University of Washington
Department of Electrical Engineering
EE 235 Lab 5:
Time Domain to Frequency Domain

In this lab, we will learn how to transform signals in Matlab from the time domain to the frequency domain. In addition, we will use Matlab to identify the frequencies ω of components $e^{j\omega t}$ that make up a signal. Once you have a good handle on identifying a signal's frequency content, we will then apply it to a fun problem of decoding a phone number based on its touch-tone signals. This will be a 2-week lab, but you are encouraged to do most of it the first week to give you more time for preparing for your exam.

Important Concepts From Lecture You Will Use In Lab 5

- Finding the Fourier Series coefficients a_k of x(t)
- Identifying the frequency content x(t), and understanding the relationship between Fourier Series coefficient index k and frequency ω

What Is Expected From You In Lab 5

- Completion of 3 pre-lab exercises (3 points)
- Completion of 2 in-lab check offs with TA (4 points)
- Completion of a lab report (3 points)

Note: all pre-lab exercises must be completed by each student individually before walking in to the lab. Each student will be checked off by the TA at the beginning of lab section.

PRE-LAB:

Read the Matlab Lab 5 tutorial, particularly the sections on the fft and new built-in Matlab functions that you will need in this lab. This will be a 2-week lab. Only the 1st question of the prelab is due in week 1, but you will get further with the lab if you do more than that.

Remember from class that periodic signals can be written as a sum of complex exponentials $e^{j\omega t}$ at different frequencies ω. When we say a signal **x(t)** has a certain list of frequencies ω, what we are really doing is identifying the frequencies of $e^{j\omega t}$ that can be summed up to produce **x(t)**. Identifying a signal's frequency content is important because working in the frequency domain is ideal for filter design and in the analysis of LTI systems. It is also a good way to automatically recognize certain classes of signals, as we will explore with touchtone phone number signals.

1) Consider the following sum of two sinusoids:

$$\boldsymbol{d0(t) = \sin(2\pi(941)t) + \sin(2\pi(1336)t)}$$

   a) Find the non-zero Fourier Series coefficients $\mathbf{c_k}$ for **d0(t)**, given that $\mathbf{w_o} = 2\pi$. Note: you do not need to prove $\mathbf{w_o} = 2\pi$.

   b) Each Fourier series coefficient index **k** with nonzero $\mathbf{c_k}$ corresponds to a frequency component $\boldsymbol{\omega = k\omega_o}$. Using this fact, identify the angular frequencies **ω** of signal **d0(t)**.

The frequencies **ω** are in rad/sec. What are the corresponding frequencies in Hz (cycles/sec)?

c) In order to find the high energy frequency components of a signal from the result of an fft, we can use the **find** command. Let's say that A, defined below, corresponds to the magnitude of the Fourier Transform of a given signal signal **x** (not **d0(t)**), i.e. it is the result of the command: **abs(fftshift(fft(x,16)**. Answer the following questions**:**

$$A = [0 \;\; 0 \;\; 1 \;\; 0 \;\; 2 \;\; 10 \;\; 3 \;\; 0 \;\; 6 \;\; 0 \;\; 3 \;\; 10 \;\; 2 \;\; 0 \;\; 1 \;\; 0]$$

i) What indices in the matrix **A** will the line of code **find(A > 8)** return?

ii) Suppose **index** = [3 4 5 13 14 15]. What will the output be for the line of code **A(index)**?

iii) What should you pass to the **find** function if you wanted to extract all the indices where the values 6 and 10 in matrix **A**? Note: there are several answers to this.

iv) If the time signal was sampled with Fs=16kHz, what frequencies (in Hz) do these indices correspond to?

v) Fill in the rest of this code to extract the 0's from matrix A:

```
index = FILL IN CODE;
A(index)
```

2) The touch-tone system on a telephone uses signals of different frequencies to indicate which key has been pushed. In particular, it uses a type of signaling scheme called dual-tone multi-frequency (DTMF) signaling. In DTMF signaling, each touch-tone on the keypad is represented as a sum of two sinusoidal tones, one tone at a low frequency and another tone at a high frequency. The DTMF keypad frequencies (in Hz) are shown in the table below:

