Kang-Wei Chang
kwc305
N17515255
Lab 2 Sep. 22

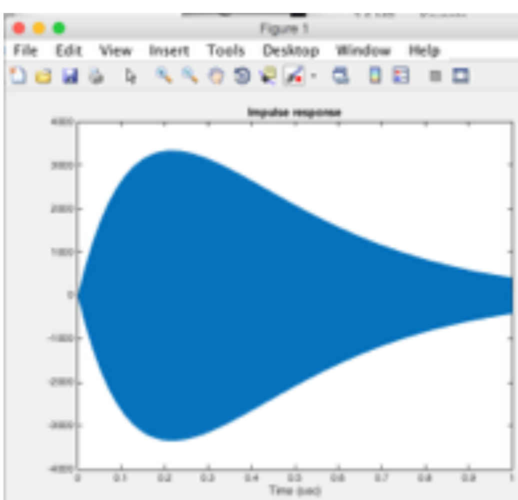
Assignment

4-10:

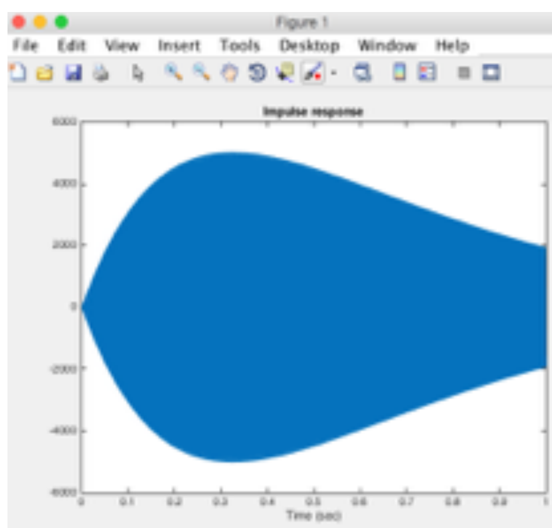
In this question, we first try to design 2 different second order filter and then do the convolution and cascade it.
On the differential equation,we found that log((1-r1)/(1-r2))/log(r2/r1). So, r2 needs to larger than r1.
So, for this equation, I have the detailed derivation in this report.
Basically, I implement the peak estimation value in the both Matlab and the python code.

For the rising/falling time

See the file Lab_2_ASGMNT_4_10_kwc305.m
First I focus on the falling time. I found if I change the value of Ta, the falling time will change. To be more specific, if I larger the Ta, the falling time will become larger. On the other hand, if I decrease the Ta value, the falling time will decrease.



original plot



Ta = 1.5, falling time decrease

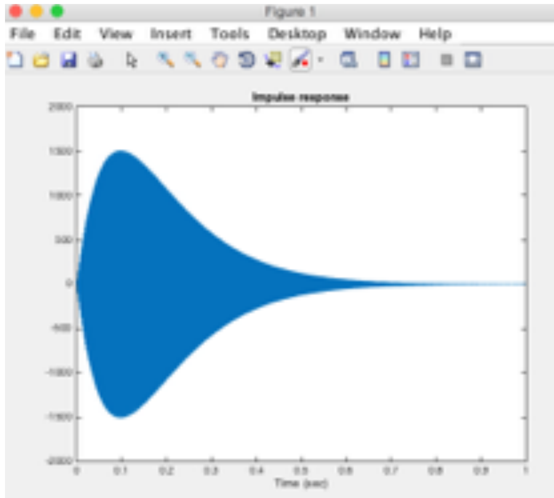


Figure 1
File Edit View Insert Tools Desktop Window Help
Impulse response
Time (sec)

Ta = 0.5

For the rising time:

My function derivation :

$$h[n] = \sum_{k=0}^{n} r_1^{k} r_2^{n-k}$$

$$= r_2^{n} + r_1 r_2^{n-1} + \cdots r_1^{n-1} r_2 + r_1^{n}$$

$$\frac{r_2}{r_1} h[n] = \frac{r_2}{r_1}\left(r_2^{n} + \cdots + r_2 r_1^{n-1}\right)$$

$$\left(\frac{r_2}{r_1} - 1\right) h[n] = \frac{r_2^{n+1}}{r_1} - r_1^{n}$$

$$h[n] = \frac{\frac{r_2^{n+1}}{r_1} - r_1^{n}}{\frac{r_2}{r_1} - 1}$$

$$h[n] - h[n-1] = \frac{\frac{r_2^{n+1}}{r_1} - r_1^{n}}{\frac{r_2}{r_1} - 1} - \frac{\frac{r_2^{n}}{r_1} - r_1^{n-1}}{\frac{r_2}{r_1} - 1} = 0$$

$$\frac{r_2^{n+1}}{r_1} - r_1^{n} - \frac{r_2^{n}}{r_1} + r_1^{n-1} = 0 \xrightarrow{\times r_1} r_2^{n+1} - r_1^{n+1} - r_2^{n} + r_1^{n} = 0$$

