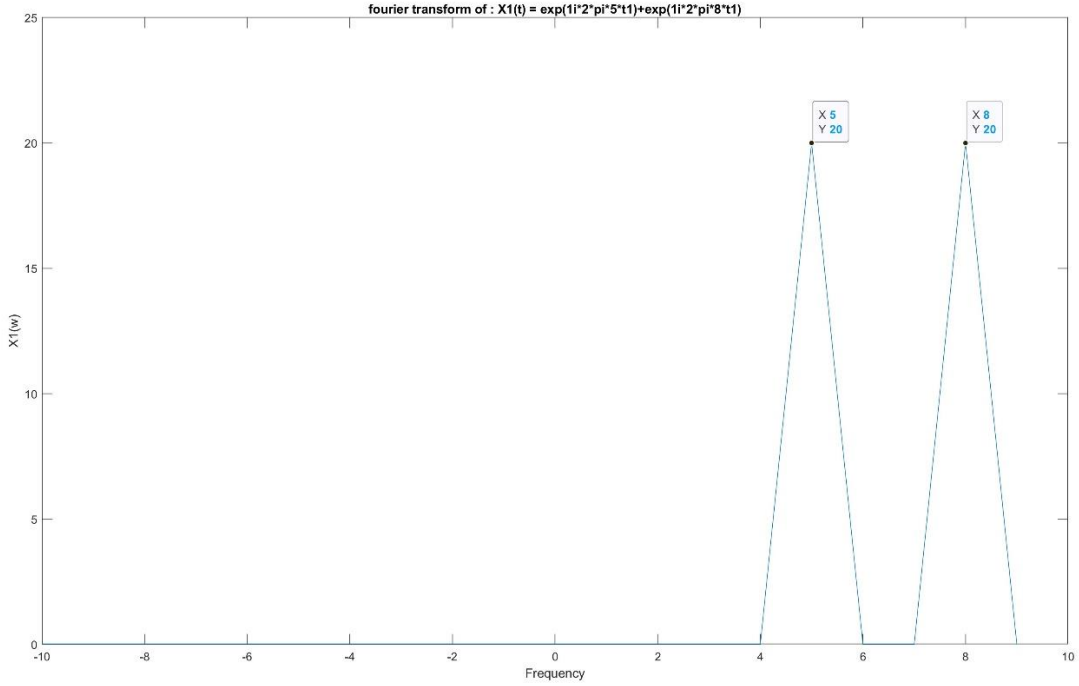
CA 5

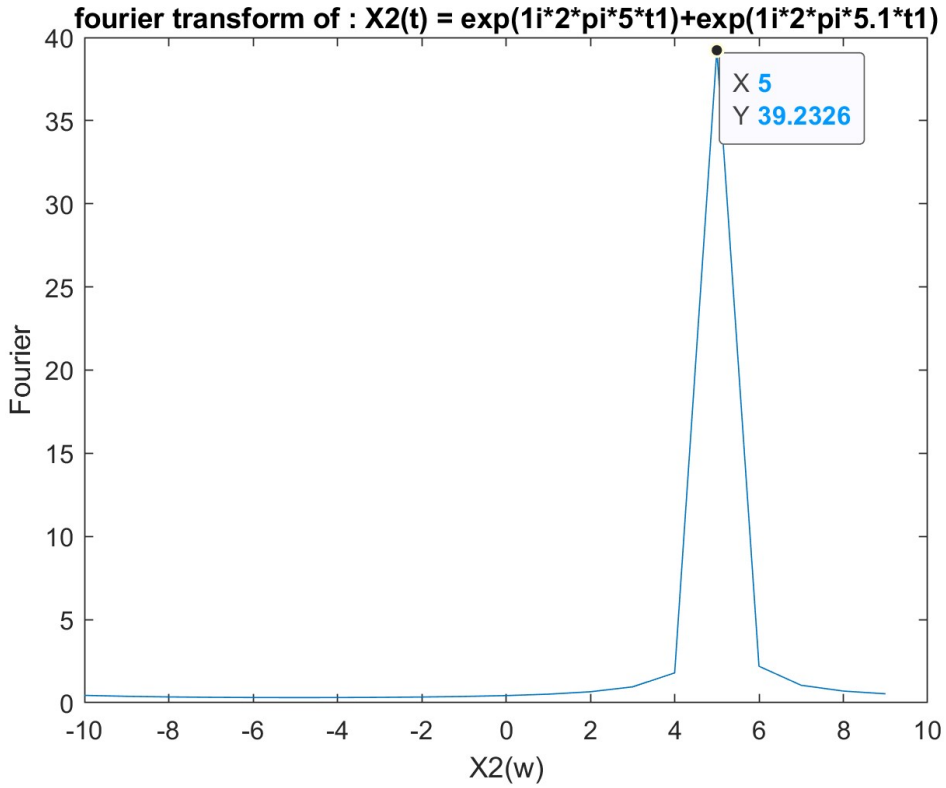
Signals and systems

Rouja Aghajani - 810101380

بخش 1-0)

```
frequency=-10:1:9;
fs = 20;
time= 0:(1/fs):0.95;

x1=exp(1i*2*pi*5*time)+exp(1i*2*pi*8*time);
y1=fftshift(fft(x1));
plot(frequency,abs(y1))
xlabel('Frequency')
ylabel('X1(w)')
title('fourier transform of : X1(t) = exp(1i*2*pi*5*t1)+exp(1i*2*pi*8*t1)')

figure()
x2=exp(1i*2*pi*5*time)+exp(1i*2*pi*5.1*time);
y2=fftshift(fft(x2));
plot(frequency,abs(y2))
xlabel('X2(w)')
ylabel('Fourier')
title('fourier transform of : X2(t) = exp(1i*2*pi*5*t1)+exp(1i*2*pi*5.1*t1)')
```



