**به نام خدا**

**سید محمد حسین مظهری 810101520 پروژه 5 درس سیگنال و سیستم**

**پارسا دقیق 810101419**

**بخش اول :**

**تمرین 1-0)**

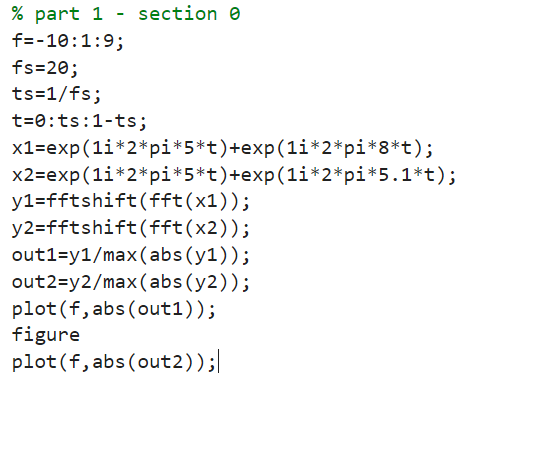
 **ایجاد بردار زمانی**: بردار زمانی از 0 تا 1 با گام زمانی محاسبه شده ایجاد می‌شود. این بردار زمانی برای نمونه‌برداری سیگنال‌ها استفاده می‌شود.

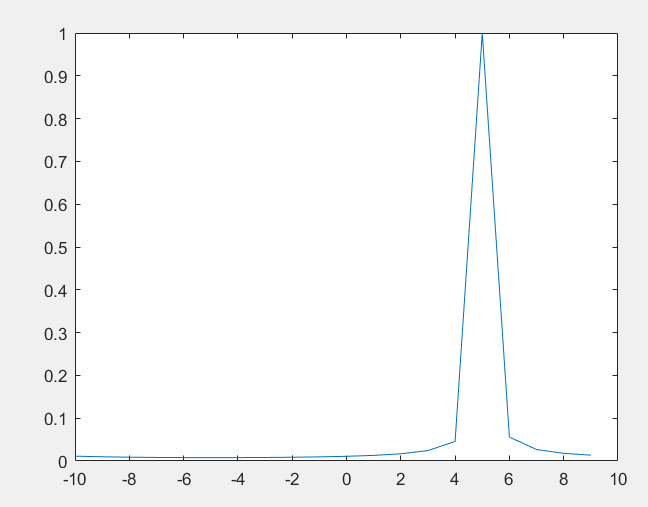
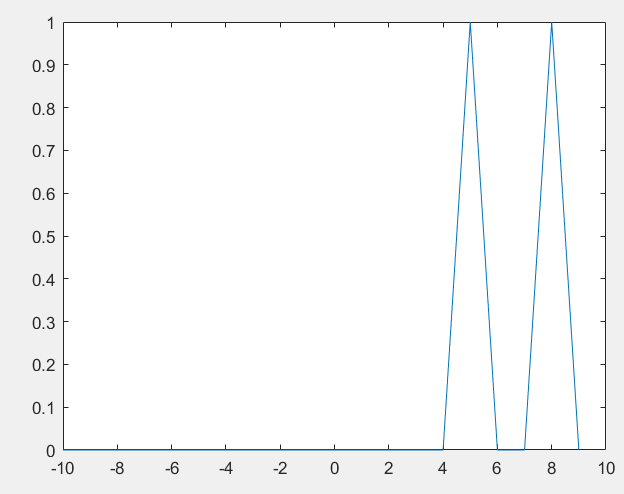
 **تولید سیگنال‌های ترکیبی**: دو سیگنال ترکیبی ایجاد می‌شوند. سیگنال اول از دو موج سینوسی با فرکانس‌های 5 و 8 هرتز تشکیل شده است و سیگنال دوم از دو موج سینوسی با فرکانس‌های 5 و 5.1 هرتز تشکیل شده است.

 **محاسبه تبدیل فوریه**: تبدیل فوریه سریع (FFT) بر روی هر دو سیگنال انجام می‌شود تا طیف فرکانسی آنها به دست آید. سپس این طیف‌ها با استفاده از تابع fftshift شیفت داده می‌شوند تا مرکز فرکانسی در وسط قرار گیرد.

 **نرمال‌سازی طیف‌ها**: هر یک از طیف‌های فرکانسی به دست آمده نرمال‌سازی می‌شوند تا ماکزیمم مقدار طیف برابر با 1 شود. این نرمال‌سازی به منظور مقایسه بهتر و نمایش بهتر طیف‌ها انجام می‌شود.

 **نمایش نمودارها**: ابتدا طیف نرمال‌سازی شده سیگنال اول بر حسب فرکانس رسم می‌شود. سپس یک پنجره جدید ایجاد می‌شود و طیف نرمال‌سازی شده سیگنال دوم نیز بر حسب فرکانس رسم می‌شود. این نمودارها به ما امکان می‌دهند که تفاوت‌های فرکانسی دو سیگنال را بصری مشاهده و مقایسه کنیم.

