ELEC 3004 / 7312 – Digital Linear Systems: Signals & Controls **2014**

Problem Set 3: Filters

Total marks: 80 **Due Date:** Wednesday, May 14, 2014 (at 11:59pm, AEST)

Note: This assignment is worth **15%** of the final course mark. Please submit answers via Platypus. It is requested that solutions, including equations, should be typed please. The final grade is the median of the marks from the peer reviews and the staff (with provisions for review). Finally, the tutors will ***not*** assist you further unless there is real evidence you have attempted the questions. Thank you very much. :--)

Short Questions

(Please keep it simple)

Q1. Discrete Transfer Functions [5 points]

Find discrete transfer functions, using the Z-Transform, for the following difference equations. The algebra alone is sufficient. No discussion or MATLAB required.

1. $y[n]+22y[n-12]=12x[n-2]-16x[n-9]$
2. $y[n]=x[n-9]+16x[n]$

Q2. Calculus: You Can't Beat Discrete [5 points]

Briefly derive the transfer functions for both the:

1. Digital integrator; and
2. Digital differentiator.

Do your transfer functions uphold the fundamental theorem of calculus? In particular, the first part of the theorem, called the **first fundamental theorem of calculus;** i.e. that integration can be reversed by differentiation. We know this in continuous time as: $\int \frac{dy}{dx}dx=y$.

Hint: Start with the difference equations for the integrator and differentiator. Then use the Z-Transform to get the transfer functions as in Q1. See what happens when multiply the transfer functions together.

Use the backward, causal, discrete-time versions of the differentiator and integrator, being sure to include the sampling interval. No MATLAB required.

Q3. Derivative Stability **[10 points]**

In MATLAB, generate a 50-vector of white Gaussian noise: `noise = wgn(1, 50, 1)`. Run the noise through a 1kHz digital differentiator (hint: see Q2). Do not use MATLAB's `diff()` function.

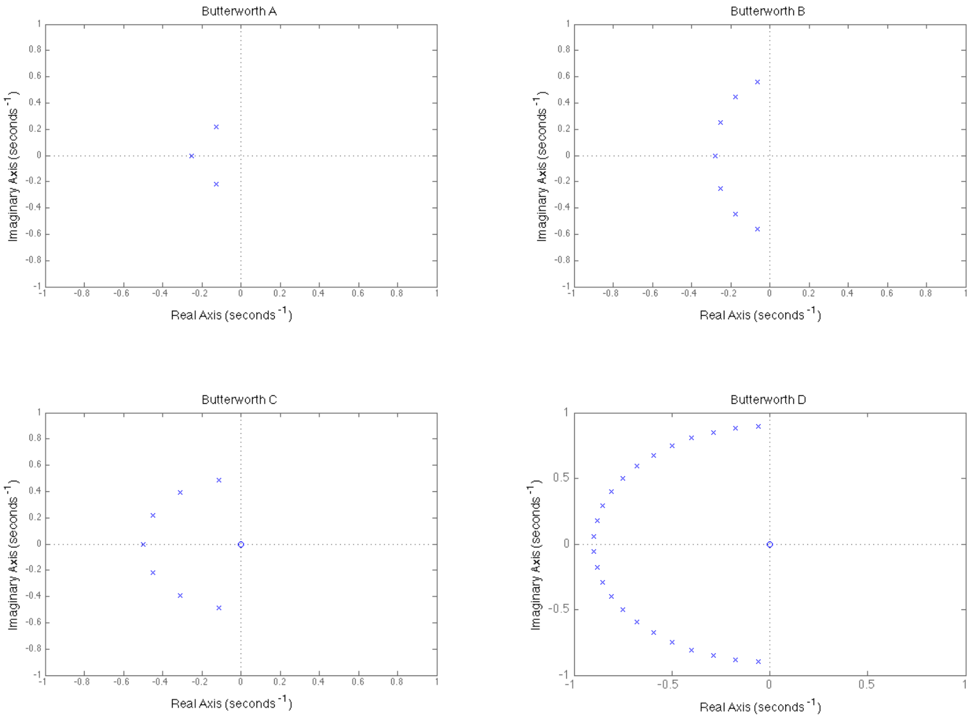
1. Now using this FIR filter[1], can you make the differentiator more stable? Present before and after plots to demonstrate the difference.
   Hints:
   `load('LPF.mat')`; then use the `filter(...);` command to apply LPF as a filter.
   Also use `filter(...)` to apply your differentiator to the noise from `wgn(...)`

2. Briefly discuss what the filter (LPF) is doing. Provide a plot of the frequency response of this filter, using `freqz()`.

[1] http://robotics.itee.uq.edu.au/~elec3004/assignments/PS3_Q3_LPF.mat

Q4. For What It's Butterworth [15 points]

You are given the following pole-zero maps[2] for four analog Butterworth filters A, B, C and D.



