تمرین نرم‌افزاری درس DSP

- ابتدا یک شاخه برای تمرین‌های نرم‌افزاری ایجاد کنید
- توابع داده شده به همراه این تمرین را در آن شاخه کپی کنید
- برای هر قسمت از تمرین‌ها که نیاز به نوشتن برنامهٔ Matlab دارد، یک فایل m. با عنوان مثلاً ex_1_a.m ایجاد کنید
- پاسخ شما به صورت یک فایل pdf. و یک فایل zip. خواهد بود که حتماً شامل موارد زیر است:
  - برنامه‌هایی که شما نوشته اید
  - خروجی‌های نرم‌افزار
  - **تحلیل پاسخ داده شده**
  - در صورت نیاز، انجام محاسبات تئوری و تأیید نتیجهٔ به دست آمده
- برای دریافت تصویر از نمودارهای خواسته شده، پس از رسم نمودار، یا تصویر رسم شده را با پسوند jpg. ذخیره کنید (File>>Save) و آن را در گزارش خود بیاورید و یا تصویر را کپی کنید (Edit>>Copy Figure) و آن را در گزارش خود بچسبانید. از عکس‌گرفتن از صفحهٔ نمایش یا استفاده از Print Screen خودداری کنید!
- در کدهای Matlab، نحوهٔ نام‌گذاری متغیرها، استفادهٔ صحیح از indent، استفاده از comment و موارد مشابه اهمیت دارد.

بخش اول: تبدیل فوریهٔ گسسته در زمان (DTFT)

تمرین ۱

مطمئن شوید که در شاخهٔ مربوط به تمرین‌ها (که توابع داده شده در آن قرار دارد) هستید. برنامهٔ زیر را در Matlab اجرا کنید:

```
n=-3:10;
x1=zeros(size(n));
x1(5:9)=1;

w=-2.5*pi:0.01:2.5*pi;
X=dtft(n,x1,w);
x2=idtft(w,X,n);



subplot(3,2,1);
stem(n,x1)

subplot(3,2,2)
stem(n,real(x2));
hold on
stem(n,imag(x2))

subplot(3,1,2)
plot(w,abs(X))

subplot(3,1,3)
plot(w,angle(X))
```

۱-الف) در مورد متن برنامه و متن تابع `dtft` و `idtft` توضیح دهید.

۱-ب) برنامهٔ ex_01_b.m را اجرا کنید. از شما انتظار می‌رود برنامه‌هایی که تحویل می‌دهید به این شکل باشد.

توجه: در تمام طول انجام تمرین‌ها با استفاده از اینترنت و راهنمای Matlab با طرز کار توابع مختلف Matlab آشنا شوید. در مورد خطوط مهم برنامه و توابع خاصی که از آن استفاده شده است، توضیح مختصری بدهید.