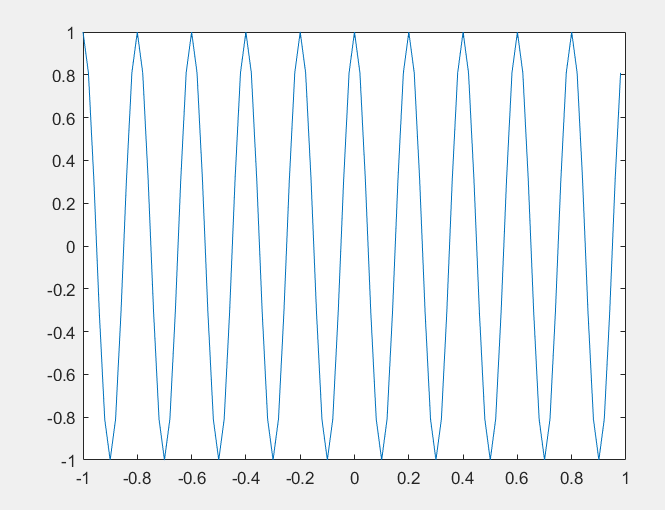


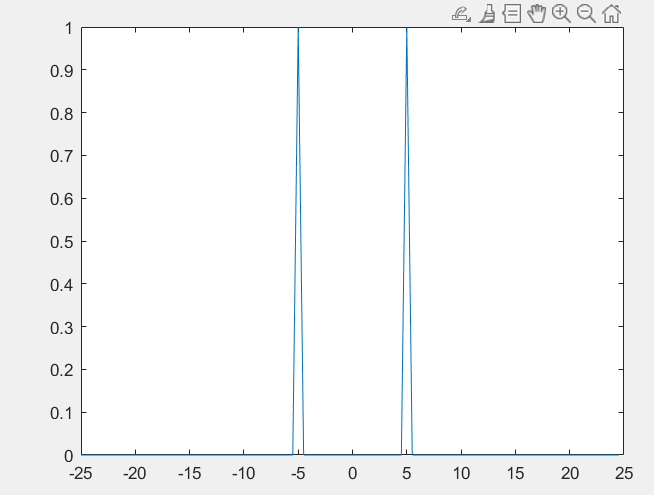
 

همان‌طور که مشاهده می‌شود، در نمودار اول، دو سیگنال به‌وضوح از یکدیگر تفکیک شده‌اند. اما در مورد x2، به دلیل این که اختلاف فرکانس سیگنال‌ها کمتر از رزولوشن سیستم است (یعنی 0.1 که کمتر از 1 است)، شناسایی آن‌ها به طور جداگانه ممکن نیست و تنها یک قله در فرکانس 5 هرتز قابل تشخیص است. این نشان می‌دهد که رزولوشن سیستم برای جداسازی فرکانس‌های نزدیک به هم کافی نیست.

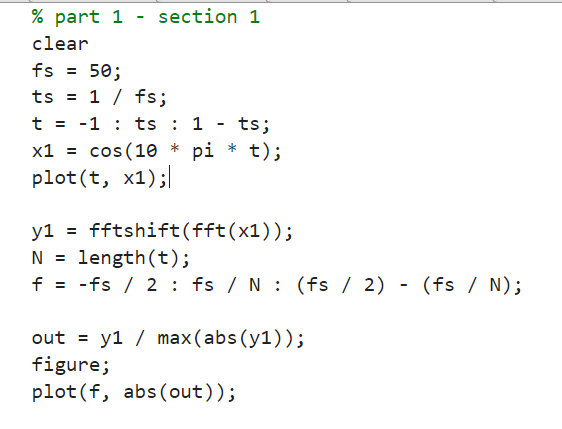
**تمرین 1-1)**

ابتدا سیگنال cos(10\*pi\*t) را رسم می‌کنیم.



