گزارش تمرین کامپیوتری ۱ درس سیگنال‌ها و سیستم‌ها

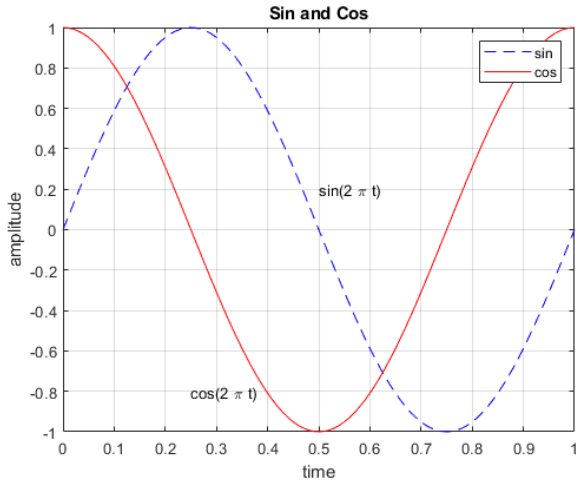
بابک حسینی محتشم

810101408

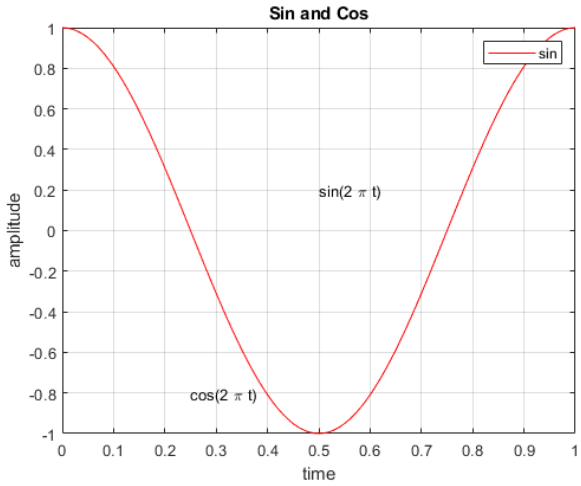
1403/7

بخش اول:

تمرین ۱-۱) در نهایت پس از اجرای کد شکل زیر تولید می‌شود.



با حذف خط 7 وقتی بار دوم به دستور کشیدن نمودار می‌رسد نمودار قبلی پاک می‌شود و نمودار جدید جایگزین آن می‌شود در صورتی که با وجود خط 7 همان طور که در شکل مشخص بود هر دو نمودار را داشتیم.



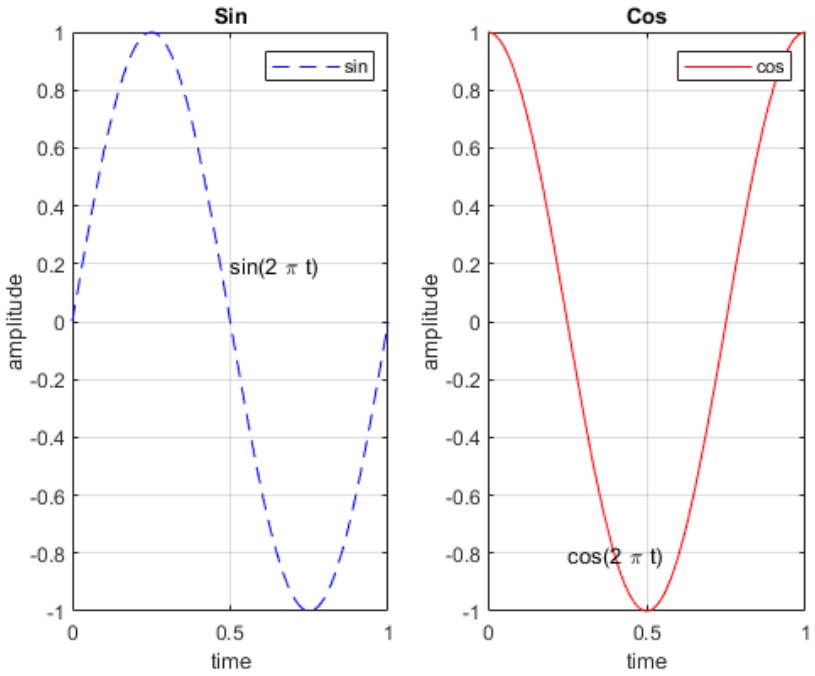
همچنین وارنینگ زیر داده می‌شود که ناشی از دادن دو ورودی به legend ولی داشتن تنها یک سیگنال است.

```
>> p1_1
Warning: Ignoring extra legend entries.
> In legend>process_inputs (line 590)
In legend>make_legend (line 319)
In legend (line 263)
In p1_1 (line 16)
```

