: Resynthesized vowel.

To solve this problem, it is possible to estimate the glottal flow from a natural speech sound. Using a technique called *linear prediction*, we can estimate the spectral envelope of a speech sound (the peaks of which represent the formants). Once we have a representation of the spectral envelope, we can use it as an inverse filter, effectively canceling out the effect of the vocal tract resonances and leaving us with a representation of the voicing source. Figure 2 gives an example of this process, where I have taken a natural token of [əʊ] and lowered the second formant to create a backer, monophthongal vowel [oː]. The top left-hand image shows a spectrogram of the natural speech sample. The smoothed spectrogram in the top right-hand image is the result of linear prediction, and represents the estimated contribution of the vocal tract resonances to the speech signal. The bottom left-hand image shows the result of inverse filtering – a canceling out of the formants seen in the original spectrogram. Finally, the bottom left-hand image shows the result of passing the estimated source signal through a modified filter, producing a vowel with a lower F2.

In reality, there are a couple of difficulties with the inverse-filtering method. Firstly, achieving full separation of the source and filter representations is very difficult. If this stage doesn't work properly, the stimuli may contain 'phantom' formants, left over from the original speech token. To solve this problem, Alku (1992) has developed a method for highly accurate source estimation, which basically involves using the inverse filtering process iteratively. Figure three shows the difference – the dark bands highlighted in the left-hand panel are examples of residual formant structure. You should see that Alku's (1992) method results in a much 'whiter' spectrogram, with very little evidence of residual formant-like bands.

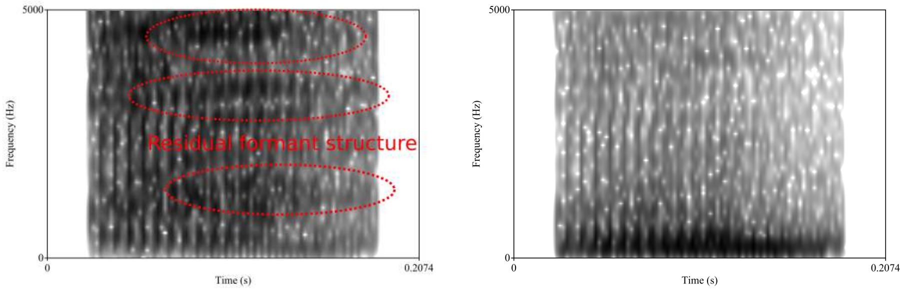


Figure 3: Left: spectrogram of inverse-filtered glottal source; Right: spectrogram of glottal source estimated using Alku's (1992) method.

A second problem is that LPC decomposition usually results in the loss of the higher frequency component of the original sound. While this doesn't affect our perception of vowel quality, it can make the stimuli sound quite muffled, and causes issues when attempting to embed the synthesized vowels in carrier tokens. To solve this problem, the high frequency component of the original sound can be added to the synthesis output (credit for this idea goes to Matt Winn – `mattwinn.com`)

Restoring the higher frequency component of the original signal improves the naturalness of the resulting stimuli considerably, and allows the resynthesized vowels to be placed in consonantal contexts without the need to downsample the carrier tokens.

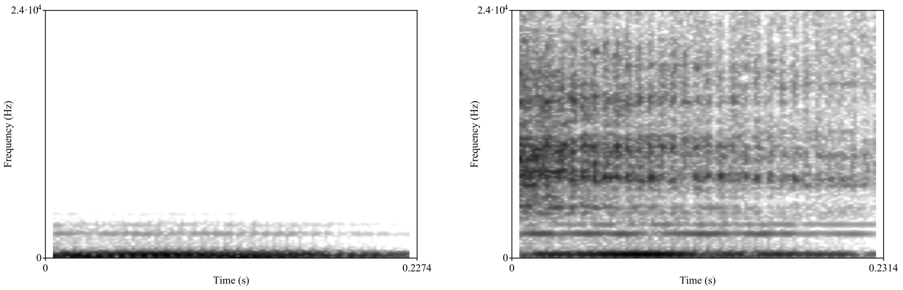


Figure 4: Left: spectrogram of resynthesized vowel showing frequencies up to 24KHz; Right: resynthesized vowel with higher frequencies restored.

