The University of Texas at Austin
Dept. of Electrical and Computer Engineering
Midterm #1 ***Solutions 4.0***

Date: March 13, 2019 Course: EE 445S Evans

Name: **Games,** **Brain**
Last, First

- The exam is scheduled to last 50 minutes.
- Open books and open notes. You may refer to your homework assignments and the homework solution sets.
- Calculators are allowed.
- You may use any standalone computer system, i.e. one that is not connected to a network. ***Please disable all wireless connections on your computer system(s).***
- Please turn off all cell phones.
- No headphones allowed.
- All work should be performed on the quiz itself. If more space is needed, then use the backs of the pages.
- **Fully justify your answers.** If you decide to quote text from a source, please give the quote, page number and source citation**.**

Problem	*Point Value*	*Your score*	*Topic*
1	28		Filter Analysis
2	24		Sampling & Aliasing
3	24		Filter Design
4	24		Potpourri
Total	100		

Lectures 3 5 & 6	Lab #3
HW 1.1 2.1 2.2. 2.3 & 3.2	
JSK Ch. 7	
Fall 2018 Midterm 1 Prob 1	
Fall 2016 Midterm 1 Prob 1	

Problem 1.1 *Filter Analysis*. 28 points.

Consider the following causal linear time-invariant (LTI) discrete-time filter with input $x[n]$ and output $y[n]$ described by

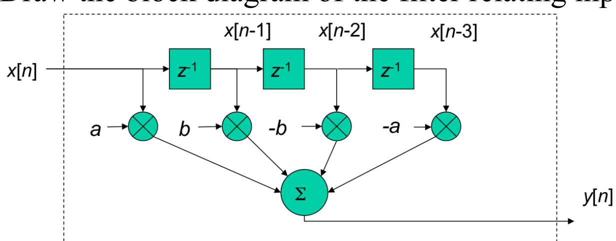
$$y[n] = a\,x[n] + b\,x[n-1] - b\,x[n-2] - a\,x[n-3]$$

for $n \geq 0$, where a and b are real-valued positive coefficients.

(a) Is this a finite impulse response (FIR) or infinite impulse response (IIR) filter? Why? *3 points*.
FIR filter. ***Any of the following reasons would provide sufficient justification*:**
1. The impulse response extends for 4 samples from $n = 0$ to $n = 3$, which is finite in duration.
2. The output $y[n]$ does not depend on previous output values; i.e., there is no feedback.
3. In the transfer function in the z-domain in part (d), the only poles are trivial poles at $z = 0$.

(b) What are the initial conditions and their values? Why? *6 points*.
Let n=0: $y[0] = a\,x[0] + b\,x[-1] - b\,x[-2] - a\,x[-3]$.
Let n=1: $y[1] = a\,x[1] + b\,x[0] - b\,x[-1] - a\,x[-2]$.
Let n=2: $y[2] = a\,x[2] + b\,x[1] - b\,x[0] - a\,x[-1]$. Etc.
Initial conditions are $x[-1]$, $x[-2]$, $x[-3]$ and must be zero for linearity and time-invariant properties to hold. Note that $x[0]$ is the first input value and not an initial condition.
Note: A causal system does not depend on future input values or future output values.

(c) Draw the block diagram of the filter relating input $x[n]$ and output $y[n]$. *6 points*.



Note: The three initial conditions are visible here as the initial condition for each unit delay block.

Lecture slides 3-15 & 5-4

(d) Derive a formula for the transfer function in the z-domain and the region of convergence. *4 points*.
***Z*-transform both sides of difference equation, knowing that all initial conditions are zero:**
$Y(z) = a\,X(z) + b\,z^{-1}\,X(z) - b\,z^{-2}\,X(z) - a\,z^{-3}\,X(z)$ which means

$$H(z) = \frac{Y(z)}{X(z)} = a + b\,z^{-1} - b\,z^{-2} - a\,z^{-3} \text{ for } z \neq 0$$

Lecture slide 5-11

(e) Give a formula for the discrete-time frequency response of the filter. *3 points*.
We can convert the transfer function $H(z)$ into the discrete-time frequency domain by substituting $z = \exp(j\,\omega)$ because FIR LTI systems are always Bounded-Input Bounded-Output stable, or equivalently, because the region of convergence includes the unit circle:

$$H_{freq}(\omega) = H(z)]_{z=e^{j\omega}} = a + b\,e^{-j\omega} - b\,e^{-2j\omega} - a\,e^{-3j\omega}$$

(f) Give a formula for the phase response vs. discrete-time frequency and the group delay vs. discrete-time frequency. Does the filter have linear phase over all frequencies? Why or why not? *6 points*.

