```
clear;
clc;
close all;
```

Mohammad Javad Amin 401211193 Problem 1 , exercise 1

definition

d : desired signal N :length of filter M : length of input signal alpha : learning rate e : errors w : weights of filter m_error : mean squared error p : power of input signal alpha_int :alpha initiate

```
a=1;
b=[1,1.8,0.81];          % impulse response
inputs=randn(1,100);
d=filter(b,a,inputs);
M=length(inputs);

N=4;

[w,~]=RLS(inputs,d,N,M);
disp("weights for  N=4 :");
disp(w');

k=5;
m_error=zeros(1,M-N+1);

for i=1:k
    [~,cost]=RLS(inputs,d,N,M);
    m_error=m_error+cost;
end
m_error=m_error/5;
figure
plot(m_error);
title('squared error, N=4 ');
xlabel('Number of iterations');

[~,~,J_min,J_inf]=RLS(inputs,d,N,M);
J_ex=J_inf - J_min;
disp("excess squared error")
disp(J_ex)

N=4

for i=1:k
    [w,cost]=RLS(inputs,d,N,M);
    m_error=m_error+cost;
end
m_error=m_error/5;
disp("weights for N=4  :");
disp(w');
figure
plot(m_error);
title('squared error, N=4');
```

```
xlabel('Number of iterations');

% N=4

for i=1:k
    [w,cost]=RLS(inputs,d,N,M);
    m_error=m_error+cost;
end
m_error=m_error/5;
disp("weights for N=4  :");
disp(w');
figure
plot(m_error);
title('squared error, N=4 ');
xlabel('Number of iterations');
```

```
weights for N=4  :
    1.0000
    1.8000
    0.8100
    0.0000

weights for N=4  :
    1.0000
    1.8000
    0.8100
    0.0000
```

```
N=2

N=2;
m_error=zeros(1,M-N+1);

for i=1:k
    [w,cost]=RLS(inputs,d,N,M);
    m_error=m_error+cost;
end
m_error=m_error/5;
disp("weights N=2  :");
disp(w');
figure
plot(m_error);
title('Mean squared error, N=2 ');
xlabel('Number of iterations');

%  N=3


N=3;
m_error=zeros(1,M-N+1);

for i=1:k
    [w,cost]=RLS(inputs,d,N,M);
```

```
    m_error=m_error+cost;
end
m_error=m_error/5;
disp("weights for N=3 :");
disp(w');
figure
plot(m_error);
title('squared error, N=3');
xlabel('Number of iterations');
disp(" The LMS is more quicker than RLS algorithm but the error in RLS is much
 better than LMS ");
```

RLS algorithms

```
function[w,cost,J_min,J_inf]=RLS(inputs,d,N,M)
% z : error
% N :length of filter
% M : length of input signal

    z=zeros(1,M-N+1);
    w=zeros(1,N);
    lambda=0.6;
    delta= 1e-10;

    p=delta*eye(N);

    for i=N:M-1
        u=inputs(i:-1:i-N+1);
        y=dot(w,u);
        z(i-N+1)=d(i)-y;
        k=(p*u')/(lambda+u*p*u');
        w=w+k'*conj(z(i-N+1));
        p=(p -k*conj(u)*p)/lambda;

    end
    cost=z.^2;
    J_min=min(z);
    J_inf=sum(z(M-N-19:M-N))/20;

end
```

```
weights for  N=4 :
    1.0000
    1.8000
    0.8100
    0.0000
```

Published with MATLAB® R2022b