Estimating Continuum Steps

Now that we have established a method for generating the stimuli, all that remains is to calculate the formant values for the stimuli items. One thing we might want to do is create a continuum between two endpoints – for example, we might want to create a continuum between [i] and [u] for an experiment investigating the perception of vowel fronting.

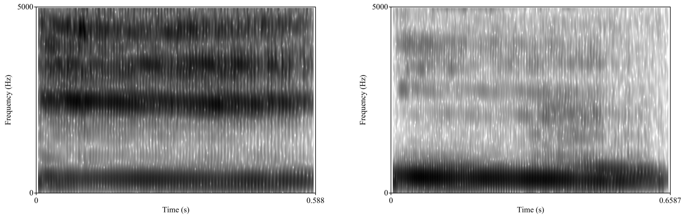


Figure 5: Left: spectrogram of natural token of [i]; Right: spectrogram of natural token of [u]

The method for doing this is fairly intuitive. First, the samples are matched in the time domain by stretching or shrinking one of them using an algorithm called *Time-Domain Pitch-Synchronous Overlap and Add*, which allows the length of a sound to be modified without changing the pitch. The formants can then be estimated using linear prediction, and the target formant contours can be estimated by interpolating between a number of points along the formant contours of the two vowels. The more measurement points used here, the more dynamic information can be captured in the interpolation.

The spectrogram below shows an eight-step continuum from [i] to [u] generated in this manner:

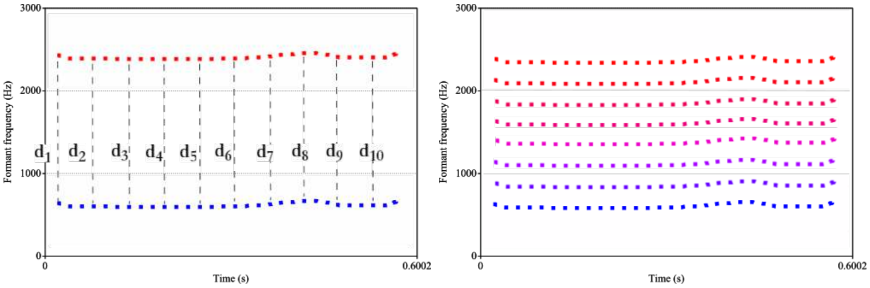


Figure 6: Example of interpolation between the second formant of a front and back vowel – distances are measured at a number of time points, allowing intermediate values to be calculated at equal steps

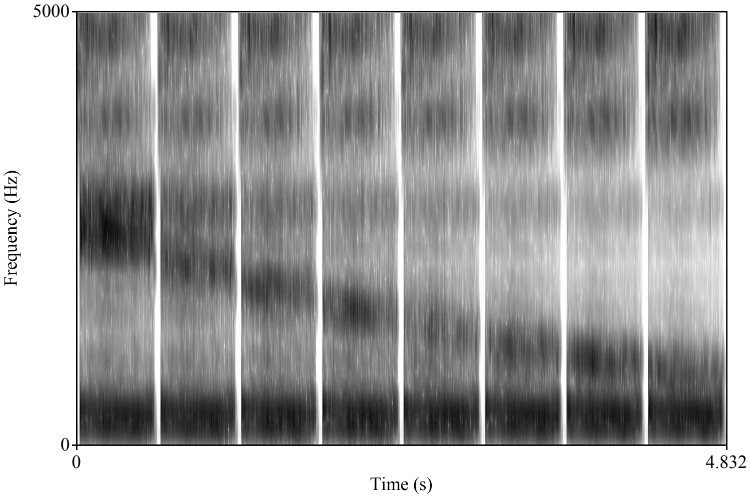


Figure 7: Eight-step semisynthetic vowel continuum from [i] to [u].