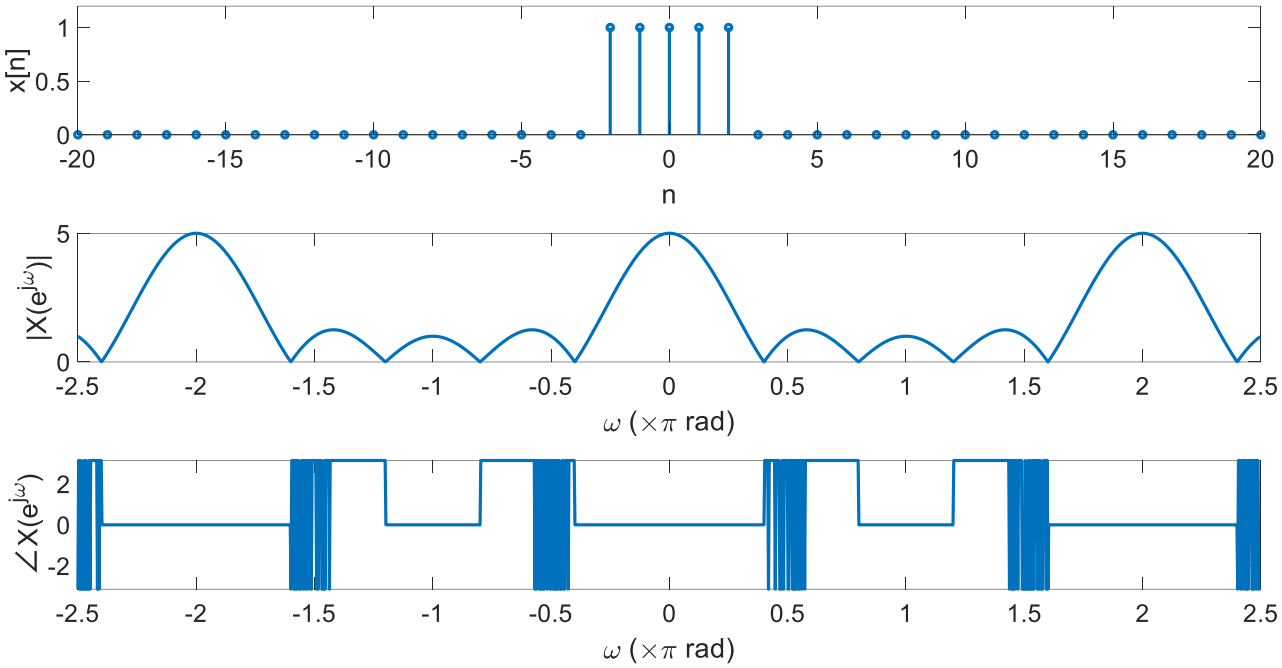
تمرین ۲

برنامهٔ زیر را در Matlab بنویسید و اجرا کنید.

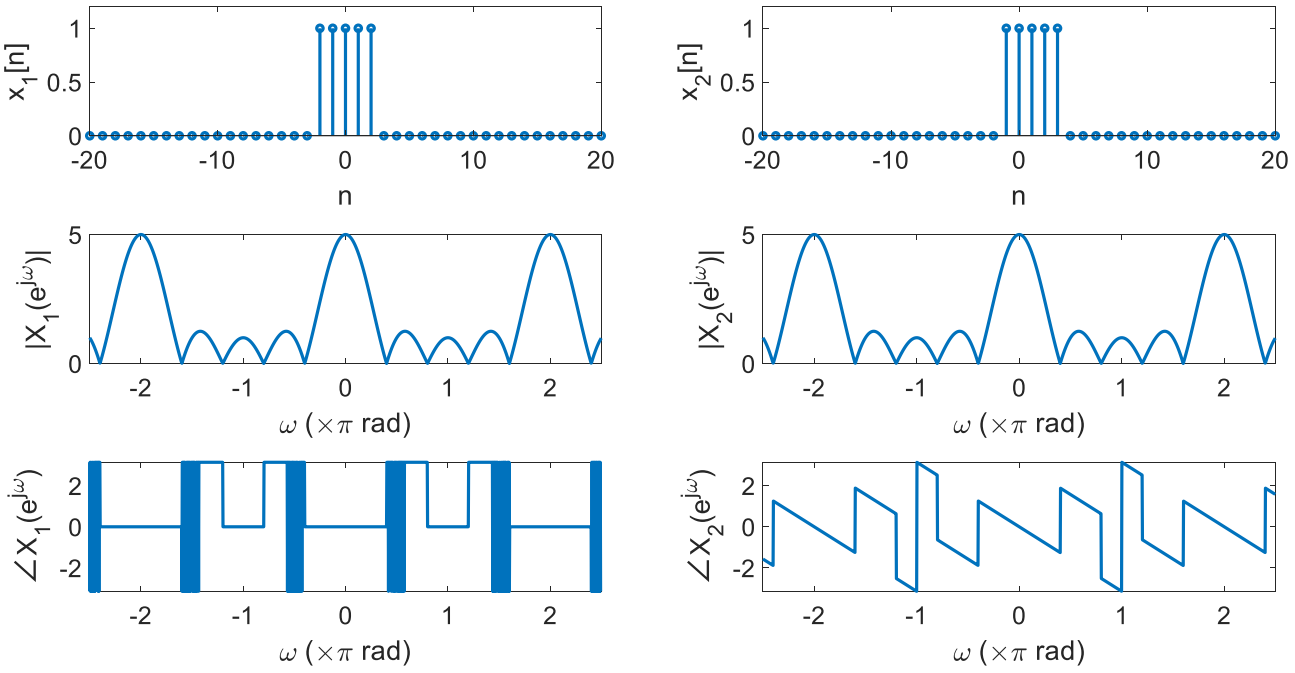
```
% Plotting the spectrum of a discrete-time rectangular pulse
n=-20:20;
w=-2.5*pi:.01:2.5*pi;
x1=zeros(size(n));
x1(n>=-2 & n<=2)=1;
% x1(19:23)=1;
X=dtft(n,x1,w);

subplot(3,1,1)
stem(n,x1,'linewidth',3);
set(gca,'fontsize',24)
axis([-20 20 0 1.2])
xlabel('n')
ylabel('x[n]')

subplot(3,1,2)
plot(w/pi,abs(X),'linewidth',3);
set(gca,'fontsize',24)
xlabel('\omega (\times\pi rad)')
ylabel('|X(e^{j\omega})|')

subplot(3,1,3)
plot(w/pi,angle(X),'linewidth',3);
set(gca,'fontsize',24)
xlabel('\omega (\times\pi rad)')
ylabel('\angle X(e^{j\omega})')
```

