Vowel Resynthesis for Speech Perception Research

Daniel Lawrence

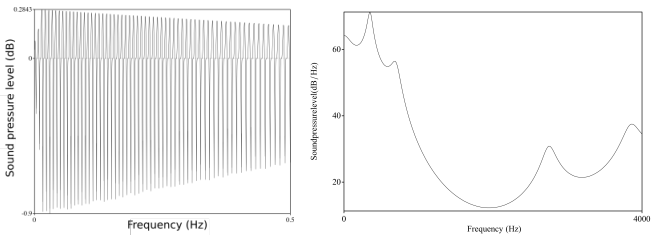
THE UNIVERSITY *of* EDINBURGH

11 March 2016

I. Basics of vowel acoustics

Source-filter model

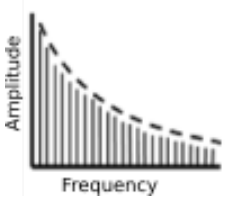
- Treat the speech signal as a combination of an excitation source and vocal tract resonances, plus the effect of radiation from the lips.
- With this intuition, all we need to do is find a way of digitally simulating these components:
  - The voicing source as a series of periodic pulses
  - The vocal tract resonances as a digital filter which boosts and attenuates frequencies of the source depending on the vocal tract setting to be simulated
  - Simulating the voicing source as the *derivative* of the glottal wave means we no longer need to worry about the lip radiation effect.



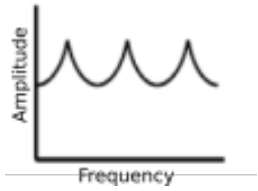
(a) Voicing source simulated in *Praat*

(b) Frequency response of digital filter created in *Praat*

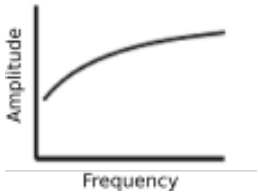
- The next section will demonstrate how these elements can be generated in *Praat*. There are (at least) two ways of doing this – using the built-in signal generation and filtering tools to generate a signal from scratch, or by using Praat's implementation of the Klatt synthesizer.



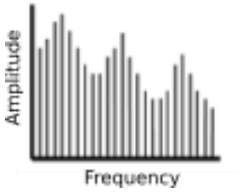
(a) Voicing source



(b) Vocal tract resonances



(c) Lip radiation effect



(d) Speech spectrum

Figure 1: Source-filter model of speech production, after Fant (1970)

II.Acoustic synthesis in Praat

Simulating the glottal source

- First, we generate an impulse train using a *PointProcess* object:

[1]

```
pitchTier = Create PitchTier: "source", 0, 0.5
Add point: 0.0, 150
Add point: 0.5, 100
pulses = To PointProcess
source = To Sound (phonation): 44100, 0.6, 0.05, 0.7, 0.03, 3.0, 4.0
removeObject: pitchTier, pulses
selectObject: source
View & Edit
```

[1] I've presented these examples in code, but because the Praat scripting language is based around editing commands, you should be able to replicate this code by finding the appropriate commands in the editor. Similarly, the meaning of each argument can be found in the editor, or by searching for the command under 'Help'

- Notice that this sound has a relatively smooth frequency responses, which is exactly what we want in order to simulate the glottal source:

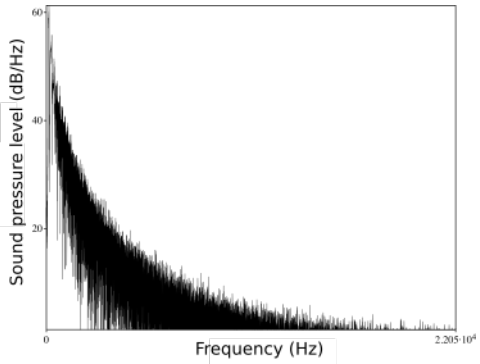


Figure 2: Spectrum of glottal wave simulated in *Praat*. The amplitude of each frequency component decays at around 6dB/Hz

