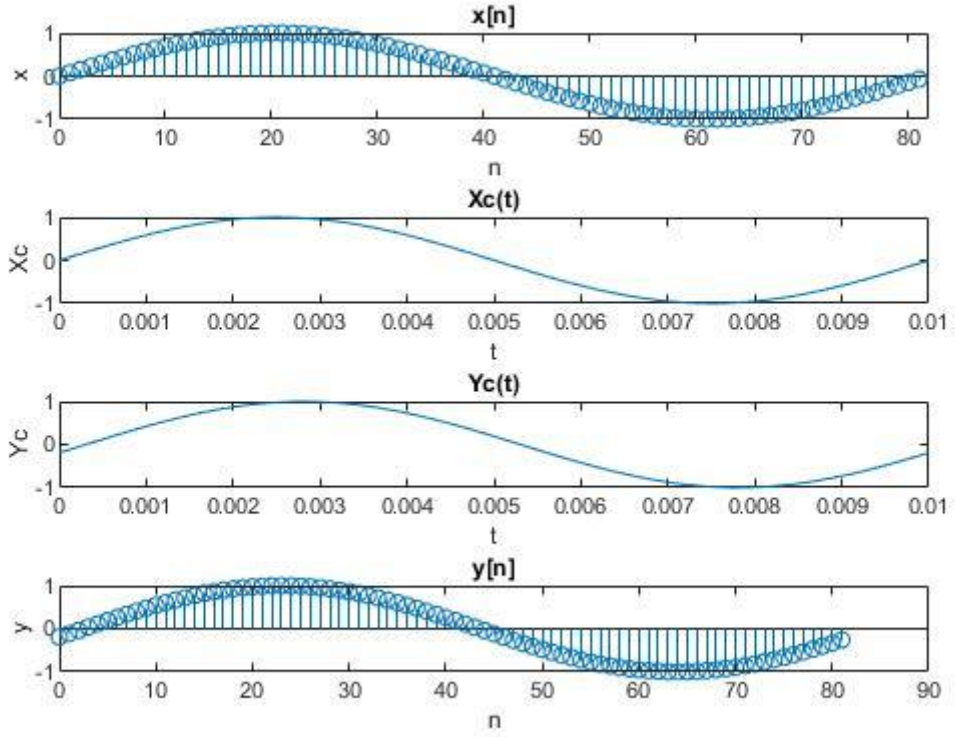
علی آراسته - ۹۶۱۰۱۱۶۵

پردازش سیگنال‌های دیجیتال – تمرین کامپیوتری سری دوم

۱

الف) با استفاده از دستورات متلب نمودارهای خواسته شده را رسم می‌کنیم.



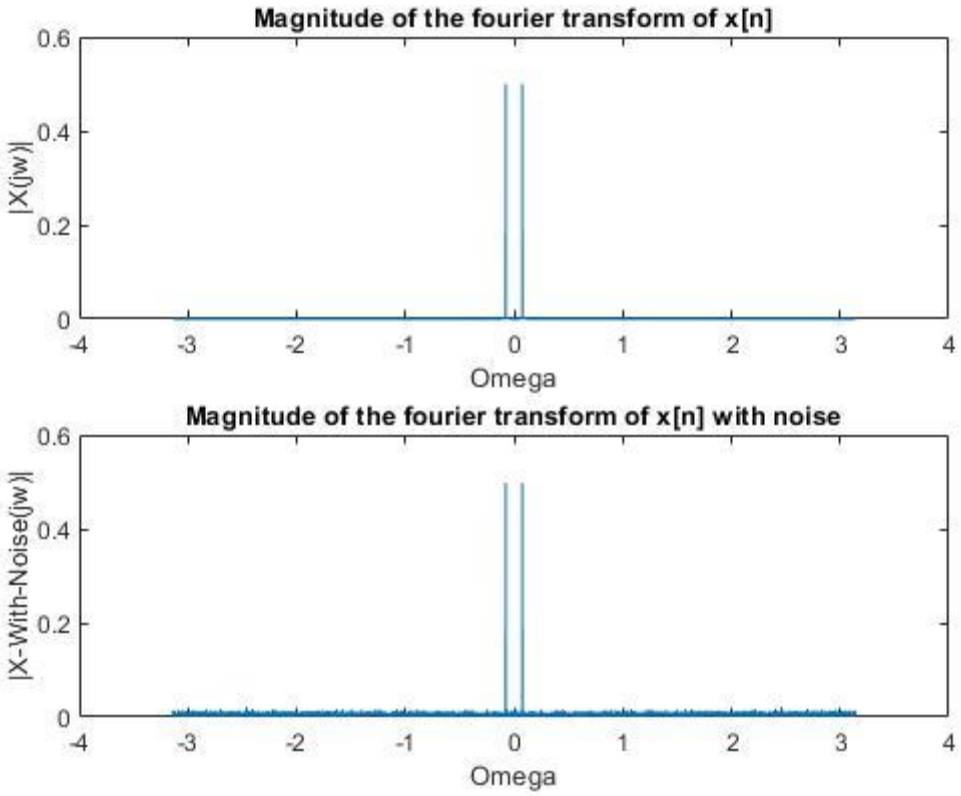

با توجه به نمودارهای رسم شده، هنگامی که سیگنال گسسته را به مقداری غیر صحیح مانند $2.5T$ شیفت می‌دهیم، پوش سیگنال گسسته دقیقا به همان مقدار شیفت می‌خورد؛ اما چون نمونه برداری در مقدارهای صحیح رخ می‌دهد، نمونه‌ها مقادیر کاملا جدیدی می‌شوند.

کد متلب:

```
fs = 8192;
T = 1/fs;
syms t Xc(t) Yc(t)
Xc(t) = sin(2*pi*100*t)
Yc(t) = Xc(t-2.5*T)
time = 0:T:0.01
n = time/T
Xn = Xc(time)
```

```
Yn = Yc(time)
subplot(4,1,1)
stem(n,Xn)
xlim([0,0.01/T])
ylabel('x')
xlabel('n')
title('x[n]')
subplot(4,1,2)
fplot(Xc,[0,0.01])
ylabel('Xc')
xlabel('t')
title('Xc(t)')
subplot(4,1,3)
fplot(Yc,[0,0.01])
ylabel('Yc')
xlabel('t')
title('Yc(t)')
subplot(4,1,4)
xlim([0,0.01/T])
stem(n,Yn)
ylabel('y')
xlabel('n')
title('y[n]')
```

ب) برای سیگنال نویز، یک توزیع یکنواخت با دامنه واحد درنظر می‌گیریم.



