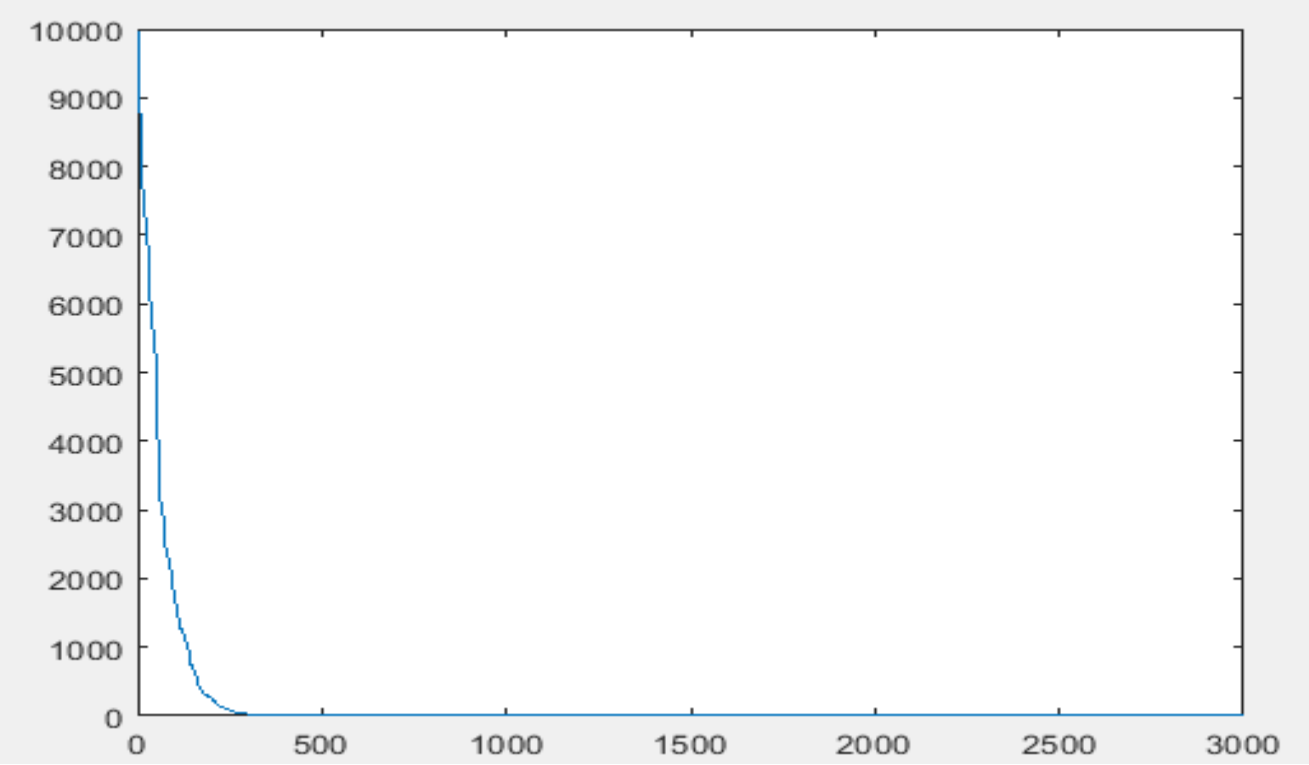
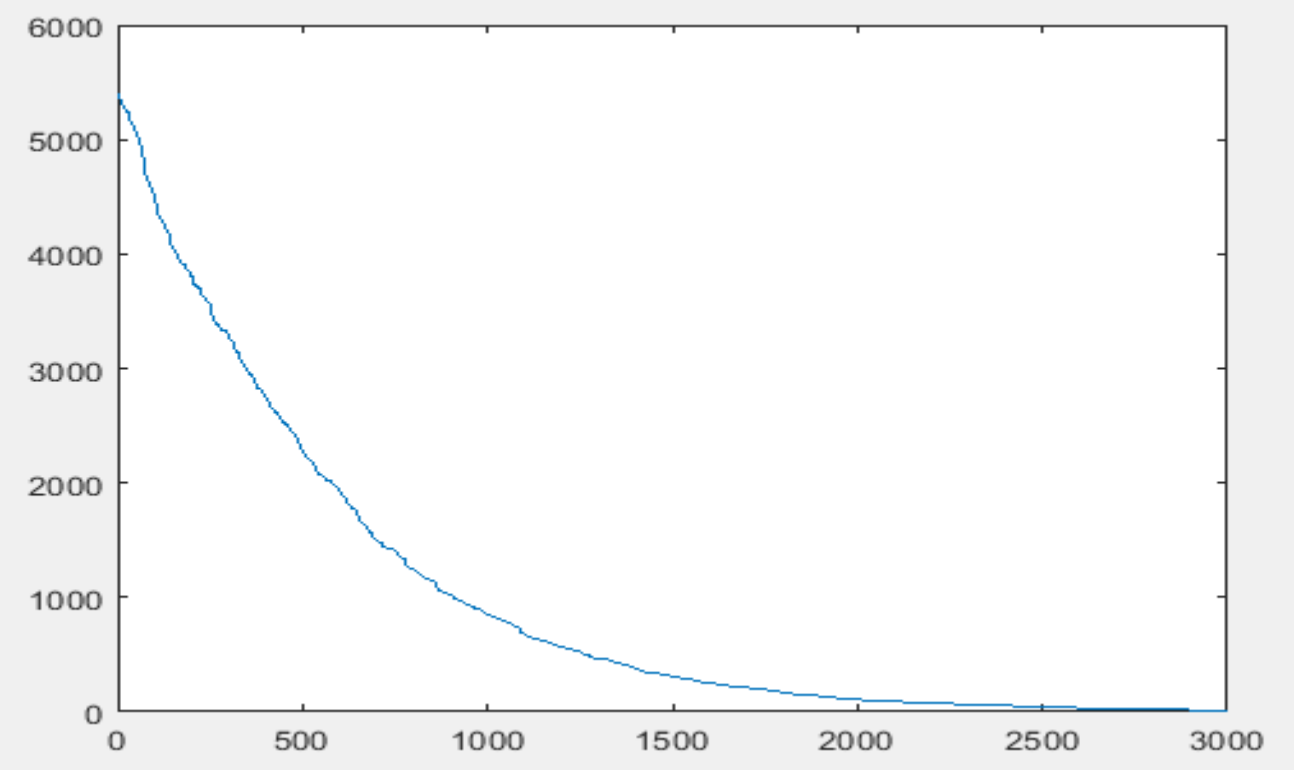
```
diary off
format compact
%Johnny Li
%EEL3135 Fall 2018
%Lab 5 Part 1

%1.1
%Use matrix multiplication to apply the first-difference filter to the
%following input:
%x[n]=4d[n]+3d[n-1]-9d[n-3]+2d[n-5]
%Code edit from given version.
K=6;
L=2;
x = [4; 3; 0; -9; 0; 2];
xx = zeros((K+L-1),L);
for i=1:L
    xx((i:(K+i-1)),i) = x;
end

%Filter Weights
w=[1,-1];
%Matrix multiplication of input vector xx and w filter weight vector.
yy=xx.*w;

%Output
yy

yy =

     4     0
     3    -4
     0    -3
    -9     0
     0     9
     2     0
     0    -2

type LMS

%LMS LMS Algorithm
%Script for 1.2

%1.2.1

%LMS algorithm to learn the desired 3-point averager filter.
%Code given.