تمرین ۱-۲) تصویر اسکریپت:

```
t=0:0.01:1;
z1=sin(2*pi*t);
z2=cos(2*pi*t);

figure;
subplot(1,2,1);
plot(t,z1,'--b')
title('Sin'); % Title
legend('sin')
xlabel('time') % the name of X-axis
ylabel('amplitude') % the name of Y-axis
grid on % Add grid

x0=[0.5 0.2];
s='sin(2 \pi t)';
text(x0(1), x0(2), s);
% hold on

subplot(1,2,2);
plot(t,z2,'r')
title('Cos'); % Title
legend('cos')
xlabel('time') % the name of X-axis
ylabel('amplitude') % the name of Y-axis
grid on % Add grid

y0=[0.25 -0.8];
s='cos(2 \pi t)';
text(y0(1), y0(2), s);
```

تصویر نمودار:

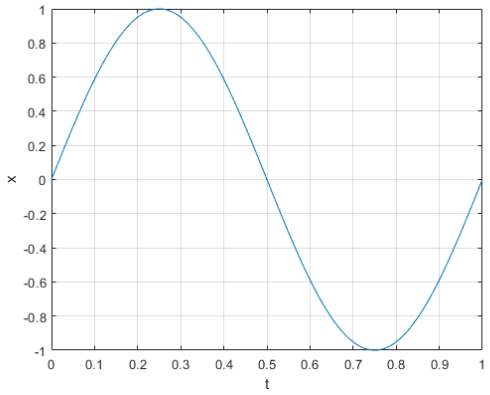


بخش دوم:

تمرین ۲-۱) تصویر اسکریپت:

```
1   p2=load('p2.mat')
2
3   figure;
4   plot(p2.t,p2.x)
5   xlabel('t') % the name of X-axis
6   ylabel('x') % the name of Y-axis
7   grid on % Add grid
```

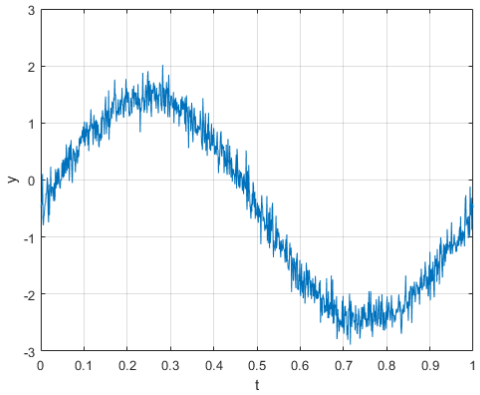
تصویر نمودار:



تمرین ۲-۲) تصویر اسکریپت:

```
p2=load('p2.mat')

figure;
plot(p2.t,p2.y)
xlabel('t') % the name of X-axis
ylabel('y') % the name of Y-axis
grid on % Add grid
```

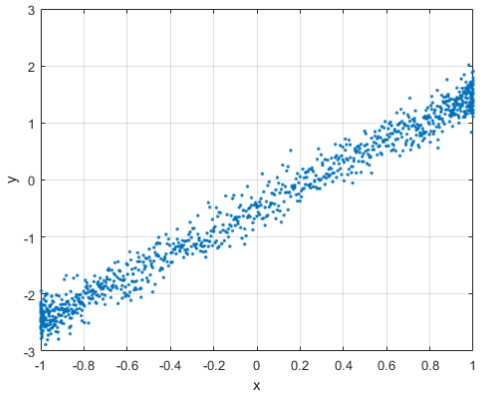
تصویر نمودار:



تمرین ۲-۳) تصویر اسکریپت:

```
p2=load('p2.mat')

figure;
plot(p2.x,p2.y,'.')
xlabel('x') % the name of X-axis
ylabel('y') % the name of Y-axis
grid on % Add grid
```

تصویر نمودار:



شیب این خط مقدار α و عرض از مبدا آن β را به ما می‌دهد.