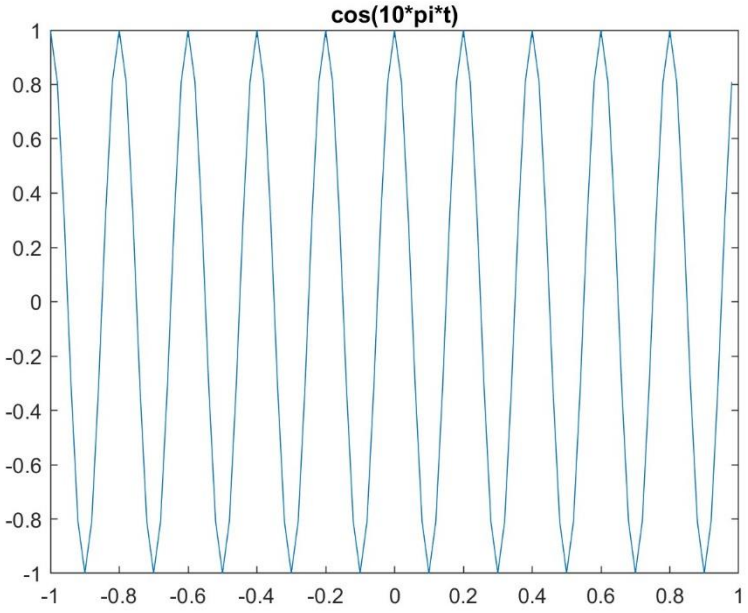


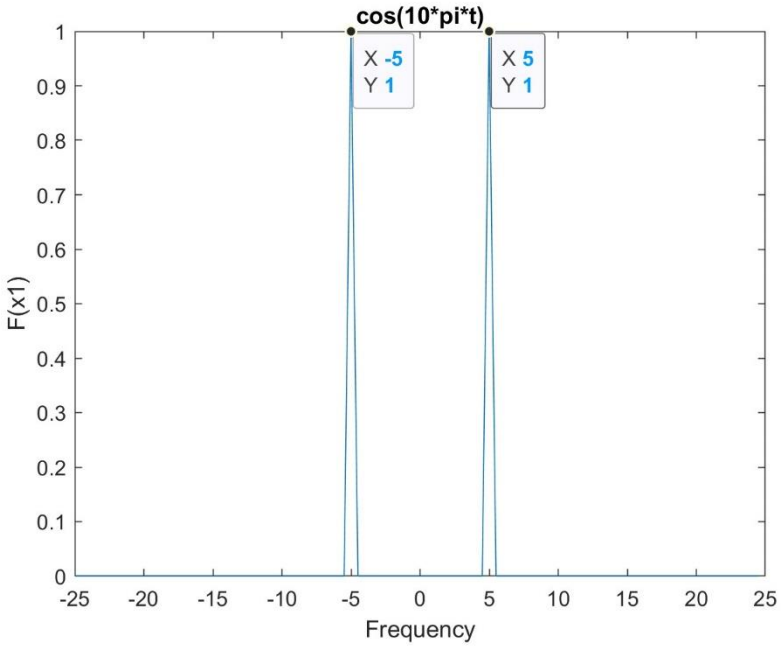
مطابق آنچه انتظار داشتیم، برای $x_1(t)$ که دارای دو سیگنال تک تن بود، دو قله دیده شد که مطابق انتظار در فرکانس های 5Hz و 8Hz می باشند.

برای $x_2(t)$ نیز چون اختلاف فرکانس دو سیگنال تک تن کمتر از 1Hz می باشد، تنها یک قله در فرکانس 5Hz دیده می شود.

تمرین 1-1)

```
fs=50;
t=-1:1/fs:0.98;
x1=cos(10*pi*t);
plot(t,x1)
title('cos(10*pi*t)')

figure()
f1=-25:1/2:24.5;