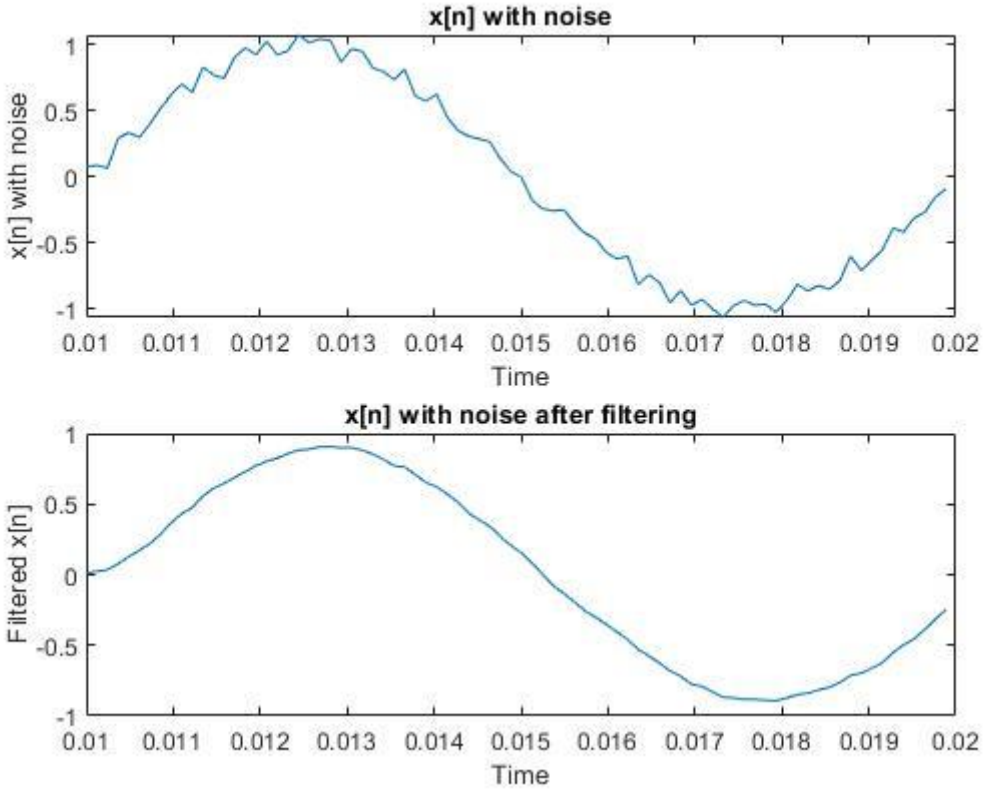

با توجه به نمودارهای رسم شده، در اثر اضافه کردن نویز به سیگنال، در همه فرکانس‌ها، مولفه‌های فرکانسی با دامنه کم و تصادفی ایجاد می‌شود.

کد متلب:

```
fs = 8192;
T = 1/fs;
syms t Xc(t) Yc(t);
Xc(t) = sin(2*pi*100*t);
time = 0:T:1;
Xn = double(Xc(time));
Xn_With_Noise = Xn + (2*rand(1,length(Xn))-1);
Fourie_Of_Xn = T*fftshift(fft(Xn));
Fourie_Of_Xn_With_Noise = T*fftshift(fft(Xn_With_Noise));
Omega = time*2*pi - pi;
subplot(2,1,1)
plot(Omega,abs(Fourie_Of_Xn))
xlabel('Omega')
ylabel('|X(jw)|')
```

```
title('Magnitude of the fourier transform of x[n]')
subplot(2,1,2)
plot(Omega,abs(Fourie_Of_Xn_With_Noise))
xlabel('Omega')
ylabel('|X-With-Noise(jw)|')
title('Magnitude of the fourier transform of x[n] with noise')
```

پ) با استفاده از دستورات متلب نمودارهای خواسته شده را رسم می‌کنیم.



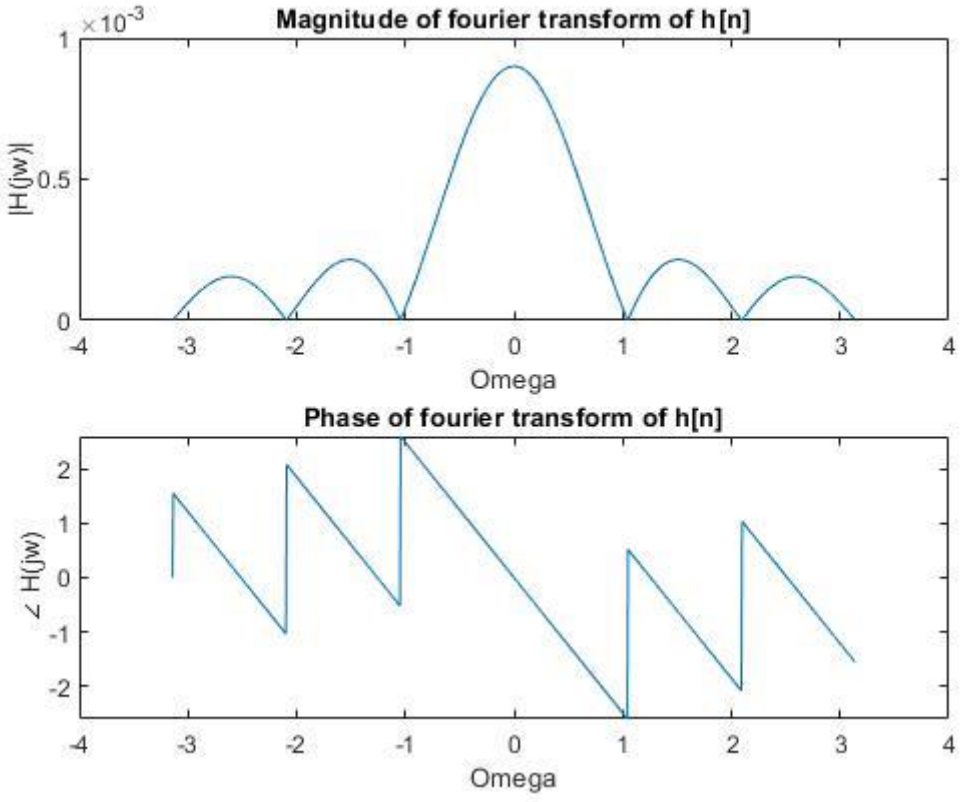

با توجه به نمودارهای رسم شده، استفاده از فیلتر تا حدودی نویز سیگنال را از بین برده است.

کد متلب:

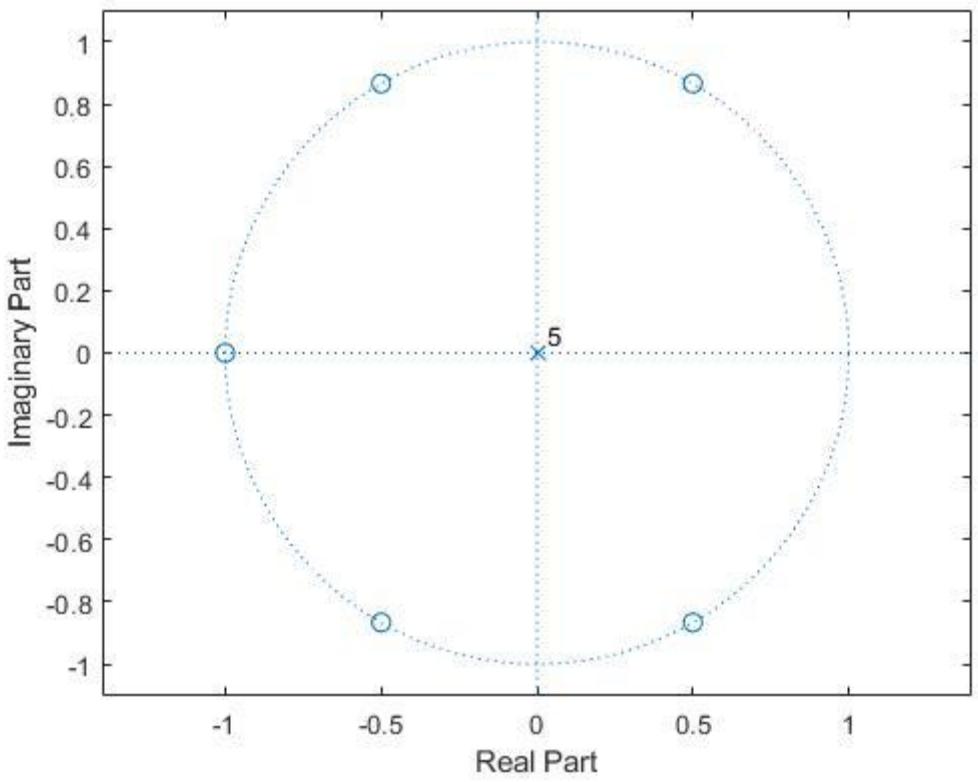
```
fs = 8192;
T = 1/fs;
syms t Xc(t) Yc(t);
Xc(t) = sin(2*pi*100*t);
time = 0.01:T:0.02;
Xn = double(Xc(time));
Xn_With_Noise = Xn + 0.1*(2*rand(1,length(Xn))-1);
Hn = zeros(1,length(time));
```

```
for i=1:6
   Hn(i) = 0.15;
end
Filtered_Xn = conv(Xn_With_Noise,Hn);
subplot(2,1,1)
plot(time,Xn_With_Noise)
xlabel('Time')
ylabel('x[n] with noise')
title('x[n] with noise')
subplot(2,1,2)
plot(time,Filtered_Xn(1,1:floor(0.01/T)+1))
xlabel('Time(second)')
ylabel('Filtered x[n]')
title('x[n] with noise after filtering')
```

ت) با استفاده از دستورات متلب نمودارهای خواسته شده را رسم می‌کنیم.




با توجه به نمودارهای رسم شده، فیلتر داده شده، پایین گذر است. به همین روی اعمال آن بر سیگنال نویز دار باعث حذف نویز می‌شود.

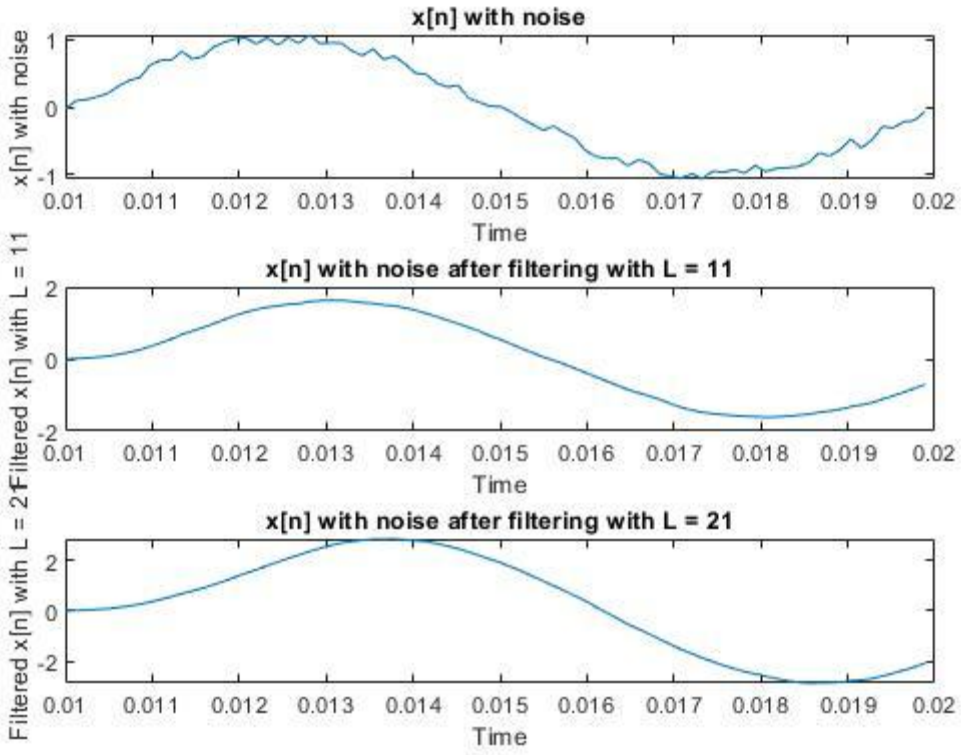



کد متلب:

```
Hn = zeros(1,1000);
for i=1:6
   Hn(i) = 0.15;
end
Fourie_Of_Hn = 1/1000*fftshift(fft(Hn));
Omega = -pi:2*pi/1000:pi-2*pi/1000;
figure();
subplot(2,1,1)
plot(Omega,abs(Fourie_Of_Hn))
xlabel('Omega')
ylabel('|H(jw)|')
title('Magnitude of fourier transform of h[n]')
subplot(2,1,2)
plot(Omega,unwrap(angle(Fourie_Of_Hn)))
xlabel('Omega')
ylabel('\angle H(jw)')
```

```
title('Phase of fourier transform of h[n]')
figure();
D = [0.15 0.15 0.15 0.15 0.15 0.15];
N = 1;
zplane(D,N);
```

ث) با استفاده از دستورات متلب نمودارهای خواسته شده را رسم می‌کنیم.




با توجه به نمودارهای رسم شده، هر چه طول فیلتر افزایش می‌یابد، نویز اعمال شده به سیگنال بیشتر فیلتر می‌شود و در نتیجه خروجی صاف‌تر و هموارتر می‌گردد. همچنین از طرفی با افزایش طول فیلتر، اندازه سیگنال خروجی در هر لحظه بزرگتر می‌شود. به عبارت دیگر از ورودی به خروجی دامنه سیگنال تغییر می‌کند.

ک متلب:

```
fs = 8192;
T = 1/fs;
syms t Xc(t) Yc(t);
Xc(t) = sin(2*pi*100*t);
time = 0.01:T:0.02;
Xn = double(Xc(time));
Xn_With_Noise = Xn + 0.1*(2*rand(1,length(Xn))-1);
```

```
Hn1 = zeros(1,length(time));
for i=1:11
   Hn1(i) = 0.15;
end
Filtered_Xn1 = conv(Xn_With_Noise,Hn1);
Hn2 = zeros(1,length(time));
for i=1:21
   Hn2(i) = 0.15;
end
Filtered_Xn2 = conv(Xn_With_Noise,Hn2);
subplot(3,1,1)
plot(time,Xn_With_Noise)
xlabel('Time')
ylabel('x[n] with noise')
title('x[n] with noise')
subplot(3,1,2)
plot(time,Filtered_Xn1(1,1:floor(0.01/T)+1))
xlabel('Time')
ylabel('Filtered x[n] with L = 11')
title('x[n] with noise after filtering with L = 11')
subplot(3,1,3)
plot(time,Filtered_Xn2(1,1:floor(0.01/T)+1))
xlabel('Time')
ylabel('Filtered x[n] with L = 21')
title('x[n] with noise after filtering with L = 21')
```

