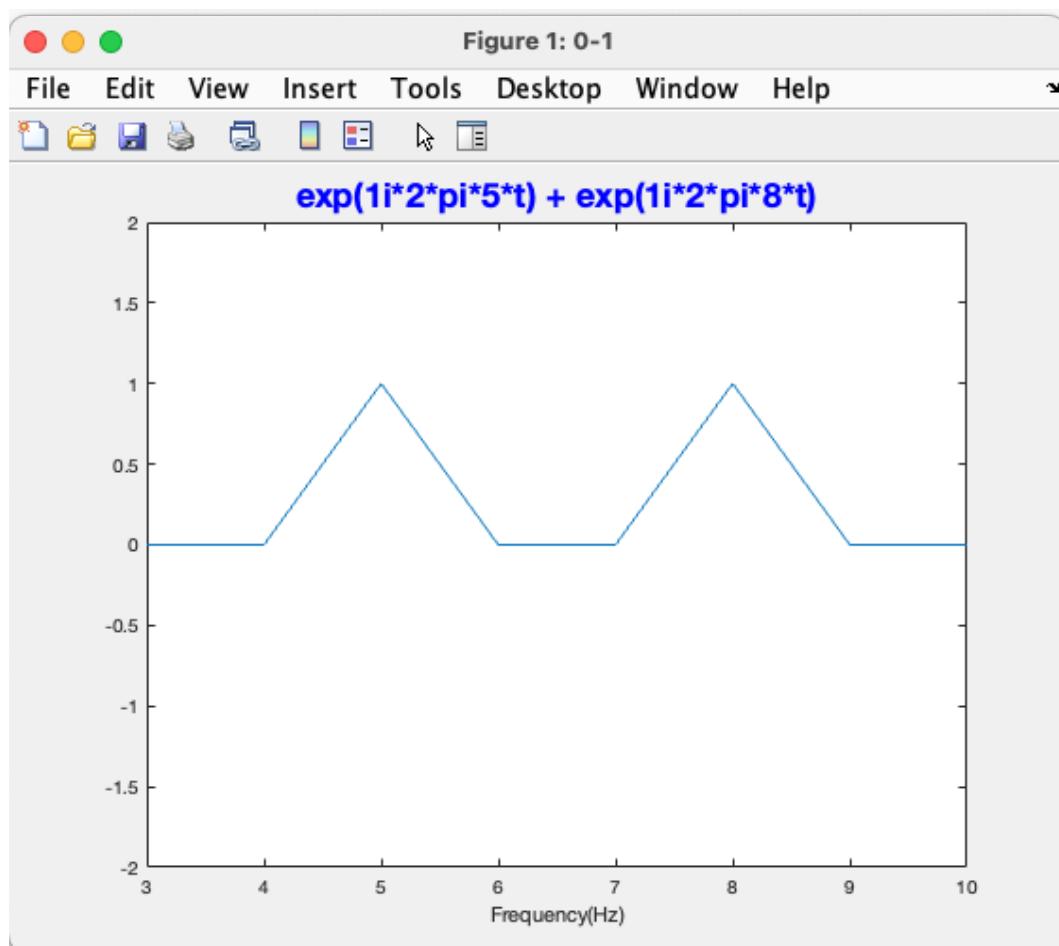


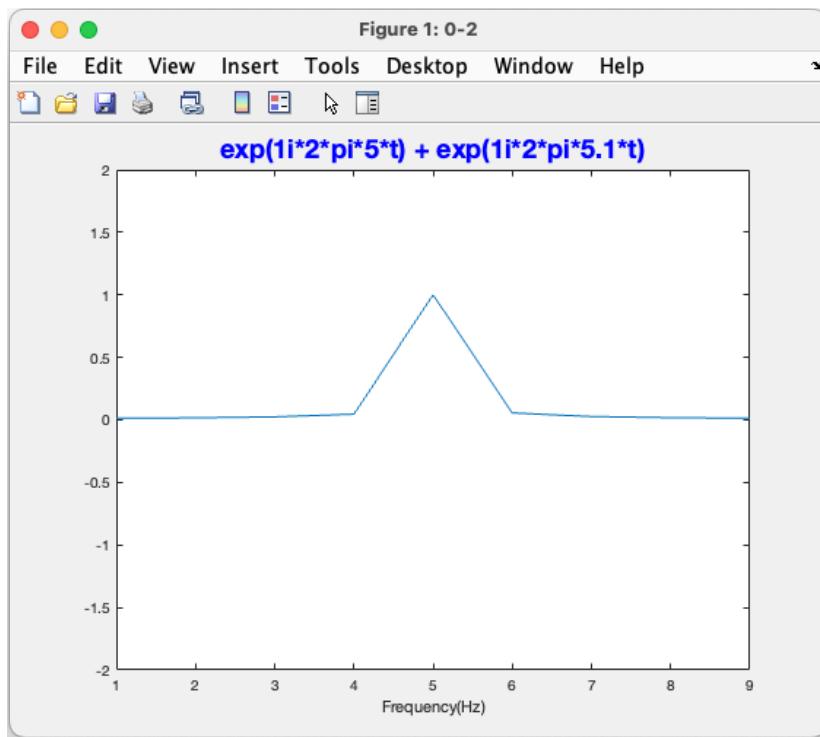


## تمرین

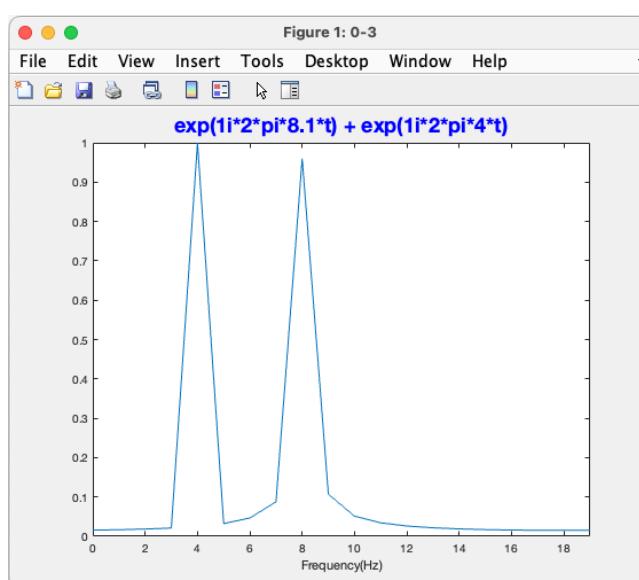
تمرین‌های گفته شده در ویدیو توضیح پروژه:



سیگنال ورودی مون حاصل دو تا سیگنال ویژه‌س، به همین حاطر در فرکانس‌های ۵، ۸ ضربه می‌بینیم.



باز جمع دو سیگنال ویژه رو داریم، یکی با فرکانس ۵، یکی با فرکانس ۰.۱ به خاطر سیگنال فرکانس ۵ یه پیک در ۵ هرتز می بینیم.  
و به خاطر سیگنال فرکانس ۰.۱ لازم است فرکانس ۵ هرتز بیشترین مشارکت را داشته باشد و هر چقدر از ۵ دور می شویم مشارکت سیگنال ها برای ساختن سیگنال ۰.۱ هرتز کم می شود.



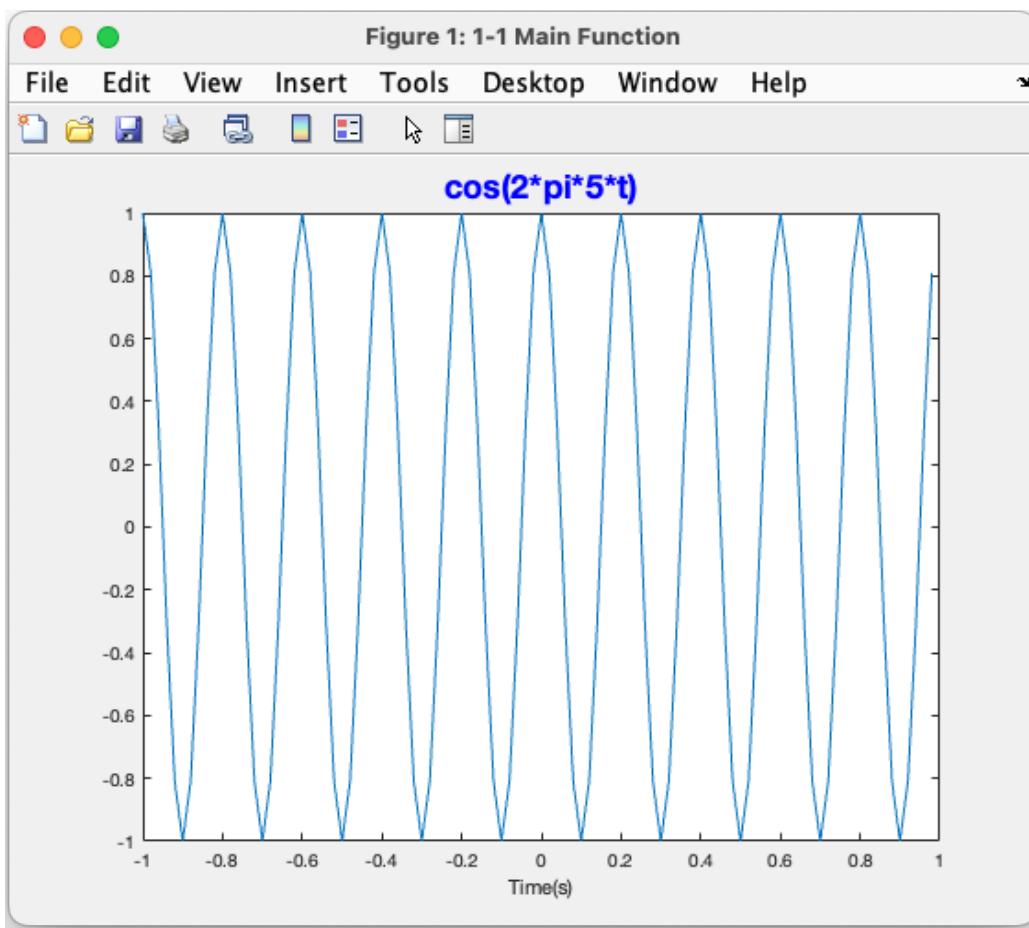
جمع دو سیگنال ویژه رو داریم، یکی با فرکانس ۴، یکی با فرکانس ۸.۱

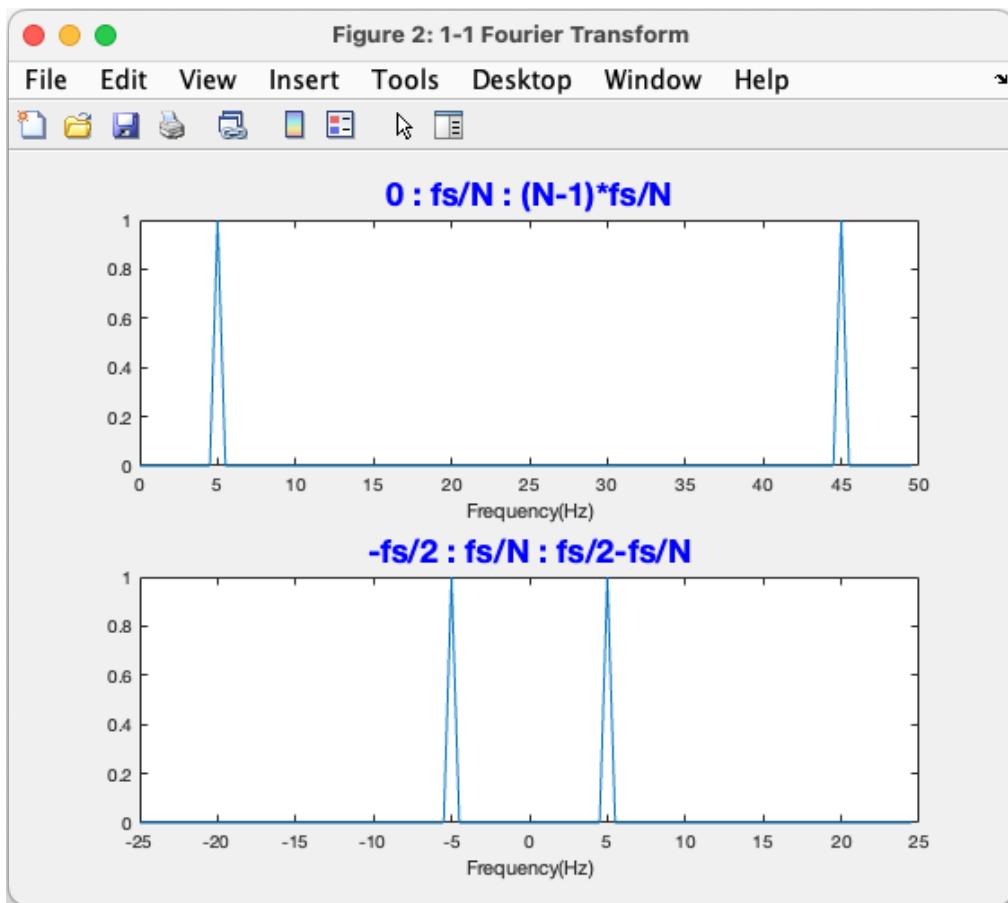
به خاطر سیگنال فرکانس ۴ یه پیک در ۴ هرتز می بینیم.