تمرین ۲-٤) می‌شود با استفاده از جبرخطی پارامترها را یافت.

$$y = X\beta \rightarrow X^T y = X^T X\beta \rightarrow \beta = (X^T X)^{-1} X^T y$$

و به همین ترتیب با نوشتن تابع زیر مقدار پارامترها را یافت.

```
function b = linear_parameter_estimator(x,y)
    x=[x;ones(size(x))];
    x=x';
    y=y';
    b = (x'*x)\(x'*y);
end
```

برای تست درستی تابع داده‌هایی با پارمترهای $\alpha = 3$ و $\beta = 1$ ایجاد کرده و یک بدون نویز و بار دیگر با نویز به تابع می‌دهیم.

```
p2=load('p2.mat')

x=[-1:0.01:1];
a=3;
b=1;
y0=a*x+b;
linear_parameter_estimator(x,y0)
y1=y0+randn(size(y0));
linear_parameter_estimator(x,y1)
linear_parameter_estimator(p2.x,p2.y)
```

می‌توان دید که با دقت خوبی پارامترها درست حساب شده‌اند.

```
ans =

    3.0000
    1.0000


ans =

    3.1169
    0.8413


ans =

    1.9736
   -0.4983
```

پس برای داده شده پارامترهای تخمین زده شده می‌شود:

$$\alpha \approx 2\,, \beta \approx -0.5$$

همچنین می‌توان با کمینه کردن رابطه داده شده به پاسخ رسید برای این کار از تابع fminsearch می‌توان استفاده کرد که تابع هزینه را کمینه می‌کند.

```
% function b = linear_parameter_estimator(x,y)
%       x=[x;ones(size(x))];
%       x=x';
%       y=y';
%       b = (x'*x)\(x'*y);
% end
function ab = linear_parameter_estimator(x,y)
    fun=@(ab)sum((y-ab(1)*x-ab(2)).^2);
    ab0=[1,0];
    ab=fminsearch(fun,ab0);
end
```

می‌توان دید که خروجی مشابه با روش قبل می‌گیریم.

```
ans =

    3.0000
    1.0000


ans =

    3.0863
    1.0350


ans =

    1.9736
   -0.4983
```

تمرین ۲-۵)

Coefficients and 95% Confidence Bounds

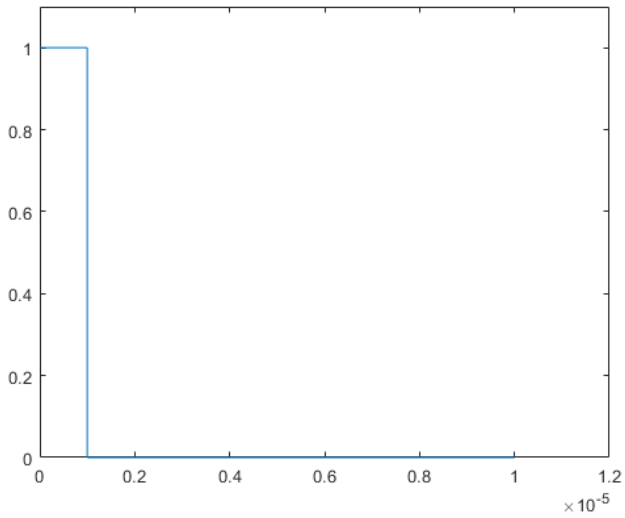
	Value	Lower	Upper
p1	1.9736	1.9560	1.9911
p2	-0.4983	-0.5107	-0.4859

می‌توان دید که نتایج مطابقت دارد.