y1=fftshift(fft(x1));
y1=y1/max(abs(y1));
plot(f1,abs(y1))
xlabel('Frequency')
ylabel("F(x1)")
title("cos(10*pi*t)")
```





با نوشتن cos(10πt) در حوزه فرکانس، مشاهده می شود که تنها در فرکانس های 5 و -5 مقدار دارد که نتیجه بدست آمده نیز این موضوع را تایید می کند.

در واقع داریم:

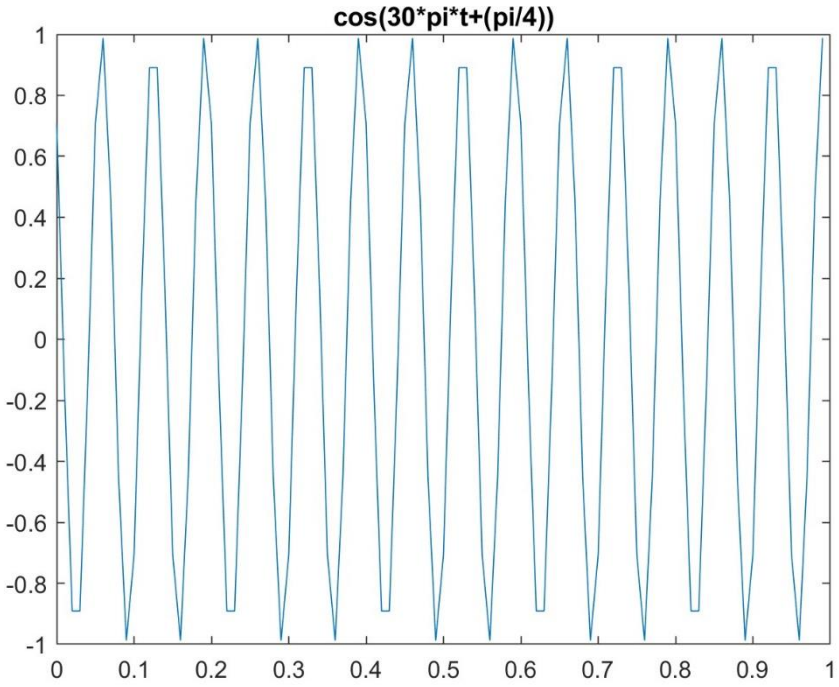
$$\cos(10\pi t) = \frac{e^{10\pi jt} + e^{-10\pi jt}}{2} \ , 2\pi f_1 = 10\pi \rightarrow f_1 = 5, 2\pi f_2 = -10 \rightarrow f_2 = -5$$