و به خاطر سیگنال فرکانس ۸.۰۱ لازم است فرکانس ۸ هرتز بیشترین مشارکت را داشته باشد و هر چقدر از ۸ دور می شویم مشارکت سیگنال ها برای ساختن سیگنال ۸.۰۱ هرتز کم می شود.

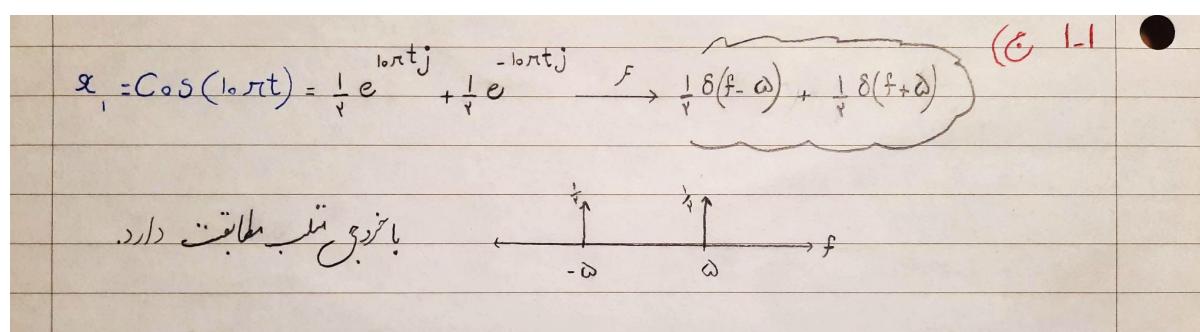
## تمرین ۱

### تمرین ۱-۱)

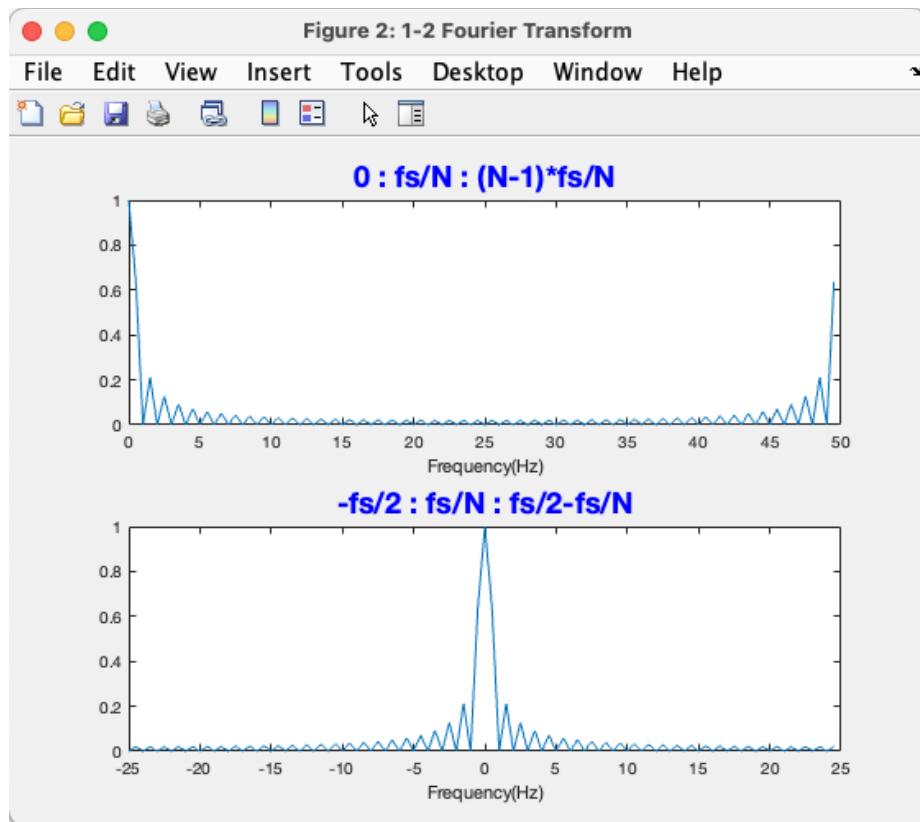
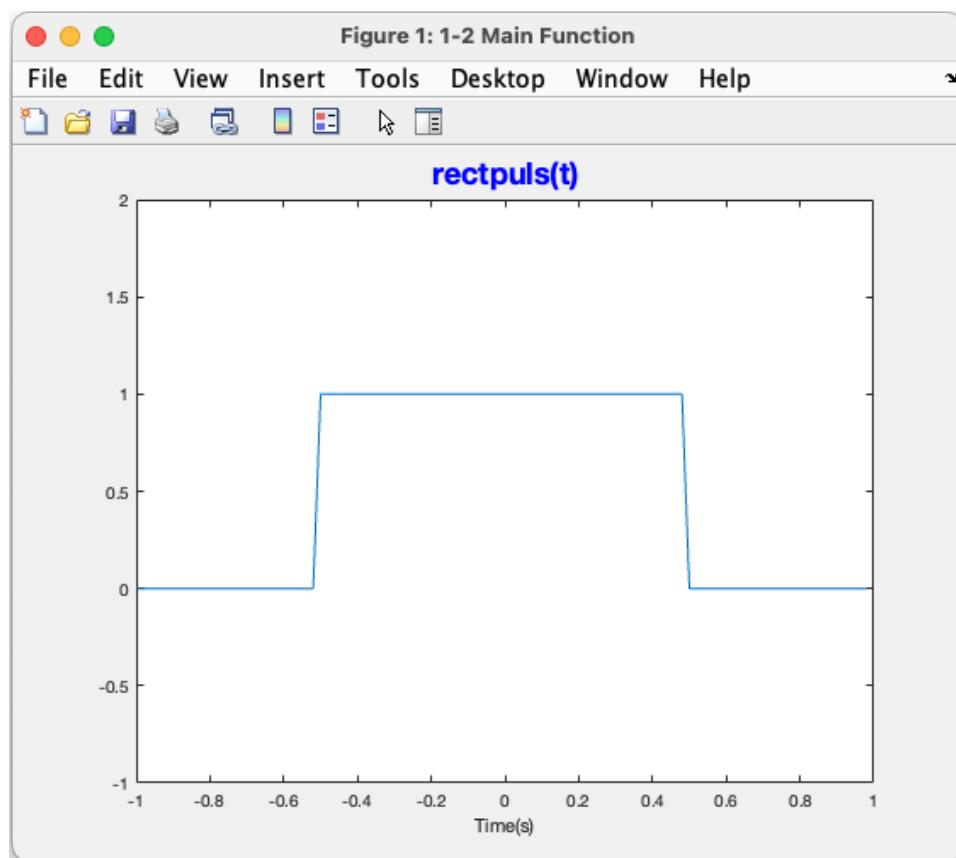




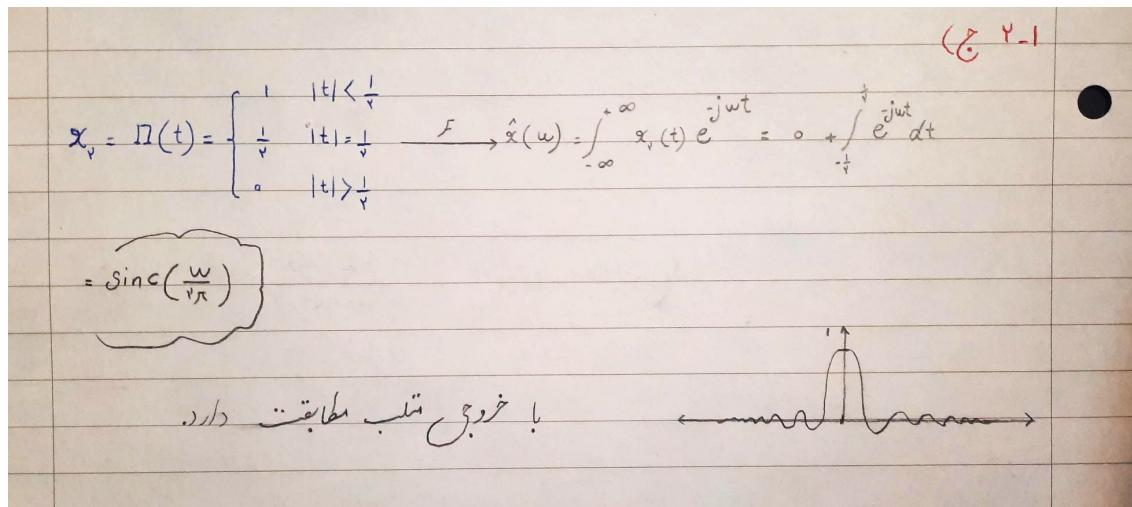
می توان  $\cos$  را بر حسب دو سیگنال ویژه نوشت.  $\cos$  با فرکانس ۵ به ما دو سیگنال ویژه با فرکانس های ۵ و -۵ می دهد. در نتیجه در تبدیل فوریه دو پیک در ۵، -۵ می بینیم.