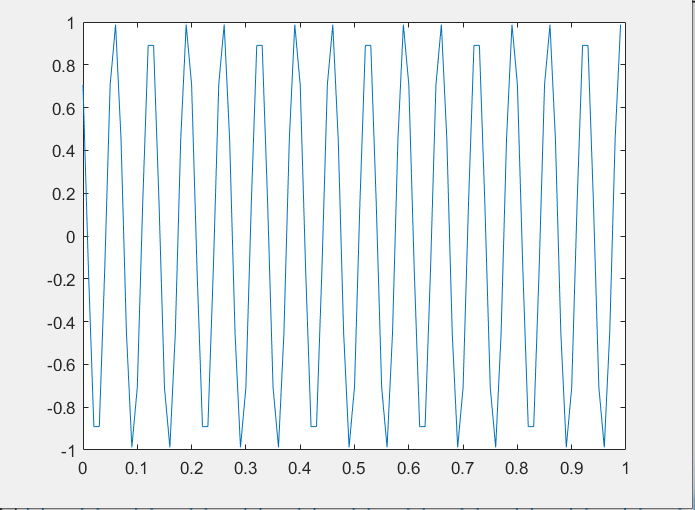


پس از تولید سیگنال مورد نظر، برای تحلیل آن در حوزه فرکانس، از تبدیل فوریه استفاده می‌کنیم. به این منظور، از تابع fftshift برای انتقال سیگنال به مرکز و از fft برای انجام تبدیل فوریه سریع بهره می‌بریم. سپس، با تقسیم نتیجه بر بزرگترین مقدار مطلق آن، طیف را نرمال‌سازی می‌کنیم تا ماکزیمم مقدار آن برابر 1 شود. در پایان، برای نمایش اندازه طیف در حوزه فرکانس، از تابع abs استفاده کرده و نمودار حاصل را با plot رسم می‌کنیم. مشاهده می‌کنیم که همان‌طور که انتظار داریم از تبدیل فوریه یک سیگنال کسینوسی، در فرکانس‌های ۵ و -۵ اوج‌های مشخصی وجود دارد. این نشان می‌دهد که سیگنال ما دارای این مؤلفه‌های فرکانسی است.

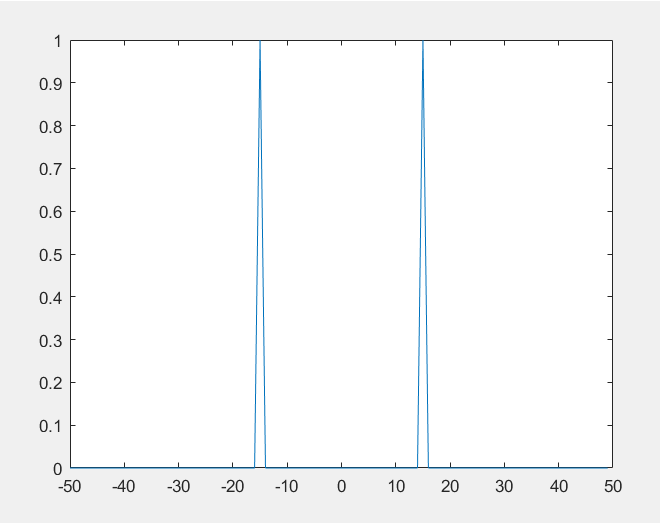


**تمرین 2-1)**

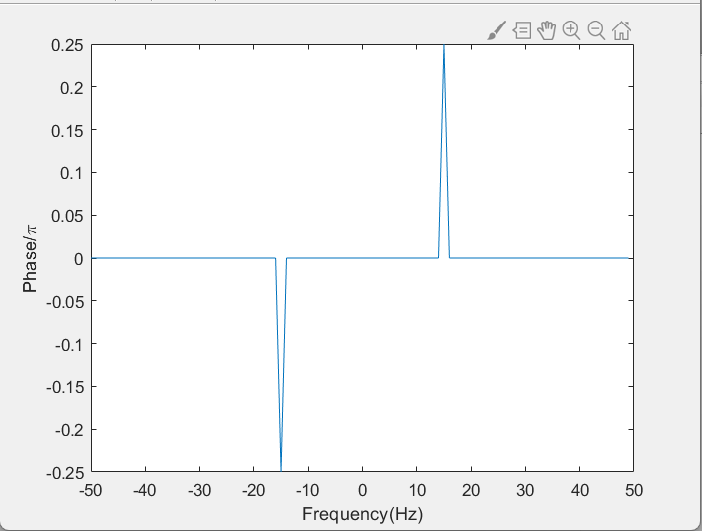
ابتدا سیگنال 𝑥2(𝑡)=cos (30 𝜋𝑡+𝜋4) را رسم میکنیم.

