GEORGIA INSTITUTE OF TECHNOLOGY
SCHOOL OF ELECTRICAL ENGINEERING

ECE 4271 SPRING 2016
MATLAB PROJECT #1

Dual-Tone Multi-Frequency (DTMF) Signal Decoding

Assigned: Sunday, January 24, 2016 (t-square)
Due: Tuesday, February 9, 2016:

a) **Report: At *beginning* of lecture**
b) **Code: electronically in t-square by 3:05 pm (please see section 2 and 6)**

- This project is to be done *individually*. Collaboration on projects is not OK. Each student must work on the projects independently. Each student must develop his or her own analyses and computer code in its entirety. Any project-specific help needed should be sought from Dr. Marenco. Students are not to discuss the theory or approaches to coding the theory with one another, nor are they to assist in debugging each other's work. ***The Georgia Tech honor code and its policies apply.***

  You may ask Dr. Marenco questions regarding theory and implementation of the project, including asking them at the beginning or end of class or during office hours, when others can benefit as well.
- Data required for this project are available for download from t-square, in Resources under Project #1. You will also find reference materials and code there. ***Whether you begin working right away or not, be sure you download the data and make sure you can load it into MATLAB as soon as possible to avoid last minute difficulties.***
- Reports and code will be graded primarily on completeness in addressing the assignment and quality of results. Reports must be typed, not handwritten; should be well-organized; and must clearly explain your design and the decisions and analysis that went into it. Code must be provided in the form specified below, and should be commented well-enough to be clearly understandable.
- Questions or clarifications should be directed to Dr. Marenco.[1] Errata, revisions and hints (if any) will be made available via e-mail, t-square, or during class.

[1]Office: 404-407-6303. Office hours: Thursday 1:45 – 2:45 PM, or by appointment, else e-mail: alvaro.marenco@gtri.gatech.edu.

1. PROBLEM

Write a MATLAB function that accepts as input a `.mat` file containing an audio signal of a DTMF (touch-tone) dialing sequence and correctly identifies and prints out the number that was dialed. Your processing architecture must be based on using the Goertzel algorithm to identify the tones, but may vary in the details of the Goertzel parameters and the other functions and steps that you use. The MATLAB Signal Processing Toolbox includes a function `goertzel` that you may use. T-square includes another goertzel function, `gfft.m`, from S. K. Mitra that you can use if you are working on a machine that lacks access to the Signal Processing Toolbox.

2. REQUIREMENTS

You must submit a report of your methods and findings that must include:

1. An overview description of how your processing algorithm works. This must include a block diagram of the major processing steps with a description of the rationale for each step. It should identify the major MATLAB functions or code blocks corresponding to each major function in your block diagram.

2. A brief description of the theory underlying the processing algorithm. Explain the analyses and design decisions that led to each of your parameter choices.

3. A complete listing of the MATLAB program used to process the signals. Code must be sufficiently modular and well-commented to be understood readily. Listings must also be included for any functions used, except for those which are built into either MATLAB or its Signal Processing Toolbox, or were provided in class.

In addition to the written report, you must submit your MATLAB code in t-square in a single `.zip` file containing MATLAB code ***by the deadline (Feb 9, 2016-3:05 pm)***. See below for further instructions on naming the zip file and how your code must be organized. Your program will be tested by applying it to at least three test signals generated by Dr. Marenco (*not* the same three provided for development of your algorithm.)

Start working early. If you encounter problems, please ask questions to clarify anything that is vague or incomplete.

3. REFERENCES

Several reference papers are provided in the section for this project on t-square (Resources->Project #1). You should read these for ideas about processing approaches and pitfalls. Remember that you are required to use a Goertzel-based approach, but the other details are up to you.

4. DATA

Sample data sequences are available in the Winzip file `tt_samples.zip` available from t-square in the `Project #1` folder on the `Resources` page. When unzipped, this will produce the following files:

- Three sample touch tone audio files. Each is a MATLAB `.mat` file, with a name of the form `nnn-nnn-nnnn_nn.mat`. The portion of the file name before the underscore character is the phone number encoded in the file. The portion after is the signal-to-noise ratio in dB. For example, a file named `404-894-2714_30.mat` would be an audio sequence dialing the number 404-894-2714 and having an SNR of 30 dB. The data is always sampled at 8 kHz. Use MATLAB's `load` command to load the data, which will appear as a single vector with the name `signal`. You can use the command `sound(signal,8000)` or `soundsc(signal,8000)` to listen to it if your computer has speakers or headphones. Each sample signal includes a one second dial tone at the beginning, followed by the ten digits of the phone number, followed by one second of silence. The digits are separated by quiet spaces. The duration of the digits varies randomly between 80 ms and 400 ms; the duration of the quiet spaces varies randomly between 50 ms and 200 ms. Noise is added at the specified *SNR* to the entire signal, including the "quiet" spaces between the digits and the one second of "silence" at the end. The signals will have no short interruptions in them, nor any frequency "twist" (imbalance between the amplitude of the two tones). There will always be exactly 10 digits (standard U.S. telephone number), but you should not rely on this fact in your code.
- The M file `tt_create.m` used to generate the sample data. You don't necessarily need this. You can examine it to see how the test sequences were generated, and you can use it to generate new test sequences of your own. You can also use it as a base to modify if you want to generate test sequences with different parameters, *e.g.* by introducing "twist" into the data, or allowing interruptions in digits, *etc.*

Even if you don't start work right away, be sure to download your data file(s) and make sure you can load and work with it as soon as possible!