Example scripts

These scripts are included to encourage others to learn about the methods discussed here. Please feel free to use them in your teaching and research, and do get in touch if you have any questions or comments regarding them.

klatt_ipa_vowels.praat

I have included this to demonstrate the output of typical (fully-synthetic) speech synthesis. This script generates a set of synthetic IPA vowels based on the formant values provided on Bruce Hayes' website (`http://www.linguistics.ucla.edu/people/hayes/103/Charts/VChart/`). To use the script, just run it in Praat – a sound and TextGrid containing the vowels should appear in the objects window.

iaif_ipa_vowels.praat

This script does the same thing as the one above, but uses a voicing source estimated from natural speech. I have included it to demonstrate the considerable increase in naturalness that this method brings. It is very quick to use, so ideal for demonstrating the method. To use the script:

1. Record yourself saying a word with a clear vowel
2. Click 'View & Edit' to view your recording, and select the vowel portion.
3. Click 'File', then 'Extract selected sound (time from 0)'
4. Close the editor window
5. Select the extracted sound in the objects window
6. Run the script

You should find a sound and TextGrid object in the objects window, which contain the semisynthetic vowels.

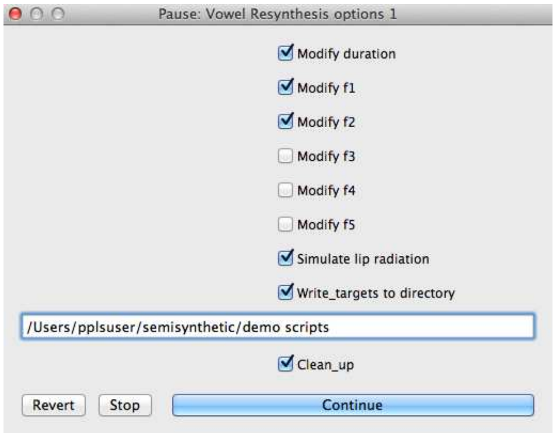
vowel_resynthesis_iaif.praat

This script implements the full procedure described in the previous section, using the method described in Alku et al. (1999). The script creates semisynthetic vowel continua and embeds them in natural consonantal contexts, ready to be used in perceptual experiments. There are few extra steps involved, including estimating the pitch and amplitude contours of the original speech sound and ensuring that they are constant across the output tokens. All parameters used in the synthesis procedure can be written to a set of log files, making it easy for the researcher to accurately report their methods.

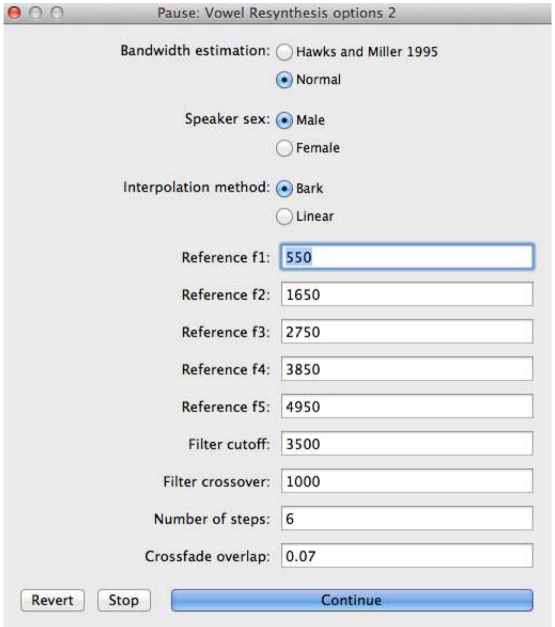
To use the script, you need two tokens of the desired word which represent the endpoints of the continuum. Record or open those in Praat, then run the script.