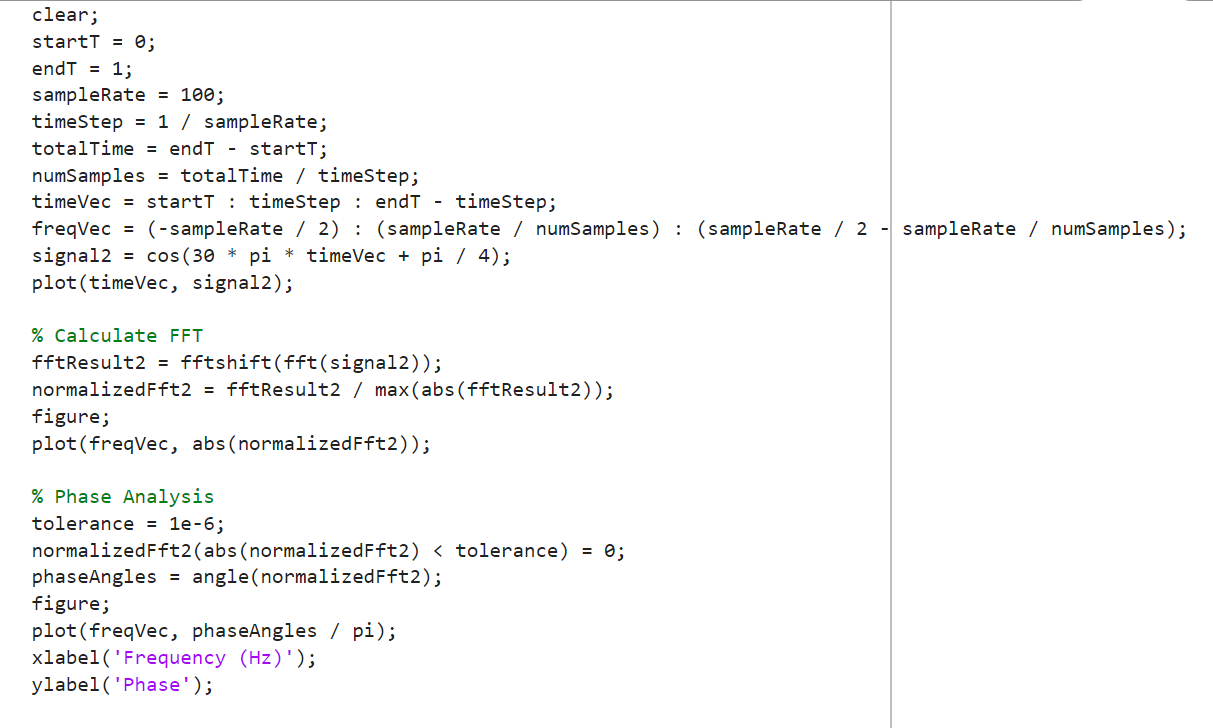


**سپس اندازه تبدیل فوریه را میکشیم که مطابق انتظارات ما در -15 و 15 مقدار دارد.**



همان‌طور که انتظار داریم، فاز یک سیگنال سینوسی در دو فرکانس با مقادیر متقابل وجود دارد. در این مورد، در فرکانس‌های 15 و -15 مشاهده می‌شود که فازها به ترتیب 0.25 و -0.25 هستند. این الگو نشان‌دهنده تقارن فاز در سیگنال‌های سینوسی است که به خوبی در داده‌ها دیده می‌شود.

****



1. تنظیمات اولیه و ایجاد بردار زمان:در ابتدا، مقادیر زمان شروع و پایان و نرخ نمونه‌برداری تعریف می‌شود. سپس گام زمانی محاسبه شده و بردار زمانی ایجاد می‌شود. این بردار زمانی نشان‌دهنده نقاط زمانی است که در آن‌ها سیگنال نمونه‌برداری می‌شود.

2. تولید سیگنال: یک سیگنال کسینوسی با فرکانس ۱۵ هرتز و یک فاز مشخص (π/4) تولید می‌شود. این سیگنال در حوزه زمان رسم می‌شود تا رفتار آن در زمان قابل مشاهده باشد.

3. محاسبه تبدیل فوریه: با استفاده از تابع `fft` و `fftshift`، تبدیل فوریه سیگنال محاسبه می‌شود. `fftshift` سیگنال را به مرکز منتقل می‌کند تا بتوان فرکانس‌های منفی و مثبت را به درستی مشاهده کرد. سپس طیف فرکانسی نرمال‌سازی می‌شود تا بیشترین مقدار برابر با ۱ باشد و اندازه طیف نرمال‌سازی شده رسم می‌شود.

4. تحلیل فاز: یک آستانه کوچک برای حذف مقادیر ناچیز تعیین می‌شود. پس از حذف این مقادیر، فاز سیگنال در حوزه فرکانس محاسبه می‌شود و نمودار فاز رسم می‌شود. برچسب‌های محورهای افقی و عمودی نیز به ترتیب به فرکانس و فاز تنظیم می‌شود.

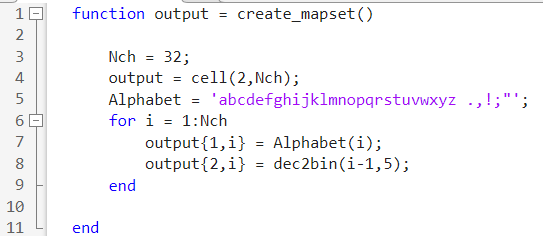
به طور خلاصه، این کد ابتدا سیگنال را تولید و در حوزه زمان نمایش می‌دهد، سپس تبدیل فوریه را محاسبه و طیف نرمال‌سازی شده را رسم می‌کند، و در نهایت فاز سیگنال را تحلیل و نمایش می‌دهد. این مراحل به ما کمک می‌کند تا تحلیل کاملی از سیگنال در حوزه زمان و فرکانس داشته باشیم.

**بخش دوم :**

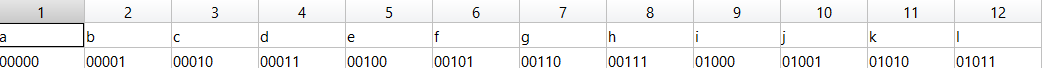
**تمرین 2-1)**

برای ایجاد mapset یک تابع ایجاد میکنیم که در سطر اول کاراکتر ها را قرار دهد و در سطر دوم عدد متناظر باینری آنها را بگذارد .