۲-الف) در برنامهٔ فوق می‌توانستیم به‌جای عبارت `x1(n>=-2 & n<=2)=1;` در خط پنجم، از عبارت داده شده در خط ششم استفاده کنیم و خروجی برنامه هیچ تغییری نمی‌کرد. چرا؟

۲-ب) در شکل خروجی برنامه که در ابتدای صفحهٔ بعد داده شده منحنی فاز، پرش‌های بسیار زیادی دارد که با آن‌چه در تئوری انتظار داشتیم متفاوت به نظر می‌رسد. این پرش‌ها را توجیه کنید.



اکنون می‌خواهیم خاصیت جابه‌جایی زمانی در تبدیل فوریهٔ گسسته در زمان را بررسی کنیم. برای این منظور، برنامهٔ داده شده در صفحهٔ بعد را در Matlab اجرا کنید. خروجی شکل زیر به دست می‌آید:



۲-ج) نتایج به دست آمده را تحلیل کنید. به طور خاص شیب منحنی فاز را در شکل سمت راست محاسبه کنید و در مورد تمام پرش‌های فاز در این شکل توضیح دهید.
۲-د) این برنامه را به ازای یک جابه‌جایی زمانی دیگر اجرا کنید.

```
% Plotting the spectrum of a discrete-time rectangular pulse
n=-20:20;
w=-2.5*pi:.01:2.5*pi;
x1=zeros(size(n));
x2=zeros(size(n));
x1(n>=-2 & n<=2)=1;
x2(20:24)=1;
X1=dtft(n,x1,w);
X2=dtft(n,x2,w);

subplot(3,2,1)
stem(n,x1,'linewidth',3);
set(gca,'fontsize',24)
axis([-20 20 0 1.2])
xlabel('n')
ylabel('x_1[n]')

subplot(3,2,2)
stem(n,x2,'linewidth',3);
set(gca,'fontsize',24)
axis([-20 20 0 1.2])
xlabel('n')
ylabel('x_2[n]')

subplot(3,2,3)
plot(w/pi,abs(X1),'linewidth',3);
set(gca,'fontsize',24)
xlabel('\omega (\times\pi rad)')
ylabel('|X_1(e^{j\omega})|')

subplot(3,2,5)
plot(w/pi,angle(X1),'linewidth',3);
set(gca,'fontsize',24)
xlabel('\omega (\times\pi rad)')
ylabel('\angle X_1(e^{j\omega})')

subplot(3,2,4)
plot(w/pi,abs(X2),'linewidth',3);
set(gca,'fontsize',24)
xlabel('\omega (\times\pi rad)')
ylabel('|X_2(e^{j\omega})|')

subplot(3,2,6)
plot(w/pi,angle(X2),'linewidth',3);
set(gca,'fontsize',24)
xlabel('\omega (\times\pi rad)')
ylabel('\angle X_2(e^{j\omega})')
```

متن برنامه مربوط به قسمت‌های ج و د

بخش دوم: نمونه برداری

تمرین ۳

بارها گفته‌ایم که به دلیل ماهیت گسستهٔ کامپوترها، توابع Matlab همگی بر روی سیگنال‌ها و سیستم‌های گسسته در زمان عمل می‌کنند. این یعنی در واقع، حتی وقتی در Matlab مشغول کار بر روی سیگنال‌های پیوسته در زمان هستیم، عملاً سعی می‌کنیم با بزرگ گرفتن فرکانس نمونه‌برداری (کوچک گرفتن فاصلهٔ زمانی بین نمونه‌ها) یک سیگنال پیوسته را با یک سیگنال گسسته تقریب بزنیم.

در این تمرین این موضوع را بررسی می‌کنیم. برنامهٔ زیر عیناً برنامه‌ای است که در تمرین اول اجرا کردید، فقط خط ۲۳ آن تغییر کرده است تا نمودار از ابتدا تا انتها رسم شود.

```
% Plotting the spectrum of a rectangular pulse

t=-10:.01:10;
w=-4*pi:.01:4*pi;
x=rectpuls(t/4);
X=ctft(t,x,w);


subplot(2,1,1)
plot(t,x,'linewidth',3);
set(gca,'fontsize',24)
axis([-10 10 0 1.2])
xlabel('t (sec)')
ylabel('x(t)')


subplot(2,1,2)
plot(w,real(X),'linewidth',3);
hold on
plot(w,imag(X),'linewidth',3);
set(gca,'fontsize',24)
legend('Real part','Imaginary part')
axis([min(w) max(w) -1 4.5])
xlabel('\omega (rad/s)')
ylabel('H(j\omega)')
```