Simulating the vocal tract resonances

- Praat can represent the vocal tract filter as a FormantGrid object. The code below creates one of these with a length of 0.5s, 10 formants, an initial formant frequency of 550Hz, with higher formants spaced at 1100Hz.
- The code also makes modifications to the onset of the vowel to create the impression of a transition from some (bilablial?) consonant.

```
filter=Create FormantGrid: "filter", 0, 0.5, 10, 550, 1100, 60, 50
Remove formant points between: 1, 0, 0.5
Add formant point: 1, 0.00, 350
Add formant point: 1, 0.05, 700
Remove formant points between: 2, 0, 0.5
Add formant point: 2, 0.00, 700
Add formant point: 2, 0.05, 1100
selectObject: filter
View & Edit
```

Filtering the FormantGrid and the source together produces a vowel-like sound:

```
selectObject: source,filter
vowel=Filter
selectObject: vowel
View & Edit
```

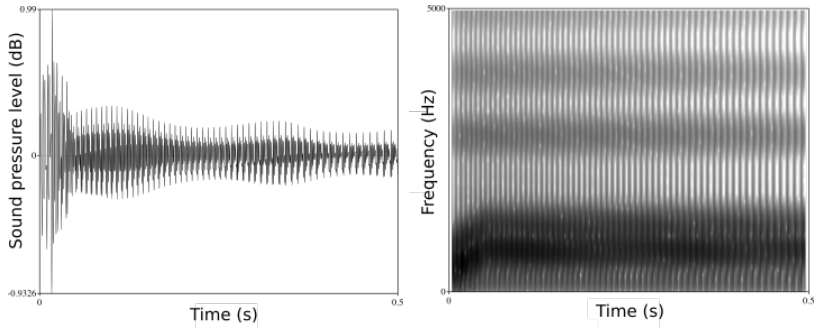


Figure 3: Synthetic vowel created through source-filter synthesis in *Praat*

Klatt synthesis in Praat

- Praat also includes an implementation of the Klatt synthesizer.
- To use this, specify parameters as a function of time in a 'KlattGrid' object.
- First, create the KlattGrid:

```
#Create a KlattGrid
Create KlattGrid... aa 0 0.5 6 1 1 6 1 1 1
```

- We can add points to the KlattGrid which represent speech parameters at a given time point.
- Given a point, the synthesizer will assume that this value extrapolates constantly unless told otherwise
- The code below adds pitch and amplitude information:

```
#Add voicing amplitude, vowel formants, and pitch targets
Add voicing amplitude point... 0.0 0
Add voicing amplitude point... 0.04 90
Add voicing amplitude point... 0.25 90
Add voicing amplitude point... 0.5 90
Add pitch point... 0.0 150
Add pitch point... 0.5 150
```

- Now we can add points representing the formant frequency and bandwidth values of a vowel:

```
Add oral formant frequency point... 1 0.1 750
Add oral formant bandwidth point... 1 0.1 70
Add oral formant frequency point... 2 0.1 1250
Add oral formant bandwidth point... 2 0.1 120
Add oral formant frequency point... 3 0.1 2500
Add oral formant bandwidth point... 3 0.1 200
Add oral formant frequency point... 4 0.1 3900
Add oral formant bandwidth point... 4 0.1 300
#Synthesis
Play
To Sound
```

- This paper gives details of all of the many parameters which can be set: `http://www.fon.hum.uva.nl/david/ba_shs/2009/is2009_klattgrid.pdf`

Acoustic synthesis: review tasks

- Try modifying the code in source_filter.praat to alter the pitch contour of the vowel
- The vowel produced by source_filter.praat has a unnaturally flat intensity contour. Could you modify this using an IntensityTier? (see ba-da_continuum.praat for an example)
- Try modifying the code in source_filter.praat to create different vowels.[2]
- Try modifying the code in ba-da_continuum.praat to create a different kind of continuum (e.g. [i] to [u])
- Try running klatt_cardinal.praat to generate a set of vowels using the Klatt synthesizer

[2] Here are some formant frequencies to try:

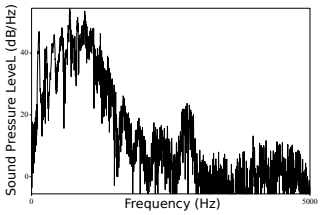
	F1	F2	F3
[i]	294	2343	3251
[e]	434	2148	2763
[ae]	766	1782	2398
[a]	781	1065	2158
[o]	334	910	2300
[u]	295	750	2342

III. Spectral envelope estimation and inverse filtering

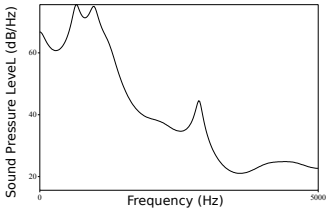
- The methods discussed so far have relied on us specifying speech parameters manually. This is particularly problematic for modeling the glottal source – if we treat it simply as an impulse train it tends to sound quite robotic.
- The intuition behind inverse-filtering is that we can attempt to separate the source and filter of a natural speech sound, and use the natural source to excite a digital filter.

Inverse filtering intuition

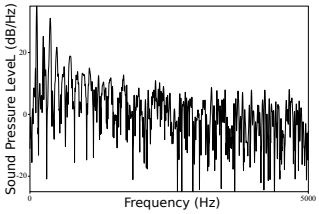
- Intuition: estimate the spectral envelope of a natural sample, then use it as an *inverse filter* to 'undo' the filtering effect of the supralaryngeal vocal tract.



(a) Natural speech



(b) Spectral envelope



(c) Inverse filtered spectrum

- The signal (c) can now be used in place of the digital simulation of the glottal wave we used in the previous section.

Linear Predictive Coding

- The value of a speech signal at any given time tends to be correlated with the directly previous values
- This means that the signal can be represented as a linear function, predicting the value at time t as a combination of the values at time $t-1$, $t-2$, ...

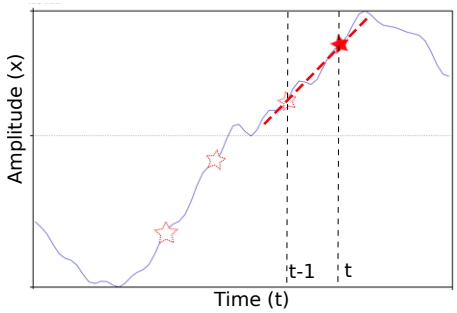


Figure 4: LPC of order 1: Predicting the value of a signal from a previous sample

$$x(t) = ax(t-1)$$

- Figure 4 visualises linear prediction of order 1 – using only the previous sample. For spectral envelope estimation, we estimate functions of orders around 10 (roughly twice the number of peaks we want to detect)
- The linear model estimated will be more or less in error as we move through the signal, as shown in Figure 5:

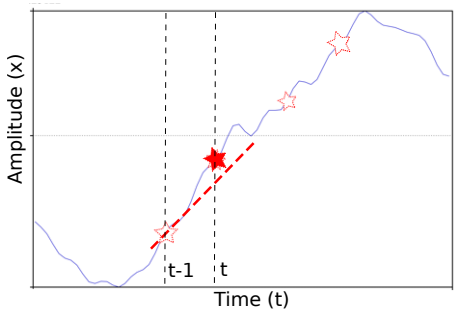


Figure 5: Example of prediction error

- This means that the LPC coefficients cannot represent the whole signal on their own.
- However, the coefficients plus the LPC *residual*, which is a measure of error at each time point, encode the whole signal