تابع ایجاد کردن mapset :

:

بخشی از خروجی تابع :



**تمرین2-2)**

در این تابع ابتدا پیامی که قرار است کدگذاری شود را با استفاده از mapset به یک عدد باینری تبدیل میکنیم و آن عدد را در یک آرایه قرار میدهیم که در هر دور حلقه یک عدد به انتهای این لوپ اضافه می شود تا در نهایت کل پیام به شکل باینری تبدیل شود .

حال پس از باینری شدن پیام میتوانیم سیگنال مورد نظر را ایجاد کنیم .

برای ایجاد سیگنال باید ابتدا یکی یکی بیت های سیگنال را به اندازه ی bit rate جدا کنیم. اگر تعداد بیت های پیام بر bit rate بخش پذیر باشد کار به سادگی انجام میشود . در غیر این صورت باید بیت های انتهای پیام را extend کنیم (به انتهای پیام صفر اضافه میکنیم. ) تا ایجاد سیگنال به درستی انجام شود . مثلا اگر در همین تمرین bit rate = 4 باشد و در انتهای پیام دو رقم 11 دااشته باشیم ، بدون extend کردن این 2 رقم معادل 3 در نظر گرفته میشوند در صورتی که با extend کردن به 1100 (12 ) میرسیم .

اگر در لوپ آخر تعداد ارقام باقی مانده بر bit rate بخش پذیر باشند که همان روند قبلی طی میشود . در غیر این صورت ارقام باقی مانده extend می شوند .

در هر دور حلقه بیت های پیام به تعداد bit rate از پیام جدا میشوند . سپس این بیت ها به یک عدد دسیمال تبدیل میشوند تا فرکانس متناظر با آنها محاسبه شود .

نحوه ی محاسبه فرکانس در این تمرین :

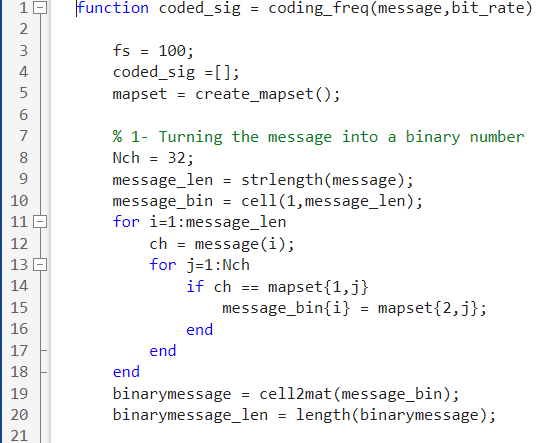
کمترین فرکانس در 5 هرتز و بیشترین فرکانس در 44 هرتز قرار میگیرد . بقیه فرکانس ها میان 5 و44 با فواصل مساوی قرار میگیرند . مقادیر threshold نیز دقیقا وسط هر دو فرکانس در نظر گرفته میشود .

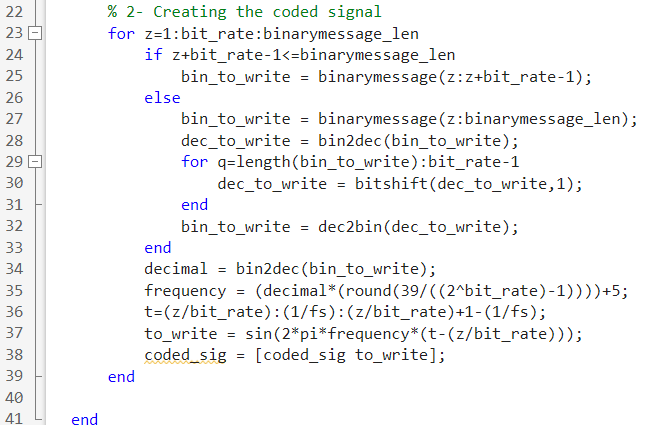
با توجه به روش بیان شده داریم :

Frequency = 5 + decimal number \*

در هر لوپ فرکانس متناظر را محاسبه میکنیم و سیگنالی با آن فرکانس به مدت یک ثانیه تولید میکنیم . در هر لوپ یک ثانیه دیگر از این سیگنال ایجاد شده و به سیگنال قبلی اضافه میشود تا در انتهای لوپ تمام سیگنال مورد نظر درست شده باشد .

