CA6

عباس خشدونی فراهانی

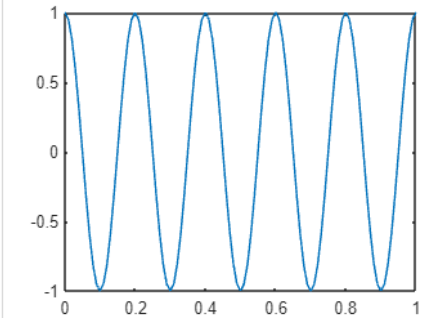
810101415

بخش اول)

1_1

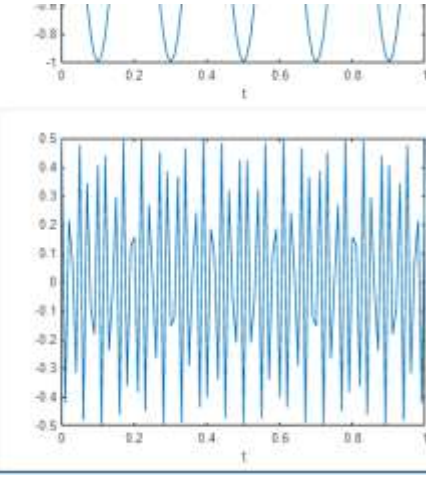
q1_1.mlx | untitled2.mlx

```
t_start = 0;
t_end = 1;
f_c = 5;
f_s = 100;

t_s = 1/f_s;
t = t_start: t_s:t_end;
x = cos(2*pi*f_c*t);

plot(t,x);
```

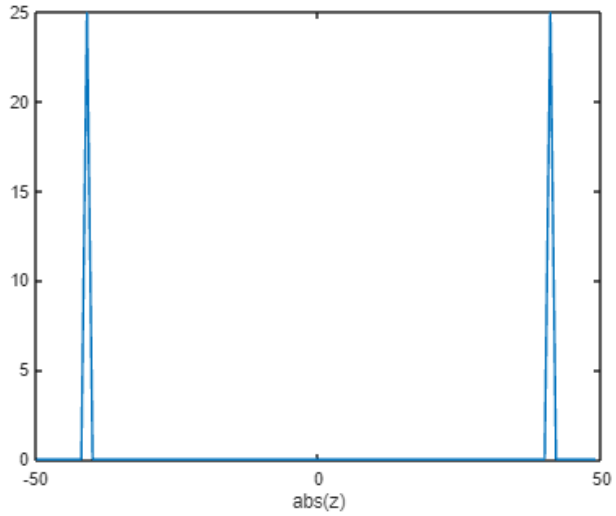


1_2

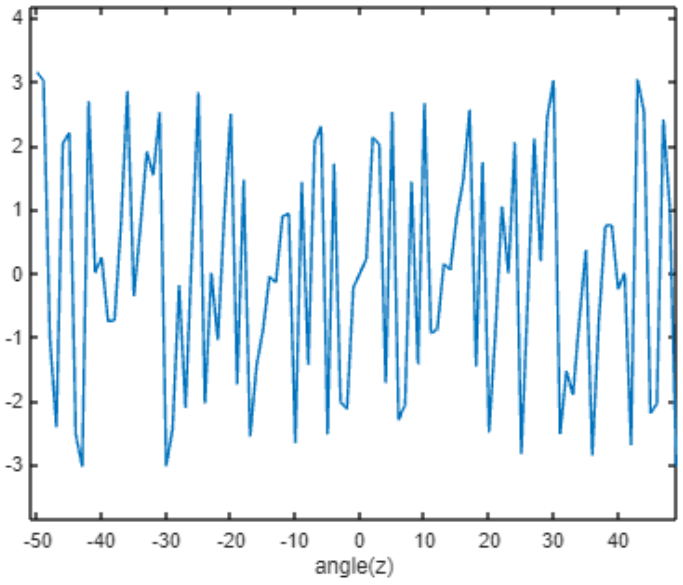
```
t_start = 0;
t_end = 1;
f_c = 5;
f_s = 100;

t_s = 1/f_s;
t = t_start: t_s:t_end;
x = cos(2*pi*f_c*t);

plot(t,x);

xlabel("t")

v = 188;
R = 250;
beta = 0.3;
alpha = 0.5;
f_d = beta*v;
t_d = (2*R)/(3*10e8);
y = alpha * cos(2*pi*(f_c+f_d)*(t-t_d));

plot(t, y);

xlim([0.000 1.000])
ylim([-0.50 0.50])
xlabel("t")
```



```
f=-f_s/2:1:(f_s/2)-1;
z = fftshift(fft(y));
plot(f, abs(z));

xlim([-50.0 50.0])
ylim([0.0 25.0])
xlabel("abs(z)")
```




```
plot(f, angle(z));

xlim([-50.0 50.0])
ylim([-4.00 4.00])
xlabel("angle(z)")
```