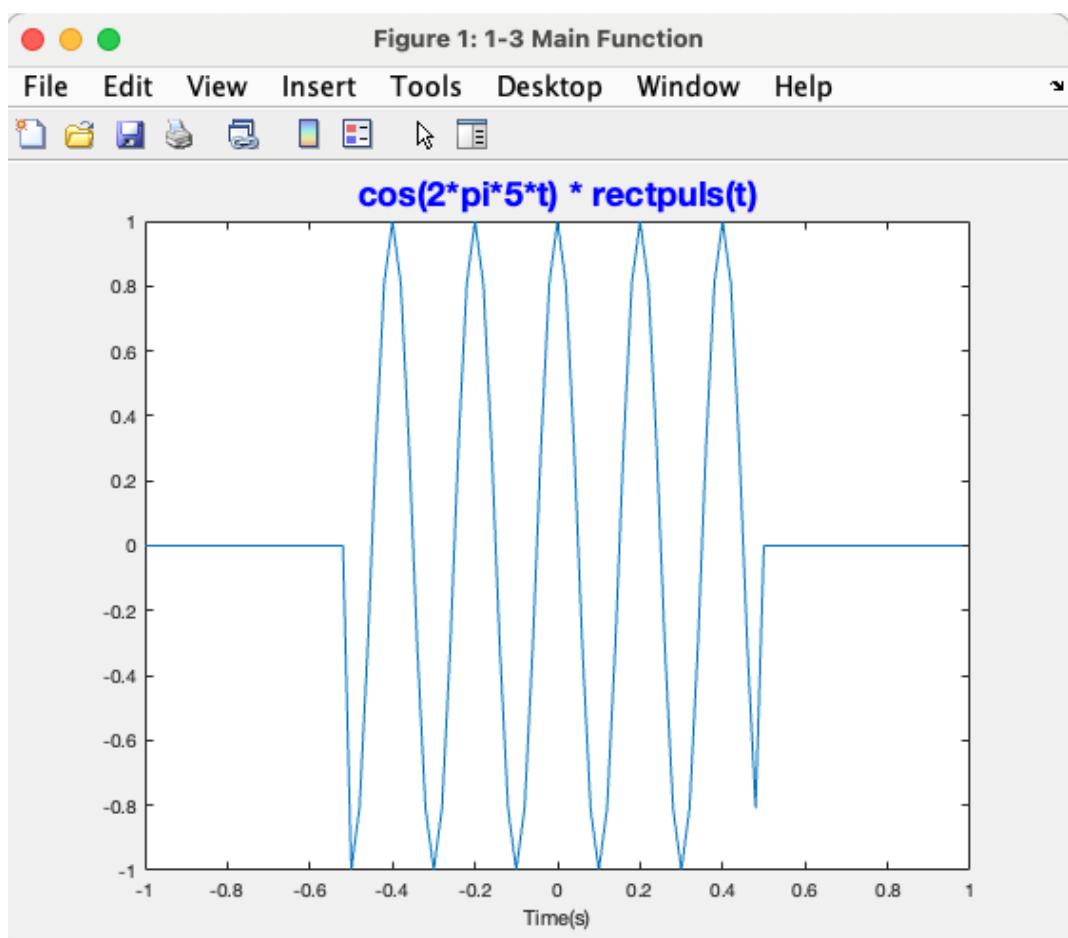
## تمرين ٢-١

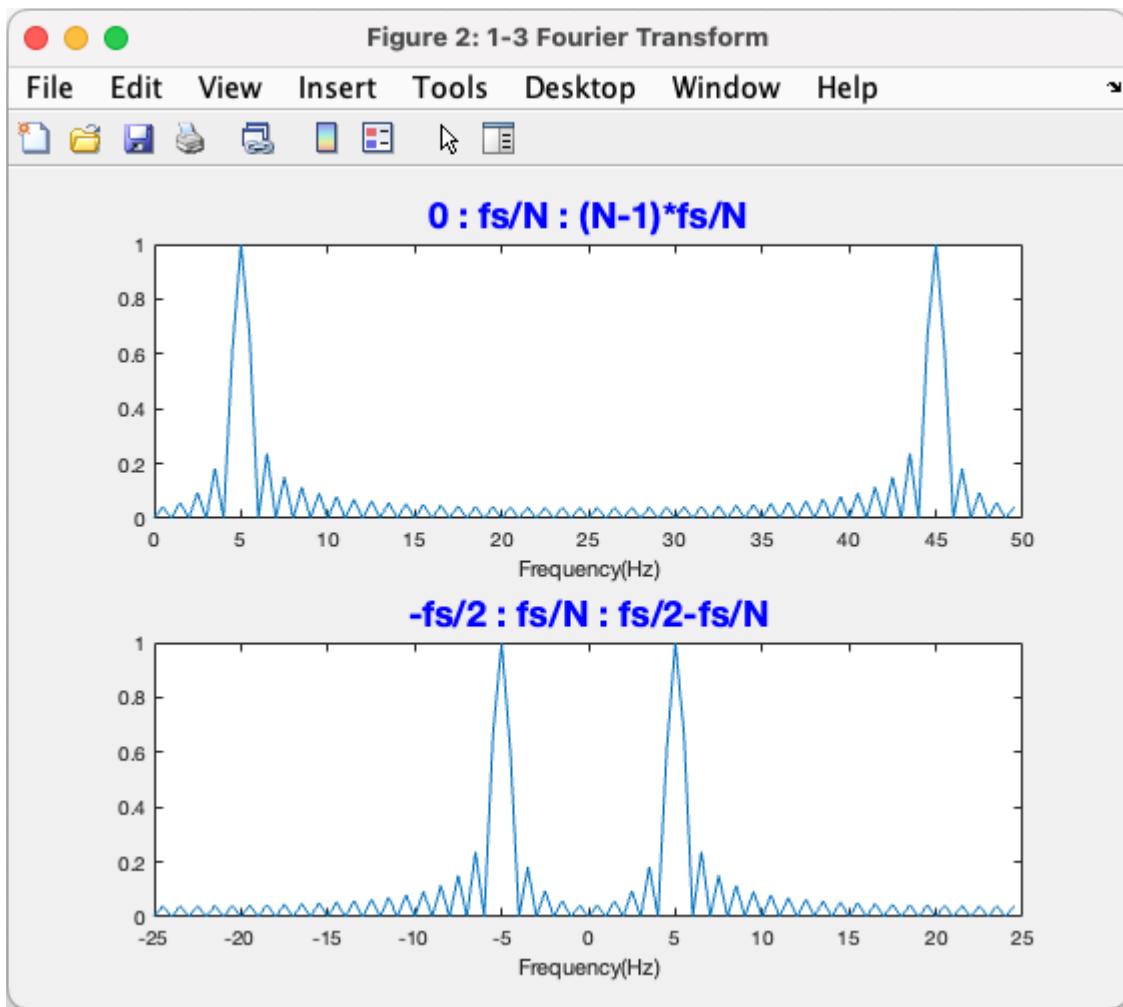


به خاطر بخش ثابت ۱ در بازه  $[0.5, 0.5]$  توقع داریم که در فرکانس ۰، یه پیک بینیم و به خاطر دو تغییرات ۰، به ۱ و ۱ به ۰، فرکانس هایی حول ۰ می بینیم که با دور شدن از نقطه ۰، به ۰ میل می کنند.



### تمرین ۳-۱





سیگنال ورودی مون حاصل ضرب دو پالس و کسینوسی است به همین خاطر در تبدیل فوریه، کانولوشن تبدیل فوریه این دو سیگنال را مشاهده می کنیم.

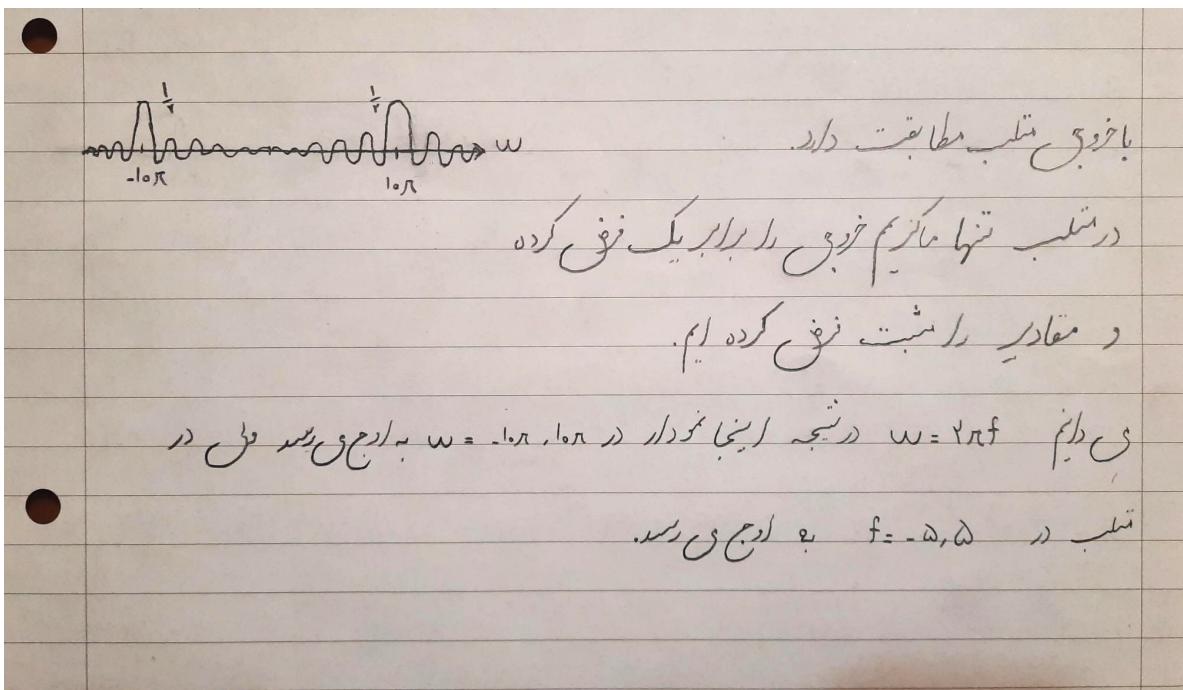
(C ۳-۱)

$$\hat{x}_p(t) = x_1(t)x_2(t) = \cos(10\pi t) \Pi(t) \xrightarrow{\mathcal{F}} \hat{x}_p(w) = \int_{-\infty}^{+\infty} x_p(t) e^{-j\omega t} dt \Rightarrow$$

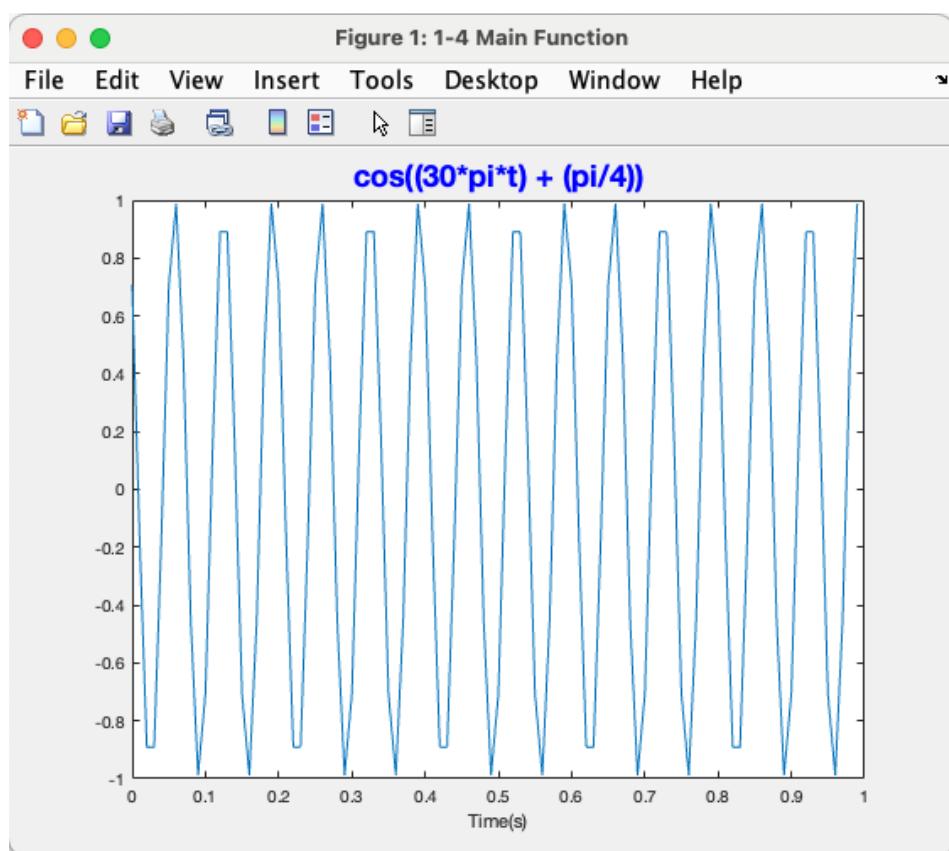
$$\hat{x}_p(w) = 0 + \int_{-\frac{1}{2}}^{\frac{1}{2}} \cos(10\pi t) \times 1 \times e^{-j\omega t} dt + 0 \quad \begin{matrix} \text{تابع زوج} \\ \text{وزیر} \end{matrix} \quad \int_{-\frac{1}{2}}^{\frac{1}{2}} \cos(10\pi t) \cos(\omega t) dt \quad \begin{matrix} \text{مجموع} \\ \text{کسر} \end{matrix} \Rightarrow \cos$$

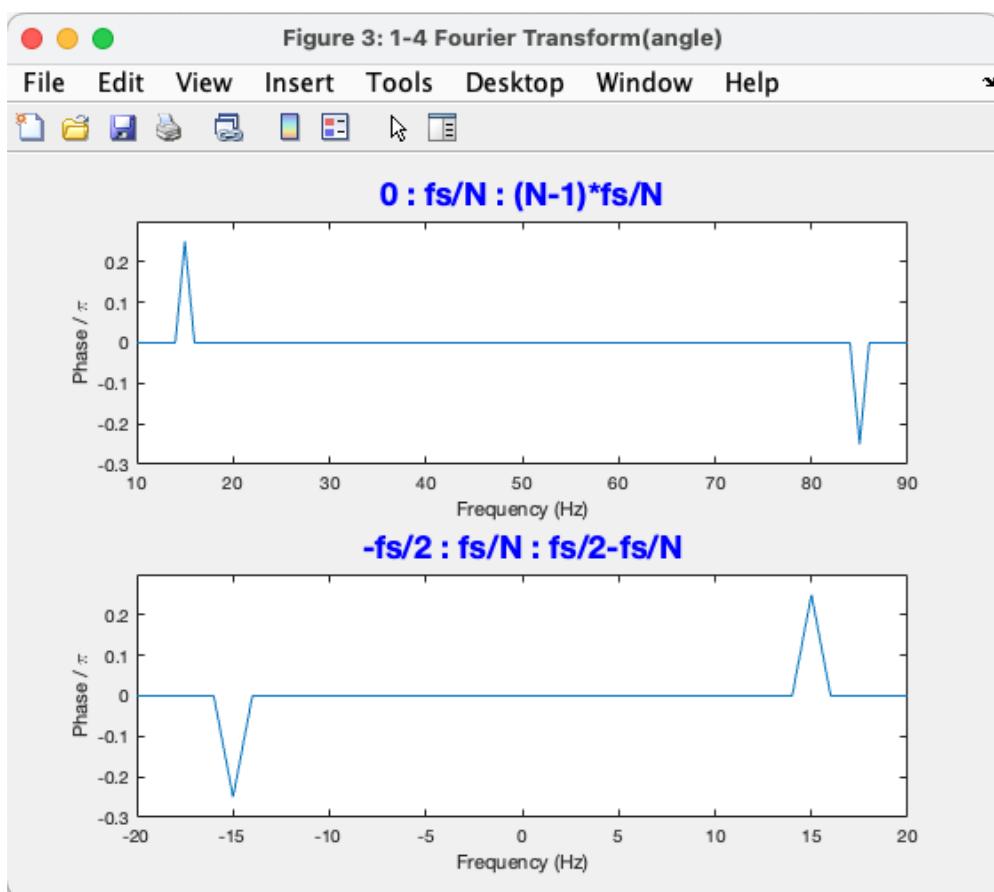
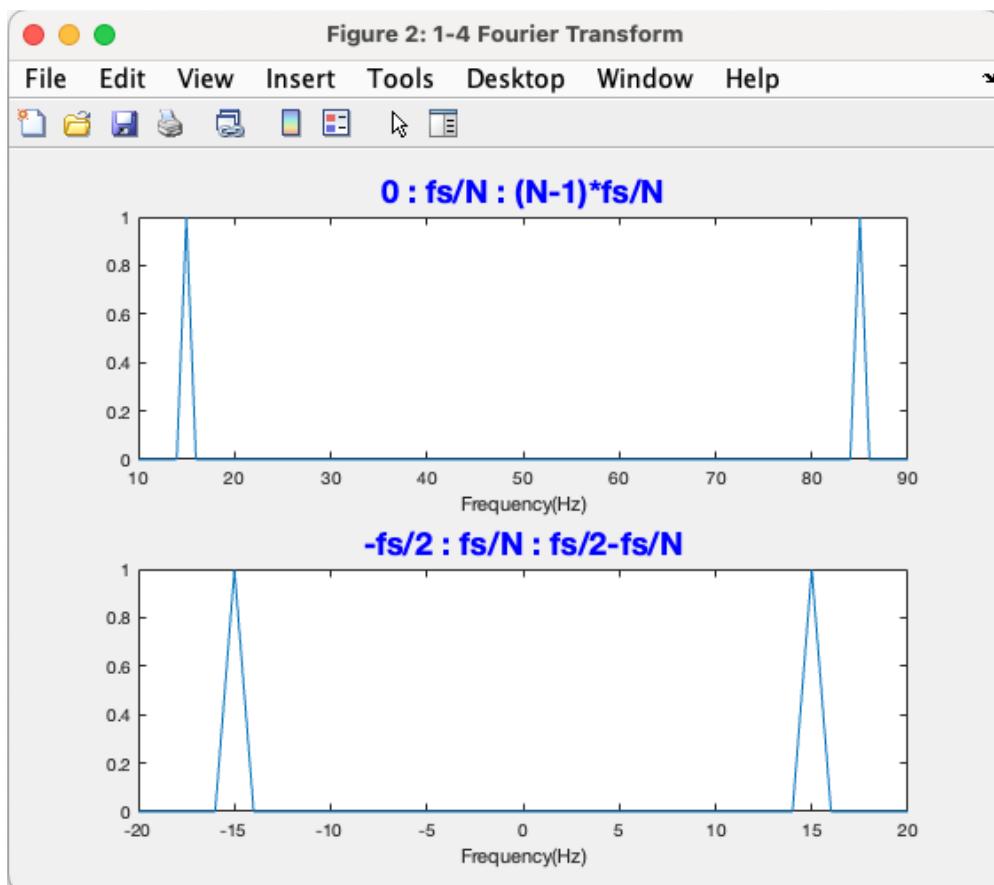
$$\hat{x}_p(w) = \frac{1}{2} \int_{-\frac{1}{2}}^{\frac{1}{2}} \cos((10\pi + \omega)t) + \cos((10\pi - \omega)t) dt = \frac{1}{2} \left[ \int_{-\frac{1}{2}}^{\frac{1}{2}} \cos((10\pi + \omega)t) dt + \int_{-\frac{1}{2}}^{\frac{1}{2}} \cos((10\pi - \omega)t) dt \right]$$