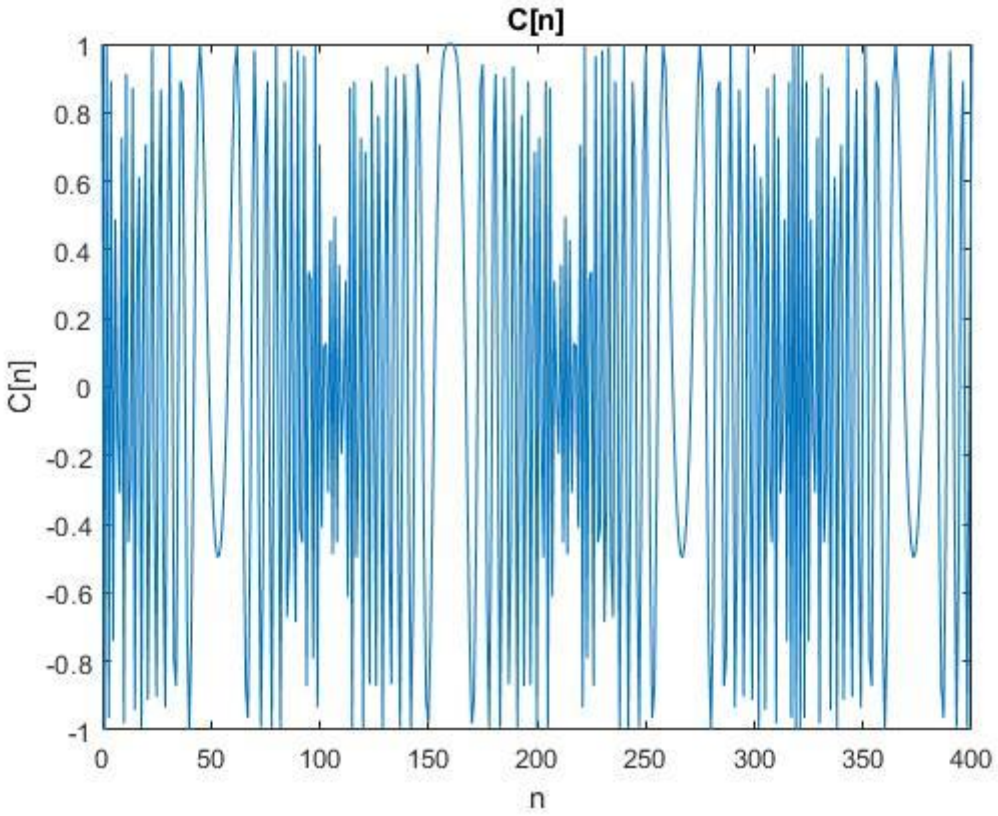
۲

الف) با توجه به صورت مسئله داریم:

$$f_i(t) = \mu t + f_1, \mu = 600kHz, f_1 = 4kHz, 0 < t < 50ms$$

$$4kHz < f_i(t) < 34kHz$$

ب) با استفاده از دستورات متلب، سیگنال گسسته مورد نظر را در حوزه زمان رسم می‌کنیم.




با توجه به روابط نوشته شده، فرکانس لحظه‌ای سیگنال به صورت زیر است:

$$f_i(t) = \mu t + f_1 = 6kHz * t + 4kHz$$

با توجه به این رابطه با افزایش زمان، فرکانس سیگنال باید به طور یکنوا افزایش یابد. اما بر اساس نمودار بدست آمده، فرکانس سیگنال نمونه برداری شده مرتبا افزایش و کاهش می‌یابد. پس پدیده aliasing رخ داده است.

از طرف دیگر بر اساس قضیه نمونه برداری، اگر از یک سیگنال با فرکانس $f_s$ نمونه برداری شود، تنها زمانی سیگنال نمونه برداری شده حاوی تمام اطلاعات سیگنال اصلی است و می‌توان از روی آن سیگنال اصلی را بازسازی کرد که پهای باند سیگنال اصلی کوچکتر از $\frac{f_s}{2}$ باشد؛ یعنی داشته باشیم:

$$X(f) = 0 \qquad |f| > \frac{f_s}{2}$$

طبق این قضیه برای فرکانس نمونه برداری 8kHz فقط برای فرکانس های کوچکتر از 4kHz پدیده aliasing رخ نمی‌دهد. لذا با توجه به بازه فرکانسی سیگنال اصلی، برای تمام فرکانس‌های این سیگنال و در تمام زمان‌ها، پدیده aliasing اتفاق می‌افتد.

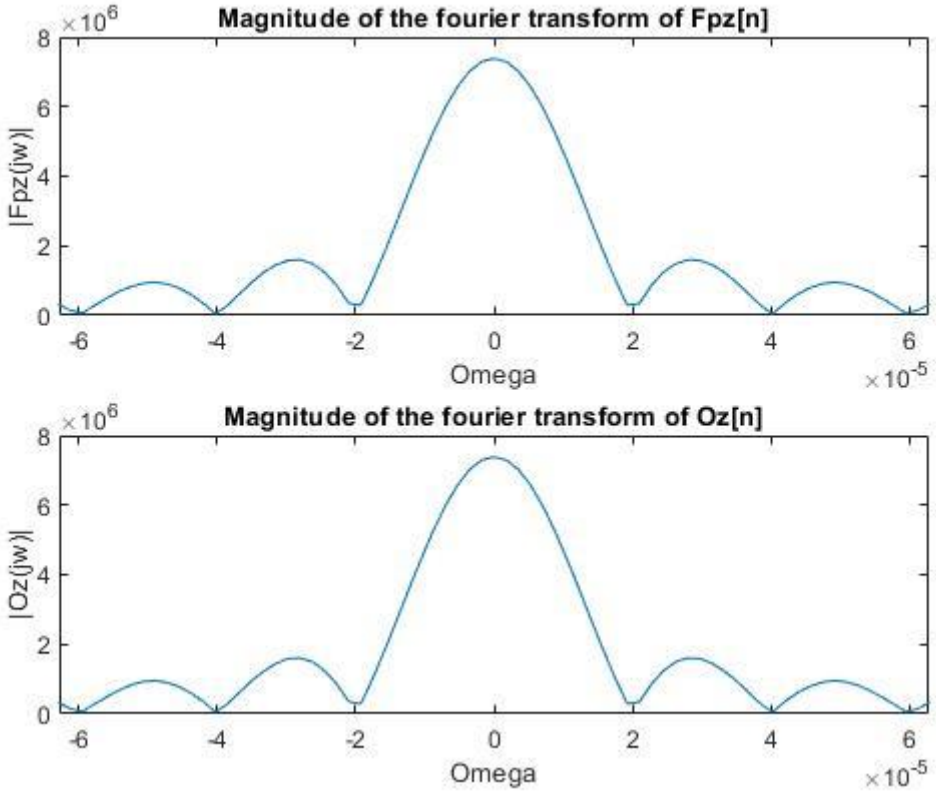
کد متلب:

```
fs = 8000;
T = 1/fs;
M = 600000;
```

```
f1 = 4000;
S = 0;
time = 0:T:0.05;
n = time/T;
syms t C(t);
C(t) = cos(pi*M*t^2+2*pi*f1*t+S);
c = double(C(time));
fplot(t,C(t))
xlabel('n')
ylabel('C[n]')
title('C[n]')
```

۳

الف) با استفاده از دستورات متلب نمودارهای خواسته شده را رسم می‌کنیم.