5. SIGNAL PROCESSING ISSUES

Following are some considerations and questions you must deal with in designing your decoding algorithm.

1. *Analysis window*. You will want to analyze the frequency content of a short segment of the signal at a time. You must decide how long of a segment of the signal to analyze at a time, and whether and by how much to overlap these segments. Examples of some of the considerations:

   - the segment length should be long enough to provide adequate frequency resolution between DTMF tones; and
   - the segments should be short enough to recognize that there are two different tones (digits) when they are separated by the minimum digit spacing of at least 10 ms.

   You must document and give the rationale for your parameter choices. How did you arrive at the values you selected?

2. *DFT Size N*: What DFT size should you use to guarantee that you are able to measure the frequencies in your data with the 1.5% accuracy required by the DTMF specification? Do you want to use one value of *N* for all eight frequencies, or one for the low band and one for the high band, or a different *N* optimized for each frequency you are trying to detect? You must document and give the rationale for your choice of *N*.

3. *Detection threshold:* Since there will be noise in the signals, you must provide a mechanism to determine if there is a signal present, or if a segment represents “quiet” space. Since the signal starts out with a dial tone, you do not have the luxury of a guaranteed segment of noise only at the beginning. One approach might be to compare the output of different Goertzel filters to see if one is above all of the others, or perhaps the average of all of them, by some minimum amount. You must document and give the rationale for your thresholding mechanism.

4. *High and low bands*. Many DTMF structures first filter the data into two bands, one passing all of the high tones, the other all of the low tones. If you choose to do this, you must document this fact and give the rationale for your parameter choices. How did you arrive at the passband values you selected?

5. *Decision logic.* Your basic filtering and frequency analysis should allow you to identify the frequencies present in the data as a function of time. You will then have to devise some decision logic to convert this information into a

series of digits. You must document and give the rationale for your decision logic.

6. MATLAB CODE SUBMISSION AND EVALUATION

Your MATLAB code must be submitted (uploaded) to t-square (Assignments section) in a single zip file by the project deadline. The following conditions apply to this zip file and your code:

1. The file must contain all of the MATLAB code, including all functions and subfunctions, data files, *etc.* required to run your algorithm on a test signal. Place all of these files in a single flat (no subdirectories inside of it) directory with the name `FirstInitialLastName.zip`. For example, my file would be AMarenco.zip. Do not include any functions that are built into MATLAB or its Signal Processing Toolbox.
2. Your top level function *must* have as its first line

   ```
   function digits = tt_decode(x)
   ```

   Furthermore, it must work when all of the files in your zip file have been unzipped and placed in the MATLAB `work` directory.
3. Your program should produce an output that is simply the phone number that is detected, in phone number format, *i.e.* the output should look something like

   ```
   404-894-2714
   ```

   if that was the signal I passed into it.
4. I will test your program by simply executing the command

   ```
   tt_decode(signal)
   ```

   using my various test signals as the input variable `signal`. ***Be absolutely certain that your program will run correctly when I unzip your files into the MATLAB*** `work` ***directory and type this command into MATLAB*****.**

   My test will consist of running your algorithm with several sound files representing various phone numbers **different** from those in `tt_samples` with fairly good signal-to-noise ratios. I will also apply a sequence of signals corresponding to a single new phone numbers with SNRs decreasing from 50 dB to 20 dB in 10 dB steps, and then down to 0 dB in 5 dB steps. The purpose of this test is to see at what noise level your algorithm "breaks" (either crashes or returns an incorrect phone number). All algorithms will break at some noise level, so I’m not saying in advance what breakage level I consider really good or really bad. Also, in past years I have seen some algorithms that work fine at moderate noise levels, but break at very high SNRs (because the algorithm depended too much in some way on noise being present). Failure to decode correctly at high SNRs is not a good thing. Remember that you can use `tt_create` to create your own test signals with various SNRs, in exactly the same way I will create all of my test signals.

7. GRADING

Your grade on this project will be allocated as follows:

- Code Performance: 70%
    - Performance on assorted high-SNR signals: 30%
    - Performance on decreasing-SNR series: 30%
    - Output format correct: 10%
- Report: 30%
    - Algorithm and parameter description and rationale: 10%
    - Code listing and readability: 10%
    - General report quality (complete but succinct, figures, writing, *etc.*): 10%