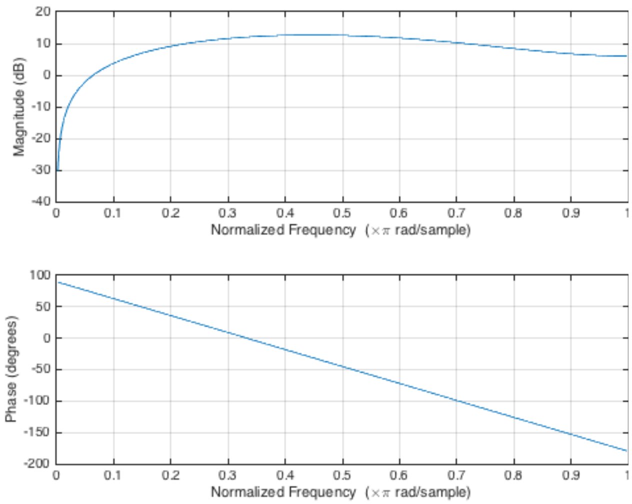
$$H_{freq}(\omega) = e^{-j\frac{3}{2}\omega}\left(a\,e^{j\frac{3}{2}\omega} + b\,e^{j\frac{\omega}{2}} - b\,e^{-j\frac{\omega}{2}} - a\,e^{-j\frac{3}{2}\omega}\right) = 2\left(a\sin\left(\frac{3}{2}\omega\right) + b\sin\left(\frac{\omega}{2}\right)\right) j\,e^{-j\frac{3}{2}\omega}$$

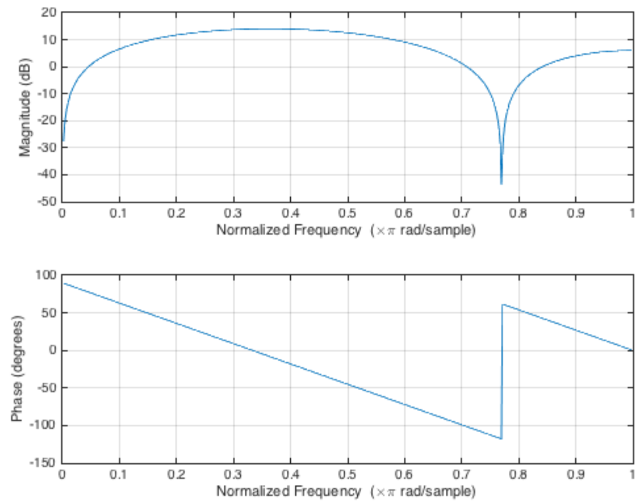
With $j = e^{j\frac{\pi}{2}}$, $\angle H_{freq}(\omega) = \frac{\pi}{2} - \frac{3}{2}\omega$ except for phase jumps (discontinuities) of π at frequencies that are zeroed out, which is generalized linear phase. $GD(\omega) = -\frac{d}{d\omega}\angle H_{freq}(\omega) = \frac{3}{2}$ samples.

Lecture slides 5-15 & 5-18	Spring 2013 Midterm 1 Prob 1(f)	Fall 2013 Midterm 1 Prob 3(b)

Problem 1.1 Supplemental information not expected for students to have provided in their answers.

Matlab plots using freqz([a b -b -a]) for a = 1 and b = 2 (left) and a = 2 and b = 1 (right)





Lecture 1 & 4 | Lab #2
HW 0.1 0.2 0.3
Fall 2016 Midterm 1 Prob 2

Problem 1.2 *Sampling and Aliasing.* 24 points.

A frequency of 46 kHz is higher than the normal audible range of 20 Hz to 20 kHz for a human being.

Consider a continuous-time signal $x(t) = \cos(2\,\pi\,f_0\,t)$ where $f_0 = 46$ kHz.

Sample the signal using a sampling rate of $f_s = 48$ kHz.