در دو تصویر بالا خروجی سوال برای بخش دوم در حوزه فوریه ترسیم شده ات و میخواهیم خواسته سوال را برای مثال در قسمت قبلی بدست اوریم.

```
phaz = zeros(size(locs));
for i = 1:size(locs,2);
    phaz(i) =angle(z(locs(i)));
end


% محاسبه زمان پرواز از فاز
t_d_out = phaz(2) / (2 * pi * f_d)
```

```
t_d_out = 1.8210e-07
```

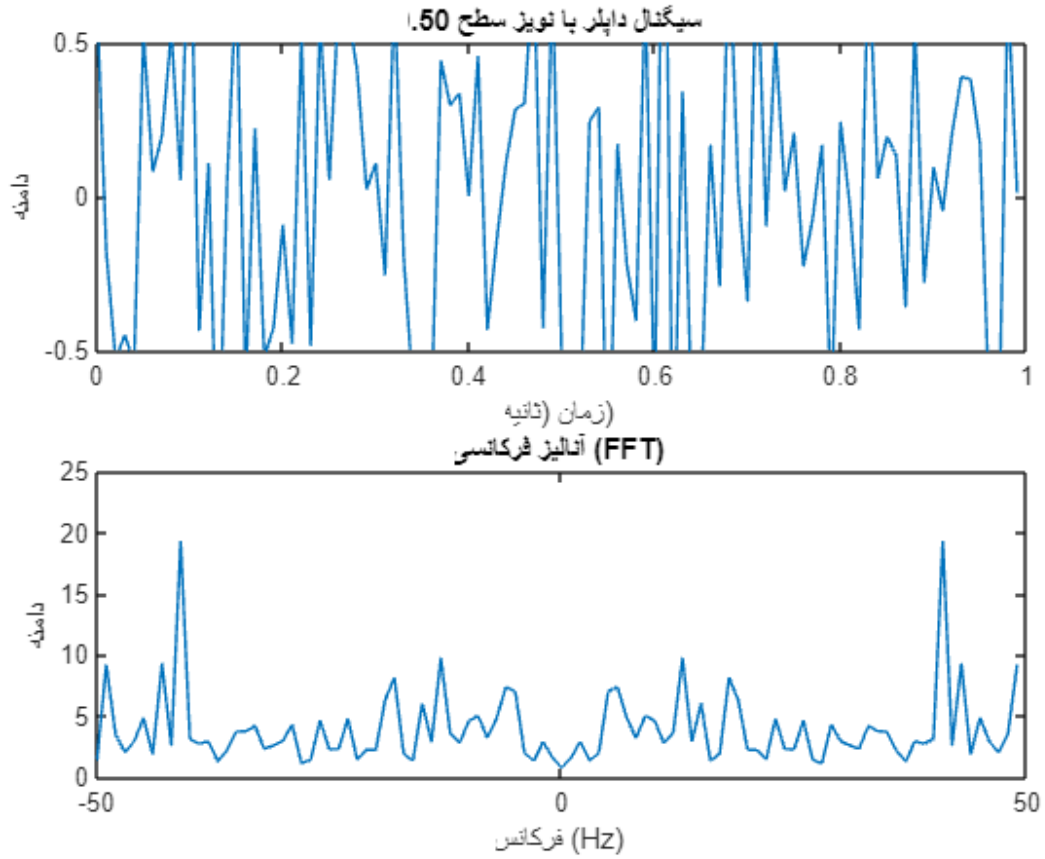
```
% محاسبه فاصله تخمینی
d_estimated = (3 * 10^8 * t_d_out) / 2
```

```
d estimated = 27.3148
```