تمرین ۳-۱) تصویر اسکریپت:

```
clc;
clearvars;
tau=1e-6;
T=1e-5;
ts=1e-9;
t=0:ts:T;
s_s=zeros(1,length(t));
s_s(1:floor(tau/ts))=1;
figure;
plot(t,s_s);
ylim([0 1.1])
```

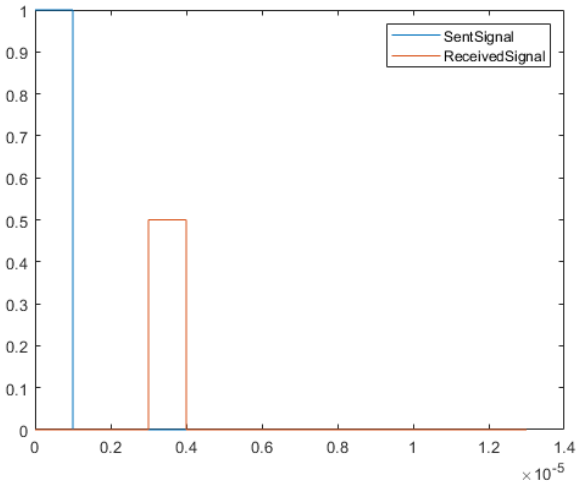
تصویر سیگنال ارسالی:



<u>تمرین ۳-۲)</u> تصویر اسکریپت:

```
tau=1e-6;
T=1e-5;
ts=1e-9;
t=0:ts:T;
s_s=zeros(1,length(t));
s_s(1:floor(tau/ts))=1;
figure;
plot(t,s_s)
hold on
alpha=0.5;
R=450;
c=3e8;
td=2*R/(c);
t=0:ts:T+td;
r_s=zeros(1,length(t));
r_s(floor(td/ts):floor((tau+td)/ts))=alpha;
plot(t,r_s)
legend('SentSignal','ReceivedSignal')
```

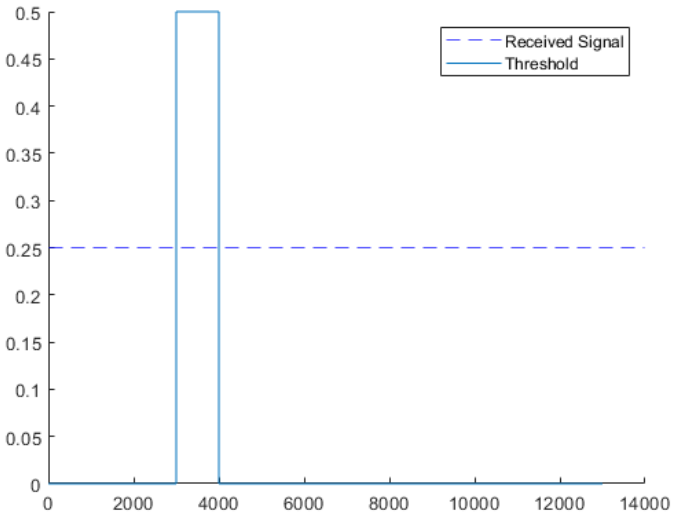
تصویر سیگنال دریافتی:



<u>تمرین ۳-۳)</u> ایده من این است که اگر مقدارسیگنال از حد مشخصی بیشتر شد که آن حد ضریبی از حداکثر مقدار سیگنال است، می‌توان حدس زد که سیگنال بازگشته و می‌توان فاصله را حساب کرد. تصویر تابع:

```
function R=distance_from_signal_using_threshold(s,ts,thresh)
    c=3e8;
    td=ts*find(s>=thresh*max(s),1);
    figure;
    yline(thresh*max(s),'--b')
    hold on
    plot(s)
    legend('Received Signal','Threshold')
    R=td*c/2;
end
```

تصویر نمودار خروجی تابع:



تصویر اسکریپت:

```
tau=1e-6;
T=1e-5;
ts=1e-9;
t=0:ts:T;
s_s=zeros(1,length(t));
s_s(1:floor(tau/ts))=1;
alpha=0.5;
R=450;
c=3e8;
td=2*R/(c);
t=0:ts:T+td;
r_s=zeros(1,length(t));
r_s(floor(td/ts):floor((tau+td)/ts))=alpha;
distance_from_signal_using_threshold(r_s,ts,0.5)
```