(a) Derive a formula for the discrete-time signal $x[n]$ that results from sampling $x(t)$. *3 points.*

Sampling in the time domain can be modeled as an instantaneous closing and opening of a switch. Each time that the switch is closed, the input is gated to the output. In practice, this could be implemented by a pass transistor with a sampling clock feeding the gate terminal.

$$x[n] = x(t)|_{t=nT_s} = \cos(2\,\pi\,f_0\,(n\,T_s)) = \cos\left(2\,\pi\,f_0\,\left(\frac{n}{f_s}\right)\right) = \cos\left(2\pi\left(\frac{f_0}{f_s}\right)n\right)$$

The discrete-time frequency corresponding to continuous-time frequency f_0 is $\omega_0 = 2\pi\frac{f_0}{f_s}$

(b) Using only analysis of $x[n]$ in the discrete-time domain, determine the discrete-time frequency to which the continuous-time frequency of f_0 will alias. *6 points.*

March 11th Lecture

$$x[n] = \cos\left(2\pi\left(\frac{f_0}{f_s}\right)n\right) = \cos\left(2\pi\left(\frac{46\text{ kHz}}{48\text{ kHz}}\right)n\right) = \cos\left(2\pi\left(\frac{23}{24}\right)n\right)$$

We can subtract an offset in the argument of $2\,\pi\,n$ without changing $x[n]$:

$$\cos\left(2\pi\left(\frac{23}{24}\right)n - 2\pi n\right) = \cos\left(2\pi\left(\frac{23}{24} - 1\right)n\right) = \cos\left(2\pi\left(-\frac{1}{24}\right)n\right) = \cos\left(2\pi\left(\frac{1}{24}\right)n\right)$$

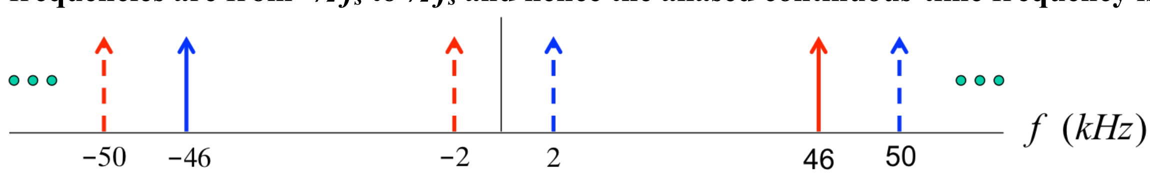
Continuous-time frequency of f_0 will alias to a discrete-time frequency of $2\pi\frac{1}{24}$ rad/sample.

(c) What is the equivalent continuous-time frequency for the aliased discrete-time frequency in (b)? *6 points.*

With $\omega_1 = 2\pi\frac{f_1}{f_s}$ and $f_s = 48$ kHz, $f_1 = 2$ kHz.

(d) Using only analysis in the continuous-time frequency domain of sampling applied to $x(t)$, determine the continuous-time frequency to which the continuous-time frequency f_0 will alias. The answer should be the same as part (c). *6 points.*

In the time domain, we model instantaneous gating of input to output every T_s seconds as a multiplication of the input signal by an impulse train with impulses every T_s seconds. The output spectrum is the convolution of the input spectrum and an impulse train with impulses separated by f_s with area f_s. In the frequency domain, sampling creates replicas of the input spectrum at offsets of integer multiples of f_s. The Fourier transform of $\cos(2\,\pi\,f_0\,t)$ is $\frac{1}{2}\delta(f+f_0) + \frac{1}{2}\delta(f-f_0)$. Replicas are shown as dashed impulses below. Reconstructed frequencies are from $-\frac{1}{2}f_s$ to $\frac{1}{2}f_s$ and hence the aliased continuous-time frequency is 2 kHz.



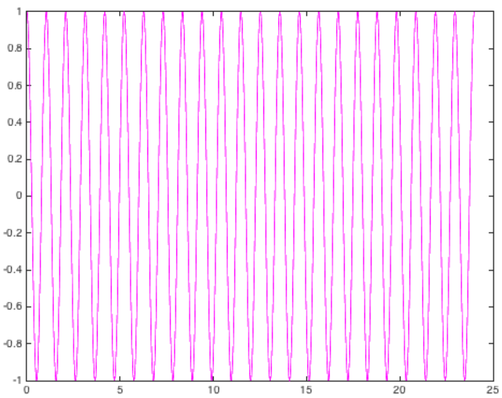
(e) Is the aliased frequency audible? *3 points.*

Yes, the aliased frequency of 2 kHz is in the audible range of 20 Hz to 20 kHz.