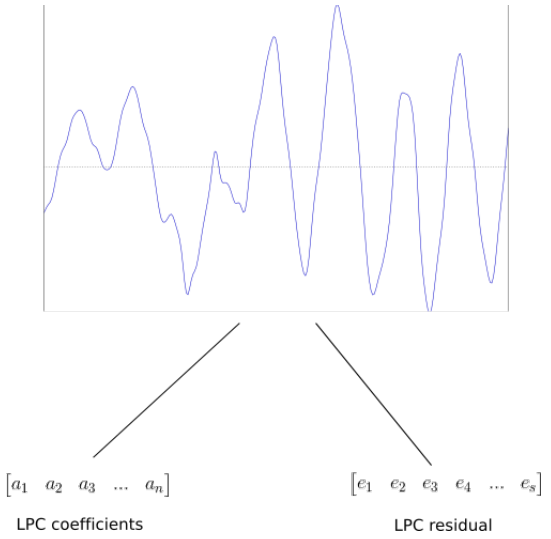
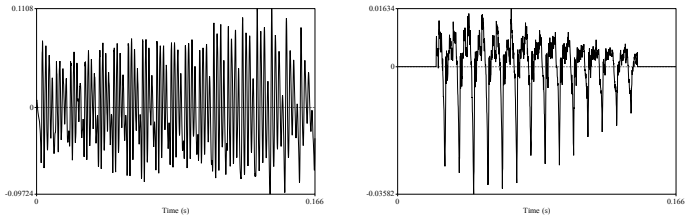
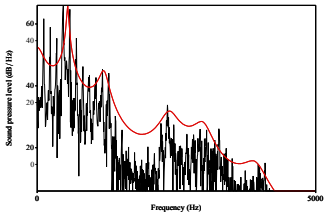


Figure 6: The LPC coefficients and prediction residual encode the entire signal

- Because errors tend to happen where the signal is changing the fastest, LPC has the property of tending to separate spectral information and glottal pulses, as the largest changes in the signal correspond to pitch periods.



(a) Natural speech (b) LPC residual



(c) Natural spectrum and LPC-estimated spectral envelope

Figure 7: Estimation of glottal wave and spectral envelope using LPC analysis

- The estimated glottal wave can then be used to excite a digital filter, resulting in (hopefully!) more natural-sounding samples than would be produced using a digitally-simulated wave.

Inverse filtering using Praat

- In order to perform inverse filtering in *Praat*, the following steps are required:
  1. Convert the natural sample to mono
  2. Resample to around 10000 Hz ($\sim 2 * nf * 1000$)
  3. Perform LPC analysis with order $\sim 2 * nf + 2$
  4. Inverse filter resampled original with LPC object
  5. Estimate formants using 'To Formant...'
  6. Manipulate formants using 'To FormantGrid'
  7. Filter modified FormantGrid with inverse-filtered source
- Code for doing this might look as follows:

Resampling and LPC analysis:

```
#Estimate the LPC filter for a selected sound
#First we need to convert to mono and resample
Convert to mono
Resample: 10000, 50
To LPC (burg): 10, 0.025, 0.005, 50
```

Inverse filtering:

```
#Take the inverse of this filter to get a representation of the source
selectObject: "Sound untitled_10000"
plusObject: "LPC untitled_10000"
Filter (inverse)
```

Estimating a digital filter from the original sample and modifying it:[3]

```
#Generate a formant object and add 400 Hz to F2
selectObject: "LPC untitled_10000"
To Formant
selectObject: "Formant untitled_10000"
Formula (frequencies): "if row = 2 then self + 400 else self fi"
```

[3] An alternative here would be to convert this Formant object into a FormantGrid, which would allow manual modification of frequencies over time.

Exciting the modified filter with the LPC-estimated glottal wave:

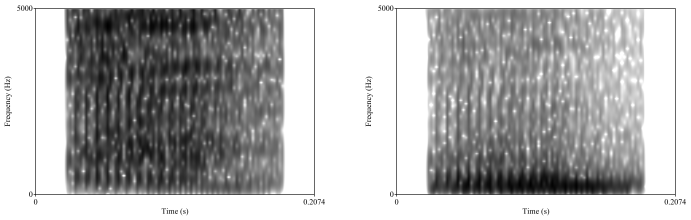
```
#Combine the source and filter representations to make a new vowel
selectObject: "LPC untitled_10000"
selectObject: "Sound untitled_10000"
plusObject: "Formant untitled_10000"
Filter
Play
```

Advanced inverse filtering methods

- Two issues with inverse filtering:
  a) Difficult to get completely 'white' source representation
  b) We lose the higher-frequency component of the sound during LPC decomposition

Iterative Adaptive Inverse Filtering

- Method described in Alku (1992)
- Involves performing LPC analysis of different orders to achieve very good source-filter separation