Initial options

The script will prompt you for two sets of options. If you are trying the program for the first time, you will probably get away with just using the defaults. However, tinkering with these options can result in higher quality synthesis output.


Pause: Vowel Resynthesis options 1

☑ Modify duration
☑ Modify f1
☑ Modify f2
□ Modify f3
□ Modify f4
□ Modify f5
☑ Simulate lip radiation
☑ Write_targets to directory
/Users/pplsuser/semisynthetic/demo scripts
☑ Clean_up


Pause: Vowel Resynthesis options 2

Bandwidth estimation: ○ Hawks and Miller 1995 ● Normal
Speaker sex: ● Male ○ Female
Interpolation method: ● Bark ○ Linear
Reference f1: 550
Reference f2: 1650
Reference f3: 2750
Reference f4: 3850
Reference f5: 4950
Filter cutoff: 3500
Filter crossover: 1000
Number of steps: 6
Crossfade overlap: 0.07

- **Modify duration:** Selecting this option will modify the duration of each token at equal steps from the A endpoint to B endpoint of the continuum.

- **Modify f1-f5:** Allows the use to decide which formants will be modified when creating the continuum. Any formants which are successfully tracked in the source token will be included in the synthesis output, but they will only be modified if selected here

- **Simulate lip radiation:** Attempts to estimate a more accurate model of the glottal source by factoring out the lip radiation effect. This is achieved by integrating the original signal (following Alku et al. 1999). The lip radiation effect is restored after the vowel has been synthesized.

- **Write targets to directory:** Writes the synthesis output, log file, measurements and original tokens to the specified directory.

- **Clean up:** Removes everything except for essential items from the objects window after synthesis.

- **Bandwidth estimation:** Bandwidths can either be set as fixed values, or estimated from the frequency values using coefficients calculated by Hawks & Miller (1995). Despite the suggestions of that paper, I have found that 'Normal' gives the best results.

- **Speaker sex:** Used for the Hawks & Miller (1995) method above.

- **Interpolation method:** Continuum steps can be calculated in the Hz ('Linear') or Logarithmic ('Bark') domain. If 'Bark' is chosen, Traunmüller's (1990) Bark-Hz conversion formulae are used.

- **Reference f1-f5:** Used for formant tracking – can be modified if Praat has trouble estimating formants.

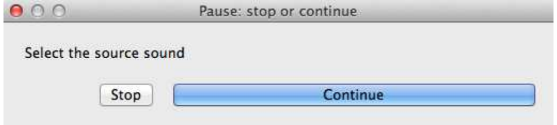
- **Filter cutoff:** This is used when restoring the HF component of the original signal. This value should be higher than the highest formant you wish to manipulate.

- **Filter crossover:** As above – this value specifies the filter skirt. I suggest experimenting with this to see what gives you the best results.

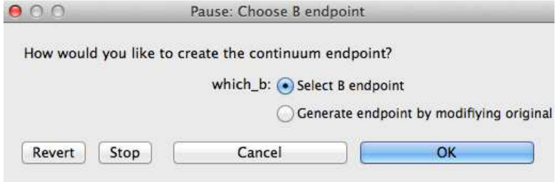
- **Number of steps:** How many continuum steps to synthesize

- **Crossfade overlap:** When embedding vowel tokens in a carrier token, the script will crossfade between them, in an attempt to avoid audible artifacts of the concatenation. This value controls how much overlap there is between the concatenated sections.