K=3000;
L=3;
x = randn(K,1); %generates 3000 random numbers as input
d = conv(x,[1/31/31/3]); %generates desired output
dd = d(1:K); %takes 3000 values of output
xx = zeros((K+L-1),3); %declare input matrix
for i=1:L
    xx((i:(K+i-1)),i) = x; %generate input matrix
end
```

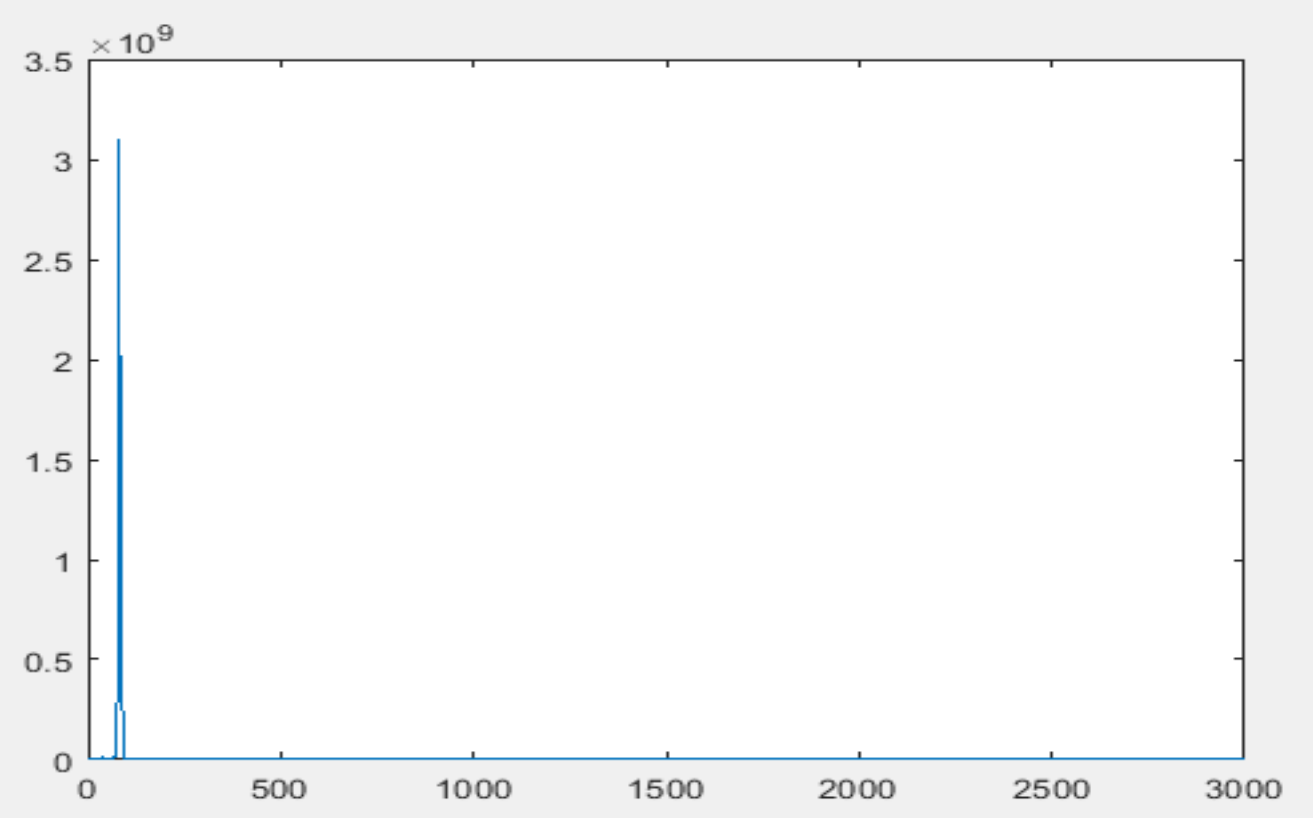
```
xx = xx(1:K,:); %takes 3000 rows of input matrix
w = randn(3,1);
mu = 0.01;

%Storage Vector
J=zeros(K,1);
%Loop to update the filter using b(n+1)=b(n)+ue[n]x[n].