	1209 Hz	1336 Hz	1477 Hz
697 Hz	**1**	**2**	**3**
770 Hz	**4**	**5**	**6**
852 Hz	**7**	**8**	**9**
941 Hz		**0**	

To generate key 2, we need the signal $\boldsymbol{d2(t) = \sin(2\pi(697)t) + \sin(2\pi(1336)t)}$**.**

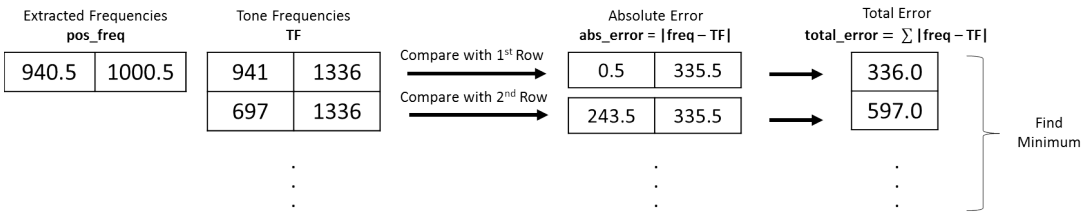
a) Write the equations for the following DTMF key signals: **d5(t)** and **d7(t)**.

b) To classify an arbitrary tone signal, we will compare its frequency content to a table of tone frequencies. Each row contains the two frequencies for one touch tone number 0 - 9. Therefore, row 1 (which will represent tone number 0) will contain the values 941 and 1336. Row 2 (tone number 1) will then have the values 697 and 1209. Write the Matlab code to create a 10 x 2 matrix that contains all the tone frequencies, and call the matrix **TF**.

c) What is the line of code to create a column vector of zeros called **total_error** that has exactly 10 elements?

d) Using **length(total_error)**, what is the for-loop statement to loop through all the rows in column vector **total_error** starting at **i = 1**?

e) The frequency content we extract from tone signal **x** will not be integers. As a result, we cannot do a simple equality check through each line of matrix **TF** to identify the correct tone number. Instead, we need to calculate and identify the minimum absolute error measurement between the frequencies we extract (**pos_freq)** and the frequencies in each row of matrix **TF**, as illustrated below:



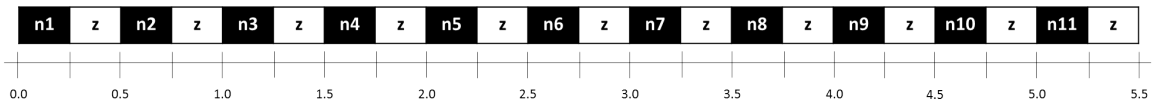
i) Using the function **abs**, write the line of code to calculate the absolute error measurement **abs_error** between the extracted frequencies **pos_freq** and **ith** row of the matrix **TF**. Note that **abs_error** is a 1 x 2 row vector.

ii) What is the line of code to sum up the 1st and 2nd elements of **abs_error**, the value of which will be stored as the **ith** element of vector **total_error**?

iii) Using the function **min** with two outputs, write the line of code to find the minimum value **min_e** and corresponding location index **index_min_e** in vector **total_error**?

iv) Note that if the correct tone number is 0, **index_min_e** will equal 1. If the correct number is 1, **index_min_e** will equal 2. If the correct number is 9, **index_min_e** will equal 10. How do we calculate the tone number **num** from **index_min_e**?

3) A long-distance phone number signal consists of 11 tone signals **n1 to n11.** You will be given a signal where tones are of duration 0.25 seconds with a silence signal **z** (of zeros) also of duration 0.25 seconds following each tone:



The numbers can be decoded by extracting each tone signal (along with its follow-on silence signal) and classifying its tone number from its frequency content.

a) We can use a loop to perform this task. Assuming you know the sampling rate **Fs** and loop index variable **i**, what is the mathematic equation for **start_index** and **end_index**, which you will use to extract the **ith** tone and silence signals?

b) To help you determine these equations, you may fill in the table below to list out the formulas to determine the start and end indices for each tone and silence signal. Then, from all these formulas, determine a general equation in terms of **i** and **Fs**.