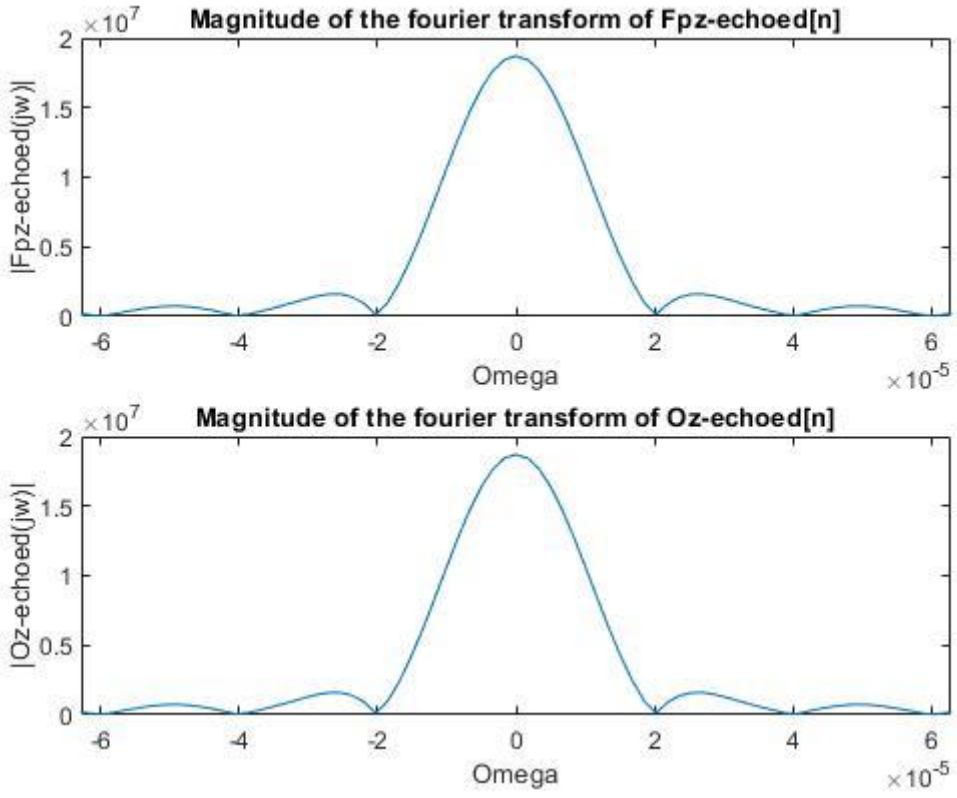



کد متلب:

```
Main();
fs = 100;
Fourie_Of_Fpz = 1/fs*fftshift(fft(Fpz));
```

```
Fourie_Of_Oz = 1/fs*fftshift(fft(Oz));
Omega = -pi:2*pi/length(Fpz):pi-2*pi/length(Fpz);
subplot(2,1,1)
plot(Omega,abs(Fourie_Of_Fpz))
xlim([-2*pi/1e5 2*pi/1e5])
xlabel('Omega')
ylabel('|Fpz(jw)|')
title('Magnitude of the fourier transform of Fpz[n]')
subplot(2,1,2)
plot(Omega,abs(Fourie_Of_Oz))
xlim([-2*pi/1e5 2*pi/1e5])
xlabel('Omega')
ylabel('|Oz(jw)|')
title('Magnitude of the fourier transform of Oz[n]')
```

ب ) با استفاده از دستورات متلب نمودارهای خواسته شده را رسم می‌کنیم.




کد متلب:

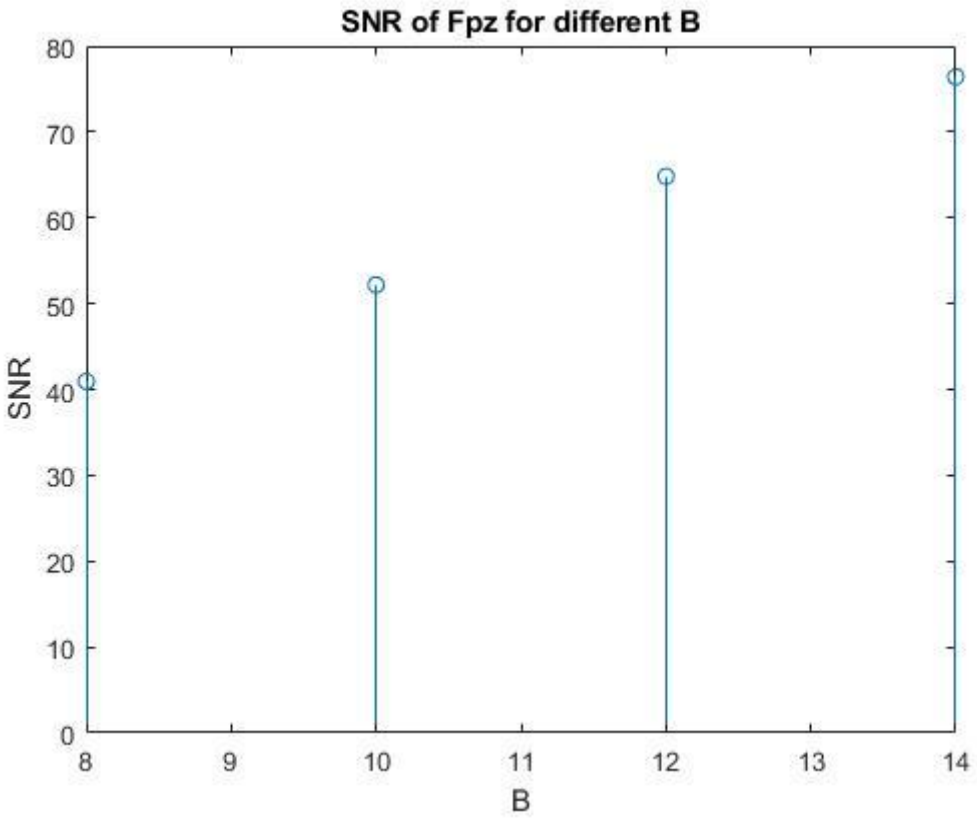
```
Main();
```

```
Aplha = 0.7;
N = 3;
M = 44100;
Fpz_echoed = zeros(1,length(Fpz) + N*M);
for i = 1:length(Fpz_echoed)
   sum = 0;
   for k = 0:N
      if (i-k*M>0 && i-k*M<length(Fpz))
         sum = sum + Aplha^k*Fpz(i-k*M);
      end
   end
   Fpz_echoed(i) = sum;
end
Oz_echoed = zeros(1,length(Oz) + N*M);
for i = 1 : length(Oz_echoed)
   sum = 0;
   for k = 0:N
      if (i-k*M>0 && i-k*M<length(Oz))
         sum = sum + Aplha^k*Oz(i-k*M);
      end
   end
   Oz_echoed(i) = sum;
end
Fourie_Of_Fpz_echoed = 1/fs*fftshift(fft(Fpz_echoed));
Fourie_Of_Oz_echoed = 1/fs*fftshift(fft(Oz_echoed));
Omega = -pi:2*pi/length(Fpz_echoed):pi-2*pi/length(Fpz_echoed);
subplot(2,1,1)
plot(Omega,abs(Fourie_Of_Fpz_echoed))
xlim([-2*pi/1e5 2*pi/1e5])
xlabel('Omega')
```