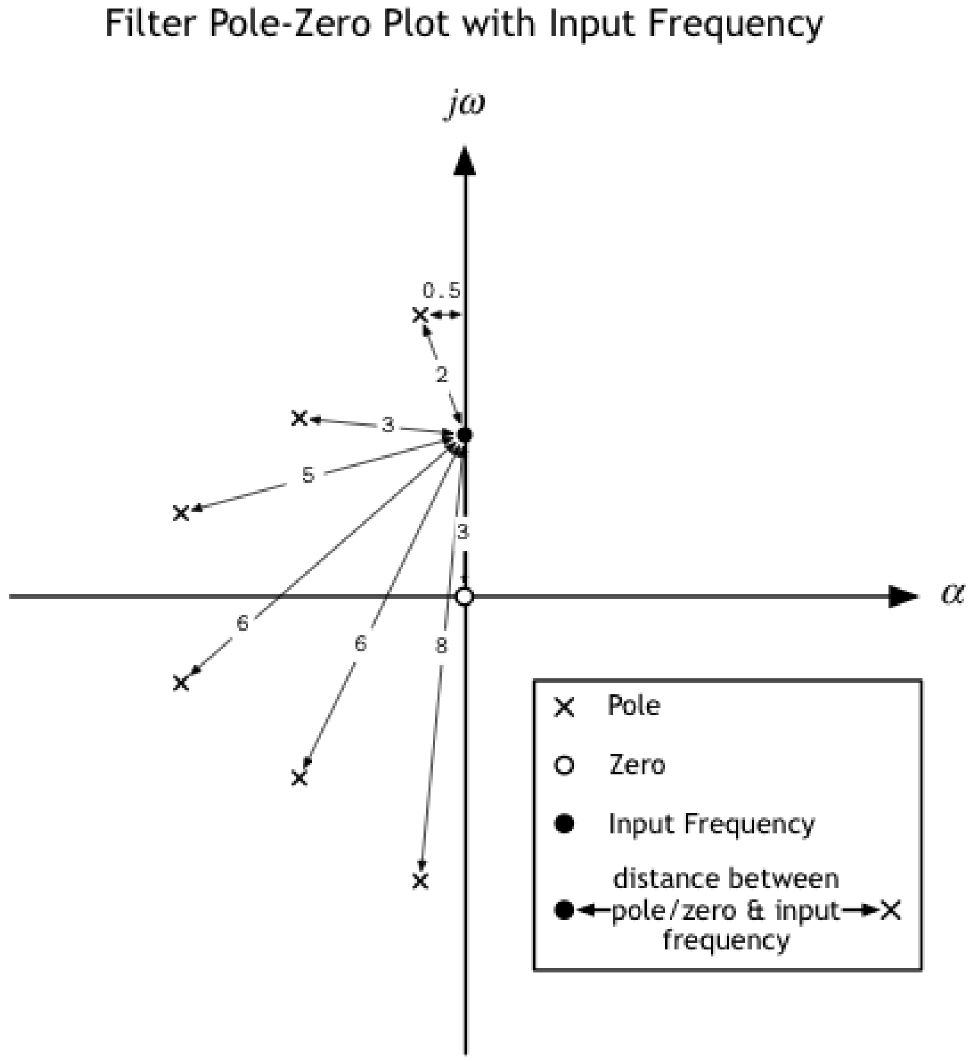
Answer the following questions about these plots. You are encouraged to experiment with MATLAB, using `butter()`, `cheby1()` and other filter design functions to see how the order, cutoff frequency etc… influence the pole-zero plots. Make sure to design analog filters by passing the `‘s’` parameter. See MATLAB's documentation.

1. Which filter has the highest order?
2. Approximately what are the cutoff frequencies of each filter?
3. Which filters are low pass, high pass or bandpass?
4. Something seems odd about one of the Butterworth filters. Which filter is the odd one out? How might its unusual shape affect its performance?

[2] http://robotics.itee.uq.edu.au/~elec3004/assignments/PS3_Q4_butters.png

Q5. Easy PZ **[15 points]**

Consider the following plot. Refer to Lathi, 1st Ed, Section 7.4 *Filter Design by Placement of Poles and Zeros of H(s)*, page 496

