# Problem Set 2: Praat for visualizing and analyzing speech

This lab has **14 questions**, each of which is clearly labeled with an initial **Q** and rendered in boldface. **Push a PDF** containing your answers to these questions to your GitHub repository for this problem set.

### 1. PROSODY

Using Praat, record yourself saying the following sentence. (I've included recordings of myself saying all of these sentences, along with TextGrid files showing word or phoneme boundaries, in `audio.zip`, if you prefer not to make your own recordings. Select the .wav and the TextGrid with the same name, then right click to open with Praat.)

(i) *Bananas are very poisonous.*

Now record yourself saying it as if you're contradicting someone. Pretend you heard someone say the first sentence, then record yourself saying only the second sentence in italics. You'll be assigning focus to, i.e., emphasizing, the word in boldface and underlined.

(ii) Blueberries are very poisonous.
***Bananas*** *are very poisonous.*

(iii) Bananas are somewhat poisonous.
*Bananas are* ***very*** *poisonous.*

(iv) Bananas are very delicious.
*Bananas are very* ***poisonous****.*

**Q1: In each version of the sentence, which word contains the highest F0 (pitch)?**

Now look at the duration of the word *bananas*. You can measure the duration of a selected area by selecting it and then reading off the duration in seconds on the bar under the spectrogram.

**Q2: How does the duration of the word *banana* change when you change the way you are saying the sentence? In which sentence is it the longest? In which is it the shortest?**

Now look at the intensity of one of your recordings (equivalent to what we've been calling energy). Display the intensity by selecting `Show intensity` from the `Intensity` menu.

**Q3: How would you characterize the intensity track in general? That is, when does intensity seem to go up and when does it go down?**

**Q4: Imagine you are designing the prosody component of a text-to-speech (TTS) system. What challenges do you think this sort of variation in prosody might present?**

Extract the maximum pitch for each of the four sentences by selecting the full sentence with the cursor, making sure that the pitch track is visible, and then selecting `Pitch->Get max pitch`.

**Q5: Compare the max pitch of the neutral sentence with the max pitch for the other sentences. Given what you see, speculate about what sort of speech-based applications might make use of max pitch, pitch range, or other pitch-based features.**

## 2. VOICE QUALITY

So far we've talked about pitch and intensity/energy, which are related to how we convey meaning using prosody, and about formants, which are what distinguish one vowel from another (more on that in a minute). Two other interesting speech features are jitter and shimmer.

These features are typically associated with voice quality. They are the features that make one voice sound different from another. They are also features that can be good indicators of an impairment affecting speech.

As you speak, the pitch at a given point (i.e., the projected number of cycles per second given the duration of that cycle) is not precisely the same from one time point to the next. Jitter is the measurement of this variability. The most common way to capture this is the Period Perturbation Quotient, which is the mean difference in pitch over a 5-cycle window between each cycle and the mean of all 5 cycles.

Shimmer is measured using a similar measurement made for amplitude (a.k.a. energy): the Amplitude Perturbation Quotient. This measures the difference in amplitude over a 5-cycle window between the amplitude of each cycle and the mean amplitude of all 5 cycles.

A third feature that can be used to describe voice quality is harmonic-to-noise ratio. As its name indicates, the is the ratio of regular, periodic noise to turbulent white noise at any moment in phonation. This is often interpreted as a measure of "breathiness".

Take one of your recordings from part 1 (or one of the recordings I provide) and select the part of the sound that contains only speech. Then from the `Pulses` menu, select `Show pulses`. Then select `Voice report`. A text window will open, containing all sorts of information about your speech file, including shimmer, jitter, harmonic-to-noise ratio, min and max pitch, and more.

**Q6: Compare the values in your own voice report with the voice report for one of the .wav files included in audio.zip. How are they different? How are they similar? Why?**

**Q7. Now compare the values in two different voice reports for the same voice (either two of your own voice or two of my voice). Are they more similar to each other than to the voice reports for the other voice? Which features are similar and which are different?**

**Q8. What speech applications do you think these sorts of features might be useful for?**

**3. PRAAT SCRIPTING**

The Praat GUI is great if you want to just analyze a few files, but it's not practical if you want to extract features from multiple files, which is the sort of thing you might being doing for this class. Praat has its own scripting language, and there is an easy way to get started scripting.

**Getting started: Using the command history**

1. From the `Praat` menu way, way, way up on the top toolbar, select `New Praat script`. This will open a little text editor.

2. From the `Edit` menu of the text editor window select `Clear history`.

3. Go back to the `Objects` window, go to the `Open` menu, and select `Read from file...`

4. Select a .wav file to open (e.g., one of the recordings you made about bananas).

5. Back in the `Objects` window, click on `Analyse Periodicity-> To Pitch (ac).`

6. Select the Pitch object, then go to `Query-> Get maximum...` then click `Okay`.

7. Go back to the script text editor window and from the `Edit` menu, select `Paste history`.

You should now have a file that looks like something like this, but with the parts in double quotes replaced with things from your workspace:

```
Read from file: "/Users/emtucker/Desktop/audio/bananas-BANANAS.wav"
To Pitch: 0, 75, 600
Get maximum: 0, 0, "Hertz", "Parabolic"
```

8. Change the name of the audio file to one of the other audio files you created or that I provided. Then go to the `Run` menu and select `Run`. This should open a window with the max pitch of the new file. You have now created and run a script that will extract the maximum pitch from a file!

9. To get jitter and shimmer, you'll need to create a Point Process. Select the Pitch object you just created, then go to `Analyze->To Point Process`.