اسکریپت این تابع :

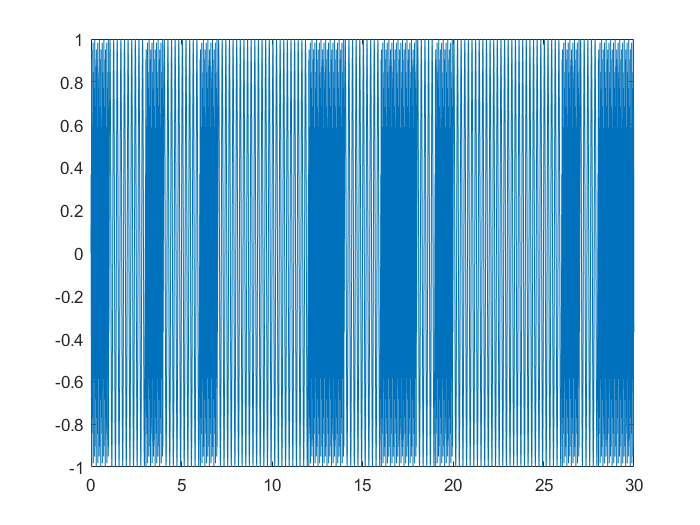




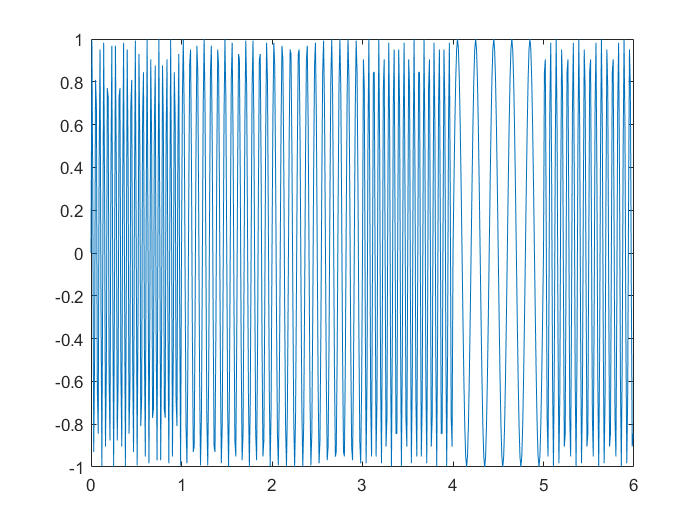
**تمرین 2-3)**

خروجی تابع coding\_freq برای پیام signal :

**Bit rate = 1**



**Bit rate = 5**

****

**تمرین 2-4 )**

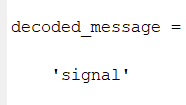
ابتدا باید threshold ها را با روش توضیح داده شده در تمرین قبل به دست آوریم . مقادیر را در یک آرایه نگهداری میکنیم. به سراغ سیگنال ورودی تابع رفته و یک حلقه ایجاد میکنیم که در هر تکرار ، یک ثانیه از سیگنال رو مورد بررسی قرار میدهد . از این یک ثانیه تبدیل فوریه گرفته و و محل قله این تبدیل فوریه را با استفاده از دستور max می یابیم . حال فرکانس متناظر با این محل قله را پیدا میکنیم . این فرکانس همان فرکانسی است که پیام در آن نهفته است . سپس این فرکانس را در یک حلقه با مقادیر threshold مقایسه میکنیم . اولین threshold ای که فرکانس از آن کمتر باشد ، threshold مد نظر است . با توجه به روش انتخاب فرکانس ها و threshold ها میدانیم :

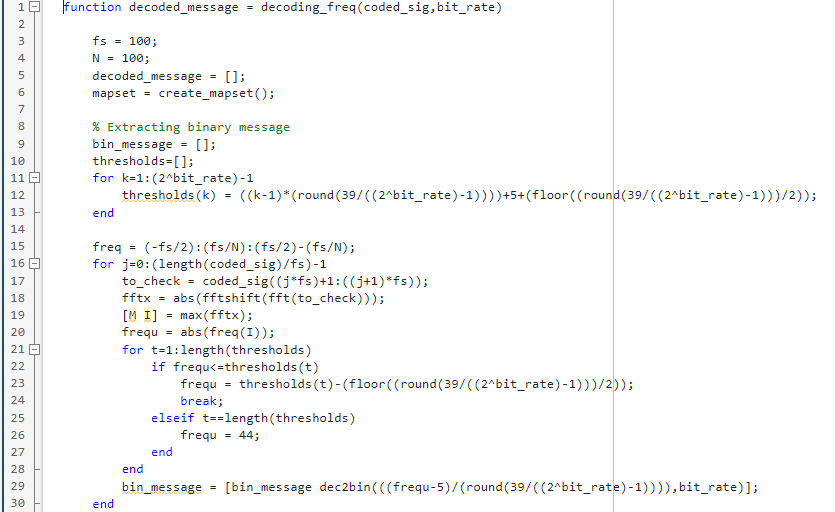
Frequency = threshold –

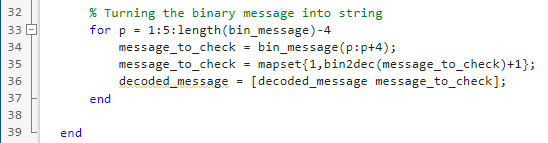
با رابطه بالا فرکانس خود را اصلاح میکنیم . هدف از این اصلاح و استفاده از threshold ، جلوگیری از تغییر پیام توسط نویز های احتمالی روی سیگنال است . حال با استفاده از فرکانس اصلاح شده میتوانیم از رابطه ی ذکر شده در تمرین قبل استفاده کنیم و عدد دسیمال متناظر با این فرکانس را به دست آوریم .