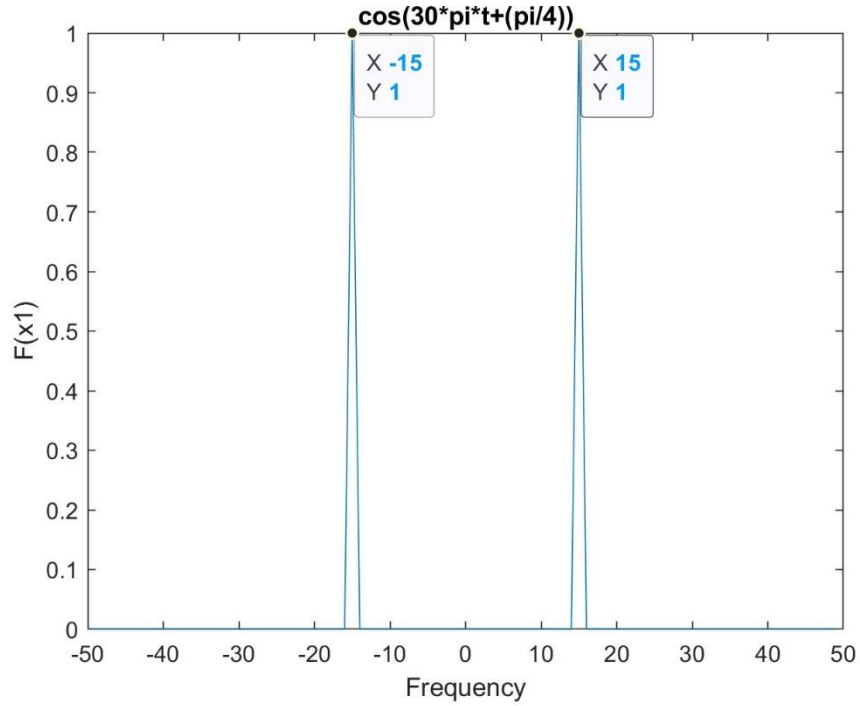
تمرین 1-2)

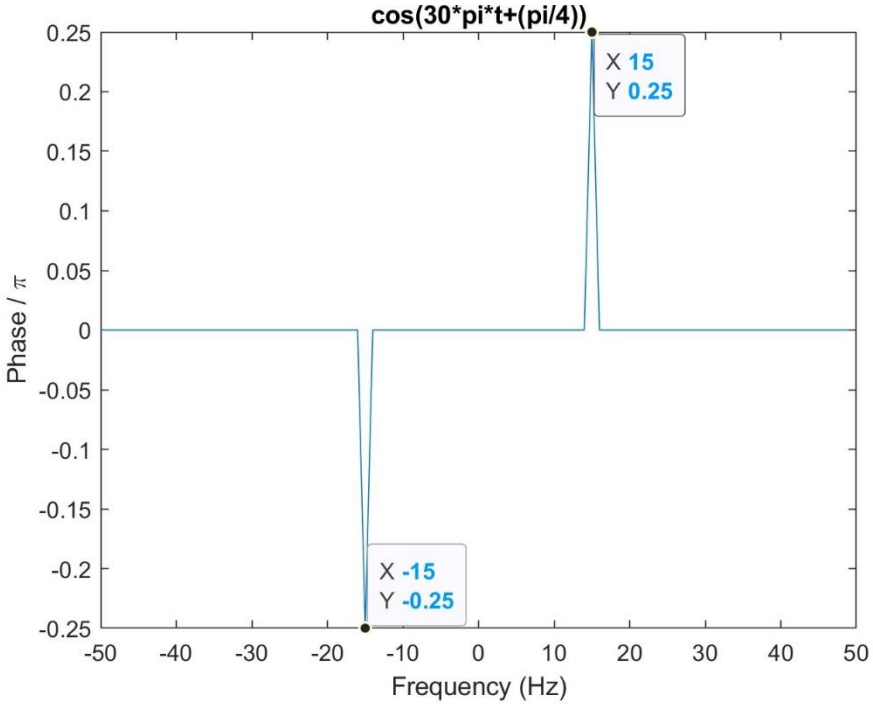
```
figure()
fs=100;
t4=0:1/fs:0.99;
x4=cos(30*pi*t4+(pi/4));
plot(t4,x4)
title('cos(30*pi*t+(pi/4))')

figure()
y1=fftshift(fft(x4));
f1=-50:1:49;
plot(f1,abs(y1)/max(abs(y1)))
xlabel('Frequency')
ylabel('F(x1)')
title("cos(30*pi*t+(pi/4))")

figure()
tol = 1e-6;
y1(abs(y1) < tol) = 0;
theta = angle(y1);
plot(f1,theta/pi)
xlabel 'Frequency (Hz)'
ylabel 'Phase / \pi'
title("cos(30*pi*t+(pi/4))")
```

