Full Name: ____

EEL 4750 / EEE 5502 (Fall 2019) – Code #05 **Due Date:** **Nov. 4, 2019**

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

Question #2: *(10 pts)* *Filter Design*

Consider a chirp signal (slightly different from the prior assignment):

$$x[n] = \cos\left(\frac{2\pi n^2}{100000}\right).$$

Included with this assignment is a `[y, xSTFT, ySTFT] = stft_func(x, W)` function to compute the STFT of an input signal `x` with window size `W`. The function outputs of a modified signal (`y` – we will use in the next question), the STFT of `x` (`xSTFT`), and the STFT output of `y` (`ySTFT`).

For this question, assume $x[n]$ has length $N = 25000$, assume a sampling rate of $F_s = 10,000$ Hz, and use an STFT window of $W_{STFT} = 50$. Furthermore, consider two ideal filter impulse responses:

$$h_1[n] = \cos((\pi/4)n) + \cos((\pi/2)n) \qquad , \qquad h_2[n] = \cos((\pi/2)n) + \cos((3\pi/4)n)$$

Throughout this problem, use the sampling frequency to plot the STFT and magnitude response dimensions with time in seconds and/or frequency in Hertz. Always label your axes.

(a) Submit a plot of the magnitude of the STFT of $x[n]$.

(b) Use the windowing method to design a length W causal, approximate FIR filters for $h_1[n]$ and $h_2[n]$. Use a Hann window, defined by

$$w[n] = \frac{1}{2}\left[1 - \cos\left(\frac{2\pi n}{W}\right)\right](u[n] - u[n-W])$$

with lengths $W_1 = 50$, $W_2 = 10$ for $h_1[n]$ and $h_2[n]$, respectively. Refer to the approximate FIR filters as $g_1[n]$ and $g_2[n]$. Submit a plot of the magnitude response for each filter.

(c) Apply each filter ($g_1[n]$ and $g_2[n]$) to $x[n]$. Refer to the outputs as $z_1[n]$ and $z_2[n]$. Submit a plot of the magnitude of STFT of $z_1[n]$ and $z_2[n]$.

(d) Use `soundsc` to listen to $x[n]$, $z_1[n]$, and $z_2[n]$. What do the filters do?

(e) Repeat parts (b)-(d) with a rectangular window rather than a Hann window. Provide all of the requested plots.

(f) Based on your result, how does the choice of window affect the filter output?

(g) Build a new filter $g_3[n]$ by concatenating $g_1[n]$ and $g_2[n]$ with the code:

```
g3 = [g2; zeros(7000,1); g1;];
```

Note that this assumes `f1` and `f2` are column vectors. Apply $g_3[n]$ to $x[n]$. Refer to the result as $z_3[n]$. Submit a plot of the magnitude of STFT of $z_3[n]$.

(h) Use `soundsc` to listen to $z_3[n]$. Explain why the STFT looks and audio sounds as it does.

Question #3: *(10 pts)* *Inverse STFT*

In early classes, filtering is often used to "denoise" data. This assumes the noise is in a particular frequency band. Yet, this is typically false. The STFT allows us to overcome this assumption and allows us to create some other interesting effects. Let $X[m,k]$ represent the STFT of some signal $x[n]$ across discrete time / frames m and discrete frequencies k. Consider the following systems:

$$\text{System 1:} \quad Y[m,k] = \begin{cases} 0 & \text{for } |X[m,k]| < 15 \\ X[m,k] & \text{for otherwise} \end{cases}$$

$$\text{System 2:} \quad Y[m,k] = \frac{1}{2}Y[m-100,k] + \frac{1}{2}X[m,k]$$

Throughout this problem, use the sampling frequency to plot the STFT and magnitude response dimensions with time in seconds and/or frequency in Hertz. Always label your axes.

(a) As previously mentioned, the `STFT_func` function computes the STFT of $x[n]$, modifies the STFT (or will after this assignment – currently it does nothing), and then computes the inverse STFT to get an output $y[n]$. Modify the `STFT_func` by implementing system 1 at these lines (this may require more than one line of code):

```
% ***** PERFORM PROCESSING HERE AND ASSIGN ySTFT *****
ySTFT(:,m) = xSTFT(:,m);                     % REPLACE THIS
% ***************************************************
```

For this question, use a window of size $W_{STFT} = 1000$.

(b) Load `chiptune_noise.wav`. Implement system 1 across the frequency domain for each frame. Plot the magnitude of the STFT for the original music signal *and* the modified music signal. Use `axis` to zoom-in on same relevant (i.e., non-zero) information for each part.

(c) Listen to the result with `soundsc`. Based on the audio and STFT, what does system 1 do?

(d) Now modify the `STFT_func` by implementing system 2 at the previous lines (this may require more than one line of code). Again, use a window of size $W_{STFT} = 1000$.

(e) Load `chiptune_nonoise.wav`. Implement system 2 across the time domain. Plot the magnitude of the STFT for the original music signal *and* the modified music signal. Use `axis` to zoom-in on same relevant (i.e., non-zero) information for each part.

(f) Listen to the result with `soundsc`. Based on the audio and STFT, what does system 2 do?