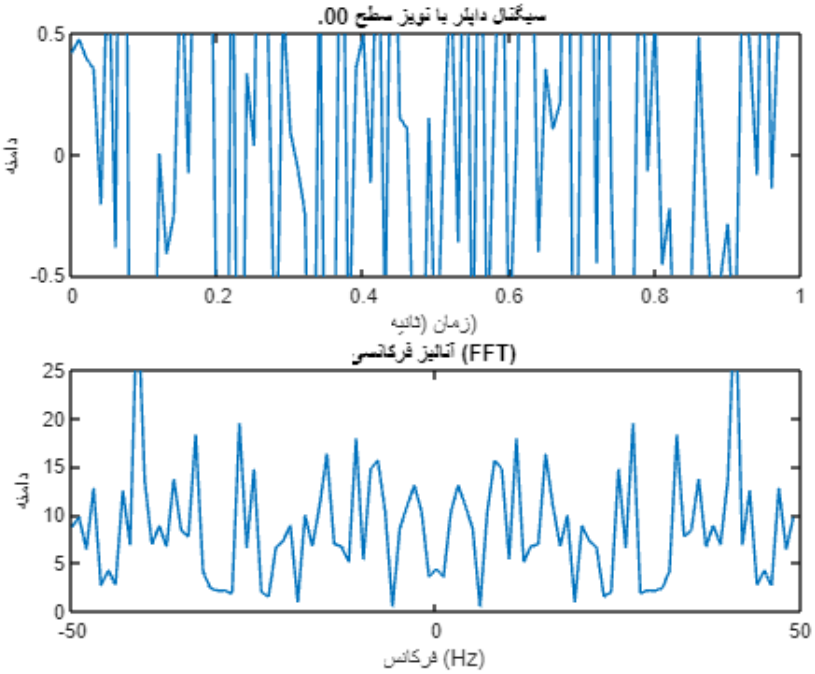
1_4

```
% سطح‌های مختلف نویز
noise_levels = [0.01, 0.05, 0.1, 0.2, 0.5, 1];

% تکرار تمرین برای سطح‌های مختلف نویز
for i = 1:length(noise_levels)
    noise_level = noise_levels(i);
    y_noisy = y + noise_level * randn(size(t)); % افزودن نویز به سیگنال

    % محاسبه FFT
    z = fftshift(fft(y_noisy));

    % پیدا کردن پیک‌های فرکانسی
    threshold_size = max(abs(z)) / 2;
    [pks, locs] = findpeaks(abs(z), "MinPeakHeight", threshold_size);

    % استخراج فاز پیک‌ها
    phaz = zeros(size(locs));
    for j = 1:length(locs)
        phaz(j) = angle(z(locs(j)));
    end

    % محاسبه زمان پرواز از فاز
    if length(phaz) >= 2
        t_d_out = phaz(2) / (2 * pi * f_d);
    else
        t_d_out = NaN; % اگر تعداد پیک‌ها کافی نباشد
    end
```

```
نویز سطح: 0.01
فرکانس داپلر تخمینی: 54.00 Hz
زمان پرواز تخمینی: -1.72e-05 s
فاصله تخمینی: -2574.17 m
نویز سطح: 0.05
فرکانس داپلر تخمینی: 54.00 Hz
زمان پرواز تخمینی: -1.96e-05 s
فاصله تخمینی: -2943.60 m
نویز سطح: 0.10
فرکانس داپلر تخمینی: 54.00 Hz
زمان پرواز تخمینی: 1.16e-04 s
فاصله تخمینی: 17380.43 m
نویز سطح: 0.20
فرکانس داپلر تخمینی: 54.00 Hz
زمان پرواز تخمینی: -7.87e-05 s
فاصله تخمینی: -11809.72 m
نویز سطح: 0.50
فرکانس داپلر تخمینی: 54.00 Hz
زمان پرواز تخمینی: -5.66e-05 s
فاصله تخمینی: -8488.38 m
```