نتایج:

cos(30πt) در حوزه فرکانس در فرکانس های 15 و 15- مقدار دارد که در شکل نیز همین موضوع نمایش داده شده است.




cos(30*pi*t+(pi/4))
X -15
Y 1
X 15
Y 1
F(x1)
Frequency


cos(30*pi*t+(pi/4))
X 15
Y 0.25
X -15
Y -0.25
Phase / π
Frequency (Hz)

بخش دوم:

تمرین 2-1)

```
function Mapset = mapsetBuilder()
    Mapset=cell(2,32);
    alphabet = 'abcdefghijklmnopqrstuvwxyz .,!";';
    for i = 1:32
        Mapset{1,i} = alphabet(i);
        Mapset{2,i} = dec2bin(i-1, 5);
    end
end
```

همانطور که در پروژه های گذشته داشتیم، یک سلول تعریف میکنیم که در ردیف اول کارکترهای مدنظر و در ردیف دوم مقدار باینری متناظر با آن ها را با عدد باینری پنج بیت مینویسیم.

تمرین 2-2)

در اینجا ابتدا پیام ورودی را با کمک مپ ست موجود به باینری تبدیل میکنیم. سپس با توجه به بیت ریت مد نظر این پیام را بخش بخش میکنیم، مثلا برای بیت ریت 1 هر رقم را جدا میکنیم. سپس به کمک بیت ریت موجود میبینیم چه حالاتی از این بلاک ها ممکن است، مثلا برای بیت ریت 2 از 0 تا 3 داریم که آن را در ماتریس x تعریف میکنیم. با کمک توضیحات دستورکار، فرکانس هایی با بیشترین فاصله از رنج 1 تا 49 برای ارسال پیام انتخاب کرده و در نهایت با مقایسه مقدار باینری پیام با x، مقدار فرکانس متناظر را ذخیره میکنیم. در آخر نیز با کمک تابع سینوس و این فرکانس ها سیگنال خروجی حاصل میگردد.

```
function encoded_message=coding_freq(message,bit_rate)
    Mapset=mapsetBuilder();
    fs=100;
    n=(strlength(message)*5)/bit_rate;
    binaryMessage = [];
    for char = message
        index = find([Mapset{1,:}] == char, 1);
        if ~isempty(index)
            binaryMessage = [binaryMessage Mapset{2, index}];
        else
            error(['Character ', char, ' does not exist in the dataset.']);
        end
    end

    stringLength = length(binaryMessage);
    loopCntr = 1;
    for k = 1 : bit_rate : stringLength
        index1 = k;
        index2 = min(k + bit_rate - 1, stringLength);
        out{loopCntr} = binaryMessage(index1 : index2);
        loopCntr = loopCntr + 1;
    end

    binaryMessage = out ;

    x = cell(1, 2^bit_rate);
    for i=0:2^bit_rate-1
        x{i+1} = dec2bin(i,bit_rate);
    end
    num_frequencies = 2^bit_rate;
    y = zeros(1, num_frequencies);
    step_size = 50 / (num_frequencies + 1);
    for i = 0:num_frequencies-1
        y(i+1) = round((i+1) * step_size);
    end
    y(y > 49) = 49;

    ts=1/fs;
    t=0:ts:1-ts;
    freq=[];
    for i=1:n
        for j=1:2^bit_rate
            if strcmp(binaryMessage(1,i),x(1,j))==1
                freq=[freq y(1,j)];
            end
        end
    end

    encoded_message=zeros(n,100);
    for i=1:n
        encoded_message(i,:)=sin(2*pi*freq(1,i)*t);
    end

    figure()
    t1=zeros(n,100);
    for i=1:n
        t1(i,:)=linspace(i-1,i,fs);
    end
```

