Eng. 100: Music Signal Processing F15 J. Fessler
Project 2: Touch-tone synthesizer and analyzer

1 Abstract

Now that you have acquired some tools for analyzing frequencies and using `Matlab`, you will apply them to a small engineering project: touch-tone phone tones. The goals of this lab are: (1) To analyze touch-tone phone signals and determine their spectral content; (2) to use `Matlab` to create a GUI *synthesizer* that functions as a touch-tone keypad and that generates the proper tones when pressed with the mouse; (3) to write a `Matlab` *transcriber* program that accepts as input a touch-tone phone signal, determines the phone number, and prints it out on the screen; and (4) to analyze the effect of noise on this transcriber. In addition to applying the tools you have acquired, this project also helps prepare you for the final project.

2 Background

Touch-tone phones create a multi-frequency tone when a button is pressed. That tone is sent over a phone line (or wirelessly) as a signal. The goal of this project is to "reverse engineer" the touch-tone system and build your own `Matlab`-based touch-tone synthesizer and transcriber from scratch. The only things you are allowed to use are: (1) the techniques you have learned so far in Engin 100; and (2) the 12 signals corresponding to each button in the keypad of your touch-tone or cell phone, as given in the file `project2.wav` on Canvas.

3 Project 2: What you have to do

The results of this project will be three m-files, one implementing a touch-tone synthesizer, and one implementing a touch-tone transcriber, and one that evaluates that transcriber's robustness to noise. You also must demonstrate to the lab instructor that they work.

You will use the first program to write a signal to a file `touch.wav` using the command `audiowrite` (as well as to produce a sound using `soundcs`). Then your transcriber will use `audioread` to decode the signal stored in `touch.wav`. You will also use the spectral analysis techniques that you have learned to analyze the touch-tone signals in the first place, like you did with the musical tones in Lab 2. Finally, you will investigate the effect of noise in the touch-tone signal on your transcriber.

3.1 Touch-tone signal analysis

Use the techniques you have learned to analyze the 12 signals generated by the 12 keys on a touch-tone phone keypad. Download `project2.wav` from the Canvas; this file contains, in succession, the signals produced by pressing keys "1,2,3,4,5,6,7,8,9,*,0,#" (in that order) for half a second each (total duration=6 seconds). Use the command `audioread` to read the signal and sampling rate from this file. All you will be told here is that you have the tools necessary to do this. Go to it!

3.2 Touch-tone synthesizer

Write a `Matlab` program (and save it as an m-file) that:

- Creates an on-screen keyboard using a sequence of `uicontrol` commands that resembles the 12-key keypad on a touch-tone phone or a cell phone (similar to what you did in Project 1);
- Produces the appropriate sound, lasting half a second, when pressed by clicking the mouse on it;
- Writes the signal to a file `touch.wav` using `audiowrite` for subsequent decoding by your transcriber.

3.3 Touch-tone transcriber

Write another `Matlab` program (and save it as an m-file) that:

- Reads a touch-tone signal (and sampling rate) produced using the program above and that was stored in `touch.wav` using `audioread`;
- Prints out on the screen the phone number that the signal represents (without the "-" in 123-4567);
- Need NOT be able to handle the "*" or "#" keys (these are not part of a phone number).

Hints

- You *could* use `abs(fft())` and look for peaks in the spectrum of each digit signal, but it is *much* faster to look *only* for those frequencies {`F1`, ..., `FM`} that you expect to see in the signal.
- Given: Row vector of sampled signal `x` where `N=length(x)` and `S` = sampling frequency, use:

```
c = x * cos(2*pi*[0:N-1]'*[F1 ... FM]/S);
s = x * sin(2*pi*[0:N-1]'*[F1 ... FM]/S);
y = c.^2 + s.^2
[~, I] = max(y);
```