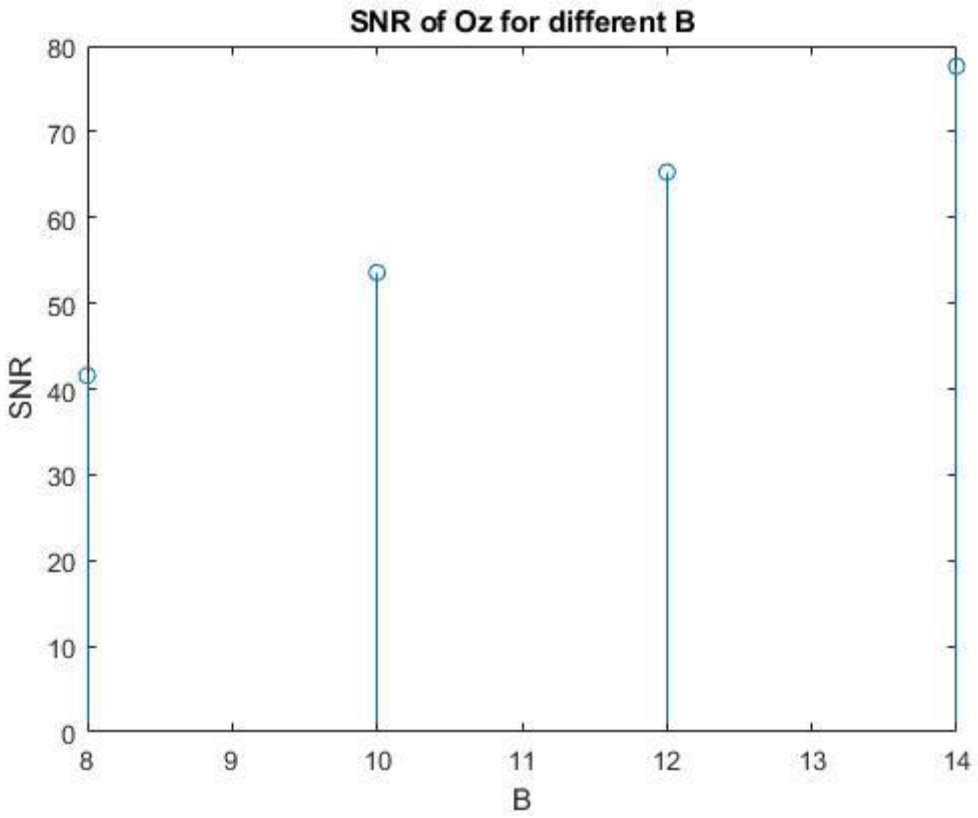
```
ylabel('|Fpz-echoed(jw)|')
title('Magnitude of the fourier transform of Fpz-echoed[n]')
subplot(2,1,2)
plot(Omega,abs(Fourie_Of_Oz_echoed))
xlim([-2*pi/1e5 2*pi/1e5])
xlabel('Omega')
ylabel('|Oz-echoed(jw)|')
title('Magnitude of the fourier transform of Oz-echoed[n]')
```

پ) با استفاده از دستورات متلب نمودارهای خواسته شده را رسم می‌کنیم.

Fpz:




Oz:



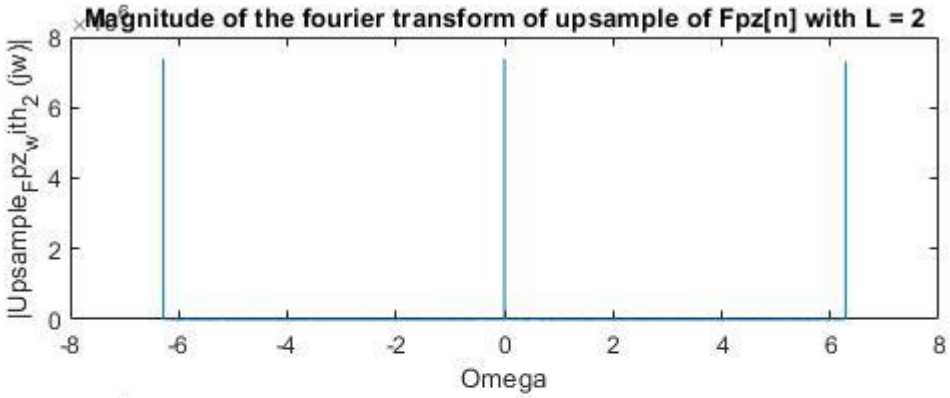

مشاهده می‌شود که با افزایش B مقدار SNR افزایش می‌یابد؛ زیرا با افزایش B، مقدار خطا در Quantization کمتر است.

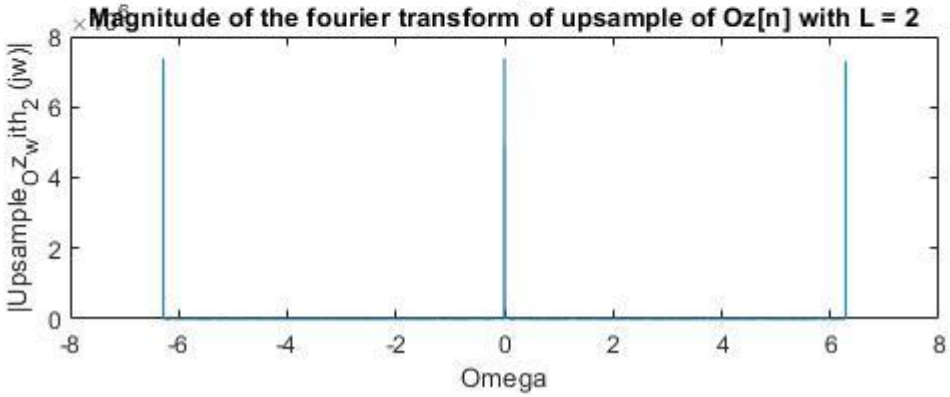
کد متلب:

```
Main();
Interval_Fpz = max(Fpz) - min(Fpz);
B = [8 10 12 14];
Quantized_Fpz_with_8 = floor(Fpz*(2^9)/Interval_Fpz)*Interval_Fpz/(2^9);
Quantized_Fpz_with_10 = floor(Fpz*(2^11)/Interval_Fpz)*Interval_Fpz/(2^11);
Quantized_Fpz_with_12 = floor(Fpz*(2^13)/Interval_Fpz)*Interval_Fpz/(2^13);
Quantized_Fpz_with_14 = floor(Fpz*(2^15)/Interval_Fpz)*Interval_Fpz/(2^15);
SNR_Fpz_8 = 10*log10(sum(abs(Fpz).^2)/sum(abs(Quantized_Fpz_with_8 - Fpz).^2));
SNR_Fpz_10 = 10*log10(sum(abs(Fpz.^2))/sum(abs((Quantized_Fpz_with_10 -
Fpz).^2)));
SNR_Fpz_12 = 10*log10(sum(abs(Fpz.^2))/sum(abs((Quantized_Fpz_with_12 -
Fpz).^2)));
SNR_Fpz_14 = 10*log10(sum(abs(Fpz.^2))/sum(abs((Quantized_Fpz_with_14 -
Fpz).^2)));
SNR_Fpz = [SNR_Fpz_8 SNR_Fpz_10 SNR_Fpz_12 SNR_Fpz_14];
figure();
```