$$\Rightarrow \hat{x}_p(w) = \frac{1}{10\pi + \omega} \sin\left(\frac{10\pi + \omega}{2\pi}\right) + \frac{1}{10\pi - \omega} \sin\left(\frac{10\pi - \omega}{2\pi}\right) = \frac{1}{\pi} \left( \sin\left(\frac{10\pi + \omega}{2\pi}\right) + \sin\left(\frac{10\pi - \omega}{2\pi}\right) \right)$$



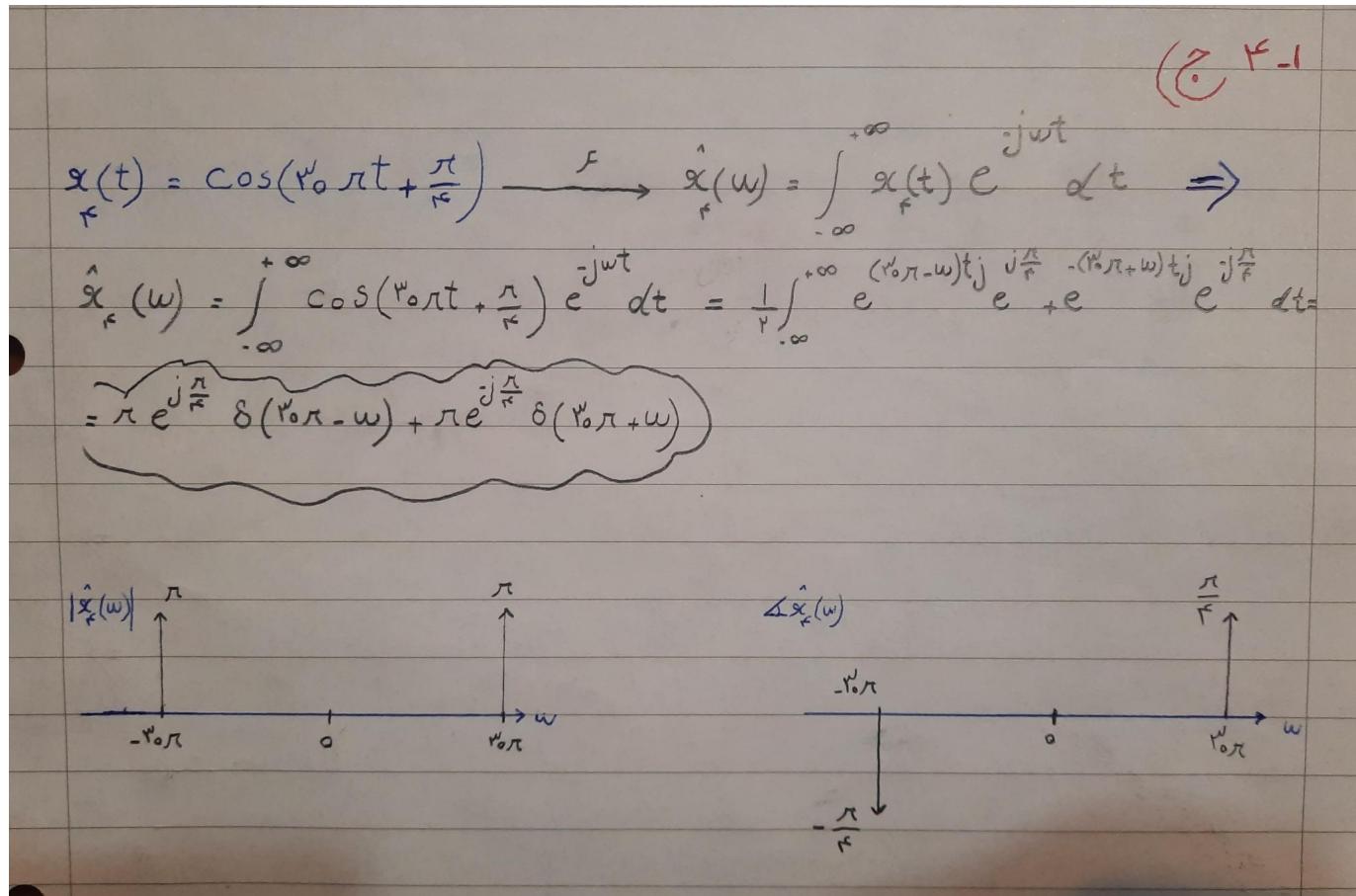
### تمرین ۴-۱)





سیگنال ورودی مون  $\cos(\omega_0 \pi t + \frac{\pi}{4})$  با فرکانس  $30$  است و می توان آن را تبدیل به دو سیگنال ویژه با فرکانس های  $15$  و  $-15$  کرد.

به دلیل وجود فاز در  $\cos$  ما شاهد وجود فاز در تبدیل فوریه مون هستیم.



آن دایم  $\omega = 2\pi f$  بیشتر داریم  $\omega = -30\pi, 30\pi$  و  $\omega = 0$  نیست  
و این درست است  $f = -15, 15$  که اینها برابر عادلند.

آن دایم صورت پردازه، دستگاه ماکریم خوب برای ۱ فریز نیست بیشتر

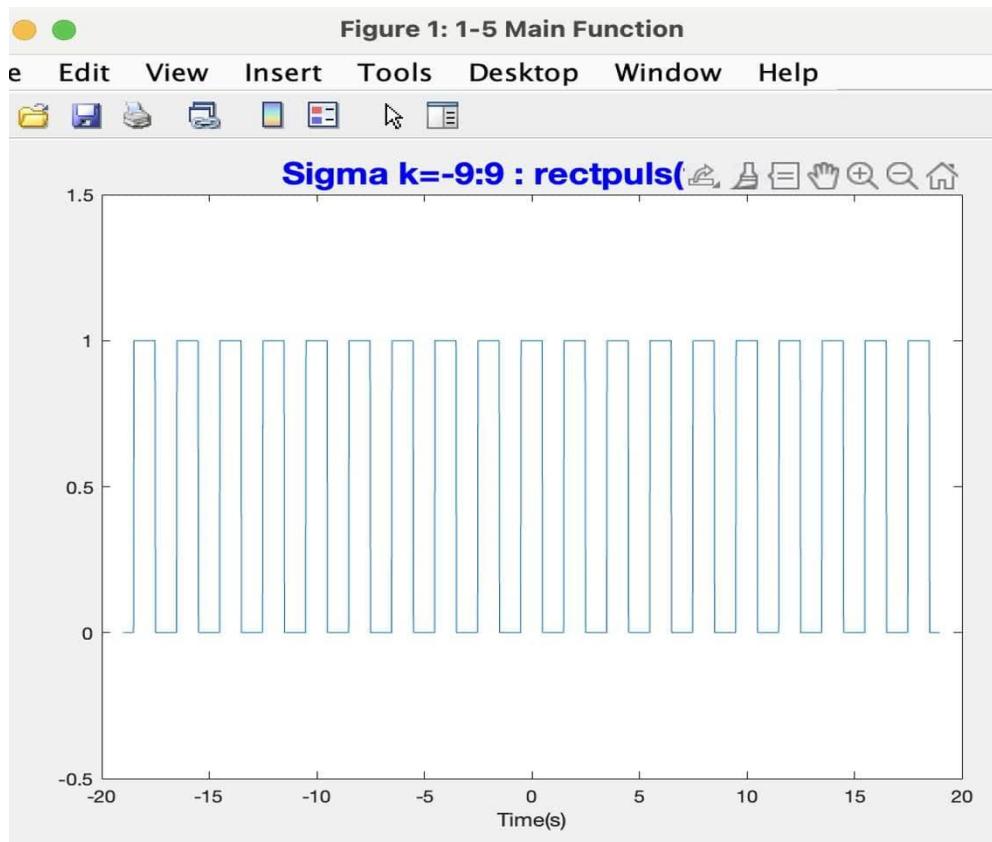
خوب داشته تبدیل فوریه دستگاه ۱ را کاغذ جوشید.

دستگاه ما زمانی تبدیل فوریه را بر حسب ضریب  $\omega$  محاسبه کرد که

نشده  $25, 0, -25$  که با مقدارهای روکاغذ  $\frac{\pi}{4}, -\frac{\pi}{4}$  مطابقت دارد.

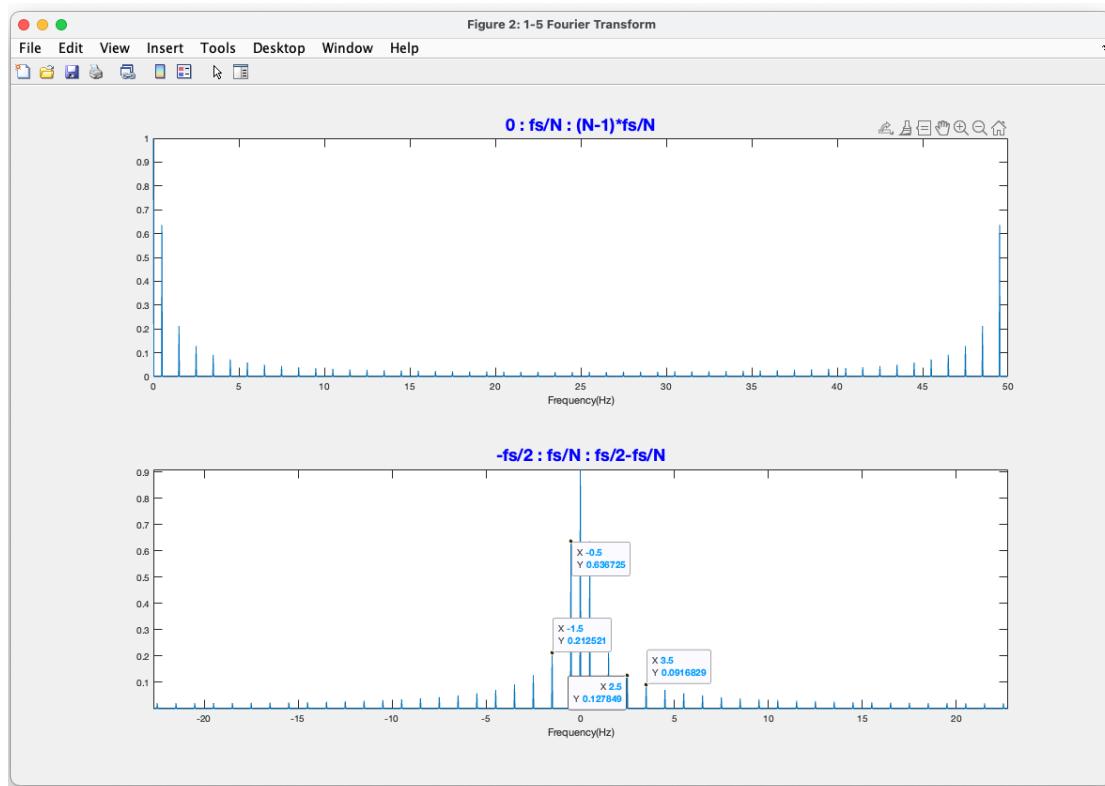
خوب دستگاه با را کاغذ مطابقت دارد.