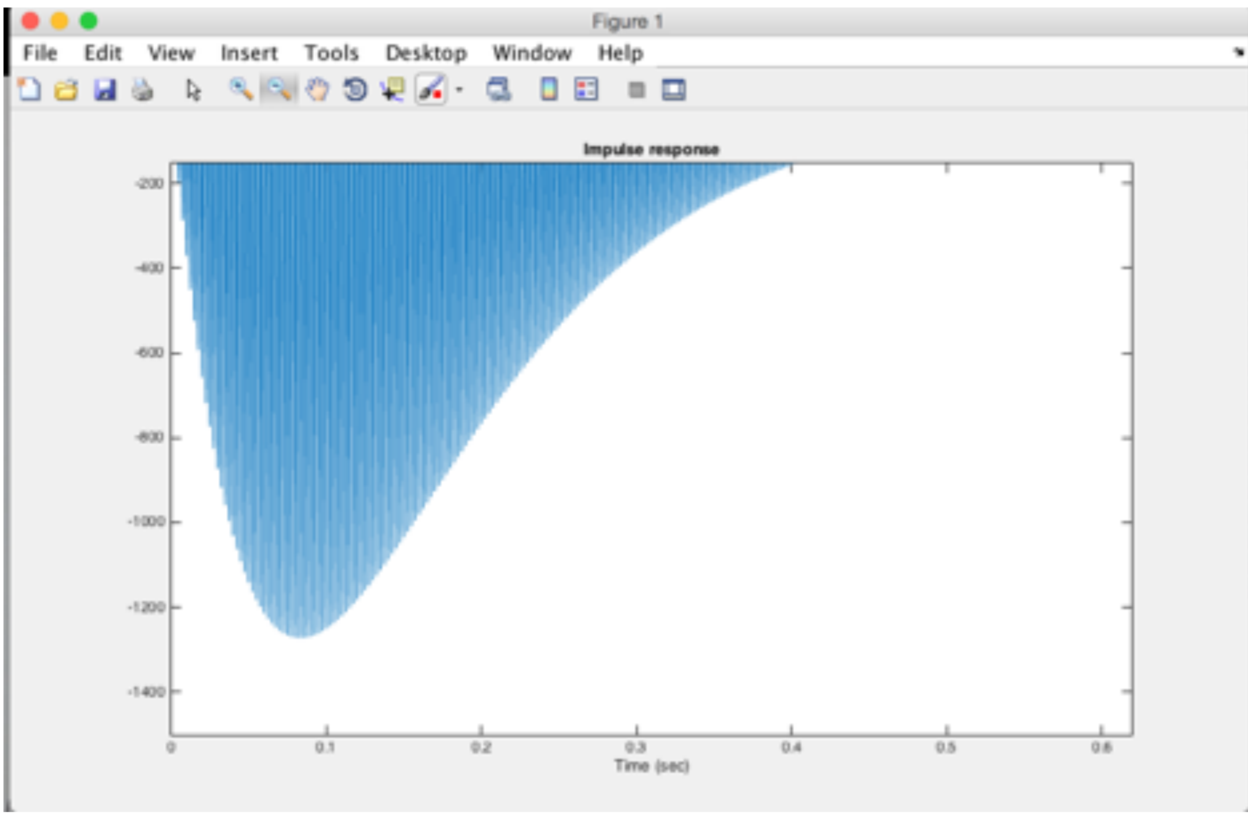
$$r_2^{n}(r_2 - 1) + r_1^{n}(1 - r_1) = 0$$

$$r_2^{n}(1 - r_2) = r_1^{n}(1 - r_1)$$

$$n \log r_2 + \log(1 - r_2) = n \log r_1 + \log(1 - r_1)$$

$$n = \frac{-\log(1 - r_2) + \log(1 - r_1)}{\log r_2 - \log r_1} = \frac{\log\left(\frac{1 - r_1}{1 - r_2}\right)}{\log\left(\frac{r_2}{r_1}\right)}$$

And I take
Ta1 = 0.5
Ta2 = 0.3
r1 = 0.9988
r2 = 0.9981
base on the result, I get that n equals to 655 and 655/8000 = 0.0081
Which pretty accurate with the result on the following:



Implement on python:

See the file Lab_2_ASGMNT_4_10_kwc305.py, I change the original file. I did with 2 second order filter and specify the Ta1, Ta2 and print the r1 and r2. Also, I have the gain check on the code.