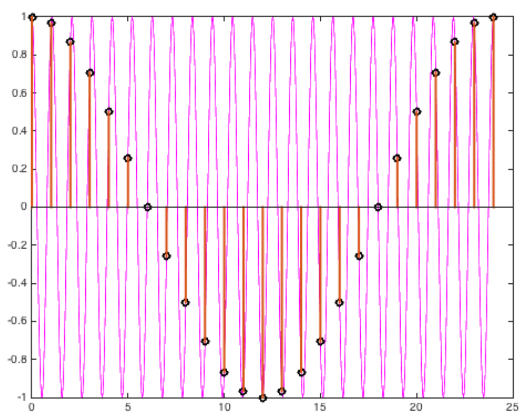
Problem 1.2 Supplemental information not expected for students to have provided in their answers.

Matlab code to show aliasing in the time domain

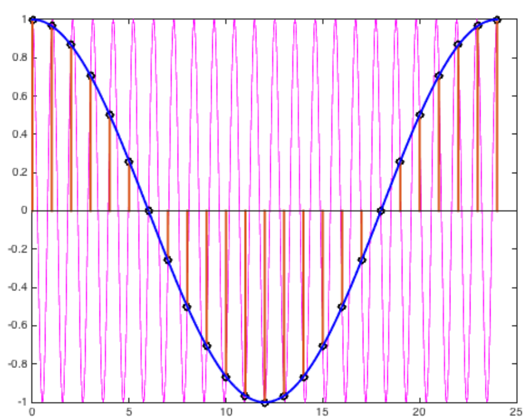
Plot $x(t) = \cos(2\pi f_0 t)$



Plot samples $x(n\, T_s)$ superimposed on $x(t) = \cos(2\pi f_0 t)$



Plot $x_1(t) = \cos(2\pi f_1 t)$ and $x(n\, T_s)$ superimposed on $x(t) = \cos(2\pi f_0 t)$




```
%% Part 1: Define Signals
wHat = 2*pi*(1/24);
nmax = 24;
n = 0:nmax;
x1 = cos(wHat*n);
x = cos(2*pi*(23/24)*n);

fs = 1;     %% fs=1 to align DT and CT
f1 = 2/48;  %% Actual fs goes in denom
w1Hat = 2*pi*f1/fs;
period = round(fs/f1);
f0 = 46/48; %% Actual fs goes in denom
w0Hat = 2*pi*f0/fs;
Ts = 1/fs;
tmax = (nmax/period)*(1/f1);
t = 0 : (Ts/100) : tmax;
x1cont = cos(2*pi*f1*t);
xcont = cos(2*pi*f0*t);

%% Part 2: Generate Plots
figure;
plot(t, xcont, 'm-', 'LineWidth', 1);

figure;
plot(t, xcont, 'm-', 'LineWidth', 1);
hold;
stem(n, x1, 'Linewidth', 2,
'MarkerEdgeColor', 'black');
stem(n, x, 'Linewidth', 2,
'MarkerEdgeColor', 'black');

figure;
plot(t, xcont, 'm-', 'LineWidth', 1);
hold;
stem(n, x1, 'Linewidth', 2,
'MarkerEdgeColor', 'black');
stem(n, x, 'Linewidth', 2,
'MarkerEdgeColor', 'black');
plot(t, x1cont, 'b-', 'LineWidth', 2);
```

Lectures 1 4 5 & 6

HW 1.1 2.1 3.1 3.3 | Lab #3

Problem 1.3 *Filter Design.* 24 points.

An electrocardiogram (ECG) device records the heart's electrical potential versus time for monitoring heart health and diagnosing heart disorders. [1]

Use a sampling rate f_s of 240 Hz for the continuous-time ECG signal for a monitoring application. [1]

Design a third-order discrete-time infinite impulse response (IIR) filter to remove baseline wander noise below 0.5 Hz and powerline interference at 60 Hz in an ECG signal. [1]

Baseline wander noise is induced by electrode changes due to perspiration, movement and respiration.

The third-order discrete-time IIR filter will be a cascade of a first-order and a second-order section.