Command Window

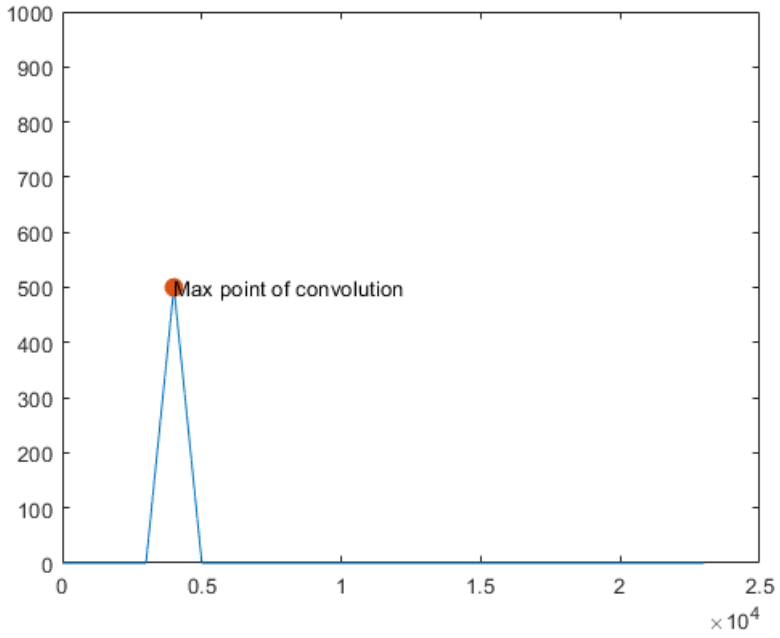
```
>> p3_3

ans =

   450
```

همچنین می‌توان با روش correlation فاصله را یافت. برای این کار از کانولوشن دو تابع استفاده کردم.

تصویر تابع:

```
function R=distance_from_signal_using_correlation(s_s,r_s,ts,tau)
    convolution=conv(s_s,r_s);
    figure;
    plot(convolution)
    hold on
    [m,i]=max(convolution);
    plot(i,m, '.', 'MarkerSize', 30)
    text(i,m,'Max point of convolution')
    ylim([0,2*m])
    td=ts*i-tau;
    c=3e8;
    R=td*c/2;
end
```

تصویر نمودارخروجی تابع:



تصویر اسکریپت:

```
tau=1e-6;
T=1e-5;
ts=1e-9;
t=0:ts:T-ts;
s_s=zeros(1,length(t));
s_s(1:int64(tau/ts))=1;
alpha=0.5;
R=450;
c=3e8;
td=2*R/(c);
t=0:ts:T+td-ts;
r_s=zeros(1,length(t));
r_s(td/ts+1:int64((tau+td)/ts)+1)=alpha;
distance_from_signal_using_threshold(r_s,ts,0.5)
distance_from_signal_using_correlation(s_s,r_s,ts,tau)
```

Command Window

```
>> p3_3

ans =

   450


ans =

  450.0000
```

<u>تمرین ۳-٤)</u> اسکریپت زیر 100 بار و هر دفعه 0.1 نویز بیشتری به سیگنال اضافه می‌کند و در نهایت نمودار ارور را رسم می‌کند.

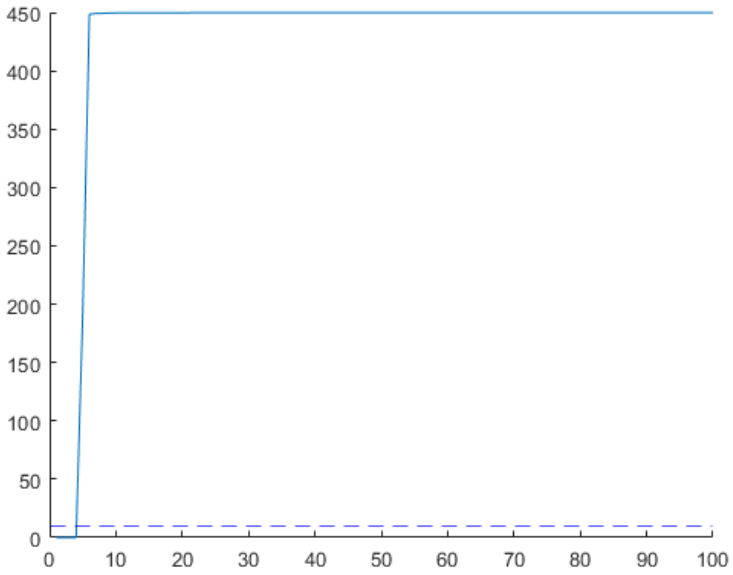
تصویر اسکریپت

```
tau=1e-6;
T=1e-5;
ts=1e-9;
alpha=0.5;
R=450;
c=3e8;
td=2*R/(c);
t=0:ts:T+td-ts;
r_s=zeros(1,length(t));
r_s(td/ts+1:int64((tau+td)/ts)+1)=alpha;

error=zeros(1,100);
for i=1:100
    for j=1:100
        s=r_s+0.1*i*rand(size(r_s));
        r=distance_from_signal_using_threshold(s,ts,0.5);
        error(i)=error(i)+abs(r-R);
    end
end
error=error./100;
figure;
yline(10,'--b')
hold on
plot(error)
wrongs=find(error>=10);
wrongs(1)
```