برنامه را ذخیره و اجرا کنید، قاعدتاً همان شکل داده شده در تمرین اول را مشاهده خواهید کرد. اکنون خط ۴ برنامه را به شکل زیر تغییر دهید و برنامه را مجدداً اجرا کنید. دقت کنید که این برنامه برای اجرا کمی بیشتر از قبل به زمان احتیاج دارد.

```
w=-400*pi:.01:400*pi;
```

با استفاده از فابلیت zoom در نمودارها، مطمئن شوید که طیف به دست آمده در بازهٔ $-4\pi \leq \omega \leq 4\pi$ همان شکل قبلی است.

۳-الف) نتایج به دست آمده را به دقت تحلیل کنید. مخصوصاً در مورد بازه‌های تکرار طیف در شکل به دست آمده توضیح دهید.

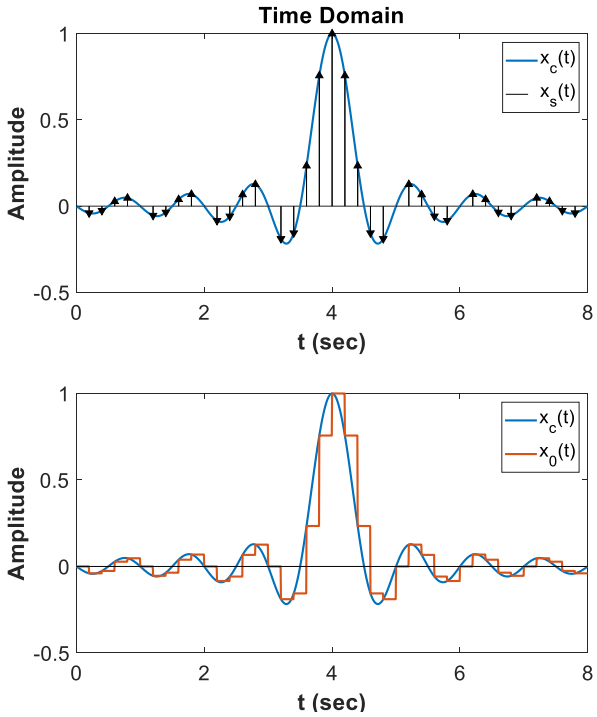
تمرین ۴

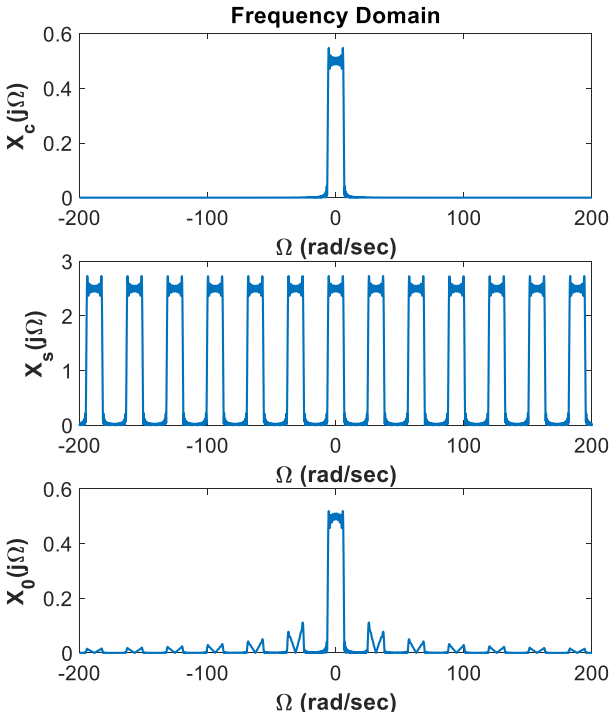
در این تمرین اثر فیلتر بازسازی غیر ایده‌آل و افزایش نرخ نمونه برداری را مشاهده می‌کنیم.