Section	start_index	end_index
n1 + z: $0.0 \le t < 0.5$	0.0 * Fs + 1	0.5 * Fs
n2 + z: $0.5 \le t < 1.0$		
n3 + z: $1.0 \le t < 1.5$		
n4 + z: $1.5 \le t < 2.0$		
n5 + z: $2.0 \le t < 2.5$		
n6 + z: $2.5 \le t < 3.0$		
n7 + z: $3.0 \le t < 3.5$		
n8 + z: $3.5 \le t < 4.0$		
n9 + z: $4.0 \le t < 4.5$		
n10 + z: $4.5 \le t < 5.0$		
n11 + z: $5.0 \le t < 5.5$		

Section	start_index	end_index
ith tone and silence signal		

c) Suppose you have a Matlab variable **x** with the value 1. Using vector concatenation, the variable **x**, and the function **num2str**, what is the line of code to create the string "The value of x is 1", which will be stored in the variable **str**?

EXERICSE #1: Identifying Frequency Content of a Signal

In this exercise, we will learn how to use Matlab to identify the frequency content of a signal **x(t)**. In particular, we will focus on the case when **x(t)** is periodic and a sum of sinusoids. The signal we will analyze represents a touch-tone telephone signal.

Lab Exercise: Let us now use Matlab to verify our pre-lab results.

a) Create a new working directory in your U Drive called **Lab 5**. In this directory, open up a script and call it **Ex1.m**. Make sure to clear all variables and close all figures.

b) The first thing we will do is generate the tone signal from pre-lab and analyze the signal in the time domain. The point of this portion of the exercise is to show that viewing signals in the time domain is sometimes not useful:

i. First, create the time samples vector **t** in the range $0 \le t < 0.25$ using sampling rate Fs = 8000. Note that t is only less than 0.25 and not less than or equal to 0.25.

ii. Using time samples vector **t**, generate your signal vector **d0** using the equation from pre-lab: $\boldsymbol{d0(t) = \sin(2\pi(941)t) + \sin(2\pi(1336)t)}$

iii. Using the **sound** and the **pause** functions, play sound signal **d0** twice with a 1-second pause in between.

iv. Run your script to hear the tone signals. The sound should resemble what you hear when dialing a number 0 twice on a touch-tone telephone.

v. Using a 2 x 1 subplot, plot **d0 vs. t** as the 1st subplot. No need to adjust your axes, but label and title your plot.

c) Now let us convert our signal from the time domain to the frequency domain, and show how useful it is to view signals in a different domain:

i. Using the **fft** function, create **N** = 4096 samples of the signal in the frequency domain and store it in a variable called **D0_fft**. Make sure to also shift the output of the **fft** using function **fftshift** so that the results are centered around w = 0.

ii. Because the output of the **fft** is complex, we will want to take the magnitude of the fft. As we will see, the magnitude of the fft contains the most useful information about the signal's frequency content. Using function **abs**, compute the magnitude of **D0_fft** and store the return value in variable **D0_abs**.

iii. Define the frequency samples **w** using the sampling rate **Fs** and the number of frequency samples **N**.

iv. On the same 2 x 1 subplot used in (b), plot **D0_abs vs. w** as the 2nd subplot. Adjust your axes so that the x-axis is between -9000 to 9000 and the y-axis is between 0 and 1000. Label and title your plots.

v. Run your script to view your plots in the frequency domain. Verify your pre-lab answer and that you have the correct number of frequencies **ω** for **d0(t)** and that the spikes roughly fall at the values of **ω** you calculated.

d) Now let us identify the frequency content of **d0(t)** by determining exactly where in frequency (in Hz) these nonzero spikes occur:

i. Using the **find** function, identify the indices in **D0_abs** where **D0_abs** is greater than a threshold value of 900. Store your results in **index_tone0**.

ii. We can now identify the frequency content of our signal. Using **index_tone0**, extract the frequencies in **w** that represent the frequency content of **d0(t)**. Convert and report these frequencies in Hz by dividing the extracted frequencies in **w** by 2π. Make sure to wrap your code for 2π in parentheses. Store your result in variable **freq_tone0** and display the output on the COMMAND window.

iii. Run your script and view the frequency content of **d0(t)** on the COMMAND window. Verify that the values of the frequencies (in Hz) match up with your answers from pre-lab.