Selecting the A and B endpoints


Pause: stop or continue

Select the source sound

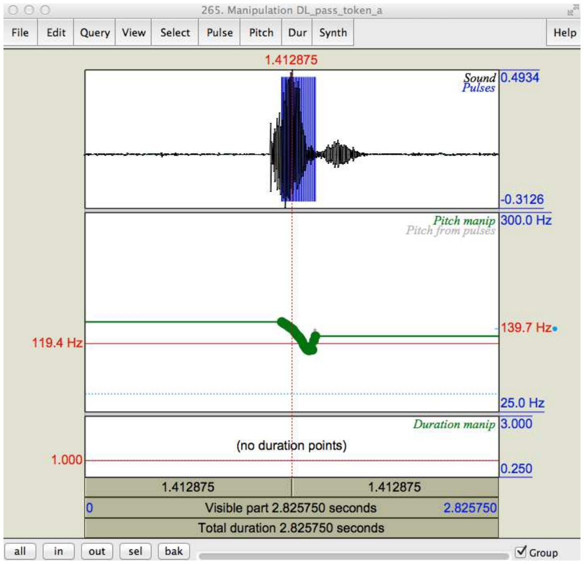

Pause: Choose B endpoint

How would you like to create the continuum endpoint?

which_b: ● Select B endpoint
○ Generate endpoint by modifying original

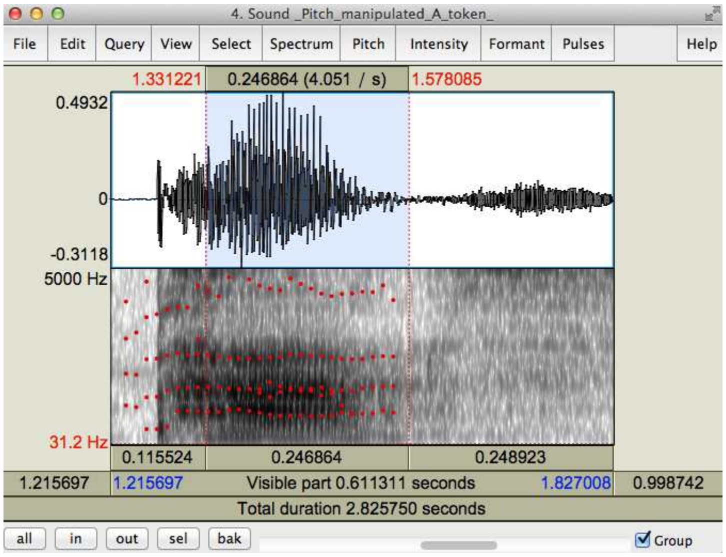
After setting the initial options, the script will prompt you to select the 'A' token for your continuum. This is the one which will be used when estimating the glottal source. Select the sound in the objects window, and click 'continue'. The script will then ask you how it should create the 'B' endpoint. You can either choose to modify the original, or choose to select an existing sound from the objects window.

Checking and modifying the pitch tracking



The script will prompt you to check the pitch tracking. This will allow you to make modifications to the pitch of the output tokens if you desire – the script will synthesize vowel tokens with this pitch contour. Click 'Continue' to proceed to the next stage.

Segmenting the source and target vowel

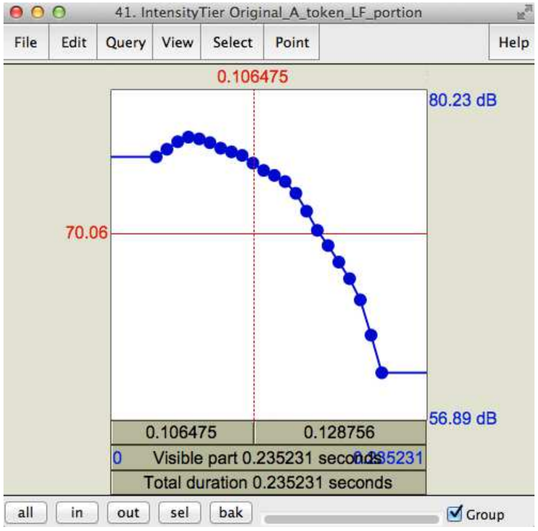


The script will prompt you to select the target segments in the 'A' and 'B' tokens. Segment the vowels carefully, and click 'Continue'. You should aim to select a clear voiced portion of the target vowels. Synthesis will still work as long as you select something with formant structure, although I suggest experimenting with different segmentation strategies to find the approach which generates the best results. Praat will automatically write the start and end time of your selections to the log file.