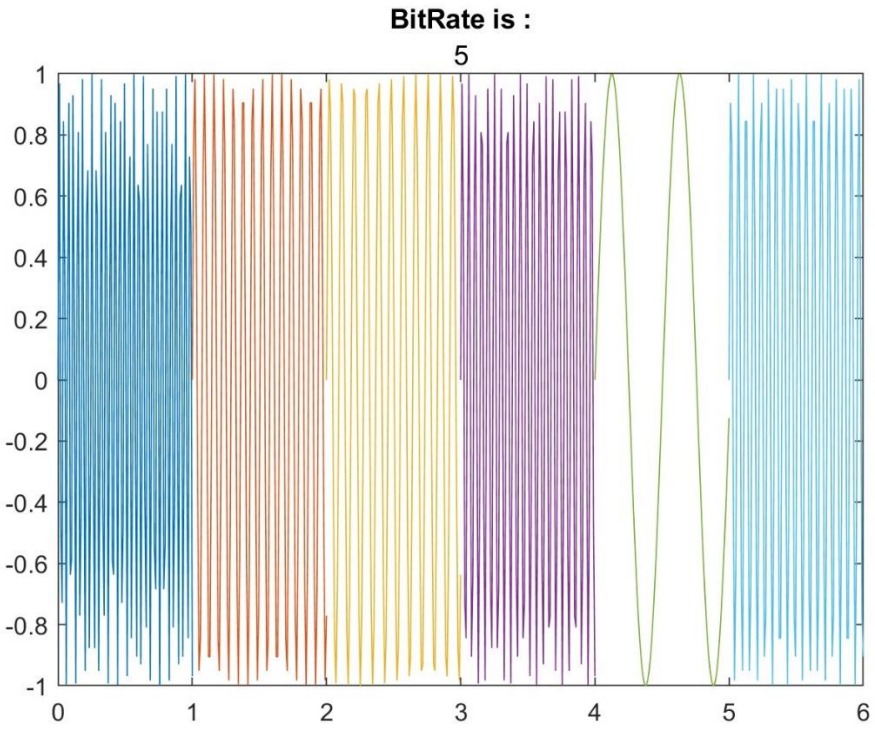
و برای رسم:

```
    for k=1:n
        plot(t1(k,:),encoded_message(k,:));
        title('BitRate is :',bit_rate)
        hold on
    end
end
```