## تمرین ۱ (۵-۱)



این تابع جمع یک سری تابع پالس شیفت یافته است. و طبق قسمت قبلی دریافتیم که تبدیل فوریه سیگنال پالس ضربه س.

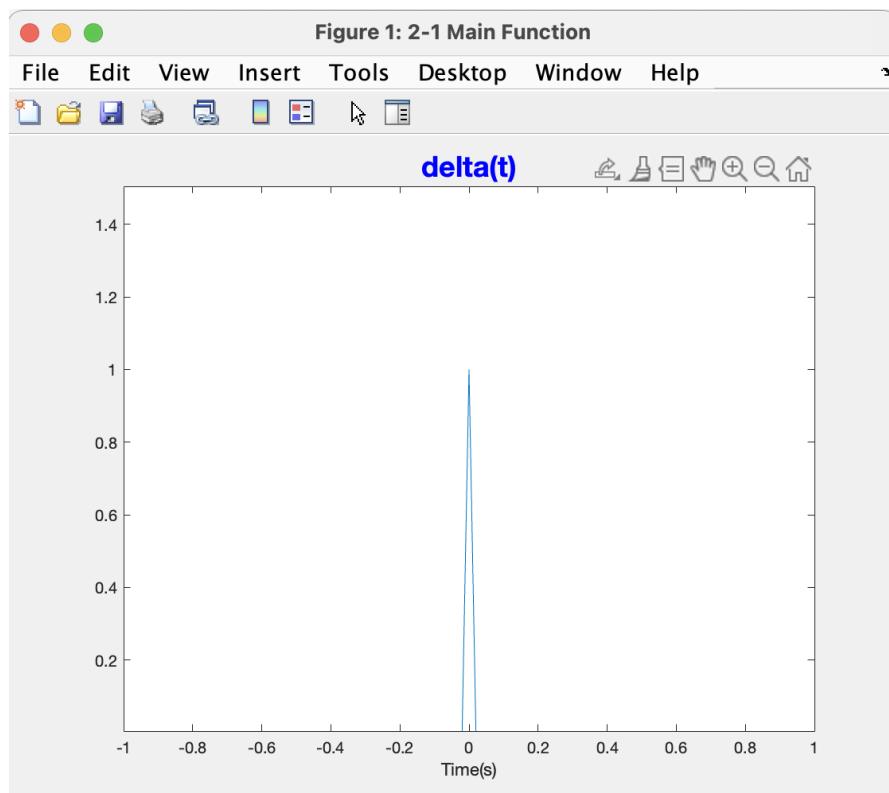
از آنجا که تبدیل فوریه یک سیستم LTI است، حاصل تبدیل فوریه  $x_5$  به شکل مجموع شیفت های تبدیل فوریه سیگنال پالس است و چون سیگنال پالس تبدیل ش ضربه است، در خروجی تبدیل  $x_5$  مجموعه ای از ضربه مشاهده می کنیم.



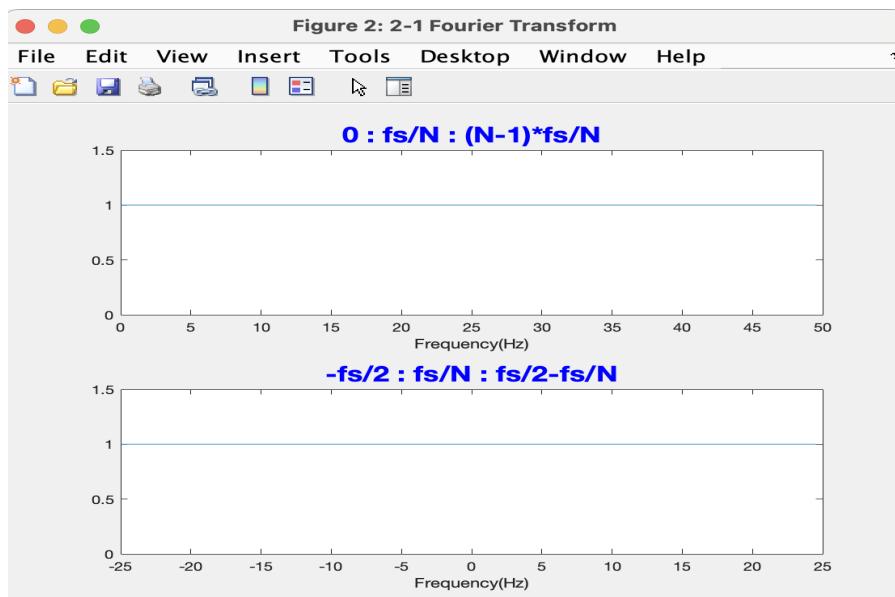
طبق عکس بالا، فواصل ضربه ها  $0.5\text{Hz}$  است.

## تمرین ۲:

### تمرین ۱-۲)



چون تابع دلتا تغییرات بسیار شدیدی دارد و ناپیوسته است، پس باید تبدیل فوریه آن شامل بزرگترین فرکانس ها باشد و چون این تغییرات با تعدادی محدودی فرکانس قابل بیان نخواهد بود باید همه ای فرکانس ها در تولید دلta شرکت کنند.



به همین خاطر تبدیل فوریه دلta یک خط ثابت است تا همه فرکانس ها در تولید دلta شرکت کنند.

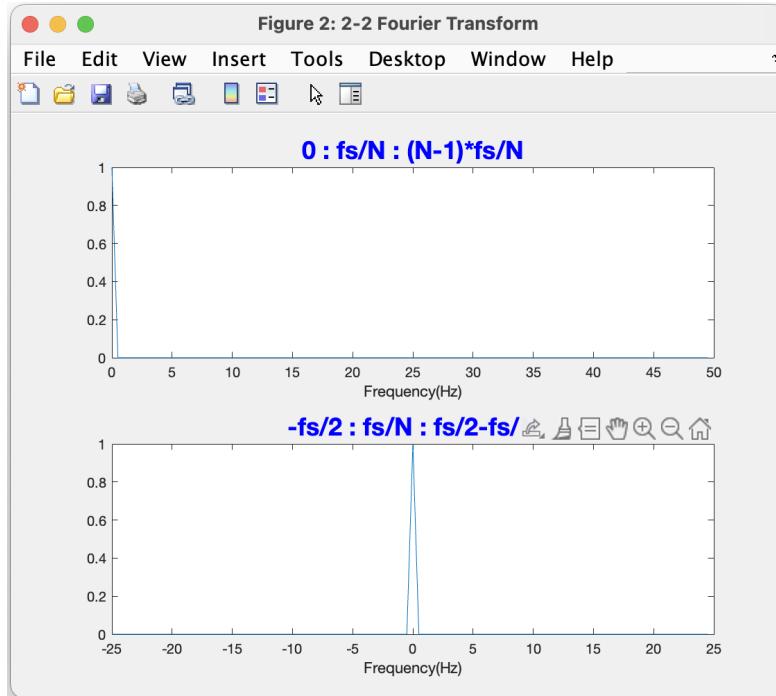
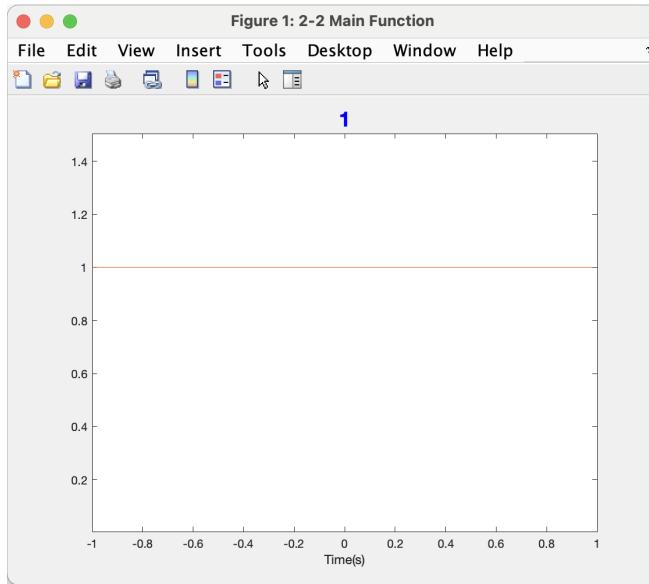
(۲-۲)

$$\hat{x}_f(t) = \delta(t) \xrightarrow{\mathcal{F}} \hat{x}_f(\omega) = \int_{-\infty}^{+\infty} \hat{x}_f(t) e^{-j\omega t} dt = \int_{-\infty}^{+\infty} \delta(t) e^{-j\omega t} dt = \int_{-\infty}^{+\infty} \delta(t) \delta(t=0) dt = 1$$

$\hat{x}_f(\omega) = 1 \Rightarrow \hat{x}_f(f) = 1$

با جزء نسب مطابقت دارد.

## تمرین ۲-۲



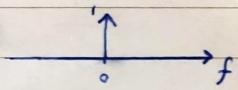
اگر تابعی تغییرات کنندی داشته باشد تبدیل فوریه آن شامل فرکانس های پایین خواهد بود.

تابع  $x$  تابع ثابت است و تغییرات و ناپیوستگی ندارد، از این جهت به فرکانس های پایین و در اینجا تنها به فرکانس نیاز داریم.

(ج) ۴-۲

$$x_v(t) = \int_{-\infty}^{+\infty} \hat{x}_v(\omega) e^{j\omega t} d\omega = \int_{-\infty}^{+\infty} e^{j\omega t} d\omega = 2\pi \delta(\omega) \Rightarrow \hat{x}_v(f) = \delta(\omega)$$

با خروج تابع مطابقت دارد.



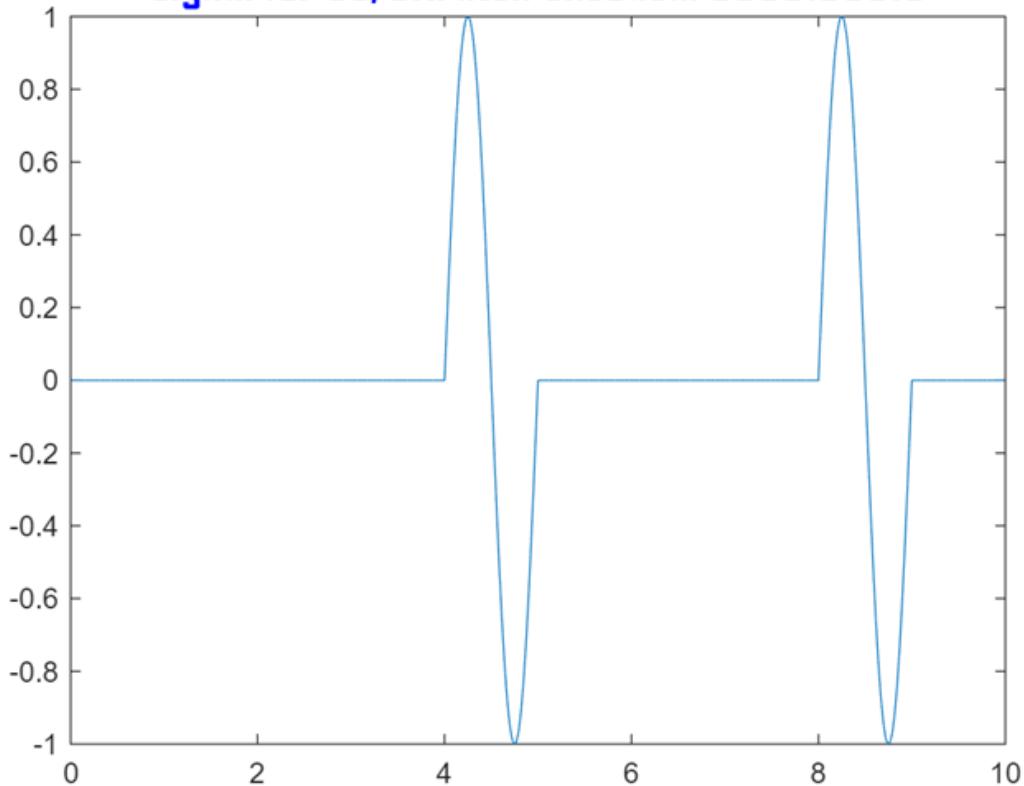
### تمرین ۳

از تابع map\_set استفاده می‌کنیم تا لیست حروف به همراه با پری شان را تولید کنیم.

پس از آن از تابع coding\_amp استفاده می‌کنیم تا string bc را به سیگنال اش تبدیل کنیم.

نتیجه را در زیر مشاهده می‌کنید:

signal for bc, bitrate:1 encoded: 0000100010



نحوه‌ی کار coding\_amp این گونه است که ابتدا کلمات msg را یک به یک خونده، سپس coded\_signal بایت‌ی

را تولید می‌کند.