for n=1:K
    yy=xx*w;
    %Calculate Error
    err=dd-yy;
    %Overall Error filter
    w=w+mu*err(n)*xx(n,:)';
    %Sum of Squared Error
    J(n)=err'*err;
end

%1.2.2

%Plot Learning Curve
plot(J);
```



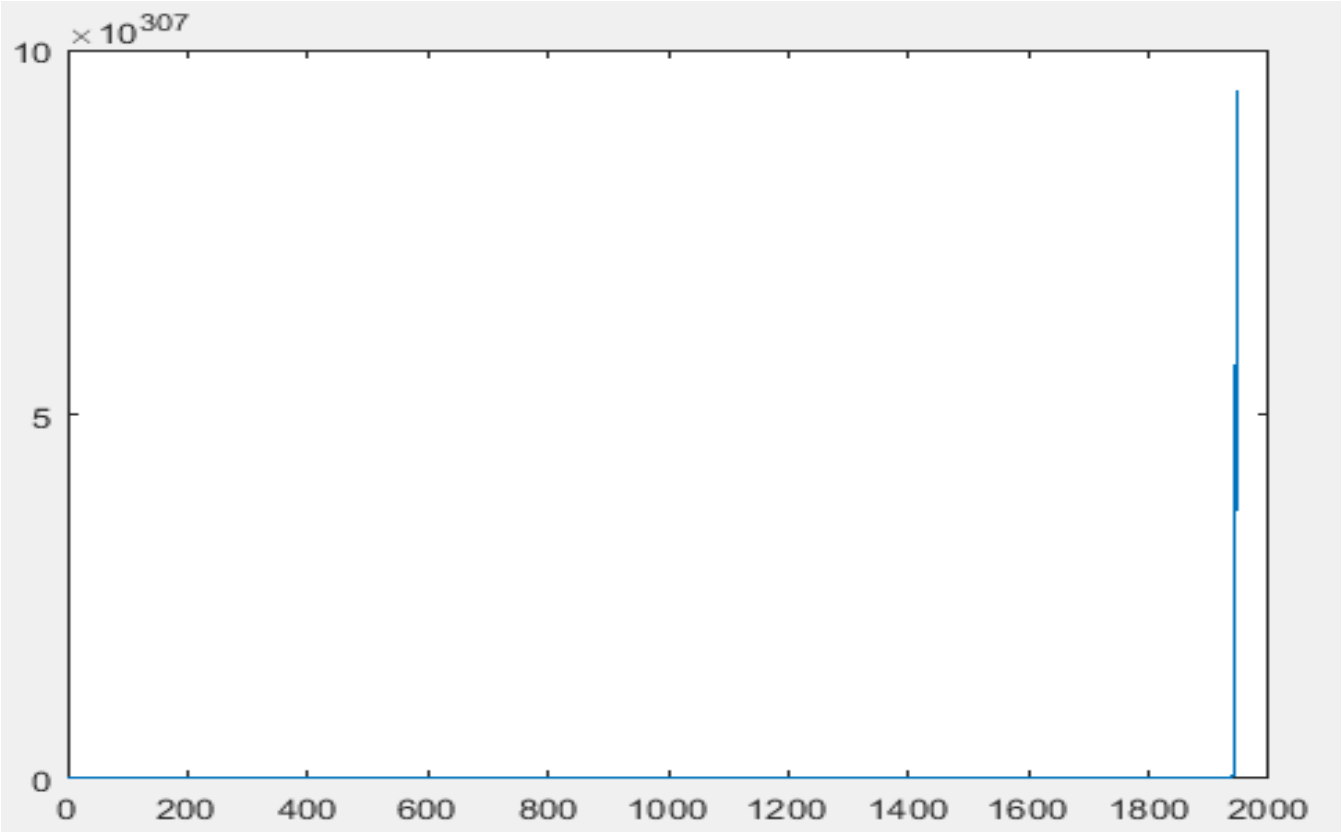
```
%1.2.3

%Quesiton: What are your final weights? Do they match the weights of the
%three-point averager?
%The final weights were the yy vector values, ranging from -0.0011 to
%0.0012 where these weights exactly match the weights of the three-point
%averager, dd vector.

%1.2.4

%Least Mean Squares algorithm with stepsizes 0.001, 0.5, 1, and 10 to adapt
%the three-point averager.


%Least Mean Squares algorithm with stepsizes 0.001
```



%Least Mean Squares algorithm with stepsizes 0.5.



%Least Mean Squares algorithm with stepsizes 1.



```
%Least Mean Squares algorithm with stepsizes 10.
```