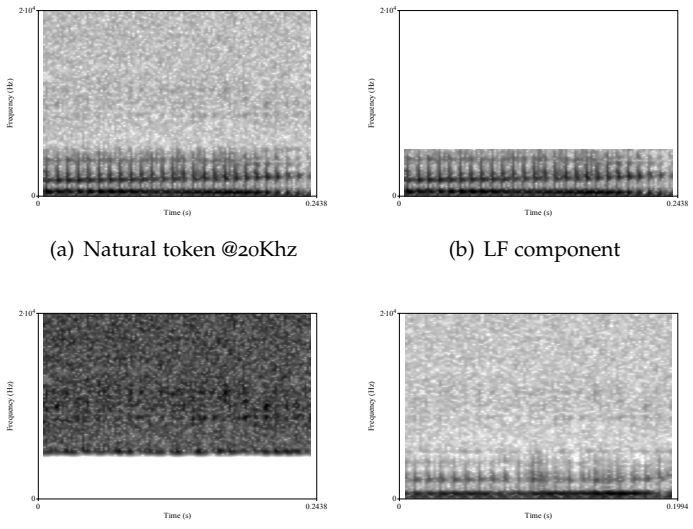
(a) Spectrogram of LPC estimated source

(b) Spectrogram of IAIF estimated source

Figure 8: Comparison of glottal estimates using basic inverse filtering and Alku's (1992) IAIF. You may notice that the lower frequencies are particularly prominent in the IAIF example. I may be implementing this method wrong – need to check!

HF component restoration

- In order to compensate for the loss of higher frequencies in LPC analysis, the HF component can be restored after manipulation of the LF component.



(a) Natural token @20Khz

(b) LF component

(c) HF component

(d) Semisynthetic token

Figure 9: Example of HF component restoration – the natural sample is filtered into higher and lower frequency components (with a cut-off of e.g. 5000Hz). Manipulation is carried out on the LF portion only, and the HF portion is restored after resynthesis.

- These techniques are implemented in the scripts formant_manipulation.praat and vowel_resynthesis_iaif.praat

Inverse filtering review tasks

(a) Try recording yourself producing a vowel (ctr+r). Select your vowel token and run 'LPC_cardinal_iaif.praat'. Compare the output with that of 'klatt_cardinal.praat'

(b) Experiment with the 'formant_manipulation.praat' script:

   i. Create a 'ba-da' continuum

   ii. Create an [i]-[u] continuum

   iii. Create a continuum between a diphthong and monophthong

IV. The TANDEM-STRAIGHT vocoder

Overview

- `http://www.wakayama-u.ac.jp/~kawahara/STRAIGHTadv/index_e.html`
- Used in many commercial text-to-speech applications
- Uses advanced techniques to estimate the glottal flow and spectral envelope, as well as the aperiodic components of natural speech.

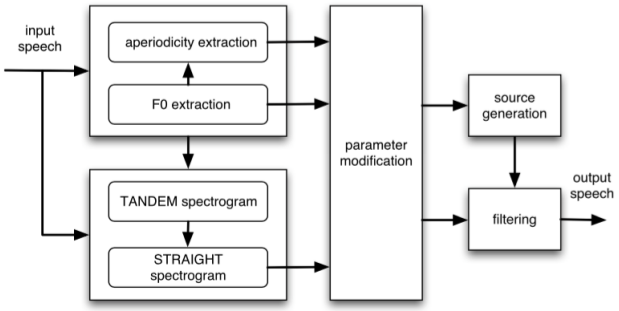


Figure 10: TANDEM-STRAIGHT schematic

Speech morphing

- Morphing GUI allows interpolation between two samples
- TANDEM-STRAIGHT GUI:

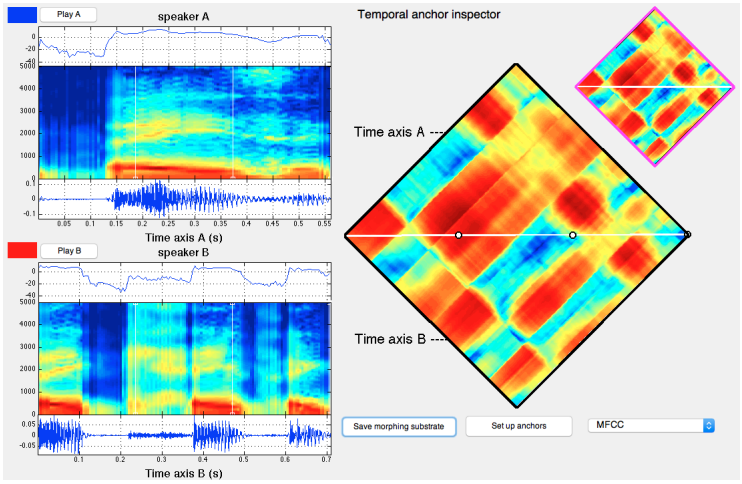


Figure 11: TANDEM-STRAIGHT Morphing GUI

- TANDEM-STRAIGHT emotional morphing demo:

- www.wakayama-u.ac.jp/~kawahara/Miraikandemo/index-e.html
- Less useful for manipulating individual formants (but still possible)

TANDEM-STRAIGHT review tasks

- Download TANDEM-STRAIGHT and see if you can get it to work!
- You will need to install the MATLAB runtime for your system from http://uk.mathworks.com/products/compiler/mcr/
- The whole set of routines is available here: https://www.dropbox.com/s/jq1butk6shnnox1/LctrEdinburgh2012.zip
- The morphing GUI is available here: www.github.com/danielplawrence/synthesis_workshop

TANDEM-STRAIGHT morphing instructions

(Courtesy of JJ Atria, `http://pinguinorodriguez.cl`)

1. Open the Morphing Menu
2. Load waveform A
3. Analyze A (This opens the STRAIGHT handler)
   (a) Fo/Fo structure extraction (This opens foExtractorGUI)
      i. Calculate
      ii. Clean LF noise
      iii. Calculate
      iv. Auto-tracking
      v. Finish/Upload (This closes foExtractorGUI)
   (b) Aperiodicity extraction
   (c) STRAIGHT spectrum
   (d) Finish/Upload (This closes the STRAIGHT handler)
4. Load sound B
5. Analyze B
6. (Repeat step 3)
7. (At least in my version, force GUI update by playing one of the sounds)
8. Open anchoring interface (This opens temporalAnchorGUI)
   (a) Calculate distance matrix
   (b) (optional) Adjust temporal anchors for both sounds; this can also be loaded from an Audacity label
   (c) Set up anchors (temporalAnchorGUI stays open, but can be closed)
9. Initialize morphing rate (Not sure if this is necessary)
10. Prepare first morphing substrate:
    (a) Edit morphing rate (This opens morphingRateGUI)
    (b) Shift knobs to A (with binded attributes they'll all move in unison, but they can be set independently as well)
    (c) Set up morphing rate (morphingRateGUI stays open, but can be closed)
    (d) (optional) Play synthesized sound; it should sound like A

(e) Save morphing substrate

11. Prepare second morphing substrate (Repeat 10 with B instead of A)

12. Continuum (This opens makeMorphingContinuumGUI)

    (a) Input continuum data

    (b) Generate continuum

        i. Open first morphing substrate (prepared in 10)

        ii. Open second morphing substrate (prepared in 11)

13. Done!

References

Alku, P. (1992). Glottal wave analysis with pitch synchronous iterative adaptive inverse filtering. *Speech communication*, 11(2-3), 109-118.

Fant, G. (1970). *Acoustic theory of speech production: with calculations based on X-ray studies of Russian articulations*. Mouton & Company.

Kawahara, H., Morise, M., Takahashi, T., Nisimura, R., Irino, T., & Banno, H. (2008). TANDEM-STRAIGHT: A temporally stable power spectral representation for periodic signals and applications to interference-free spectrum, Fo, and aperiodicity estimation. In *IEEE International Conference on Acoustics, Speech and Signal Processing*. 2008. (pp. 3933-3936).