(a) Design a first-order discrete-time IIR filter to remove DC (0 Hz) but pass as many of the other frequencies as possible with a gain of one in linear units. Please give the pole, zero, and gain. *6 points*.

Fall 2014 Midterm 1 Prob 3 notch

Spring 2015 Midterm 1 Prob 3(b)

Spring 2016 Midterm 1 Prob 1

Lecture Slide 6-7 Filter Demos

Pole p_0 = 0.95 and zero z_0 = 1.

$$H_0(z) = C_0 \frac{1 - z_0 z^{-1}}{1 - p_0 z^{-1}} = C_0 \frac{z - z_0}{z - p_0}$$

Set **$H_0(z)$ = 1 at $z = \exp(j\,\pi) = -1$ to give C_0 = 0.975.**

(b) Design a second-order discrete-time IIR filter to remove 60 Hz but pass as many of the other frequencies as possible with a gain of one in linear units. Please give the two poles, two zeros, and gain. *6 points*.

Spring 2013 Midterm 1 Prob 3(a)

Spring 2014 Midterm 1 Prob 2(b)

Lecture Slide 6-6

HW 3.1(c)

$$\omega_{60} = 2\pi \frac{f_{60}}{f_s} = 2\pi \frac{60\ Hz}{240\ Hz} = \frac{\pi}{2}$$

Poles and zeros are at the same angle ω_{60}

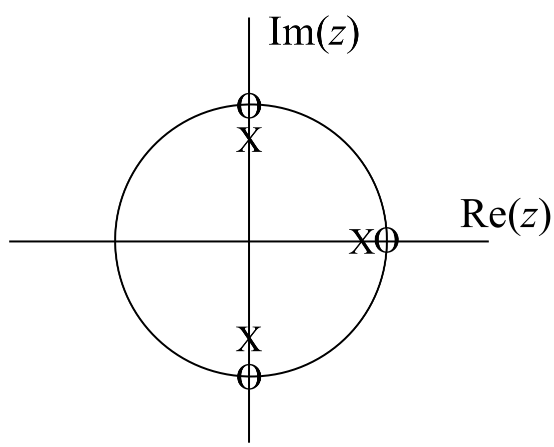
Poles at $p_1 = 0.9\, e^{j\,\omega_{60}} = j0.9$ and $p_2 = 0.9\, e^{-j\,\omega_{60}} = -j0.9$

Zeros at $z_1 = e^{j\,\omega_{60}} = j$ and $z_2 = e^{-j\,\omega_{60}} = -j$

$$H_1(z) = C_1 \frac{(1 - z_1 z^{-1})(1 - z_2 z^{-1})}{(1 - p_1 z^{-1})(1 - p_2 z^{-1})}$$

Normalize $H_1(z)$ = 1 at $z = \exp(j\,0) = 1$ to give C_0 = 0.905.

(c) Plot the poles and zeros for the third-order discrete-time IIR filter on the right. The circle on the right has a radius of 1. *6 points*.



(d) What is the response of the discrete-time IIR filter to continuous-time frequencies in the ECG signal that are odd harmonics of 60 Hz, i.e. 180 Hz, 300 Hz, etc.? Why? *6 points*.

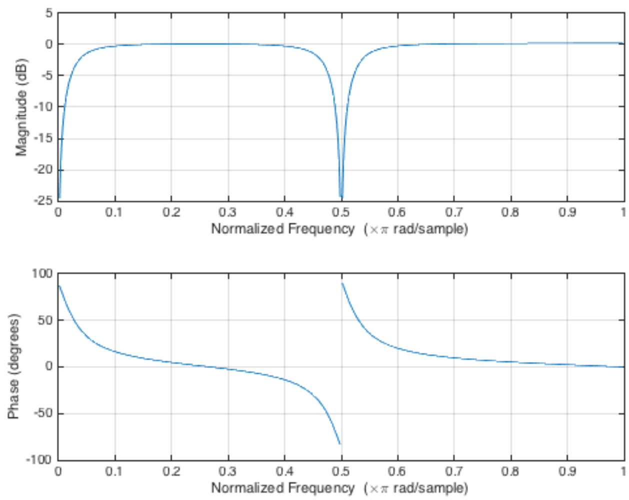
When sampled at the sampling rate of 240 Hz, continuous-time frequencies that are odd harmonics of 60 Hz will alias to the discrete-time frequency ω_{60}, and hence will be zeroed out by the discrete-time IIR filter.