Segmenting the preceding and following context

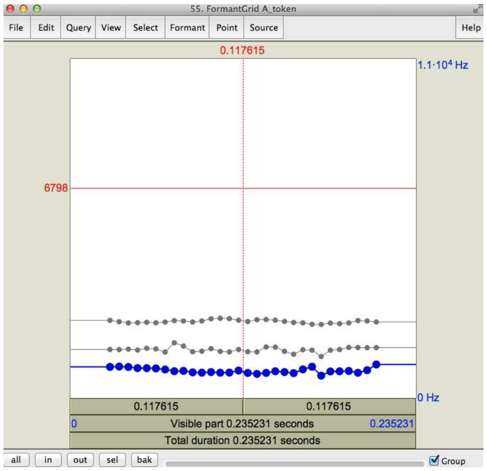
The script will ask you how you would like to generate the preceding and following context. These will be concatenated together with the synthesized vowel, allowing you to create word tokens. You can choose to use silence, or choose to extract the context from token 'A' or 'B'. This will proceed in the same way as the vowel selection in the previous stage.

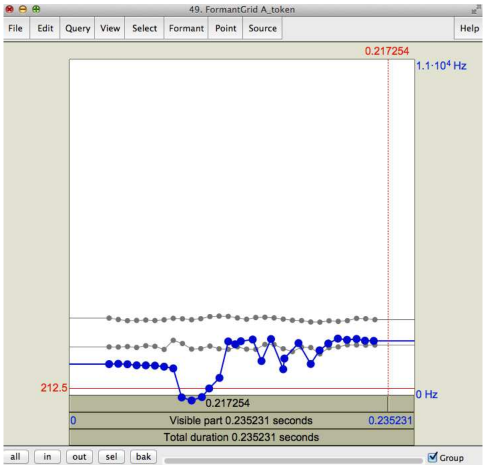
Checking and modifying the intensity contour



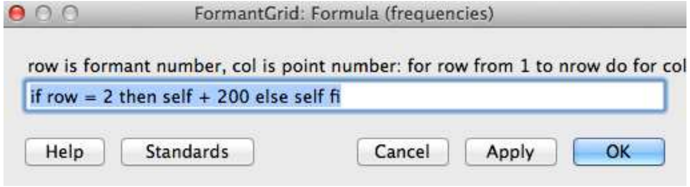
Praat will display an intensity contour which can be manipulated by clicking and dragging the blue points. The script will ensure that all tokens are matched for intensity using this contour. This can usually be left alone, although it may help in improving the naturalness of word tokens.

Checking the FormantGrids





Praat will display a FormantGrid object, representing its formant measurements for each endpoint token. It will use these values for the interpolation of the continuum, so it is very important to check that there are no errors – typical things which will cause an issue are when formants cross over, or when there are negative formant values. These errors can be seen in the right-hand panel of figure above. The formants can be modified by clicking and dragging the blue dots, or by formula. To modify a formant using a formula, go to the objects window, and click 'Modify...' then 'Formula (frequencies)'.