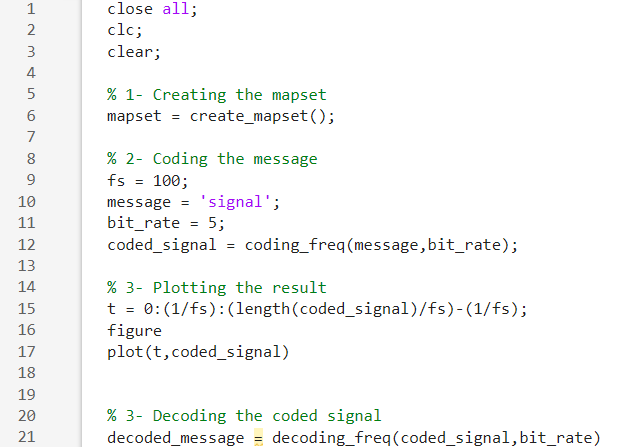
پس از به دست آوردن عدد دسیمال ، این عدد را به یک عدد باینری تبدیل میکنیم و به یک آرایه اضافه میکنیم . حلقه تا رسیدن به پایان سیگنال ادامه پیدا میکند و در پایان معادل باینری پیام نهفته شده به دست آمده است . حال با استفاده از mapset باید پیام باینری را به کاراکتر تبدیل کنیم .

حاصل تست تابع برای bit rate های 1 تا 5 :



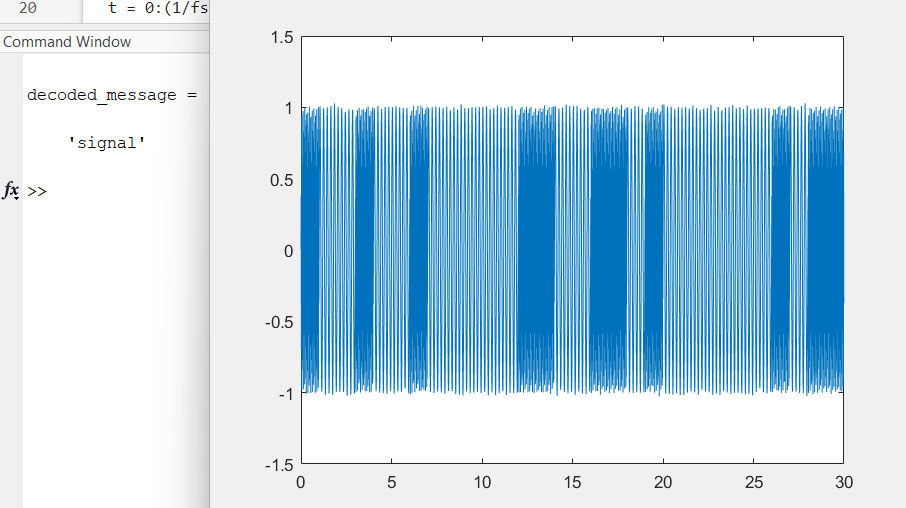
اسکریپت این تابع :

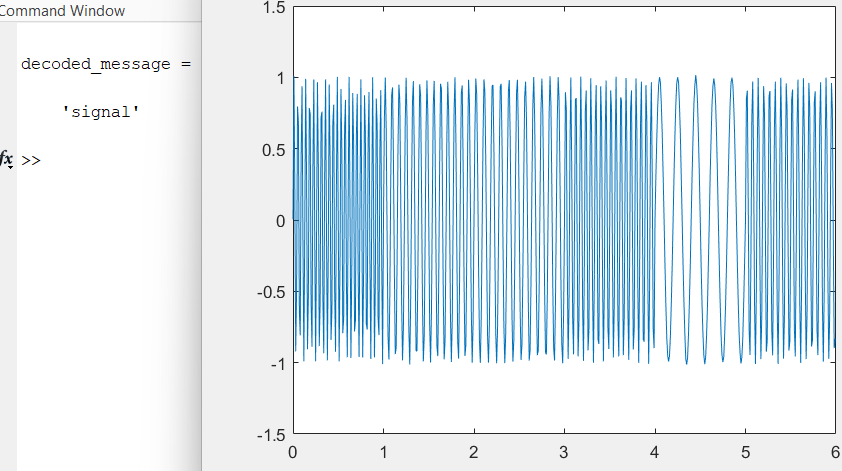


اسکریپت استفاده شده برای تمارین یک تا چهار : 

