DSP Speech Application Exercise
March 6–24, 2019
Deadline: March 24, 2019 | 11:59 PM

Guidelines:

- Save all MATLAB text files with the following header format:

```
%%  Digital Speech Processing - filename.m
    %   DSP Application Process - 2nd Semester AY 2018-2019
    %   Submitted by:   SURNAME, Firstname M. 2012-34567
    %   Submitted on:   24 March 2019
    %   Created with:   MATLAB R2019b
```

- Make sure all code is readable with well-placed comments, and all figures are properly entitled and labeled.
- Compress all deliverables into a single ZIP archive with the filename `DSPSpeech_Surname_34567.zip`. Your ZIP archive must contain ten files:
  - `nickname_34567.wav`
  - `nickname_34567.TextGrid`
  - `calcSTE.m`
  - `STEContour.m`
  - `STEContour.png`
  - `SilenceRemoval.m`
  - `SilenceRemoval.wav`
- Submit your ZIP archive as an attachment via email to `angelina.aquino@eee.upd.edu.ph` with the subject field `'DSP Speech Project - SURNAME'`. Include your feedback on the speech lecture and project in the contents of the email.
- If you have any further questions, feel free to email us, message us on Facebook, or drop by DSP 410.

---

1 Recording

In the previous exercise, you were asked to record an audio file containing an utterance of your nickname and the last five digits of your student number. Save this recording as a WAV file following the filename `nickname_34567.wav`. You will use this recording in the succeeding tasks to demonstrate your understanding of speech signal processing through practical applications.

2 Visual Analysis and Segmentation via Praat

In this part of the project, you will be using the Praat phonetics software to analyze your speech recording and segment utterances & non-speech events at the word & phone levels.

1. Download the appropriate Praat application for your operating system from this website: `http://www.fon.hum.uva.nl/praat/`
2. Open your speech recording in Praat. Annotate your recording by creating a TextGrid file with three tiers, labeled `word`, `phone`, and `voicing`, respectively. Select both files to view and edit them simultaneously.
3. Segment your recording at the word level and label each segment appropriately. Use the word `silence` to indicate non-speech events.
4. Segment your recording at the phone level and label each segment appropriately. Use the ARPAbet two-letter notation to label each phone. Use the term `sil` to indicate non-speech events.
5. Copy the phone-level segmentations onto the `voicing` tier. Determine the voicing of each uttered phone, and label each appropriately as `voiced` or `unvoiced`. Use the term `sil` to indicate non-speech events.
6. Save your TextGrid file following the filename `nickname_34567.TextGrid`. Open the TextGrid file in a text editor to see how your segmentation and labeling is parsed from a graphical interface into a text file.

3 Short-time Energy Extraction

In this part of the project, you will be using MATLAB to extract short-time energy (STE) values from your speech recording using the windowing process. You will also be using STE extraction to remove silences on your recorded speech file.

3.1 Short-time Energy Function

1. Create a MATLAB function `calcSTE.m` which accepts four inputs:
   - filename of the input speech file
   - window length (in msec)
   - window overlap (in msec)
   - window type (either `'rectangular'` or `'Hamming'`)

   The function should return three outputs:
   - vector containing the speech signal
   - sampling frequency of the input speech file
   - vector containing the STE of the speech signal
2. Use the `audioread` function to extract the speech signal vector and sampling frequency from the input speech file. Normalize your speech signal vector (i.e. limit all values to $[-1, 1]$). If your signal contains two columns (for WAV files recorded in stereo), manipulate your speech signal vector into a single-column vector. (Some possible expressions: `sig(:)`, `sig(:,1)`, `sig(:,2)`, `sum(sig,2)./2` – can you identify what each expression does?)
3. Using the given sampling frequency and the input window length & overlap, calculate the number of samples in one window (n_{length}), as well as the window jump (n_{jump}) or number of samples from one consecutive window to the next. (Hint: $n_{jump} = n_{length} - n_{overlap}$)
4. Create a vector of length n_{length} containing the window to be used, as specified in the input. You can generate this vector using the `rectwin` and `hamming` functions.
5. Iterate through your speech signal by framing (i.e. multiplying) the signal with your window vector every n_{jump} samples. Calculate the energy contained in each frame, and store each energy value in your STE vector. (Note: you may need to pad your speech signal vector with additional zeros, so that each frame contains n_{length} samples.)

3.2 STE Contour of Speech Signal

1. Create a MATLAB script `STEContour.m`.
2. Generate an STE vector from your speech file using a Hamming window with a 40-msec window length and 75% overlap.
3. Superimpose the graphs of your original speech signal vector and its energy contour into a single time-amplitude plot. Save this figure with the filename `STEContour.png`.

3.3 Removal of Silences from a Speech Recording

1. Create a MATLAB script `SilenceRemoval.m`.
2. Create two variables `newSig` and `threshold`. Set the value of `threshold` to `0.001`.
3. Iterate through your signal using a rectangular window with a 100-msec window length and zero overlap.
4. If the energy of the window is greater than the `threshold` value, append the contents of the window to `newSig`; otherwise, proceed to the next window.
5. Listen to both the original signal and `newSig` using the `sound` function. If your system works, your `newSig` should no longer contain long pauses between words. You may optimize the value of `threshold` as necessary.
6. Use the `audiowrite` function to export your `newSig` with the filename `SilenceRemoval.wav`.

Good luck with these exercises, and with the rest of your sem! Kaya mo 'to! :)