- This last command determines the `location` (*i.e.*, index) `I` of the maximum of `y`. The first output of the `max` command returns the maximum *value* of the array, which is not needed here, so we enter a tilde `~` instead of a variable name to tell `Matlab` to discard the first output.
- Note the use of the *transpose* operation `'` above. You should study what is the size of the array of signals generated by the command `z = cos(2*pi*[0:N-1]'*[F1 ... FM]/S);`
- See the `Matlab` tips section below for more about the `max` command.
- Using this array processing capability of `max`, you can `reshape` a signal `x` consisting of multiple tones into a 2D array of size number of samples by number of tones and then transpose it to make and array where each row is one tone, before performing correlation.
- Final hint: if you have an array `B` where you want to replace all the values that are 11 with the value 0, use `B(find(B == 11)) = 0;`
  There are many ways to do this part so not all of you may want to use this hint.

3.4 Transcriber robustness to noise

Now investigate the effect of noise on your transcriber, as follows.

- Add noise to the signal produced by your touch-tone synthesizer for a single key press, *e.g.*, the "1" button. Use `randn` (not `rand`) to generate pseudo-random noise.
  Use `soundsc` to listen to the signal before and after you add noise to it for the lowest and the highest noise levels.
- Compute the *Signal-to-Noise Ratio*:

$$\mathrm{SNR} = 10 \log_{10} \frac{\sum_n |\mathrm{signal}(n)|^2}{\sum_n |\mathrm{noise}(n)|^2}.$$

  This is the noise level figure-of-merit.
- For each of 10 different noise levels (multiply `5*randn` by successively larger numbers), estimate the *error rate* by counting the number of incorrectly-decoded digits out of 100. Plot the error rate (as a percentage) versus SNR.

3.5 Actual phone tones (optional)

- Press your own phone keypad keys and record the audio using a microphone and `audiorecord` as demonstrated in class with guitar sounds.
- Apply your transcriber to your recorded touch-tone phone signal.

- Does your transcriber work on this signal?
- Use your synthesizer to dial a number on your cell phone when held near the speaker.

3.6 `Matlab` tips

To see whether a variable has a certain value or not, use an `if` statement like this:

```
a = rand(1,1)
if a < 0.5
        print 'a is smaller than 0.5'
else
        print 'a is bigger than 0.5'
end
```

If you want to check if a variable differs from some value, then use the "not equal to" symbol `~=` like this:

```
a = 2
if a ~= 1
        print '1 is not equal to 2, obviously'
end
```

The `max` command is useful when working with correlations.

Try the following: `a = [10 20 30 15]`
`[big1, index1] = max(a)`

The result is `big1 = 30` and `index1 = 3` because the largest value in `a` is its 3rd element.

The `max` command can also work with arrays.

Try the following: `b = [10 20; 30 40; 50 15]`
`[big2 index2] = max(b)`

In this case `big2 = [50 40]` because the largest value in the first column of $b = \begin{bmatrix} 10 & 20 \\ 30 & 40 \\ 50 & 15 \end{bmatrix}$ is 50 and the largest value in the second column is 40. The second output is `index2 = [3 2]` because the largest value in the first column is in the 3rd row and the largest value in the second column is in the 2nd row. So for arrays, the `max` command works on each *column*. If you only need the second output, then use `[~, index2] = max(b)`.

3.7 Project report

Write the results of your lab as a report, designed for a fellow ENGN 100 student as the audience, and upload a pdf file to Canvas. Include the following parts.

- [10] A diagram of the frequencies associated with each touch-tone key.
- [5] A summary of the work you did to determine those frequencies.
- [10] The error rate versus SNR plot.

Upload to Canvas a zip file of your three m-files, named `p2_`*teamname*`1.m` and `p2_`*teamname*`2.m` and `p2_`*teamname*`3.m`, for your synthesizer, transcriber, and error rate versus SNR analysis, for grading.

- [10] points for working synthesizer
- [10] points for working transcriber
- [5] points for working error rate script

This will be graded by your lab instructor.

RQ Proj2.1. What value is displayed by the following `Matlab` command line?
`c = [3 1 4 1 5 9 2 6]; [big index] = max(c); disp(index)`