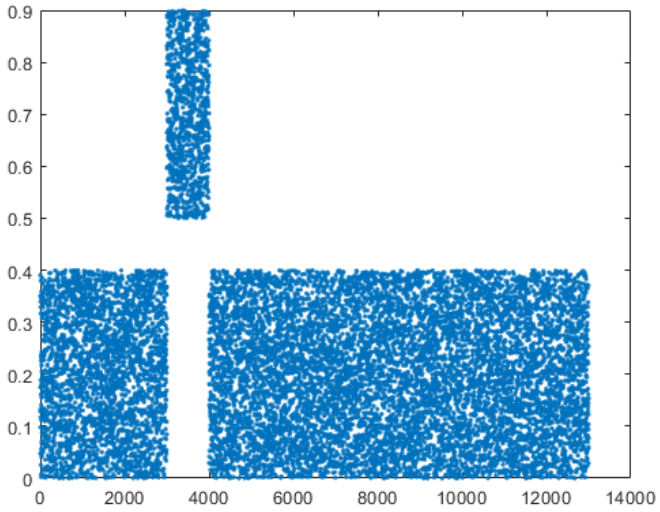
```
ommand Window
>> p3_4

ans =

     5
```

می‌توان دید روش اول تنها قادر است تا حداکثر 0.4 برابر نویز دریافت کند تا خطایش قابل چشم پوشی باشد.

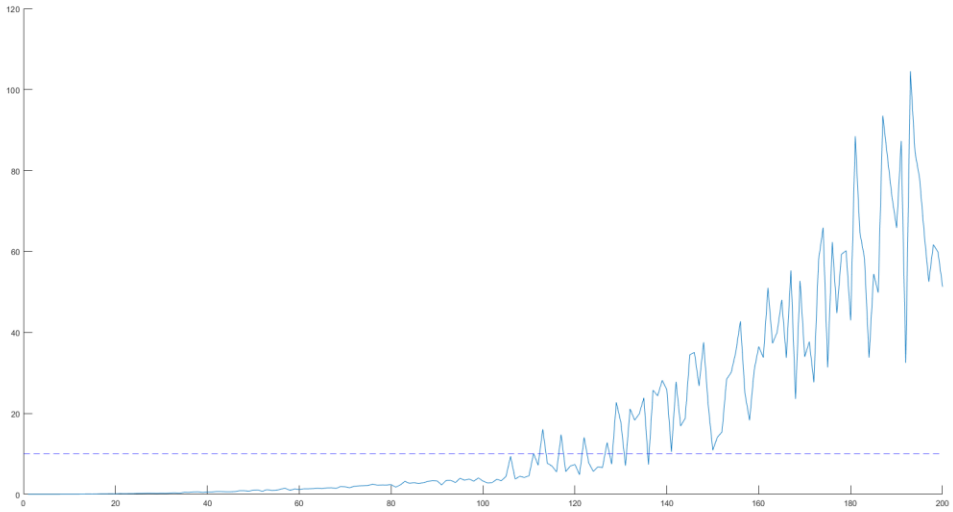
تصویر خروجی اسکریپت:



تصویر سیگنالی با 0.4 نویز:

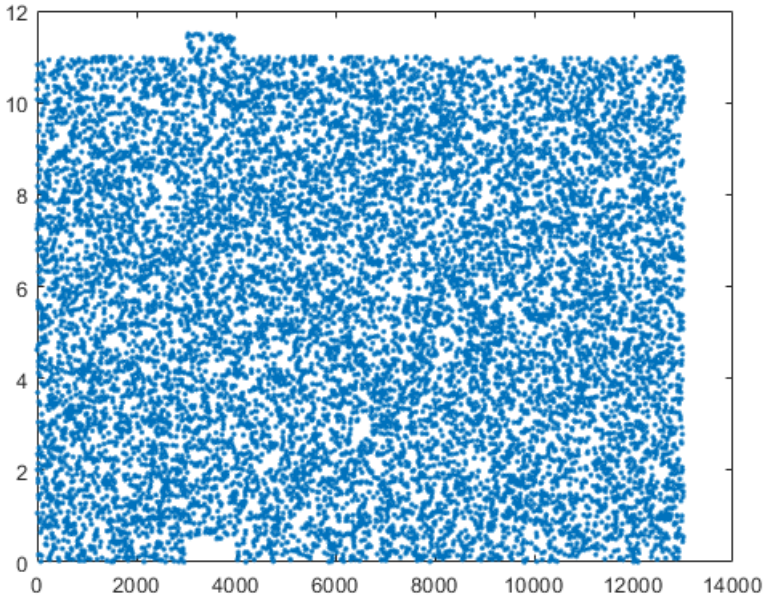


استفاده از روش دوم نمودار خطای زیر را می‌دهد.



اولین بار با نویز 11.1 برابر خطا بیش از 10 متر می‌شود که نشان می‌دهد چقدر روش دوم بهتر از روش اول در این جا جواب می‌دهد.

تصویر سیگنالی با 11.1 نویز:



تمرین ٤-١)

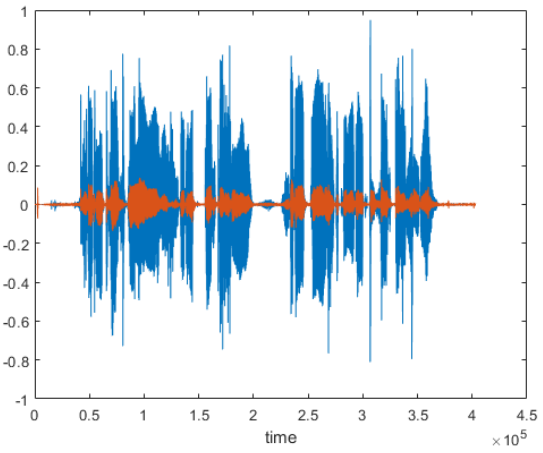
```
1    [x fs]=audioread('audio.wav');
2    fs
```

```
Command Window
>> p4_1

fs =

        48000
```

تمرین ٤-٢) تصویر اسکریپت:

```
1    [x fs]=audioread('audio.wav');
2    figure;
3    plot(x);
4    xlabel('time');
5    audiowrite('x.wav',x,fs);
```

تصویر سیگنال:



تمرین ٤-٣) تصویر تابع:

```
function p4_3(x,speed)
    if speed==2
       sound(x(1:2:end,:),48000)
    elseif speed==0.5
        y=[x(1)];
        for i=2:size(x,1)
            y=[y;(x(i-1)+x(i))/2;x(i)];
        end
        sound(y,48000)
    else
        error('Invalid value for speed');
    end
end
```

تمرین ٤-٤) تصویر تابع:

```
function p4_4(x,speed)
    sz=size(x);
    sz(1)=floor(sz(1)/speed);
    y=zeros(sz);
    if speed>1
        z=speed;
        for i=1:size(y,1)
            y(i)=x(floor(z));
            z=z+speed;
        end
    else
        speed=1/speed;
        z=speed;
        j=2;
        for i=2:size(y,1)
            if i==ceil(z)
                y(i)=x(j);
                z=z+speed;
                j=j+1;
            else
                y(i)=(x(j-1)*(i-ceil(z-speed))+x(j)*(ceil(z)-i))/(ceil(z)-ceil(z-speed));
            end
        end
    end
    sound(y,48000)
end
```