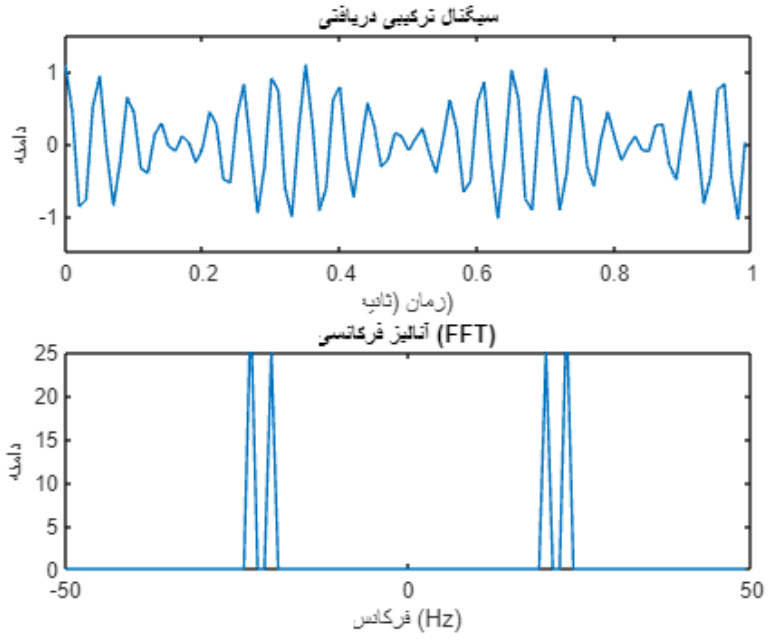


```
نویز سطح: 1.00
فرکانس داپلر تخمینی: 54.00 Hz
زمان پرواز تخمینی: 6.06e-03 s
فاصله تخمینی: 909535.39 m
```



تا نویز سطح 2. با تقریب نتبتا خوبی خروجی را درست دریافت می کنیم.

1_5, 1_6



```
فرکانس داپلر جسم 1 تخمینی: 15.00 Hz

زمان پرواز جسم 1 تخمینی: 2.04e-03 s

فاصله جسم 1 تخمینی: 306666.67 m

فرکانس داپلر جسم 2 تخمینی: 18.00 Hz

زمان پرواز جسم 2 تخمینی: 1.85e-03 s

فاصله جسم 2 تخمینی: 277777.78 m
```

```
% پارامترهای اولیه
t_start = 0;
t_end = 1;
f_c = 5; % فرکانس سیگنال اصلی
f_s = 100; % فرکانس نمونه‌برداری


% تولید بردار زمان
t_s = 1/f_s;
t = t_start: t_s:t_end-t_s;


% پارامترهای جسم 1
R1 = 250e3; % فاصله (متر)
V1 = 180 * 1000 / 3600; % سرعت (متر بر ثانیه)
alpha1 = 0.5; % دامنه
f_d1 = V1 * 0.3; % فرکانس داپلر
t_d1 = (2 * R1) / (3 * 10^8); % زمان تأخیر


% پارامترهای جسم 2
R2 = 200e3; % فاصله (متر)
V2 = 216 * 1000 / 3600; % سرعت (متر بر ثانیه)
alpha2 = 0.6; % دامنه
f_d2 = V2 * 0.3; % فرکانس داپلر
t_d2 = (2 * R2) / (3 * 10^8); % زمان تأخیر


% تولید سیگنال‌های جسم 1 و 2
y1 = alpha1 * cos(2*pi*(f_c + f_d1) * (t - t_d1));
y2 = alpha2 * cos(2*pi*(f_c + f_d2) * (t - t_d2));


% ترکیب سیگنال‌ها
y = y1 + y2;
```

```
% نمایش سیگنال ترکیبی
figure;
subplot(2,1,1);
plot(t, y);
xlim([0 1]);
ylim([-1.5 1.5]);
xlabel("زمان (ثانیه)");
ylabel("دامنه");
title("سیگنال ترکیبی دریافتی");


% محاسبه FFT
f = -f_s/2 : 1 : (f_s/2) - 1;
z = fftshift(fft(y));


% نمایش FFT
subplot(2,1,2);
plot(f, abs(z));
xlim([-50 50]);
ylim([0 25]);
xlabel("فرکانس (Hz)");
ylabel("دامنه");
title("آنالیز فرکانسی (FFT)");



% پیدا کردن پیک‌های فرکانسی
threshold_size = max(abs(z)) / 2;
[pks, locs] = findpeaks(abs(z), "MinPeakHeight", threshold_size);


% استخراج فاز پیک‌ها
phaz = zeros(size(locs));
for i = 1:length(locs)
    phaz(i) = angle(z(locs(i)));
end


% محاسبه زمان پرواز از فاز
if length(phaz) >= 2
    t_d_out1 = phaz(1) / (2 * pi * f_d1);
    t_d_out2 = phaz(2) / (2 * pi * f_d2);
else
    t_d_out1 = NaN; % اگر تعداد پیک‌ها کافی نباشد
    t_d_out2 = NaN; % اگر تعداد پیک‌ها کافی نباشد
end
```

```
% محاسبه فاصله تخمینی
d_estimated1 = (3 * 10^8 * t_d_out1) / 2;
d_estimated2 = (3 * 10^8 * t_d_out2) / 2;
```