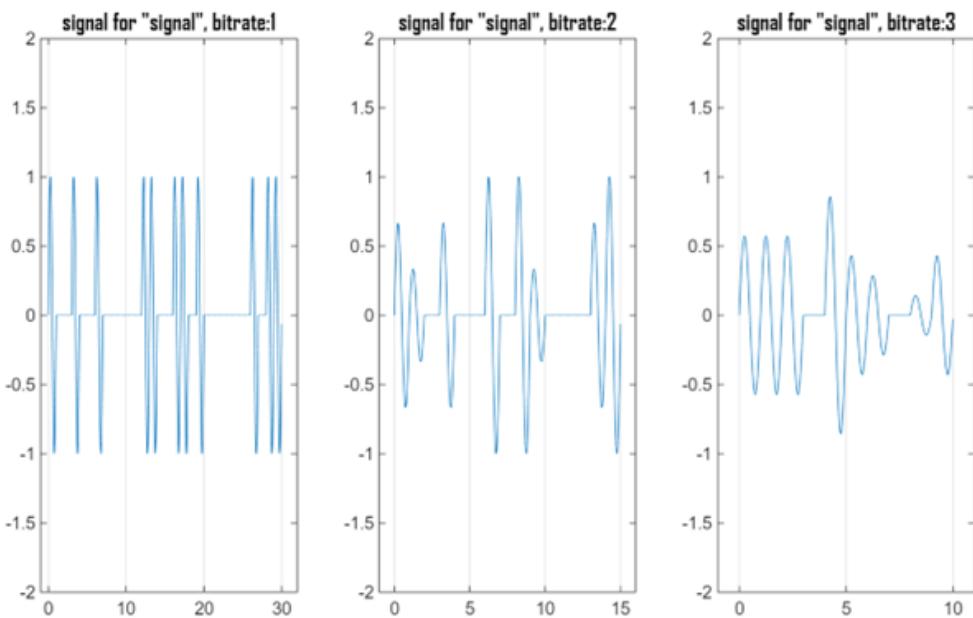
پس از آن به آخر coded\_signal به تعداد صفر اضافه می‌کند تا بتوان آن را به تکه‌های صحیح به اندازه‌ی bit\_rate

برای تبدیل شدن به سیگنال تبدیل کرد.

پس از آن coded\_signal را تکه تکه می‌خوانیم و تبدیل به سیگنال مشخص با ارتفاع مشخص می‌کنیم و آن سیگنال

را بر می‌گردانیم.

در قسمت بعدی، با استفاده از همان تابع، سیگنال با  $bit\_rate=1, 2, 3$  برای  $msg = \text{signal}$  محاسبه می‌شود.



از تابع `decoding_amp` برای تبدیل سیگنال به کلمه استفاده می‌کنیم.

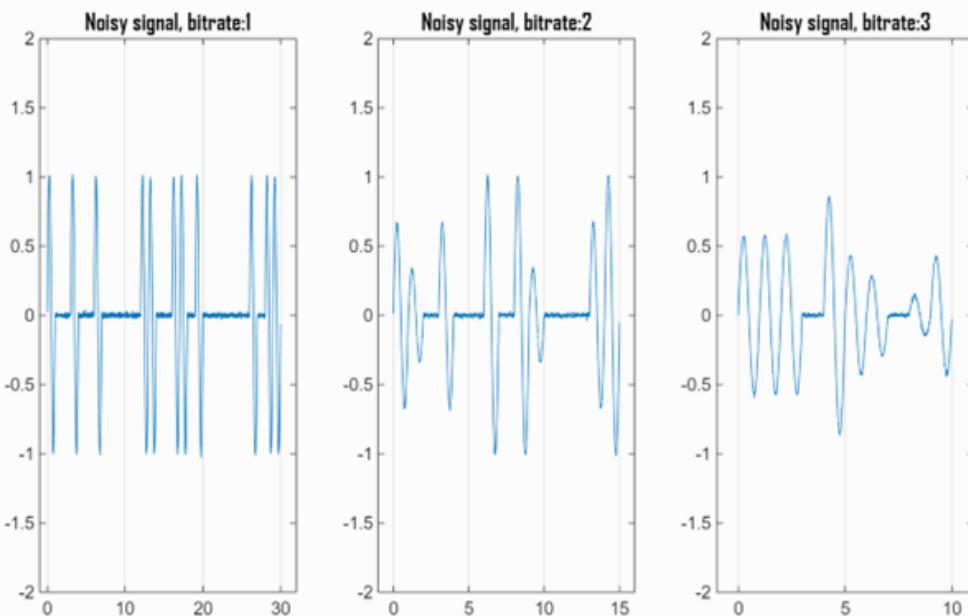
بدین گونه که سیگنال را یک ثانیه و در سمپل های ۱۰۰ تایی جدا می کنیم و `corrolation` اش را با سیگنال  $2\sin(2\pi*t)$  محاسبه می کنیم و بر حسب `bit_rate` بهترین `coded_signal` را برای آن می‌یابیم.

پس از آن با استفاده از `map_set` نمایش باینری `coded_signal` را به `string` تبدیل می‌کنیم.

این تابع روی سیگنال های قسمت بعد تست شد و نتیجه ها درست هستند.

حال زمان اضافه شدن نویز به سیگنال های ورودی است. نتیجه و تصویر سیگنال ها با آلفا های متفاوت نویز ها را می‌توانید در زیر مشاهده نمایید:

`alpha = 0.01`



signal

signal

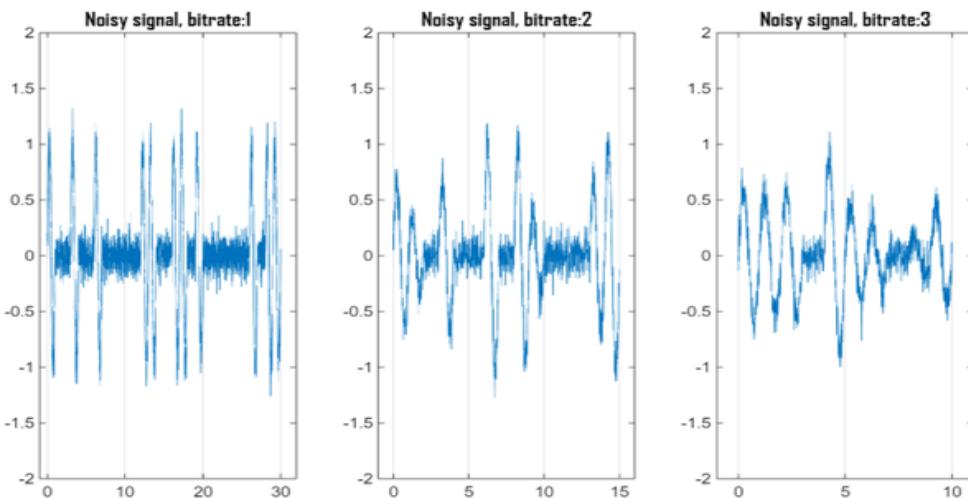
signal

bitRate1 CORRECT

bitRate2 CORRECT

bitRate3 CORRECT

$\alpha = 0.1$



signal

signal

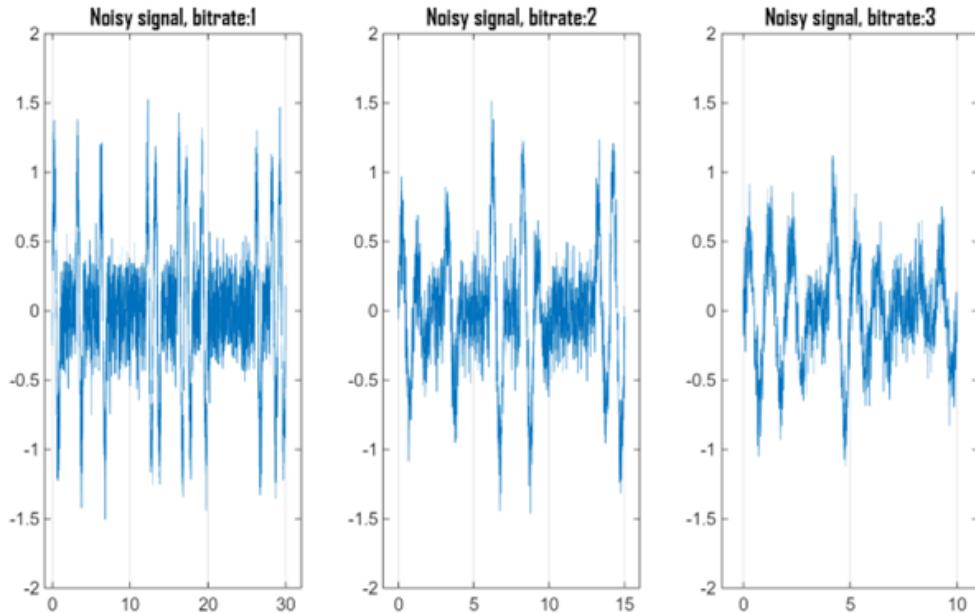
signal

bitRate1 CORRECT

bitRate2 CORRECT

bitRate3 CORRECT

alpha = 0.2



signal

signal

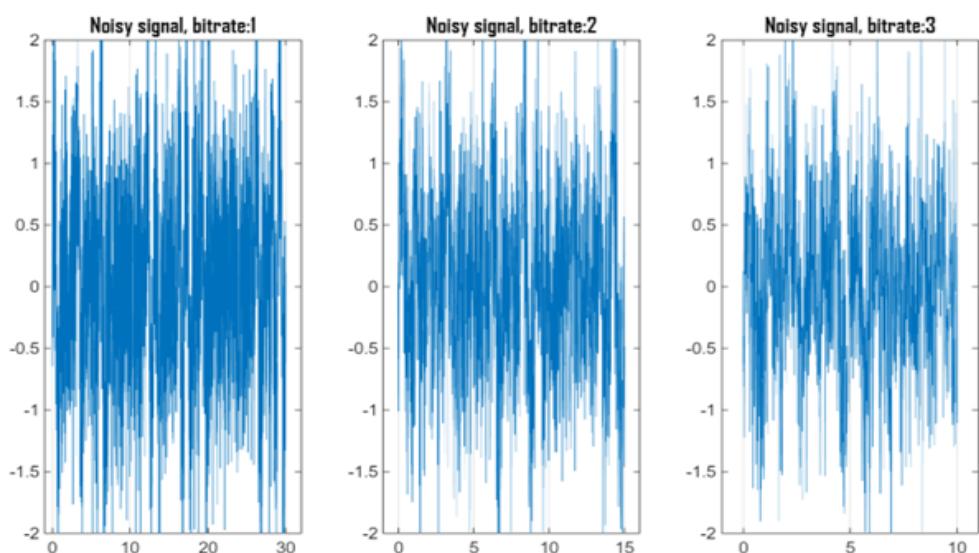
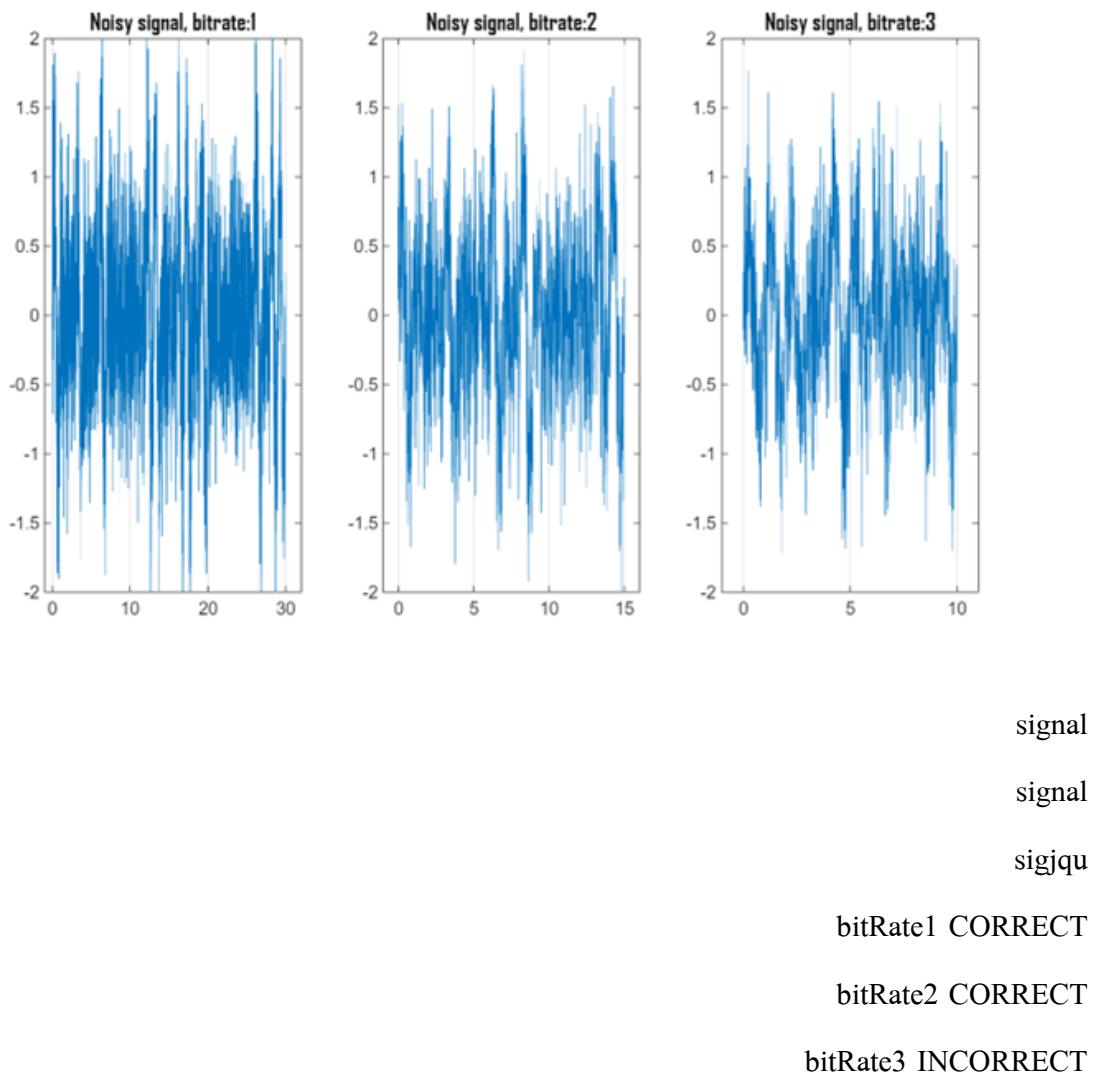
sggnal