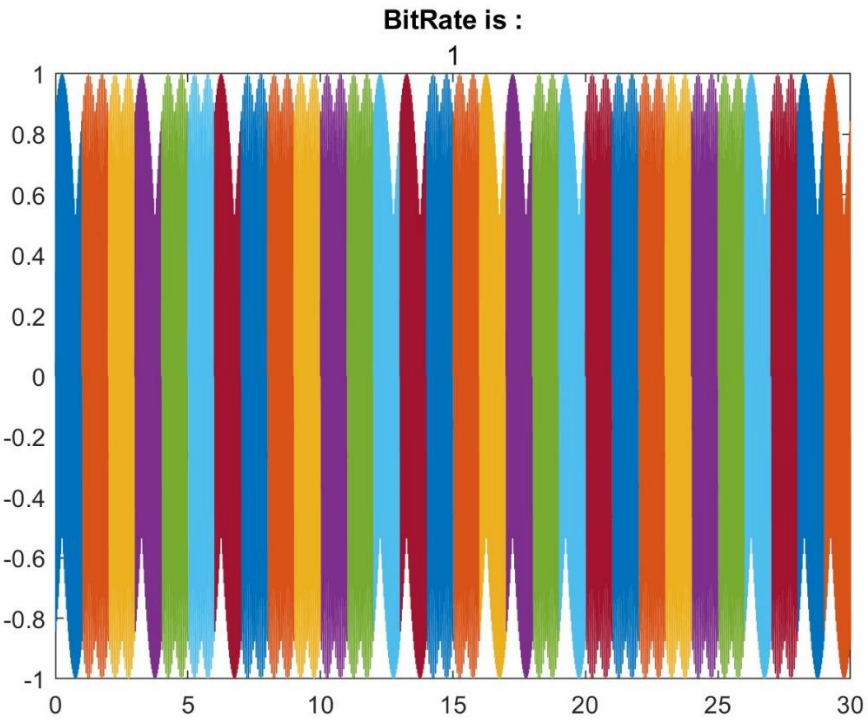
تمرین 2-3)

```
for counter=0.0:0.01:2
    test1 = coding_freq('signal' , 1);
    test1wNoise = noiseAdder(test1,counter);
    result1 = decoding_freq(test1wNoise,1);
  %  test2 = coding_freq('signal' , 2);
  %  test2wNoise = noiseAdder(test2,counter);
  %  result2 = decoding_freq(test2wNoise,2);
    test3 = coding_freq('signal' , 5);
    test3wNoise = noiseAdder(test3,counter);
    result3 = decoding_freq(test3wNoise,5);
    disp([" Sigma = " , counter ]);
    disp(["bit rate = 1, docoded: " , result1]);
   % disp(["bit rate = 2, docoded: " , result2]);
    disp(["bit rate = 3, docoded: " , result3]);
end
```

رسم برای بیت ریت 5:



و برای بیت ریت 1:



در واقع شکل حالت سینوسی است که با تغییرات زیاد است.

تمرین 2-4)

در اینجا نیز همان سلول x با مقادیر ممکن تشکیل می شود و در y نیز فرکانس های مربوطه را داریم. پیام را به حوزه فوریه میبریم و قله های آن را ذخیرع میکنیم و حداکثرشان را میابیم. حال با استفاده تابع mean بررسی میکنیم این قله بین کدام دو عدد است و عدد نزدیکتر را پیدا میکنیم. در نهایت با استفاده از مقادیر بدست امده در مپ ست جست و جو کرده وپیغام نهایی را بدست می آوریم.

```
function decodedMsg=decoding_freq(encodedMessage,bit_rate)
    x = cell(1,(2^bit_rate));
    for i=0:2^bit_rate-1
        x{i+1} = dec2bin(i,bit_rate);
    end

    Mapset=mapsetBuilder();
    num=size(encodedMessage);
    fs=100;

    num_frequencies = 2^bit_rate;
    y = zeros(1, num_frequencies);
    step_size = 50 / (num_frequencies + 1);
    for i = 0:num_frequencies-1
        y(i+1) = round((i+1) * step_size);
    end
    y(y > 49) = 49;

    freq=-fs/2:1:fs/2;
    I=[];
    for k=1:num(1)
        emF=abs(fftshift(fft(encodedMessage(k,:))));
        [~,loc]=max(emF);
        fhat=abs(freq(loc));
        I=[I fhat];
    end

    meanY=zeros(1,2^bit_rate);
    for l=1:2^bit_rate-1
        meanY(1,l+1)=(y(1,l+1)-y(1,l))/2+y(1,l);
    end

    indexFinder=[];
    for i=1:num(1)
        for h=1:2^bit_rate-1
            if (I(1,i)>= meanY(1,h) && I(1,i)< meanY(1,h+1))|| I(1,i)==meanY(1,h)
                indexFinder=[indexFinder x(1,h)];
            end
        end
        if I(1,i)>meanY(1,2^bit_rate)
            indexFinder=[indexFinder x(1,2^bit_rate)];
        end
    end

    indexFinder=cell2mat(indexFinder);

    charSize = 5;
    stringLength = length(indexFinder);
    loopCntr = 1;
    for k = 1 : charSize : stringLength
        index1 = k;
        index2 = min(k + charSize - 1, stringLength);
        out{loopCntr} = indexFinder(index1 : index2);
        loopCntr = loopCntr + 1;
    end

    indexFinder = out ;