برنامهٔ زیر را در Matlab اجرا کنید.

```
clear;clc;
dt=0.005;
t=0:dt:8-dt;
w=-200:0.1:200;
xc=sinc((t-4)*2);
Xc=ctft(t,xc,w);
M=40;
T=M*dt;
x=xc(1:M:end);
Xs=dtft(0:length(x)-1,x,w*T);
x0=kron(x,ones(1,M));
X0=ctft(t,x0,w);

subplot(2,2,1);
plot(t,xc,'linewidth',2);
hold on
n=0:length(x)-1;
nPositive=n(x>1e-10);
nNegative=n(x<-1e-10);
stem(n*T,x,'k','Marker','none')                  %20
stem(nPositive*T,x(x>1e-10),'k^','filled');      %21
stem(nNegative*T,x(x<-1e-10),'kv','filled');     %22
set(gca,'fontsize',20)
xlabel('t (sec)','fontweight','bold')
ylabel('Amplitude','fontweight','bold')
legend('x_c(t)','x_s(t)')
title('Time Domain')

subplot(2,2,3)
plot(t,xc,'linewidth',2);
hold on
plot(t,x0,'linewidth',2)
line([min(t) max(t)],[0 0],'color','k')
set(gca,'fontsize',20)
xlabel('t (sec)','fontweight','bold')
ylabel('Amplitude','fontweight','bold')
legend('x_c(t)','x_0(t)')

subplot(3,2,2);
plot(w,abs(Xc),'linewidth',2)
set(gca,'fontsize',20)
xlabel('\Omega (rad/sec)','fontweight','bold')
ylabel('|X_c(j\Omega)|','fontweight','bold')
title('Frequency Domain')

subplot(3,2,4)
plot(w,abs(Xs),'linewidth',2)
set(gca,'fontsize',20)
xlabel('\Omega (rad/sec)','fontweight','bold')
ylabel('|X_s(j\Omega)|','fontweight','bold')

subplot(3,2,6)
plot(w,abs(X0),'linewidth',2)
set(gca,'fontsize',20)
xlabel('\Omega (rad/sec)','fontweight','bold')
ylabel('|X_0(j\Omega)|','fontweight','bold')
```

شکل خروجی مطابق شکل زیر خواهد بود:





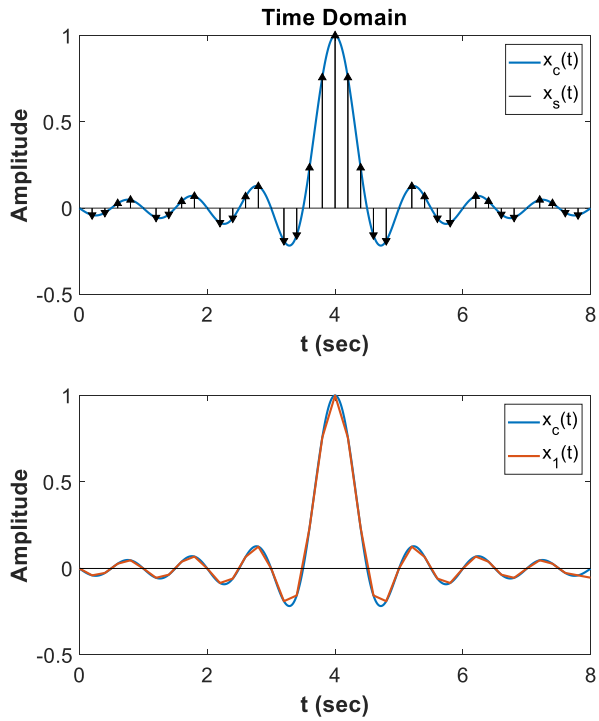
۴-الف) خروجی را در حوزهٔ زمان و فرکانس تحلیل کنید. مقدار فرکانس نمونه برداری را از روی شکل زمانی، شکل فرکانسی و متن برنامه به دست آورید. پاسخ فرکانسی فیلتر بازسازی را روی نمودار $X_s(j\Omega)$ رسم کنید.

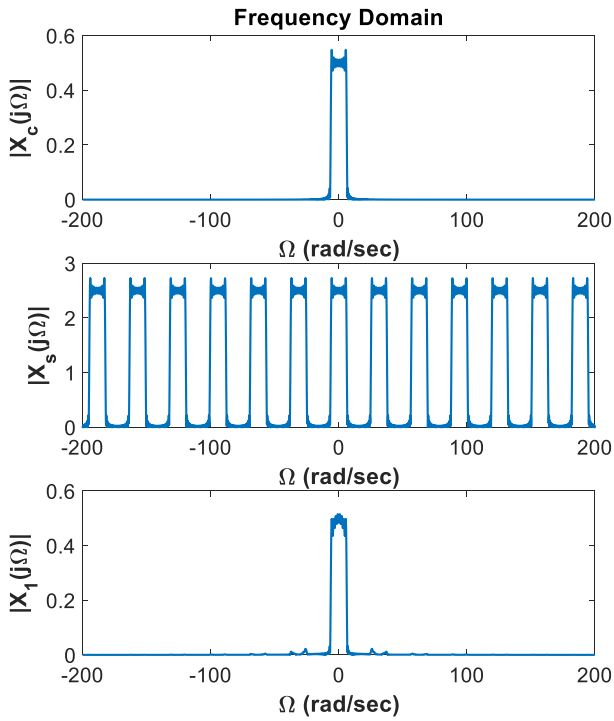
۴-ب) در مورد متن برنامه به دقت توضیح دهید، مخصوصاً:

- نحوهٔ محاسبهٔ طیف سیگنال نمونه برداری شده، $X_s(j\Omega)$ با استفاده از تابع `dtft`
- نحوهٔ محاسبهٔ سیگنال بازسازی شده با استفاده از zero-order hold در حوزهٔ زمان
- کاربرد خطوط 20 تا 22

۴-ج) برنامه را طوری تغییر دهید که فرکانس نمونه‌برداری برابر با 20π رادیان بر ثانیه باشد

۴-د) برنامهٔ اولیه را طوری تغییر دهید که به‌جای zero-order hold از درون‌یابی خطی استفاده کند.





بخش سوم: تبدیل فوریهٔ گسسته (DFT)

در این بخش با DFT آشنا می‌شویم و چند مورد از ویژگی‌های آن را بررسی می‌کنیم. توجه کنید که در Matlab برای محاسبهٔ DFT از تابع `fft` استفاده می‌کنیم.

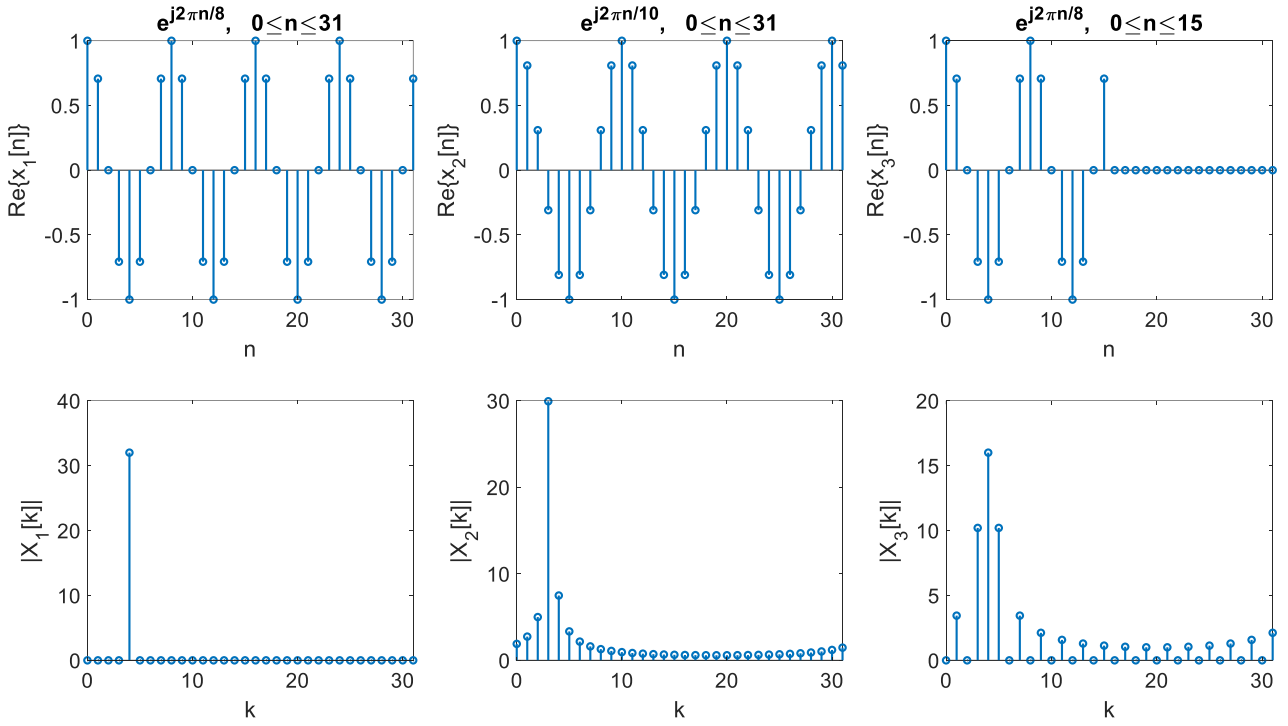
تمرین ۵

در این تمرین با DFT آشنا می‌شویم و چند مورد از تفاوت‌های آن با تبدیل فوریه را مشاهده می‌کنیم.