bitRate1 CORRECT

bitRate2 CORRECT

bitRate3 INCORRECT

alpha = 0.5



signal

sienal

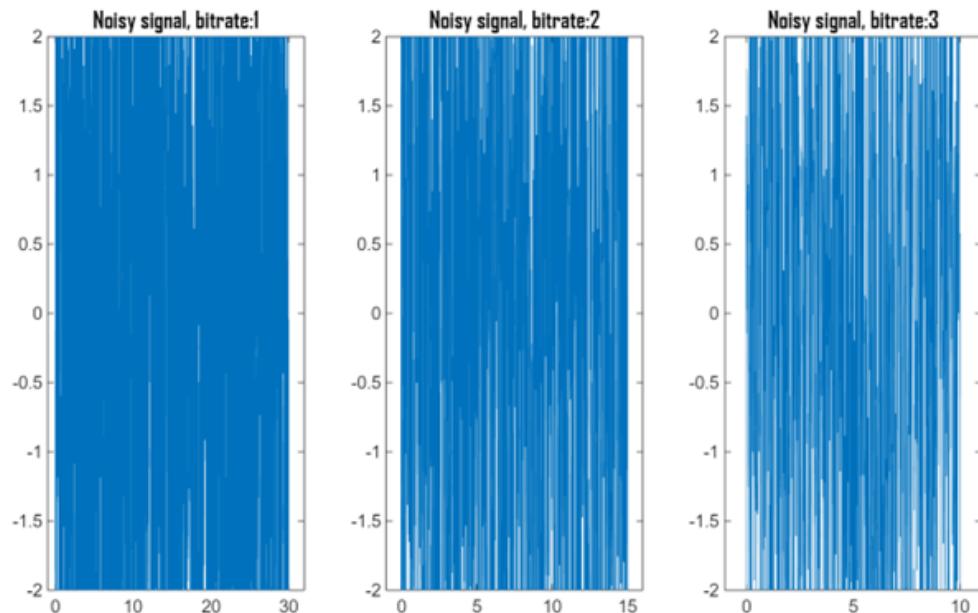
rinnac

bitRate1 CORRECT

bitRate2 INCORRECT

bitRate3 INCORRECT

alpha = 2.5



agpal

iaemmo

rip!ee

bitRate1 INCORRECT

bitRate2 INCORRECT

bitRate3 INCORRECT

نتیجہ ی کلی:

البته نمی‌توان به طور قطعی اظهار نظر نمود که در کدام آلفا کدامیک درست تشخیص داد و کدامیک خیر ولی در حالت کلی، نتایج زیر برقرارند و طبق توضیحات خود صورت پژوهه هر چه bit\_rate افزایش یابد سرعت انتقال اطلاعات افزایش می‌یابد ولی نسب به نویز مقاومت کمتر می‌شود.

alpha = 0.1 -> ALL CORRECT

alpha = 0.2 -> bitrate 3 INCORRECT

alpha = 0.5 -> bitrate 3 INCORRECT

alpha = 0.7 -> bitrate 3&2 INCORRECT

alpha = 2.5, 3.5 -> ALL INCORRECT

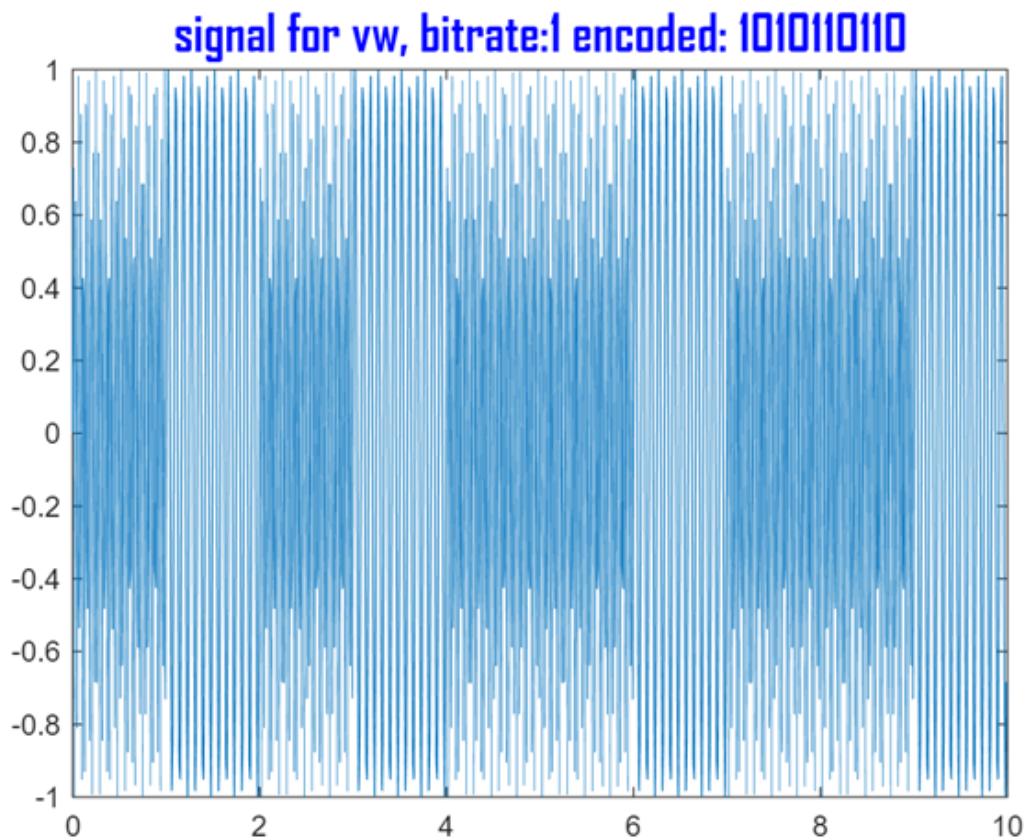
برای مقام کردن روش کد گزاری دامنه، می‌توانیم ماکریم دامنه را افزایش دهیم که در آن صورت بهتر می‌توان نویز‌ها را پیش‌بینی کرد ولی این به معنای افزایش توان مصرفی فرستنده خواهد بود.

## تمرین ۴

از تابع map\_set استفاده می‌کنیم تا لیست حروف به همراه باینری شان را تولید کنیم.

پس از آن از تابع coding\_& string استفاده می‌کنیم تا vw را به سیگنال اش تبدیل کنیم.

نتیجه را در زیر مشاهده می‌کنید:



نحوه‌ی کار coding\_amp این گونه است که ابتدا کلمات msg را یک به یک خونده، سپس coded\_signal بایت‌ی را تولید می‌کند.

پس از آن به آخر coded\_signal به تعداد صفر اضافه می‌کند تا بتوان آن را به تکه‌های صحیح به اندازه‌ی bit\_rate برای تبدیل شدن به سیگنال تبدیل کرد.

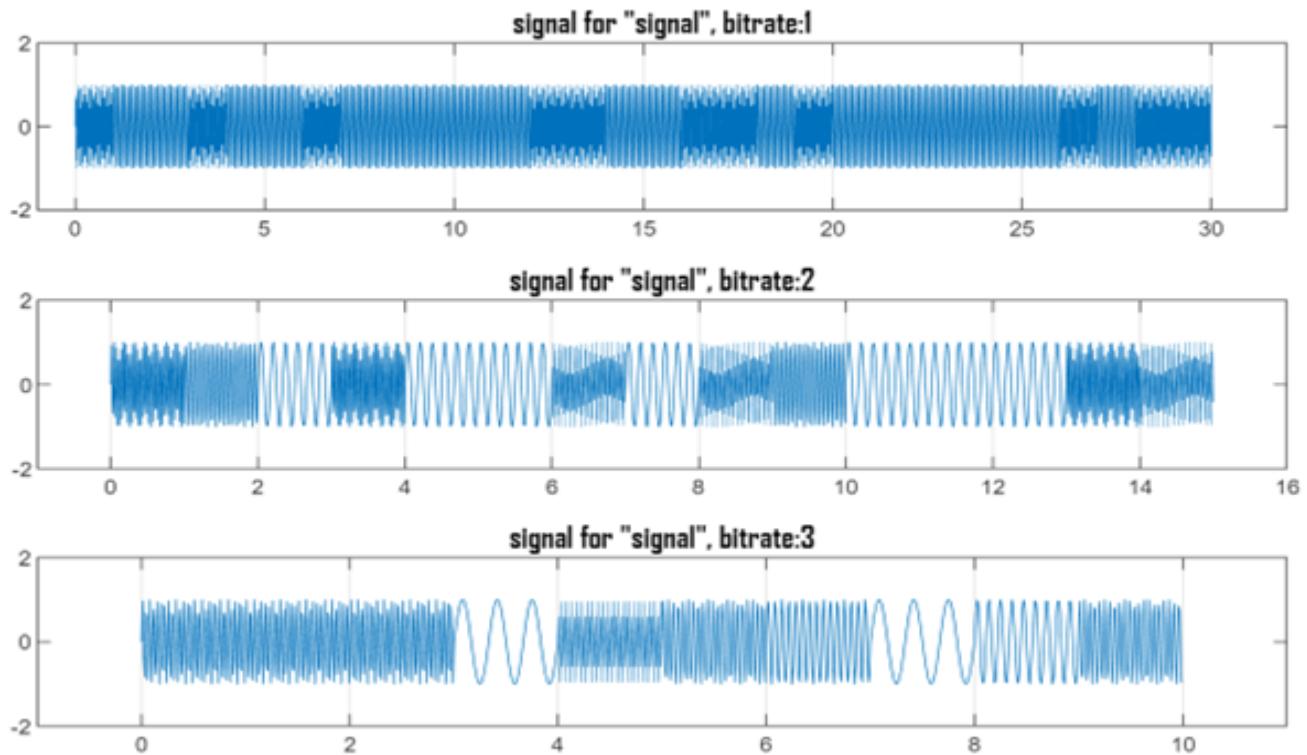
پس از آن coded\_signal را تکه تکه می‌خوانیم و تبدیل به سیگنال مشخص با فرکانس مشخص می‌کنیم و آن سیگنال را بر می‌گردانیم.

برای پیدا کردن فرکانس از فرمول زیر استفاده می‌شود که در آن، step برابر فاصله‌ی اولین فرکانس از صفر است و پس از آن سیگنال‌ها باهم به اندازه‌ی  $2 * \text{step}$  فاصله دارند. val برابر شماره‌ی فرکانس است که از 0 تا  $\text{max} - 1$  شماره گذاری می‌شود. و فرکانس برابر  $\text{round}(\text{step} * (2 * \text{val} + 1))$  می‌باشد.

```
max = 2 & bit_rate;
step = 49/(max *2);
val = bin2dec(coded_signal((i - 1)*bit_rate+1 : i*bit_rate));
```

```
freq = round(step * (2*val + 1));
```

در قسمت بعدی، با استفاده از همان تابع، سیگنال با  $\text{bit\_rate}=1, 2, 3$  برای  $\text{msg} = \text{signal}$  محاسبه می شود.



از تابع `decoding_amp` برای تبدیل سیگنال به کلمه استفاده می کنیم.

بدین گونه که سیگنال را یک ثانیه یک ثانیه و در سمپل های ۱۰۰ تایی جدا می کیم و `fft` اش را محاسبه می کیم و

بر حسب `bit_rate` بهترین `coded_signal` را برای آن می یابیم. برای انجام این کار از تابع `find_nearest` استفاده

می شود که نزدیکترین فرکانس را به ماکریم فرکانس مثبت را به عنوان فرکانس مورد استفاده انتخاب می کند.