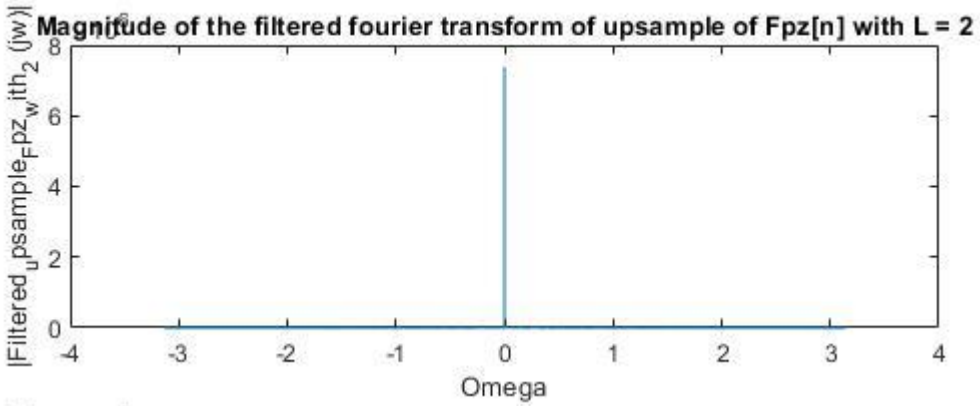
```
stem(B,SNR_Fpz)
xlabel('B')
ylabel('SNR')
title('SNR of Fpz for different B')
Interval_Oz = max(Oz) - min(Oz);
B = [8 10 12 14];
Quantized_Oz_with_8 = floor(Oz*(2^9)/Interval_Oz)*Interval_Oz/(2^9);
Quantized_Oz_with_10 = floor(Oz*(2^11)/Interval_Oz)*Interval_Oz/(2^11);
Quantized_Oz_with_12 = floor(Oz*(2^13)/Interval_Oz)*Interval_Oz/(2^13);
Quantized_Oz_with_14 = floor(Oz*(2^15)/Interval_Oz)*Interval_Oz/(2^15);
SNR_Oz_8 = 10*log10(sum(abs(Oz).^2)/sum(abs(Quantized_Oz_with_8 - Oz).^2));
SNR_Oz_10 = 10*log10(sum(abs(Oz.^2))/sum(abs((Quantized_Oz_with_10 -
Oz).^2)));
SNR_Oz_12 = 10*log10(sum(abs(Oz.^2))/sum(abs((Quantized_Oz_with_12 -
Oz).^2)));
SNR_Oz_14 = 10*log10(sum(abs(Oz.^2))/sum(abs((Quantized_Oz_with_14 -
Oz).^2)));
SNR_Oz = [SNR_Oz_8 SNR_Oz_10 SNR_Oz_12 SNR_Oz_14];
figure();
stem(B,SNR_Oz)
xlabel('B')
ylabel('SNR')
title('SNR of Oz for different B')
```

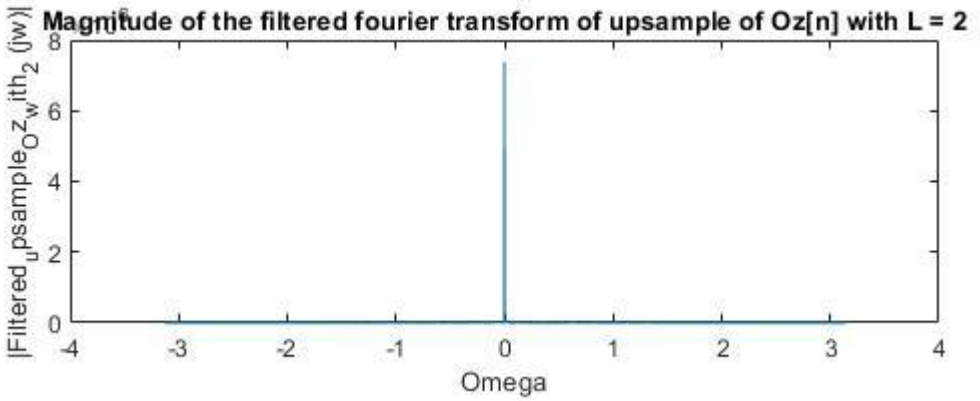
ت) ابتدا سیگنال را با $L = 2$، Upsample می‌کنیم.



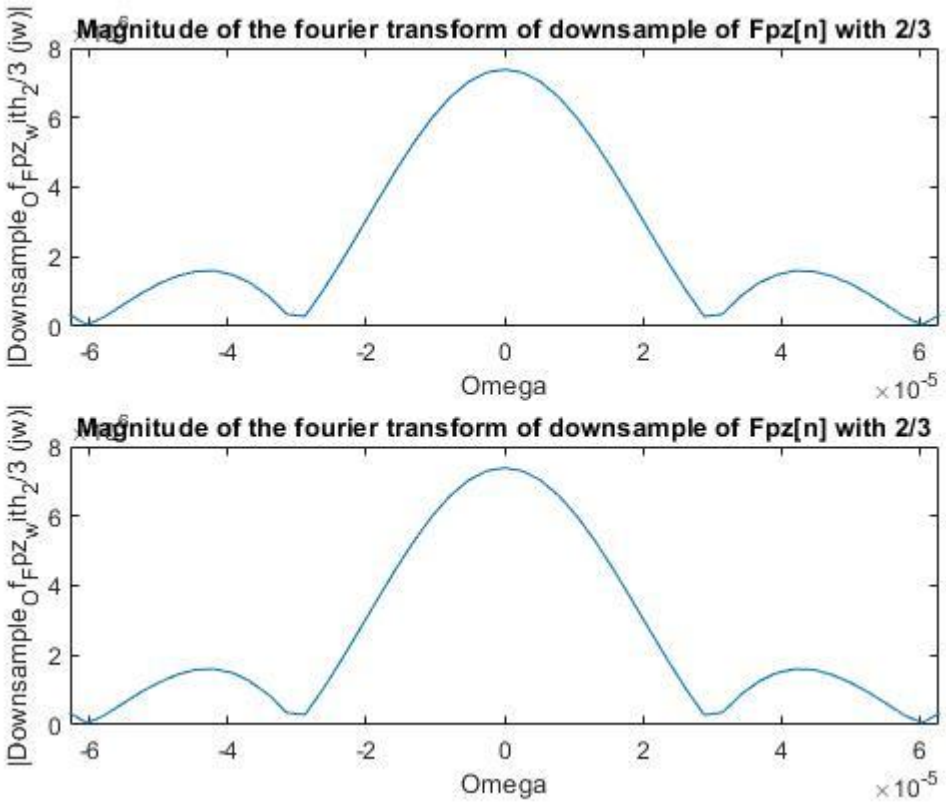




سپس سیگنال بدست آمده را از یک فیلتر پایین گذر عبور می‌دهیم.




