Signal & System Theory (CSD:5224)

Instructor: Shawn S. Goodman, Ph.D.

**Matlab Project 2**

Create Sinusoids

**Submission**: Upload your function as an M-file to the ICON dropbox. Also upload your MS Word document. Be sure to save your function in a safe and accessible place. You will use it again in the future.

**Instructions**:

**Part 1.**

Write a MATLAB function to create sinusoids. The function should have a header section with the following information:

1) Name of the function and usage (i.e. how to call the function)

2) Short description of what the function does.

3) List of input / output arguments with short descriptions.

4) Name of the author

5) Date

Name the function **mySine**.

There should be four input arguments: amplitude, frequency, phase, and duration.

There should be two output variables: the sine wave, and time.

Your function should be able to create sinusoids at frequencies up to 10000 Hz.

The formula for computing a discrete sinusoid is *x*[*n*] = *A* sin(2*fnts* + **), where

*A* is amplitude,

*f*is frequency in Hz,

*n* is a vector of integers,

*ts* is the sampling period, and

** is the starting phase.

Use the MATLAB function sin() to calculate the sine function.

**Part 2**

Create a1000-Hz sinusoid in *cosine* phase. The peak amplitude should be 2.5. Plot the waveform (time on the x-axis and amplitude on the y-axis) using the MATLAB function plot(). Zoom in on the waveform so that you can see a few (5-10) periods. Label the x and y axes appropriately (you can use xlabel() and ylabel()). **Copy the figure and paste it into a MS Word document.**



**Part 3**

Using your function, create two sine waves, each of a different frequency. Listen to the waveforms using the MATLAB function soundsc(). (Note: In the computer lab you will probably need headphones.) Answer these questions in the Word document: **3a. What happens perceptually if you specify a lower sampling rate (when calling soundsc) than your function actually used to create the waves? 3b. Why does this happen?**

A). A lower sampling rate causes the sound to be lower pitch and longer in duration.

B). This is because the same amplitude values get assigned to a new time series based on the sampling rate passed to `soundsc()`. If the sampling rate matches the rate we used when generating the sound, then the amplitudes will be assigned to the expected times and play the desired sound. Otherwise, they will be spread out differently; the same number of time points will be used but the intervals between them will be larger if we specify a lower sampling rate, thus taking up more time and effectively modeling a lower frequency.

**Part 4.**

Combine the two sine waves from Part 3 by adding them together. Combine the two sine waves by multiplying them (using point-wise multiplication). Listen to the two different combined waveforms. **4a. Plot the two sets of combined waveforms (added and multiplied). 4b. Comment on the differences between the combined sets.**

Added:

A). 

Multiplied:



B). In the added waveforms, the positive peak and the negative peak are not symmetric (the negative peak has a larger magnitude than the positive peak) while the multiplied waveform is completely symmetric; however, the added waveform also has a positive DC shift which makes the positive peak higher than the negative peak is low. The multiplied waveform does not have a DC shift like this. The multiplied waveform appears to have some type of lower frequency component with faster components alternating about that, while the added waveform consistently alternates around 1.

**Part 5.**

Create two sine waves, both of the same frequency but different phase. Listen to the waveforms. Answer these questions in the Word document: **5a. Can you see the differences between the sine waves when you plot them? Can you hear a difference between the sine waves when you listen to them? 5b. Why or why not? Comment on your answers in 5a.**

1. No to both questions
2. You can see a difference if you zoom in at an identical time window for the two waveforms, but otherwise, they are exactly the same across time. They sound the same as well because they have the same frequency, and therefore the same pitch. Our ears are not sensitive to a difference in phase across a continuous signal.

**Part 6.**

Create a single sine wave and manipulate it in two different ways: 1) Multiply it by a scalar. 2) Add it to a scalar. **6a. Listen to the results using soundsc() and sound() What is the difference and why? 6b. Plot the results. 6c. Comment on your observations: What differences did you see visually? What differences did you hear audibly? Why did these occur?**

**A).** Using soundsc( ), all three waves sound identical. However, with sound( ), they each have different loudnesses (while maintaining the same pitch across them). This is because soundsc( ) scales the volume so that it is equivalent regardless of the waveforms’ amplitudes, while sound( ) does not.

**B).**

Multiplied:



Added:



**C).** The notable difference between these is that the multiplied waveform alternates around 0, while the waveform with an added scalar alternates around 3 (which was the value of the added scalar). Additionally, the peak to peak amplitude is different; the added to waveform has a p-p of 5, same as the waveform it is derived from, while the multiplied waveform has a p-p of 15, equal to the original p-p times the scalar used. This makes sense as added a scalar manifests as a DC shift equal to the scalar, while not affecting the AC component. Multiplying increases the size of the AC component, without any sort of flat shift.

Audibly, the two tones have the same pitch, since they have the same frequency. The multiplied one is louder, which makes sense given the higher amplitude. Interestingly, they also have a different sound quality. The added to wave has a tinnier sound to it, perhaps due to how it only contains positive amplitude values.