**تمرین 5-2)**

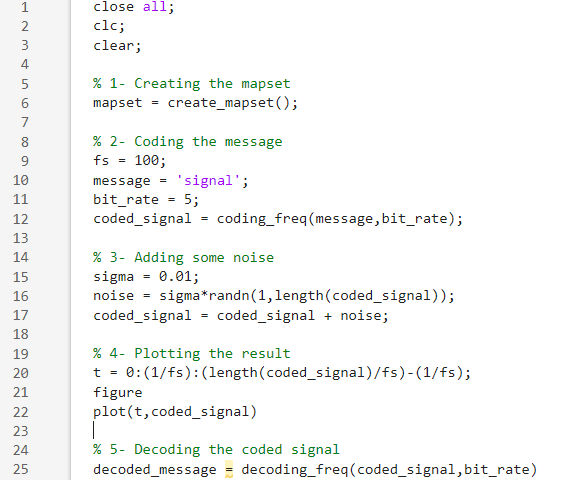
**Bit rate = 1**

****

**Bit rate = 5**

پس از اعمال نویز همچنان پیام signal به خوبی استخراج می شود .

اسکریپت این تمرین :

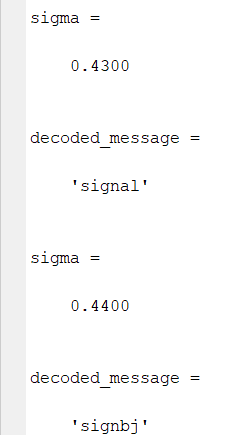


**تمرین 2-6)**

در این تمرین به تدریج قدرت نویز را از 0.01 تا 1 افزایش داده و خروجی ها را مقایسه میکنیم .( δ را تغییر میدهیم)

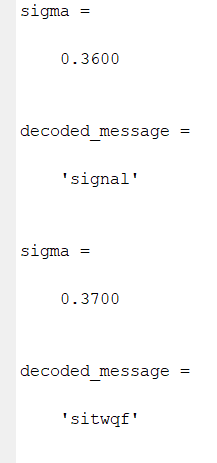
تا δ = 0.43 پیام به درستی استخراج میشود اما با بیشتر شدن قدرت نویز از این مقدار ، دیگر قادر به رمزگشایی پیام نیستیم .

Bit rate = 1

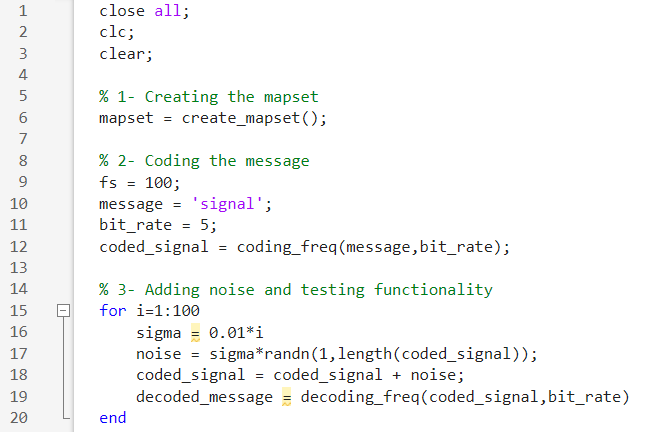


تا δ = 0.36 پیام به درستی استخراج میشود اما با بیشتر شدن قدرت نویز از این مقدار ، دیگر قادر به رمزگشایی پیام نیستیم .

bit rate = 5



همان طور که مشاهده میشود سیگنال با bit rate = 1 نسبت به سیگنال با bit rate = 5 به نویز مقاوم تر است . که مطابق با مطلب بیان شده در مقدمه است .

اسکریپت این تمرین :

**تمرین 2-7 )**

بیشترین واریانس نویز که bit rate = 5 به آن مقاوم بود تقریبا 0.1296 بود . برای bit rate = 1 این عدد تقریبا 0.1849 بود .

**تمرین 2-8)**

انجام شد .

**تمرین 2-9)**

اگر نرخ نمونه برداری را افزایش دهیم ، خود به خود پهنای باند نیز افزایش می یابد . میدانیم : 

با افزايش نرخ نمونه برداري ، رزولوشن فركانسي هيچ تغييري نمي كند ، اما با توجه به رابطه بالا ، پهناي باند

افزايش مي يابد. يعني ما دسترسي به پهناي باند بيشتري داريم اما با توجه به صورت تمرين ، مجاز به استفاده از تمام اين پهناي باند نيستيم . بنابرين اين حالت هيچ فرقي با حالت قبل ندارد و مقاومت نسبت به نويز با حالت قبل يكسان است . شايان ذكر است كه اگر صورت سوال پهناي باند را محدود نمي كرد ، با افزايش نرخ نمونه برداري ، پهناي باند هم خود به خود افزايش مي يافت و مي توانستيم مقاومت نسبت به نويز را افزايش بدهيم. اما اين اتفاق نمي افتد زيرا پهناي باند توسط صورت تمرين محدود شده است و پهناي باند مصرفي از پهناي باند قابل استفاده كمتر است.