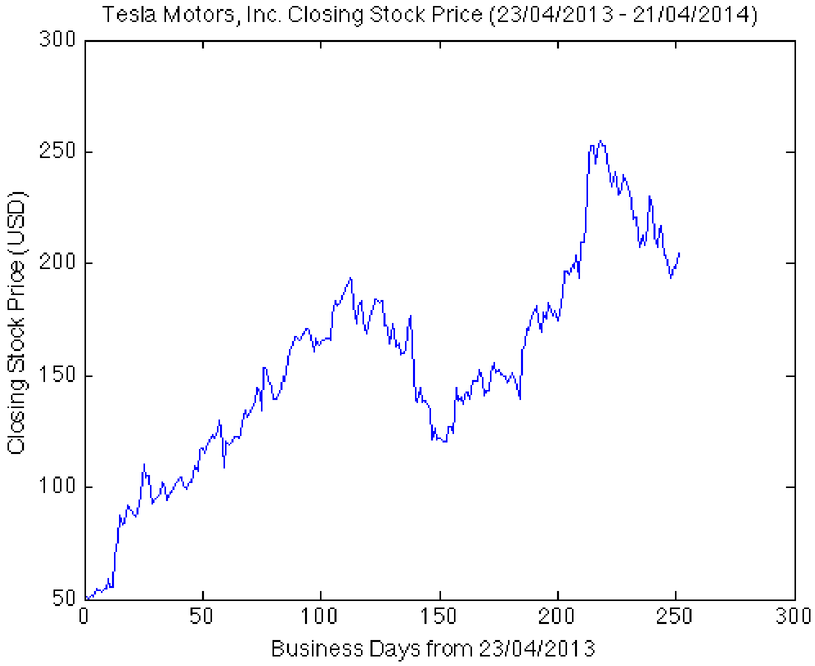


Derive the following details about the filter represented by this pole-zero plot.

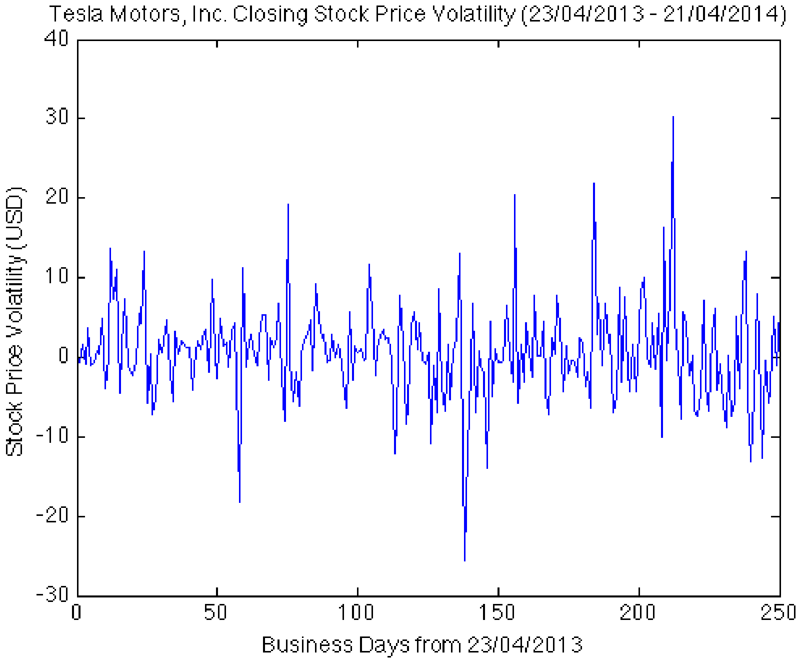
1. Input Frequency Gain (assume a constant factor of 10^4 in the transfer function numerator)
2. Cutoff frequency
3. Can the Input Frequency Phase be determined? If so, what is it? If not, why?

Q6. Filters: What More Could You Quant? [30 points]

Day trader Nick Leeson is researching Tesla Motors, Inc[3]. He wants to predict a trend using Tesla's closing stock prices from 23/04/2014 to 21/04/2014.



Nick takes the derivative to quantitatively assess the current trend of Tesla's closing stock price.



However, he finds the derivative is too volatile to be useful. Nick has contracted you to clean this data. He has provided the prices in a matlab data file[4] for you.

[3] https://www.google.com/finance?q=tesla

[4] http://robotics.itee.uq.edu.au/~elec3004/assignments/PS3_Q6_tslaClosingPrice.mat

Part I: Let's Get Moving:
A Simple Moving Average (SMA) Filter

Write a MATLAB program to do the following to clean the data with a moving average filter:

1. Develop a difference equation for a causal, five-day simple moving average filter.
2. Calculate the filter's z-domain transfer function.
3. Realise the system (make a block diagram) using delay (z^{-1}) elements.
4. Using any method you like, calculate and plot the new, five-day averaged data set. Then plot its discrete derivative, using `diff()`.

Discuss the approach of averaging before analysis. Specifically:

1. Is this approach analogous to any traditional methods in electrical engineering?
2. Is a moving average filter like a low-, high- or bandpass filter? Or is it something entirely different?

Part II. Let's Get Exponentially Rich:
An Exponential Moving Average (EMA) Filter

After contemplating the nature of the stock market, you decide to change the filter. You think that older prices are less relevant to the current trend. You decide to exponentially lower the weight of older terms in the average.

You may use the following difference equation.

$$m[n] = \alpha \cdot p[n-1] + (1-\alpha) \cdot m[n-1]$$

$$m[n] = \text{exp moving average on day n}$$

$$p[n] = \text{closing stock price on day n}$$

Follow steps 2, 3 and 4 (of Q7) again, tuning α as you see fit.

Discuss the new results. Specifically:

1. Have the results actually changed?
2. What is the effect of alpha?
3. Is it possible to implement an FIR EMA filter?
4. If you could implement an FIR EMA filter, would you? Why or why not?

Hint: Although you cannot use convolution to compute the response of an IIR filter, you can still use MATLAB's `filter()`. Always use `filter()` when filtering and `freqz()` when plotting the frequency response of a filter.