[1] Yong Lian and Jianghong Yu, "A Low Power Linear Phase Digital FIR Filter for Wearable ECG Device", *Proc. IEEE Int. Conf. on Engineering in Medicine and Biology Society*, pp. 7357-7360, 2005.

Problem 1.3 Supplemental information not expected for students to have provided in their answers.

Matlab code to specify and analyze the discrete-time IIR filter.

```
%% Zeros
z0 = 1;
z1 = j;
z2 = -j;
%% Poles
p0 = 0.9;
p1 = 0.9j;
p2 = -0.9j;
%% Gains for each stage
C0 = 0.975;
C1 = 0.905;
%% Expand factors to coefficents
zeros = [z0 z1 z2];
poles = [p0 p1 p2];
feedforwardCoeffs = C0*C1*poly(zeros);
feedbackCoeffs = poly(poles);
%% Filter frequency response
freqz( feedforwardCoeffs, feedbackCoeffs );
```



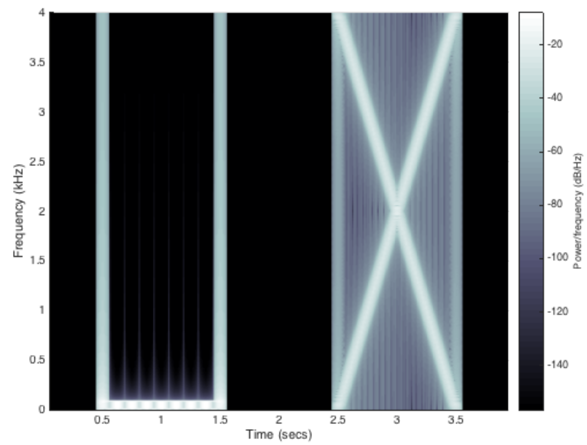
Lectures 1 3 4 5 & 6

HW 2.1 2.3 3.1 3.2 3.3 | Lab #3

Problem 1.4. *Potpourri.* 24 points.

(a) A discrete-time signal with sampling rate of f_s of 8000 Hz has the following "UX" spectrogram. The spectrogram was computed using 1000 samples per block and an overlap of 900 samples.

i. Describe frequency components vs. time. *6 points*.



By using the intensity scale shown to the right of the spectrogram plot:

t = 0.5s : all frequencies present

0.5s < t < 1.5s : Low frequencies 0 to 0.1 kHz continuously present (in white) plus six less intense short bursts of frequencies 0 to 1 kHz equally spaced in time (short rect. pulses)

t = 1.5s : all frequencies present

2.5s < t < 3.5s : chirp increasing from 0 to ½f_s plus a chirp decreasing from ½f_s to 0

ii. What would the signal sound like when played as audio signal? *6 points*.

0.5s < t < 1.5s : Bass tones 20-100 Hz plus lower intensity 0-1 kHz freq. repeated 6 times

2.5s < t < 3.5s : Note increasing 0 to 4 kHz, and note decreasing 4 to 0 kHz, with time

(b) Consider an unknown causal, time-varying, nonlinear, discrete-time system with input $x[n]$ and output $y[n]$. We will model the system as a discrete-time linear time-invariant (LTI) finite impulse response (FIR) filter. Find the FIR coefficients.

i. Give a formula for a finite-length input signal other than an impulse that contains all frequencies. *3 points*.

Fall 2017 Midterm 1 Prob 4(a)

HW 1.2

The discrete-time frequency domain has period 2π.

Input signal $x[n]$ of N samples should contain all discrete-time frequencies from -π to π. Use a chirp signal that linearly sweeps all frequencies from 0 to π:

$$x[n] = \cos\left(2\pi\left(\frac{n}{4N}\right)n\right) = \cos\left(\frac{2\pi}{4N}n^2\right) \quad for\ n = 0, 1, \ldots, N-1$$

ii. Using your answer in part i, derive a time-domain algorithm to estimate the FIR filter coefficients. Your algorithm should also be able to determine how many FIR filter coefficients are meaningful. *9 points*.