Lab Check-Off #1 of 2: Show **Ex1.m** to a lab TA and demonstrate you know how to plot a signal in the frequency domain and that you can identify the frequency content of a signal in Matlab.

Lab Report Question #1 of 6: To find the frequency content, we used the **find** function with a threshold of 900. Using your FFT magnitude plot, explain why a threshold of 100 would not work. Would that threshold identify more or fewer frequencies than the threshold of 900? Explain your answer.

Lab Report Question #2 of 6: Suppose we altered the signal for **d0(t)** and used $\boldsymbol{d0(t) = 1 + \sin(2\pi(941)t) + \sin(2\pi(1336)t)}$ instead. A student incorrectly reports that the graph for the FFT of D0 would not change. Explain why this student is incorrect.

EXERCISE #2: Classifying Touch-Tone Telephone Signals

In this exercise, we will learn more about the signals of a touch-tone telephone system by expanding on one of the signals we started analyzing in Exercise #1. We will also write a function to decode and identify the number corresponding to an arbitrary touch-tone signal from its frequency content.

Lab Exercise: We will now write and test the function to classify a tone signal.

a) Open up a new file and call it **classify.m**. Using your pre-lab, write the function header for **classify**.

b) Using pre-lab, define the tone frequency matrix **TF**.

c) Using the same procedure from Exercise #1, let us now extract the frequency content of tone signal **x**:

   i. First, we need to convert the signal from the time domain to the frequency domain. Using function **fft**, **fftshift**, and **abs**, compute the magnitude of the **fft** of **x** consisting of **N** = 4096 samples, and store the result in a variable called **X_abs**.

   ii. Define the frequency samples **w** using the sampling rate **Fs** and the number of frequency samples **N**.

   iii. Now, let us proceed to identify the frequency content of signal **x**. Using the **find** function, identify the indices in **X_abs** where **X_abs** is greater than a threshold value of 900. Store your results in **index_tone**.

   iv. Using **index_tone**, extract the frequencies in **w** that represent the frequency content of **x**. Convert and these frequencies to Hz by dividing the extracted frequencies in **w** by 2π. Make sure to wrap your code for 2π in parentheses. Store your result in variable **freq_tone**.

   v. Recall from Exercise #1 that **freq_tone** contains both the negative and positive frequencies of a signal for a total of 4 frequencies. We only need the positive frequencies, which are located in the last two elements of **freq_tone**. Grab the positive frequencies by extracting the 3rd and 4th elements of **freq_tone**, and store these frequencies in variable **pos_freq**.

d) Now that we the frequency content of signal **x**, let us classify it to the correct tone number using some of the lines of code you wrote in pre-lab:

   i. Create a column vector of zeros with 10 elements called **total_error**. You will use this to store the total absolute error measurement for each tone number.

   ii. We will now calculate **total_error** one row at a time using a for loop:

```
% LOOP through all the elements in total_error:

        % Calculate the absolute error measurement abs_error between pos_freq
        % and the ith row in matrix TF.

        % Sum up the first and second elements of abs_error to compute the total
        % error, and store the value in the ith element of total_error.

% END OF LOOP
```

   iii. Using the function **min**, find the minimum error **min_e** and the corresponding location index **index_min_e** in vector **total_error**.

   iv. From **index_min_e**, calculate the tone number and store the value in the output variable **num**.

e) Provide function header and in-line comments.

f) Let us now test your function on a separate script. Open up a new script and call it **Ex2.m**. Clear all variables and close all figures.

g) Create a vector of time samples called **t** in the range $0 \leq t < 0.25$ using sampling rate **Fs** = 8000. Note that t is only less than 0.25 and not less than or equal to 0.25.

h) Using the pre-lab results, write the equations for the tone signals **d4**, **d5**, **d8**, and **d9**.

i) Now let us test your function **classify** on each of these tone signals. To start, call your function **classify** with the tone signal **d4**, and store the output in variable **num**. Display the output on the COMMAND window. Run your script and verify your function is correct and that you get the value 4 for **num** on the COMMAND window.

j) Repeat (i) with **d5**, followed by **d8**, and then **d9**.

k) As an extra test, let us test the function and show it still works even if we modify the signal **d4** with extra zeros padded to the end of the signal:

   i. Create a row vector of zeros called **z** that lasts a duration of 0.25 seconds using sampling rate **Fs** = 8000.

ii. Append the row vector **z** by concatenating it to the end of row vector **d4**, and call the result **d4_z**.

iii. Call function **classify** with signal **d4_z** and store the output in the variable **num**.

iv. Run your script and verify you still get the correct tone number.

l) Comment your script.

