Full Name: ______________________________

EEL 4750 / EEE 5502 (Fall 2018) – Code #05 **Due Date:** **Nov. 5, 2018**

Question #1: *(1 pts)* How many hours did you spend on this homework?

Question #2: *(6 pts)* *The Chirp Signal*

Consider the signal

$$x[n] = \cos\left(\frac{2\pi n^2}{10000}\right) .$$

This is known as a linear frequency-modulated "chirp" signal whose instantaneous frequency $\omega[n]$ changes linearly with time. Chirp signals are commonly used in RADAR processing to extract time-varying time-shifts (distance from a target) and frequency/doppler-shifts (velocity of a target). In this context, the instantaneous frequency $\omega[n]$ is defined by

$$x[n] = \cos\left(\left(\omega[n]/2\right) n\right) .$$

More specifically, the instantaneous frequency $\omega[n]$ is the time-derivative of the expression inside the cosine, also known as the instantaneous phase.

(a) Determine the instantaneous frequency $\omega[n]$ for $x[n]$.

(b) Determine the instantaneous frequency of $x[n]$ specifically for $n = 0$ and $n = 2500$. Express the frequencies as a function of π (e.g., $\omega[1000] = \pi/100$).

(c) Plot $x[n]$ as a length-2501 signal. Label your horizontal axis as "Samples" and your vertical axis as "Amplitude."

(d) Use the `fft` function to compute the DFT of $x[n]$ (i.e., $X[k]$). Use the `abs` function in MATLAB to plot the magnitude response $|X[k]|$. Label your horizontal axis "Normalized frequency [rad / s]" with values $2\pi k/N$. Label your vertical axis "Magnitude." After plotting, apply the command `axis([0 2*pi 0 max(abs(X))])` (`abs(X)` is your magnitude response) to see the figure from 0 to 2π in frequency and 0 to the maximum value in magnitude.

(e) Use the `angle` function in MATLAB to plot the phase response of $\angle X[k]$. Label your horizontal axis "Normalized frequency [rad / s]" with values $2\pi k/N$. Label your vertical axis "Phase [rad]."

(f) What other well-known signal has a similar magnitude response (i.e., one that is relatively constant over frequency)? Why are the two signals significantly different in time?

Question #3: *(6 pts)* *The Discrete Fourier Transform over Time*

As we illustrated in the previous problem, the DFT is not particularly useful for describing chirp signals. In this problem, we are going to compute the DFT over time (i.e., a short-time Fourier transform) to better describe the chirp's behavior.

(a) Code the following short-time Fourier Transform process for your chirp $x[n]$:

- 1) Compute `M = floor(N/W)`, the number of length-W segments in $x[n]$.
- 2) Initialize a matrix `STFT = zeros(W, M);`.
- 3) Extract the first W samples (samples 0 to $W-1$) of the signal
- 4) Compute the DFT (using the `fft` function) of these W samples
- 5) Store the result of the DFT in `STFT(:,m)` where $m = 1$.
- 6) Iteratively repeat steps # 3 to # 5 for the next W samples (i.e., samples W to $2W-1$ and then samples $2W$ to $3W-1$ ... until you reach samples $(M-1)W$ to $MW-1$). Increase `m` with each iteration.

Use the MATLAB code

```
imagesc(0:(M-1), 2*pi*(0:(W-1))/W, abs(STFT))
xlabel('Time [samples]');
ylabel('Normalized Frequency [rad/s]');
```

to plot the short-time Fourier transform magnitude values for a W of your choosing.

(b) Plot the short-time Fourier transform values for 8 different values of W. Specifically, consider W equal to 10, 20, 40, 80, 160, 320, 640, and 1280.

(c) Your plots should illustrate the time-frequency uncertainty principle of signal processing: you cannot simultaneously measure time and frequency with fine resolution. For which value of W do you think best illustrates the chirp signal?

Question #4: *(8 pts)* *Audio DFT over Time*

Load the audio .mp4 file `rudenko_01.mp4` into MATLAB using

```
[x, Fs] = audioread(['rudenko_01.mp4']);
```

(a) Compute the short-time Fourier transform of your Rudenko music `x`. Use the MATLAB code

```
imagesc((0:(M-1))*W/Fs, 2*pi/W*(0:(W-1)), abs(STFT))
xlabel('Time [seconds]')
ylabel('Normalized Frequency [rad/s]');
```

to plot the short-time Fourier transform magnitude values for a $W = 10,000$. Use the `axis` command from Question #2 to focus only on the lower frequencies from 0 to $\pi/20$. Note that the code above it plotting time in seconds since we know the sampling rate.

(b) Use slightly different MATLAB code

```
imagesc((0:(M-1))*W/Fs, 2*pi/W*(0:(W-1)), 10*log10(abs(STFT)./
   max(max(abs(STFT)))), [-20 -5])
xlabel('Time [seconds]')
ylabel('Normalized Frequency [rad/s]');
```

to plot the short-time Fourier transform magnitude values for the same W. This normalizes the data so that the maximum value is 1 (0 dB) and plots the values in the decibel range of -20 dB to -5 dB to make the data is easier to visualize. Again, use the `axis` command from Question #2 to focus only on the lower frequencies from 0 to $\pi/20$.

(c) Just as in coding assignment #3 and #4, create a vector corresponding to the impulse response of a 10000-point running average filter

$$h[n] = \frac{1}{10000} \sum_{k=0}^{10000-1} \delta[n-k] \; .$$

Use `conv` or the textttfft to filter $x[n]$ and get an output $y[n]$. Compute the short-time Fourier transform of `y`. Plot the short-time Fourier transform magnitude values for the same W you chose wit the code in part (b).

(d) What are you observing in the plots? How is result related to the music? How does the filter affect the short-time Fourier transform?