1_7, 1_8

نمی توان به درستی تخمین زد زیرا f_d هر در ان ها برابر می شود و در محاسبه فاز مشکل ایجاد میکند و باید اختاف سرعت آن ها به گونه ای باشد که که بتوان در نمه برداری از سگنال آن ها را از هم تمایز داد.

1_9

با افزایش رزلوشن میتوان تعداد فرکانس های غالب را تشخیص داد و از آنجا که هر cos داری دو فرکانس غالب است میتوان تعداد اجسام را تشخیص داد.

بخش دوم

2_1

```
t_start=0;
t_end = 0.5;
T= 0.5;
fs = 8000;

note_cell = cell(2, 12);
notes = {'A', 'A#', 'B', 'C', 'C#', 'D', 'D#', 'E', 'F', 'F#', 'G', 'G#'};
notes_freq = [880, 932.33, 987.77, 523.25, 554.37, 587.33, 622.25, 659.25, 698.46, 739.99, 783.99, 830.61];

for i=1:12
    note_cell{1, i} = notes(i);
    note_cell{2, i} = notes_freq(i);
end

%music
rest = zeros(1, (8*25));
notes_key = [6 6 11 10 6 ,6 8 8 6 10 6 8 ,6 8 10 8 ,6 8 8 6 10 6 8 , 6 8 6 10 8 ,6 8 6 10 8, 6 6 8 10 8 10, 10 8 10 10 6];
time_keys = [1/2 1/2 1 1 1 , 1/2 1/2 1/2 1/2 1/2 1/2 1, 1 1 1 1, 1/2 1/2 1/2 1/2 1/2 1/2 1, 1 1/2 1/2 1 1, 1 1/2 1/2 1 1 ,  1/2 1/2 1 1/2 1/2 1, 1/2 1

music = [];
for i=1:1:size(notes_key, 2)

t = t_start:1/fs:(time_keys(i)*T)-(1/fs);
tone = sin(2*pi* note_cell{2, notes_key(i)}*t);
music = [music, tone, rest];
end
```

نوت داده شده را به صرت یک بردار در متلب اسفاده می کنیم و فرکانس نوت های داده شده را در یک سلول قرار میدهیم.

2_2

```
% Define the note frequencies (adjust as needed)
note_freqs = [261.60, 293.66, 329.63, 349.23, 392.00, 440.00, 493.88];

% Define the melody (sequence of note indices)
melody = [1, 2, 3, 4, 6, 5, 6, 5, 1, 2, 3, 5, 4, 2, 1, 1];

% Create the audio signal
Fs = 44100; % Sampling frequency (adjust as needed)
duration = 0.5; % Duration of each note (in seconds)
song = zeros(1, round(Fs * duration * length(melody)));

for i = 1:length(melody)
    note_idx = melody(i);
    note_signal = sin(2*pi*note_freqs(note_idx)*(0:duration*Fs-1)/Fs);
    song((i-1)*duration*Fs+1:i*duration*Fs) = note_signal;
end

sound(song, Fs);

% Save the melody as a WAV file
filename = 'my_melody.wav';
audiowrite(filename, song, Fs);
% disp(['Melody saved as "', filename, '"']);
```