Lab Check-Off #2 of 2: Show your script to a lab TA and demonstrate you can classify a tone signal to the correct tone number.

Lab Report Question #3 of 6: In (2k), you concatenated your tone and zeros signals (both row vectors) using either a space or a comma as your delimiter. If you had used a semicolon as your delimiter instead, describe how signal **d4_z** would change.

Lab Report Question #4 of 6: Suppose you created a new signal called **d(t)**, where

$$\mathbf{d(t) = d4(t) + d8(t)}$$

If you called **classify** with this signal, the values for **pos_freq** end up being negative. Explain why this happens. What are these negative frequencies?

EXERCISE #3: Decoding Phone Number from Touch-Tone Signals

In this experiment, we will put together what we learned in the last two exercises to decode an entire phone number from its touch-tone signals. Here is the scenario for this week: The phone number of Eldridge's favorite fast-food restaurant in San Diego is encoded in a sound file. Your mission is to decode the phone number and figure out what the restaurant is.

Lab Exercise: Let us now decode an entire phone number and plot all the extracted signals on one subplot.

a) Open a new script and call it **Ex3.m**. Again, clear all variables and close all figures.

b) From the website, download the data file **phonenum.mat** and then load it. The file contains two variables:
   - **Fs**: sampling rate of phone number signal
   - **x**: actual phone number signal you need to decode

c) To see the signal you will be analyzing, plot the signal **x**. Do not forget to compute the corresponding time samples of **x** first, so that you can plot **x vs. t_x**. Leave axes unchanged. Title and label your plot.

d) To prep us for the upcoming for loop that will decode the phone number, we need to do two things:

   i) Create a vector to store each individual digit in the phone number. Declare a row vector of zeros called **phone_num** with 11 elements.

   ii) Load a new figure window, which we will use to plot all extracted tone signals and their follow-on silence signal.

e) Using a for-loop, we will now fill in **phone_num** one element at a time. We will extract each tone signal, decode it, and plot it on one figure window using subplots. The final plot should have the following format, where each subplot has both the tone signal and the follow-on silence signal:

Signal #1	Signal #2	Signal #3	Signal #4
Signal #5	Signal #6	Signal #7	Signal #8
Signal #9	Signal #10	Signal #11	

The pseudocode for the for-loop is given below:

```
% LOOP through all elements in phone_num using loop variable i

    % Using pre-lab, compute start_index and end_index for the ith tone and silence signals

    % Using start_index and end_index, extract the ith tone and silence signals from phone
    % number signal x, the values of which are stored in the variable signal.

    % Call the classify function to decode the tone number from signal, and store the output
    % as the ith element in the vector phone_num.

    % Create time samples vector t_signal for the extracted signal.

    %  Plot signal vs. t_signal as the ith subplot.  Leave the axes unchanged.
    % To save space, the axes will not be labeled.  However, it will have a title.

    % Using the loop variable i and the function num2str, create the title string str,
    % which will display the decoded ith number.  As an example,
    % if the ith number is 7, then the title should say: Number = 7

    % Using str, add the title to the current subplot.

% END OF LOOP
```

f) Display the contents of **phone_num** on the COMMAND window.

g) Run your script to view the decoded phone number. Verify that the digits in the decoded phone number are displayed on the subplot titles.

Lab Report Question #5 of 6: What is the name of this well-known fast-food restaurant?

Lab Report Question #6 of 6: A student accidentally calls the **title** function outside the loop, following the **end** statement. Explain how the final plot changes and why these changes happen.

FOR NEXT WEEK: LAB REPORT DUE

Turn in a PDF of your lab report (one per team) online via the link on the class website. A sample format for the lab report is on the class website. You may vary from this format but please include the basic sections. The report is due 24 hours prior to the start of your next lab section. Lab turn-in times will be checked, so no late lab reports will be accepted unless arranged in advance with the instructor.