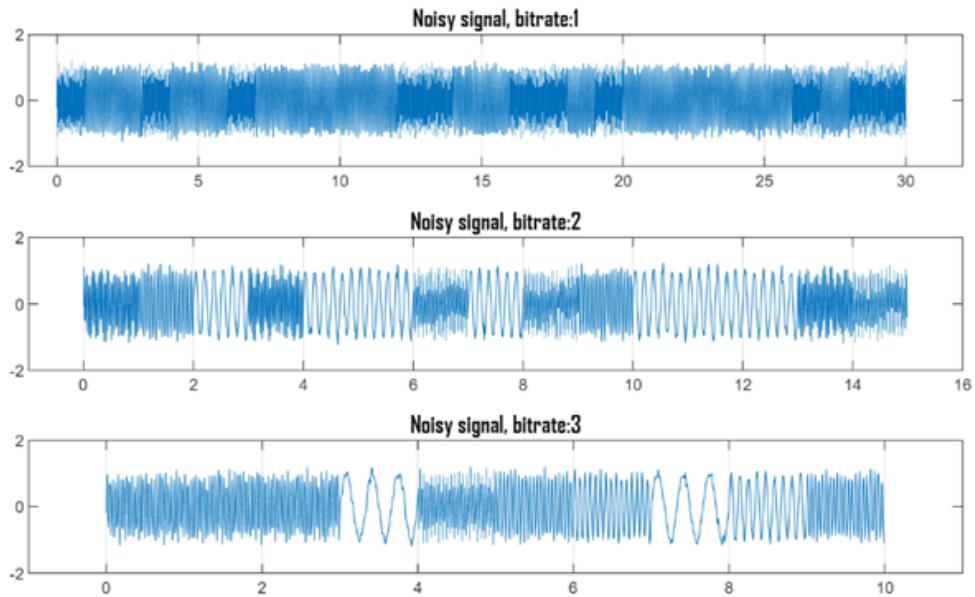
پس از آن با استفاده از `map_set` نمایش باینری `coded_signal` را به `string` تبدیل می کنیم.

این تابع روی سیگنال های قسمت قبل تست شد و نتیجه ها درست هستند.

حال زمان اضافه شدن نویز به سیگنال های ورودی است. نتیجه و تصویر سیگنال ها با آلفا های متفاوت نویز ها را می توانید در زیر مشاهده نمایید:

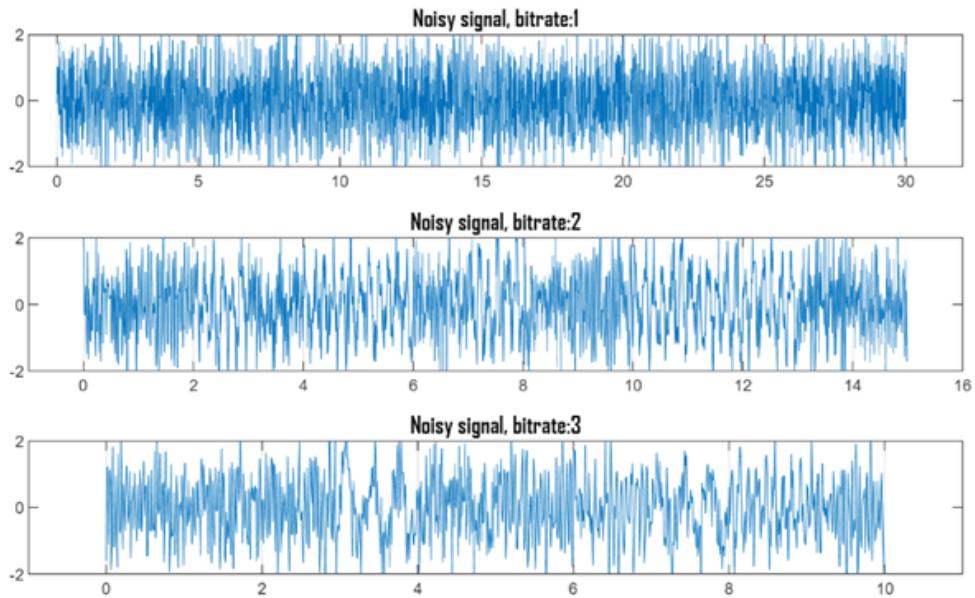
`alpha = 0.1;`

bitRate1 CORRECT    bitRate2 CORRECT    bitRate3 CORRECT



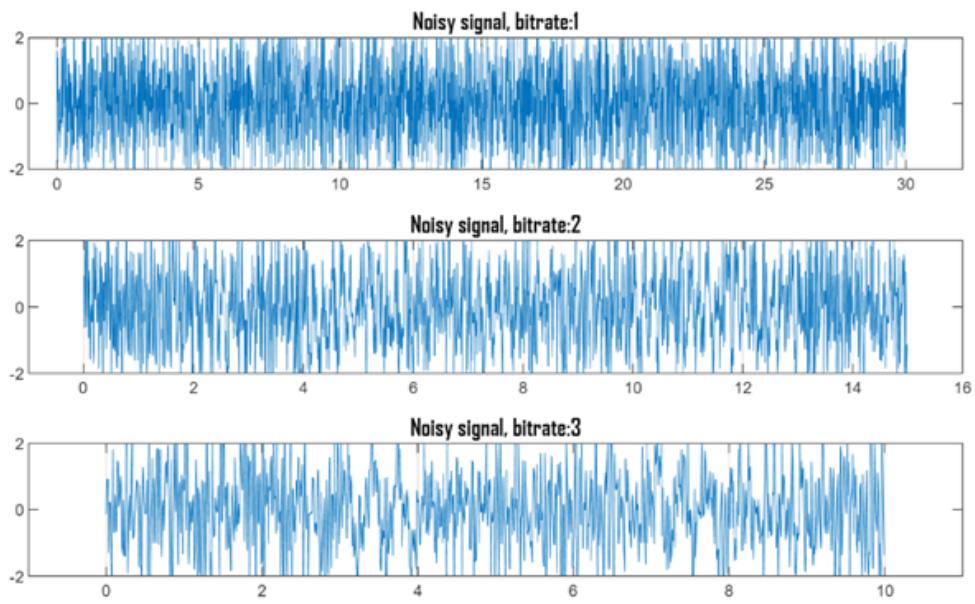
alpha = 0.7:

bitRate1 CORRECT    bitRate2 CORRECT    bitRate3 CORRECT



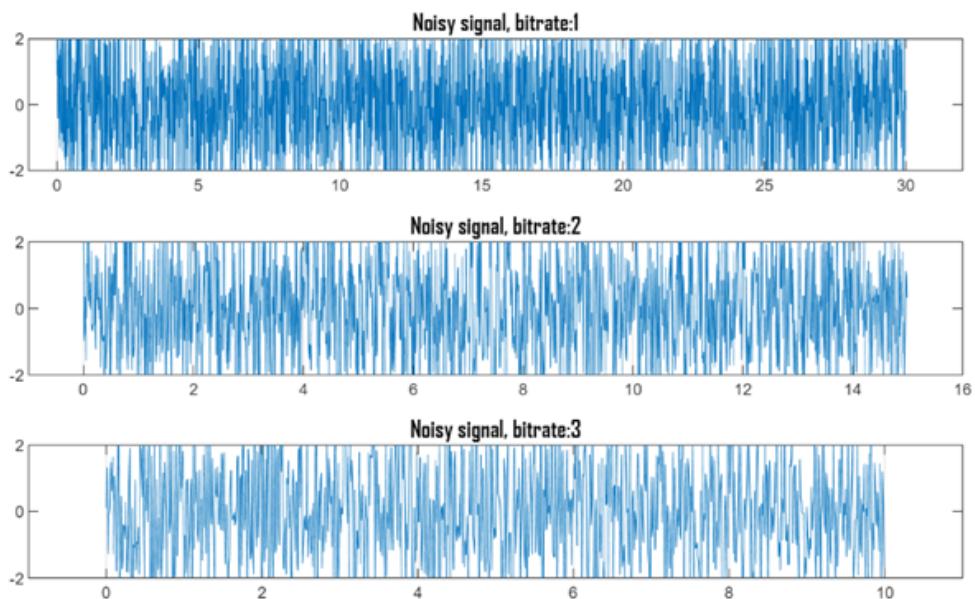
alpha = 1:

bitRate1 CORRECT    bitRate2 CORRECT    bitRate3 CORRECT



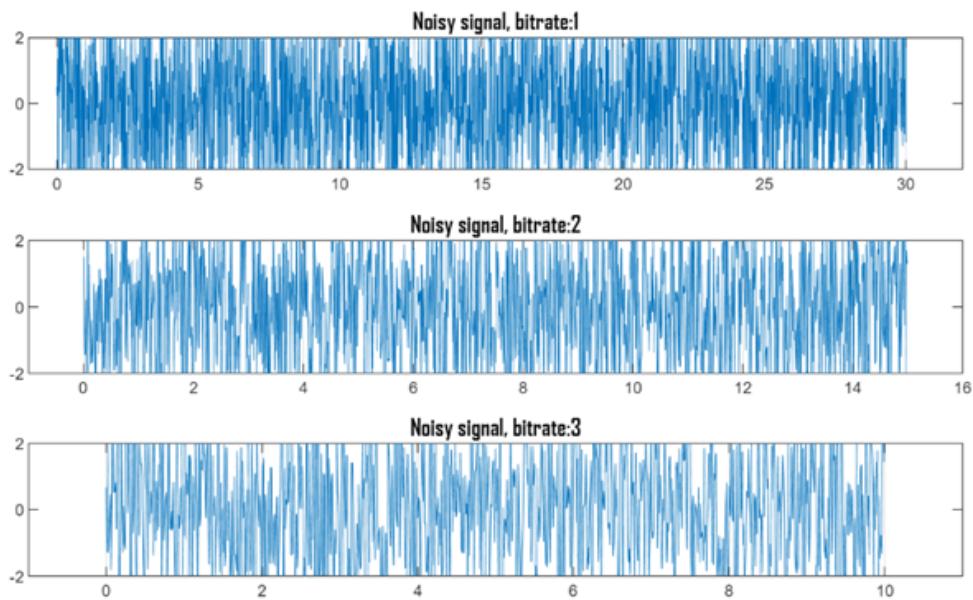
$\alpha = 1.5:$

bitRate1 CORRECT    bitRate2 CORRECT    bitRate3 INCORRECT



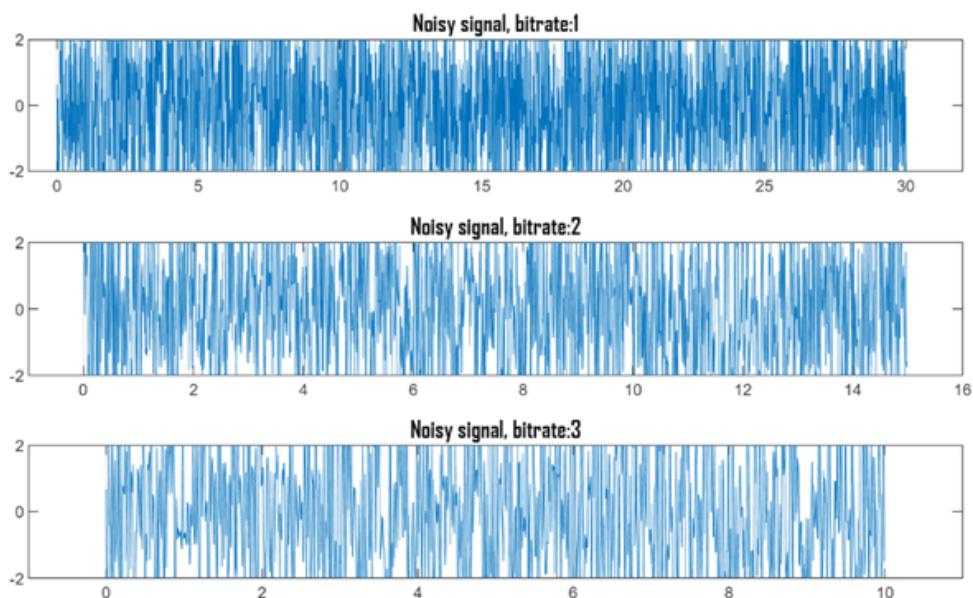
$\alpha = 1.75:$

bitRate1 INCORRECT    bitRate2 INCORRECT    bitRate3 INCORRECT



$\alpha = 2:$

bitRate1 INCORRECT      bitRate2 INCORRECT      bitRate3 INCORRECT



نتیجه ی کلی:

البته نمی توان به طور قطع اظهار نظر نمود که در کدام آلفا کدامیک درست تشخیص داد و کدامیک خیر ولي در حالت کلی، نتایج زیر برقرارند و طبق توضیحات خود صورت پروره هر چه `bit_rate` افزایش یابد سرعت انتقال اطلاعات افزایش

می‌یابد ولی نسب به نویز مقاومت کمتر می‌شود. چراکه فاصله‌ی فرکانس‌های انتخابی کمتر می‌شود و احتمال تشخیص درست کمتر.

دقت کنید در برخی مواقع بعلت سمپل  $N=100$  گاهی اوقات بدلیل مشارکت برخی دیگر از فرکانس‌های مجاور نتایج در صورت نویز می‌تواند طبق پیش‌بینی ظاهر نشود.

alpha = 0.1 -> ALL CORRECT

alpha = 0.7 -> ALL CORRECT

alpha = 1 -> ALL CORRECT

alpha = 1.5 -> bitrate 3 INCORRECT

alpha = 1.75 -> ALL INCORRECT

alpha = 2 -> ALL INCORRECT

برای مقام کردن روش کد گزاری فرکانسی، می‌توانیم فاصله‌ی فرکانس‌های انتخابی را افزایش دهیم که در آن صورت بهتر می‌توان نویز‌ها را پیش‌بینی کرد و این به معنای افزایش پهنای باند خواهد بود.