در انتها سیگنال را با $M = 3$، Downsample می‌کنیم.




همان‌طور که انتظار داشتیم طیف سیگنال با نرخ $\frac{3}{2}$ کشیده شده است.

از آن جایی که سیگنال اولیه در همه فرکانس‌های دارای مولفه بود، پدیده aliasing رخ می‌دهد؛ اما چون اندازه مولفه‌هایی که دچار aliasing می‌شوند، نسبت به اندازه مولفه‌های اصلی بسیار کوچکتر است، تغییر چندانی در سیگنال نهایی حاصل نمی‌شود. اما اگر نرخ Downsample را چند برابر کنیم، مولفه‌های اصلی نیز دار aliasing می‌شوند و سیگنال به طور کلی تغییر می‌کند. برای جلوگیری از aliasing باید در ابتدا سیگنال را از یک فیلتر پایین گذر با پهنای باند $\pi * \frac{L}{M}$ عبور داد.

کد متلب: Main();

```
fs = 100;
Upsample_Of_Fpz_with_2 = upsample(Fpz,2);
Upsample_Of_Oz_with_2 = upsample(Oz,2);
Fourie_Of_Upsample_Of_Fpz_with_2 = 1/fs*fftshift(fft(Upsample_Of_Fpz_with_2));
Fourie_Of_Upsample_Of_Oz_with_2 = 1/fs*fftshift(fft(Upsample_Of_Oz_with_2));
Omega = -pi:2*pi/length(Upsample_Of_Fpz_with_2):pi-
2*pi/length(Upsample_Of_Fpz_with_2);
figure();
subplot(2,1,1)
plot(2*Omega,abs(Fourie_Of_Upsample_Of_Fpz_with_2))
xlabel('Omega')
```

```
ylabel('|Upsample_Fpz_with_2 (jw)|')
title('Magnitude of the fourier transform of upsample of Fpz[n] with L = 2')
subplot(2,1,2)
plot(2*Omega,abs(Fourie_Of_Upsample_Of_Oz_with_2))
xlabel('Omega')
ylabel('|Upsample_Oz_with_2 (jw)|')
title('Magnitude of the fourier transform of upsample of Oz[n] with L = 2')
H = zeros(1,length(Upsample_Of_Fpz_with_2));
for i = length(Upsample_Of_Fpz_with_2)/4:3*length(Upsample_Of_Fpz_with_2)/4
   H(i) = 1;
end
Filtered_Fourie_Of_Upsample_Of_Fpz_with_2 =
abs(Fourie_Of_Upsample_Of_Fpz_with_2.*H);
Filtered_Fourie_Of_Upsample_Of_Oz_with_2 =
abs(Fourie_Of_Upsample_Of_Oz_with_2.*H);
figure();
subplot(2,1,1)
plot(Omega,Filtered_Fourie_Of_Upsample_Of_Fpz_with_2)
xlabel('Omega')
ylabel('|Filtered_upsample_Fpz_with_2 (jw)|')
title('Magnitude of the filtered fourier transform of upsample of Fpz[n] with L = 2')
subplot(2,1,2)
plot(Omega,Filtered_Fourie_Of_Upsample_Of_Oz_with_2)
xlabel('Omega')
ylabel('|Filtered_upsample_Oz_with_2 (jw)|')
title('Magnitude of the filtered fourier transform of upsample of Oz[n] with L = 2')
Fourie_Of_Downsample_Of_Fpz_with_2_3 =
Filtered_Fourie_Of_Upsample_Of_Fpz_with_2;
Fourie_Of_Downsample_Of_Oz_with_2_3 = Filtered_Fourie_Of_Upsample_Of_Oz_with_2;
```

```
for i =
floor(3*length(Filtered_Fourie_Of_Upsample_Of_Fpz_with_2)/12):floor(4*length(Filtered_Fourie_Of_Upsample_Of_Fpz_with_2)/12)

   Fourie_Of_Downsample_Of_Fpz_with_2_3(i) =
Filtered_Fourie_Of_Upsample_Of_Fpz_with_2(i) +
Filtered_Fourie_Of_Upsample_Of_Fpz_with_2(floor(8*length(Filtered_Fourie_Of_Upsample_Of_Fpz_with_2)/12) + i);

   Fourie_Of_Downsample_Of_Oz_with_2_3(i) =
Filtered_Fourie_Of_Upsample_Of_Fpz_with_2(i) +
Filtered_Fourie_Of_Upsample_Of_Fpz_with_2(floor(8*length(Filtered_Fourie_Of_Upsample_Of_Oz_with_2)/12) + i);

end

for i =
floor(8*length(Filtered_Fourie_Of_Upsample_Of_Fpz_with_2)/12):floor(9*length(Filtered_Fourie_Of_Upsample_Of_Fpz_with_2)/12)

   Fourie_Of_Downsample_Of_Fpz_with_2_3(i) =
Filtered_Fourie_Of_Upsample_Of_Fpz_with_2(i) +
Filtered_Fourie_Of_Upsample_Of_Fpz_with_2(floor(3*length(Filtered_Fourie_Of_Upsample_Of_Fpz_with_2)/12) + i);

   Fourie_Of_Downsample_Of_Oz_with_2_3(i) =
Filtered_Fourie_Of_Upsample_Of_Oz_with_2(i) +
Filtered_Fourie_Of_Upsample_Of_Oz_with_2(floor(3*length(Filtered_Fourie_Of_Upsample_Of_Oz_with_2)/12) + i);

end

Fourie_Of_Downsample_Of_Fpz_with_2_3 =
Fourie_Of_Downsample_Of_Fpz_with_2_3(length(Filtered_Fourie_Of_Upsample_Of_Fpz_with_2)/3:2*length(Filtered_Fourie_Of_Upsample_Of_Fpz_with_2)/3);

Fourie_Of_Downsample_Of_Oz_with_2_3 =
Fourie_Of_Downsample_Of_Oz_with_2_3(length(Filtered_Fourie_Of_Upsample_Of_Oz_with_2)/3:2*length(Filtered_Fourie_Of_Upsample_Of_Oz_with_2)/3);

Omega =
Omega(length(Filtered_Fourie_Of_Upsample_Of_Fpz_with_2)/3:2*length(Filtered_Fourie_Of_Upsample_Of_Fpz_with_2)/3);

figure();

subplot(2,1,1)

plot(3*Omega,abs(Fourie_Of_Downsample_Of_Fpz_with_2_3))

xlim([-2*pi/1e5 2*pi/1e5])
```

```
xlabel('Omega')
ylabel('|Downsample_Of_Fpz_with_2/3 (jw)|')
title('Magnitude of the fourier transform of downsample of Fpz[n] with 2/3')
subplot(2,1,2)
plot(3*Omega,abs(Fourie_Of_Downsample_Of_Oz_with_2_3))
xlim([-2*pi/1e5 2*pi/1e5])
xlabel('Omega')
ylabel('|Downsample_Of_Fpz_with_2/3 (jw)|')
title('Magnitude of the fourier transform of downsample of Fpz[n] with 2/3')
```