برنامهٔ زیر را در Matlab اجرا کنید:

```
n=0:31;
x1=exp(1j*2*pi/8*n);
x2=exp(1j*2*pi/10*n);
x3=x1;
x3(17:end)=0;


X1=fft(x1,32);
X2=fft(x2,32);
X3=fft(x3,32);

subplot(2,3,1)
stem(0:31,real(x1),'linewidth',2)
set(gca,'fontsize',20)
xlabel('n')
ylabel('Re\{x_1[n]\}')
title('e^{j2\pin/8},   0\leqn\leq31');

subplot(2,3,2)
stem(0:31,real(x2),'linewidth',2)
set(gca,'fontsize',20)
xlabel('n')
ylabel('Re\{x_2[n]\}')
title('e^{j2\pin/10},   0\leqn\leq31');

subplot(2,3,3)
stem(0:31,real(x3),'linewidth',2)
set(gca,'fontsize',20)
xlabel('n')
ylabel('Re\{x_3[n]\}')
title('e^{j2\pin/8},   0\leqn\leq15');

subplot(2,3,4)
stem(0:31,abs(X1),'linewidth',2)
set(gca,'fontsize',20)
xlabel('k')
ylabel('|X_1[k]|')

subplot(2,3,5)
stem(0:31,abs(X2),'linewidth',2)
set(gca,'fontsize',20)
xlabel('k')
ylabel('|X_2[k]|')

subplot(2,3,6)
stem(0:31,abs(X3),'linewidth',2)
set(gca,'fontsize',20)
xlabel('k')
ylabel('|X_3[k]|')
```

خروجی نشان داده شده در شکل زیر ایجاد می‌شود:



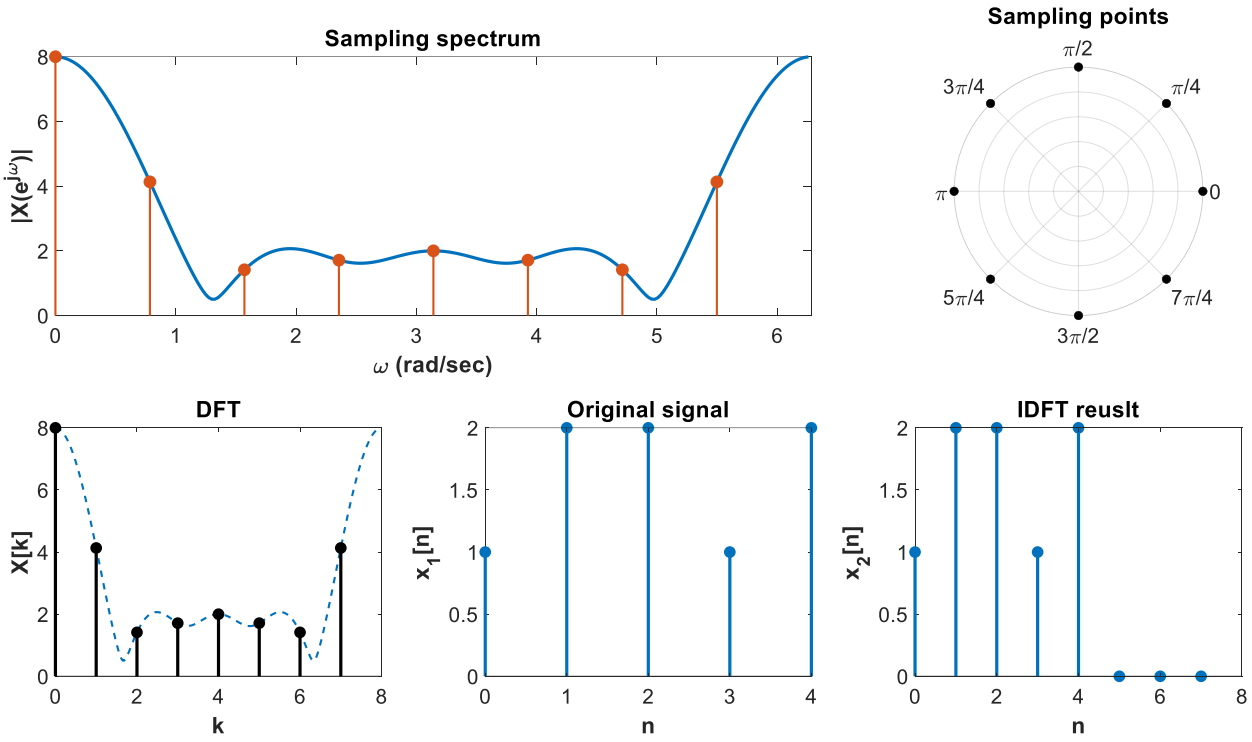
در مورد نمودارهای به دست آمده به طور کامل توضیح بدهید.