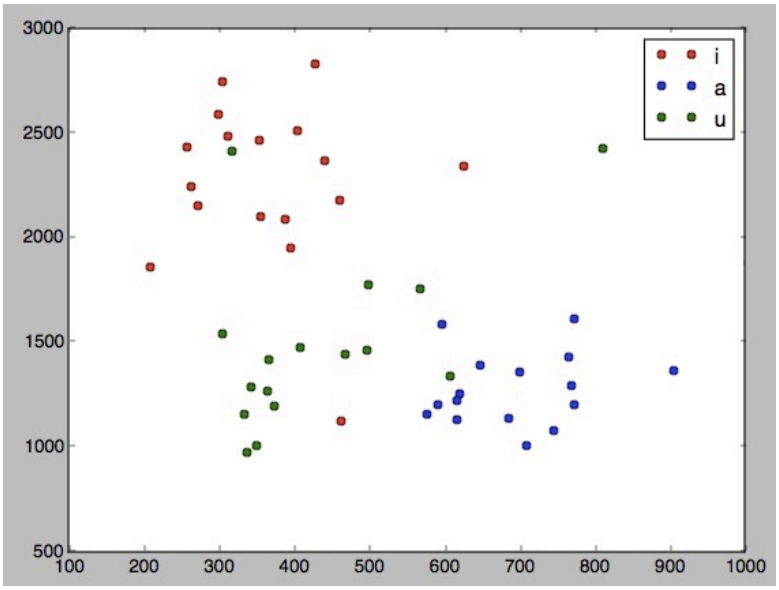
10. Select the Point Process object you created, then go to `Query->Get jitter (local),` and click okay. Now go in a paste the last few commands from the history into your window, and you can get the local jitter for that file.

11. Now, select the Sound, the Pitch, and the Point Process objects all at once by holding down the Shift key. The one available option will now be `Voice report...` Click that, paste in the last view commands in the history, and you've got a full voice report for your file.

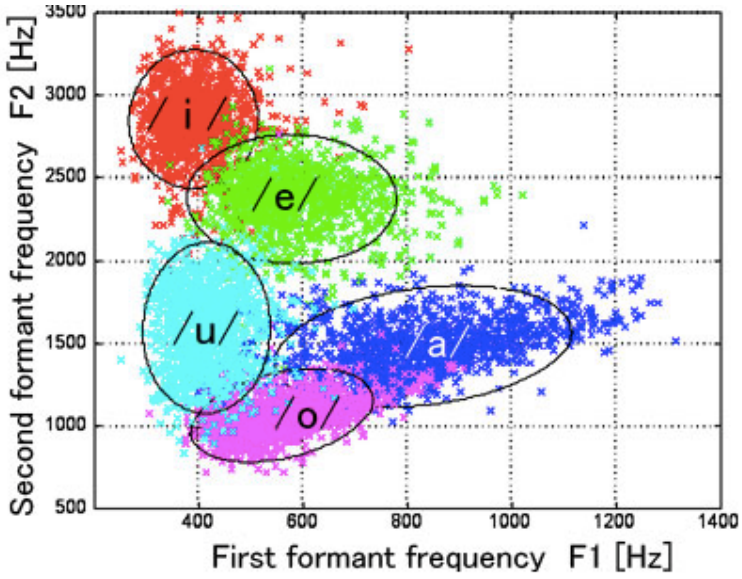
**Q9. Save your script by going to `File->Save` in the script editor window. Include the text of the script in your lab write-up.**

## 4. FORMANTS

In class, we saw that different vowels have concentrations of energy (i.e., formants) in different locations in the spectrum. Here's a plot of F1 (x-axis) against F2 (y-axis) for formant data from a previous class using the code in plotf1f2.py included in this repo.




Here's a more complete picture from http://auris.pu-toyama.ac.jp/image/F1F2E.jpg showing data similar to what we collected in class but for the five vowels of Japanese with many speakers.




Although there were only about 20 speakers, and although there were a few outliers for /u/ particularly, you can see that the small sample of /i/ /u/ and /a/ sounds from American English speakers followed a very similar pattern to those observed for those vowels in Japanese.

In this section, you will learn a little bit about Praat scripting with TextGrid files so that you can compare inter-speaker variation (variation across speakers) to intra-speaker variation (variation within a single speaker) in formant values.

In audio.zip, you'll find a recording of me saying the following sentence:

> *Did h**e** bel**ie**ve that C**ae**sar could s**ee** p**eo**ple s**ei**ze the scar**y** s**ea** am**oe**ba?*

There is also an accompanying TextGrid file that has labels and time stamps for every instance of the phoneme/i/, each of which corresponds to one of the boldfaced vowels, above. You can view the two files together by opening them in Praat, then selecting both objects in the Praat Objects window, and then clicking View & Edit.

In the repo, find the `getformants.Praat` script, and modify it so it works with your directory structure on your computer (just the first few lines). Then run it. It should extract the F1 and F2 for each segment labeled with /i/ in the TextGrid.

Plot the F1s against the F2s to make a picture like the ones above by modifying the code of `plotf1f2.py`, included in the repo, or by using your favorite plotting tool (Excel, R, Matlab...).

**Q10: Embed the resulting plot in your submission for this problem set.**

**Q11: How does the plot of your formants differ from the class plot (if at all)?**

Using numpy.std or Excel (or R, Matlab, etc.), calculate the standard deviation of the F1 and F2 of the class values for "bead" (the first two numeric columns in the `vowel-formants.csv` file) and the standard deviations of F1 and F2 for /i/ that you just extracted.

**Q12: What are the two standard deviations you calculated for each of the two formants?**

**Q13: Why do you think the standard deviation might be larger in one vs. the other?**

**Q14: How do you think the variation of F1 and F2 might be challenging for some speech applications but useful for other applications?**