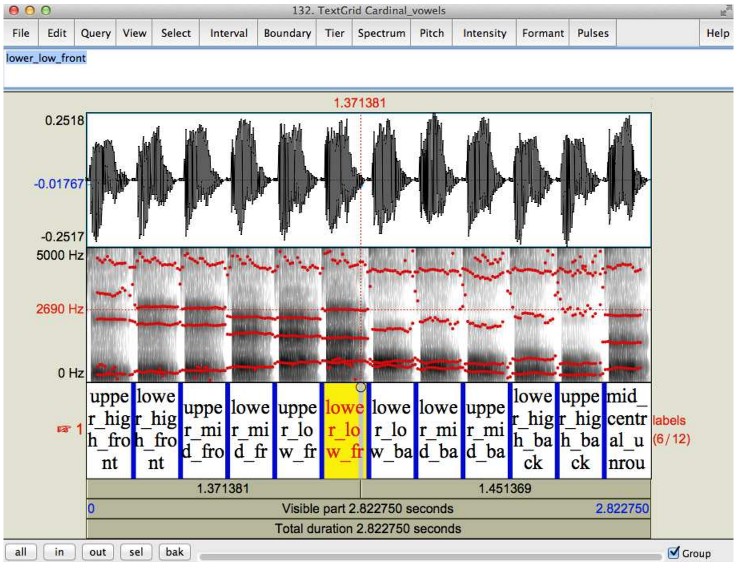
FormantGrid: Formula (frequencies)

row is formant number, col is point number: for row from 1 to nrow do for col

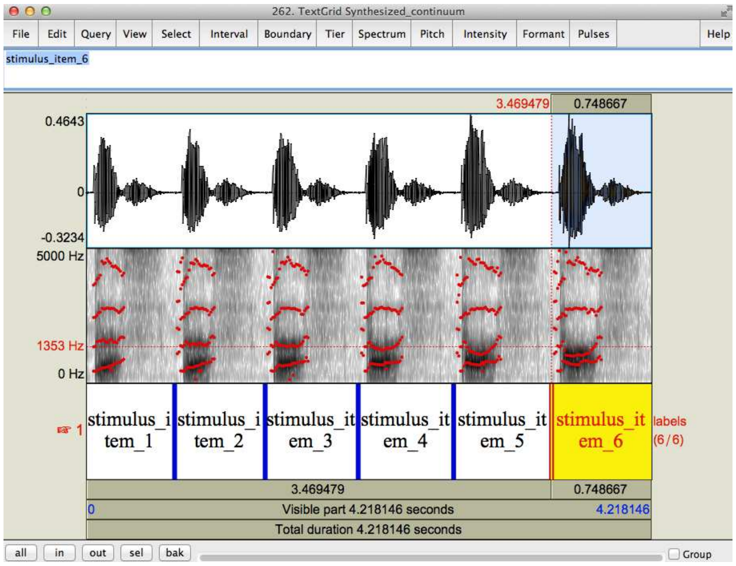
```
if row = 2 then self + 200 else self fi
```

This will allow you to make global changes to the frequencies. The formula above would raise F2 by 200Hz. Another useful trick is to delete unnecessary points in the FormantGrid by selecting the target formant (Ctrl/Cmd + formant number) and deleting the points (Ctrl/Cmd+alt+t). Praat will linearly extrapolate any points which are missing, so formants can be modified without having to click and drag every point. Pressing 'tab' while viewing the FormantGrid will give you an impression of the vowel quality this grid will produce.

Checking the quality of the source estimation



After the FormantGrids have been checked and modified, the script will produce a set of IPA vowels using the voicing source estimated from the natural speech sample you provided. Listen to these vowels – if you notice any strange sounds or extra formants, it might be worth running the script again or selecting a higher-quality source token. Clicking ‘Continue’ should create the continuum.



Enjoy!

References

- Alku, P., Tiitinen, H., & Näätänen, R. (1999). A method for generating natural-sounding speech stimuli for cognitive brain research. Clinical Neurophysiology, 110(8), 1329-1333.

- Alku, P. (1992). Glottal wave analysis with pitch synchronous iterative adaptive inverse filtering. Speech communication, 11(2), 109-118.

- Fant, G. (1960). Acoustic theory of speech production. Walter de Gruyter.

- Traunmüller, H. (1990). Analytical expressions for the tonotopic sensory scale. The Journal of the Acoustical Society of America, 88(1), 97-100.