تمرین ۶

در این تمرین نشان می‌دهیم که DFT مانند نمونه‌برداری از طیف و احتمالاً به وجود آمدن time-aliasing است و در آخر، یک روش سریع برای محاسبهٔ طیف سیگنال معرفی می‌کنیم. برنامهٔ زیر را در Matlab اجرا کنید:

```
x1=[1 2 2 1 2];
dw=2*pi/256;
w=0:dw:2*pi-dw;
Xw=dtft(0:length(x1)-1,x1,w);
wk=w(1:32:end);
Xk=Xw(1:32:end);
x2=ifft(Xk);

subplot(2,3,[1 2])
plot(w,abs(Xw))
hold on
stem(wk,abs(Xk))

subplot(2,3,3)
polarplot(wk,ones(size(wk)))

subplot(2,3,4)
stem(0:length(Xk)-1,abs(Xk))
hold on
plot(w/2/pi*8,abs(Xw),'--')

subplot(2,3,5)
stem(0:length(x1)-1,x1)

subplot(2,3,6)
stem(0:length(x2)-1,real(x2))
```

۶-الف) این برنامه را طوری تغییر دهید که خروجی آن دقیقاً مشابه شکل زیر باشد.



۶-ب) برنامه را طوری تغییر دهید که به جای 8 نمونه، از طیف در 4 نقطه نمونه برداری شود.

۶-ج) با توجه به این‌که می‌دانیم محاسبهٔ DFT با الگوریتم FFT بسیار سریع است، تابع `dtft` را با استفاده از FFT مجدداً بنویسید.

- نام تابع جدید را dtft_new بگذارید.
- مقادیر n صعودی هستند ولی ممکن است از 0 شروع نشوند و پشت سر هم نباشند.
- برای راحتی فرض کنید که مقادیر w حتماً با فاصلهٔ یکسان بین صفر و 2π هستند.

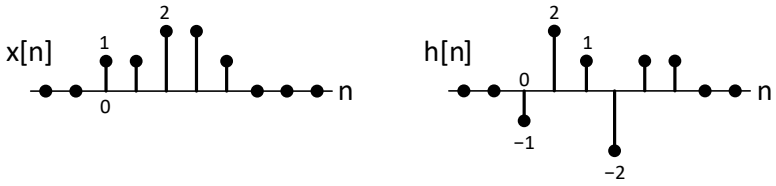
۶-د) با استفاده از تابع `dtft` که قبلاً داشتیم و تابع جدیدی که نوشته اید، برنامهٔ زیر را اجرا کنید. حتماً در برنامه تغییرات لازم را بدهید تا خروجی به شکل قابل قبول در آید.

```
n=[-2 0 1 6 10:50];
x=[1 1 1 1 1 zeros(1,20) zeros(1,20)];

dw=pi/65536;
w=0:dw:2*pi-dw;
tic
X1=dtft(n,x,w);
fprintf('dtft takes %f seconds\n',toc)
tic
X2=dtft_new(n,x,w);
fprintf('dtft_new takes %f seconds\n',toc)

subplot(1,2,1)
plot(w,abs(X1))
hold on
plot(w,abs(X2))

subplot(1,2,2)
plot(w,angle(X1))
hold on
plot(w,angle(X2))
```

تمرین ۷

در این تمرین قصد داریم ویژگی کانولوشن چرخشی برای DFT را بررسی کنیم.

۷-الف) با استفاده از Matlab، حاصل کانولوشن چرخشی دو سیگنال زیر را روی هشت نقطه به سه روش محاسبه کنید:



- با استفاده از تابع `cconv`
- با استفاده از تابع `conv` و اعمال time-aliasing
- با استفاده از DFT

۷-ب) حاصل کانولوشن چرخشی همان دو سیگنال را این‌بار روی 16 نقطه و با همان سه روش محاسبه کنید.

تمرین ۸

در این تمرین ادعایی که بارها در درس مطرح کردیم را صحت سنجی می‌کنیم:

DFT را می‌توان با استفاده از الگوریتم FFT خیلی سریع و مؤثر محاسبه کرد.

برای این منظور از تابع `dft` که به همراه این تمرین در اختیار شما قرار داده شده استفاده می‌کنیم.

۸-الف) برنامهٔ زیر را در Matlab اجرا کنید.

```
clear;
clc;
n=0:100;
x=zeros(size(n));
x(1:10)=1;
N=1024;
M=20;
tic
for i=1:M
    X1=dft(x,N);
end
fprintf('%d %d-point ordinary DFTs take %f seconds\n',M,N,toc);
tic
for i=1:M
    X2=fft(x,N);
end
fprintf('%d %d-point FFTs take %f seconds\n',M,N,toc);

plot(abs(X1))
hold on
plot(abs(X2))
```

۸-ب) برنامه را به ازای مقادیر مختلف N اجرا کنید و میزان صرفه‌جویی در زمان را برحسب N در یک نمودار رسم کنید.