```

```
58
59          decodedMsg=[];
60          Mapset_len=size(Mapset);
61          for i=1:num(1)*bit_rate/5
62              for j=1:Mapset_len(1,2)
63                  if strcmp(indexFinder(1,i),Mapset(2,j))==1
64                      decodedMsg=[decodedMsg  Mapset(1,j)];
65                  end
66              end
67          end
68
69          decodedMsg=strjoin(decodedMsg,'');
70          disp(decodedMsg)
71      end
```

و خروجی:

```
>> P2
signal
signal
    " Sigma = "     "0"

    "bit rate = 1, docoded: "     "signal"

    "bit rate = 3, docoded: "     "signal"
```

بخش 2-5)

برای افزودن نویز:

```
1   function signalwNoise=noiseAdder(CodedMsg , sigma)
2       num=size(CodedMsg);
3       fs=100;
4       encodedSignalLen=size(CodedMsg);
5
6       noise=sigma*randn(encodedSignalLen(1),encodedSignalLen(2));
7
8       signalwNoise=CodedMsg+noise;
9       t=zeros(num(1),fs);
10
11
12      for i=1:num(1)
13          t(i,:)=linspace(i-1,i,fs);
14      end
15    %  figure()
16    %  for k=1:num(1)
17    %      plot(t(k,:),signalwNoise(k,:),'b');
18    %      hold on
19    %  end
20  end
```

خروجی:

```
    " Sigma = "     "0.0001"

    "bit rate = 1, decoded: "     "signal"

    "bit rate = 5, decoded: "     "signal"
```

که مشاهده می شود پیام ورودی به درستی دیکود شد.

بخش 2-6)

```
" Sigma = "    "1.22"

"bit rate = 1, decoded: "    "sig!al"

"bit rate = 5, decoded: "    "signal"
```

```
" Sigma = "    "1.4"

"bit rate = 1, decoded: "    "sjgnal"

"bit rate = 5, decoded: "    "ssgnal"
```

بیت ریت 5 به نویز مقاومتر بود، که دلیل آن استفاده از تعداد بیشتری فرکانس برای کدگذاری بود.

بخش 2-7)

برای بیت ریت 1 حداکثر سیگمای قابل تحمل 1.22 و برای بیت ریت 5 حداکثر آن 1.4 بود.

بخش 2-8)

همانطور که در دستور کار آورده شد، به پهنای باند وابسته است. با پهنای باند بیشتر، میتوانیم فاصله فرکانس های کدگذاری را افزایش دهیم و همچنین از فرکانس های بیشتری برای کدگذاری استفاده کنیم. نتیجه آن است که میتوانیم پیام ارسالی را با سرعت بیت ریت بیشتر و مقاومتر با نویز ارسال کنیم.

بخش 2-9)

تا حدی بله، و بیشتر از آن خیر،.

من در اینجا از تعداد فرکانس بیشتری جهت تولید سیگنال با بیت ریت 5 استفاده کردم، و نتایج نسبت به نویز مقاومت بیشتری از خود نشان دادند. ولی این افزایش مقاومت تا همیشه برقرار نیست.

در واقع چون با افزایش بیت ریت بدون افزایش پهنای باند (بعد از یک حد معین) خواهیم دید که چون فرکانس های انتخابی زیاد و با فاصله کم هستند، در واقع فاصله threshold تصمیم گیری کاهش خواهد یافت و در نتیجه احتمال دیدن اثر نویز ضعیفتر بیشتر خواهد شد؛ یعنی مقاومت به نویز کم خواهد شد.