In-class discussion Feb. 20th & 25th

Fall 2013 Midterm 1 Prob 3(a)

We base the algorithm on convolution:

$$y[n] = h[n] * x[n] = \sum_{k=0}^{K-1} h[k]\, x[n-k]$$

For each output value $y[n]$, we'll have one equation and one unknown $h[n]$:

$y[0] = h[0]\, x[0]$ solve for $h[0]$ which works as long as $x[0]$ is not zero.

$y[1] = h[0]\, x[1] + h[1]\, x[0]$ solve for $h[1]$ which works as long as $x[0]$ is not zero

until **$|h[n]| < 10^{-5}$ *or* $n = N$**

Alternate criterion to $|h[n]| < 10^{-5}$: $\sum_{k=0}^{n} |h[k]|^2 \geq 0.9R$ where $R = \frac{\sum_{m=0}^{N-1} |y[m]|^2}{\sum_{m=0}^{N-1} |x[m]|^2}$

The alternate criterion will be able to handle some of the FIR coefficients values being close to zero in absolute value without stopping the update of the

Problem 1.4 Supplemental information not expected for students to have provided in their answers

1.4(a) Matlab code to generate the spectrogram.

```
fs = 8000;
Ts = 1 / fs;
tmax = 4;
utSignal = zeros(1, tmax*fs);
t1sec = 0 : Ts : (1 - Ts);
%% Spectrogram parameters
Nfft = 1000;
Noverlap = 900;
%% Generate low frequency groups
f0 = fs / Nfft;
lowfcosines = zeros(1, length(t1sec));
for n = 1 : 10
  f1 = n*f0;
  lowfcosines = lowfcosines + cos(2*pi*f1*t1sec);
end
%% Create chirp signals
fstart = 0;
fend = fs/2;
fstep = fend - fstart;
phi = pi*fstep*(t1sec.^2);
upchirp = cos(2*pi*fstart*t1sec + phi);
downchirp = cos(2*pi*fend*t1sec - phi);
%% Draw U into spectrogram
utSignal(0.5*fs+1:1.5*fs) = lowfcosines;
%% Draw X into spectrogram
utSignal(2.5*fs+1:3.5*fs)= upchirp + downchirp;
%% Plot the spectrogram
spectrogram(utSignal, hamming(Nfft), Noverlap, Nfft, fs, 'yaxis');
colormap bone;
```

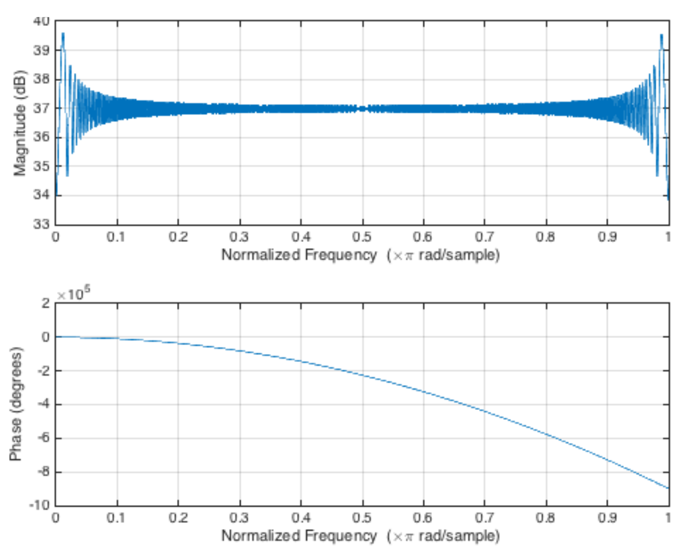
1.4(a)ii Matlab code to play the signal in the spectrogram in problem 1.4(a) as an audio signal

```
soundsc(utSignal, fs);
```

1.4(b)i Matlab code to generate chirp signal $x[n]$ of N samples in length. All frequencies are present in $x[n]$.

```
N = 10000;
n = 0 : N-1;
x = cos(((2*pi)/(4*N))*(n.^2));

%% Plot frequency content in x
freqz(x, 1, N);
```



1.4(b)ii Although not asked, here are two frequency-domain algorithms.

***Algorithm #1*: Computer $H(z) = Y(z) / X(z)$, take inverse transform to find $h[n]$, and truncate $h[n]$ to keep 90% of energy or N coefficients, whichever is smaller.**

***Algorithm #2*: Similar approach to Algorithm #1 using $H_{freq}(\omega) = Y_{freq}(\omega) / X_{freq}(\omega)$.**