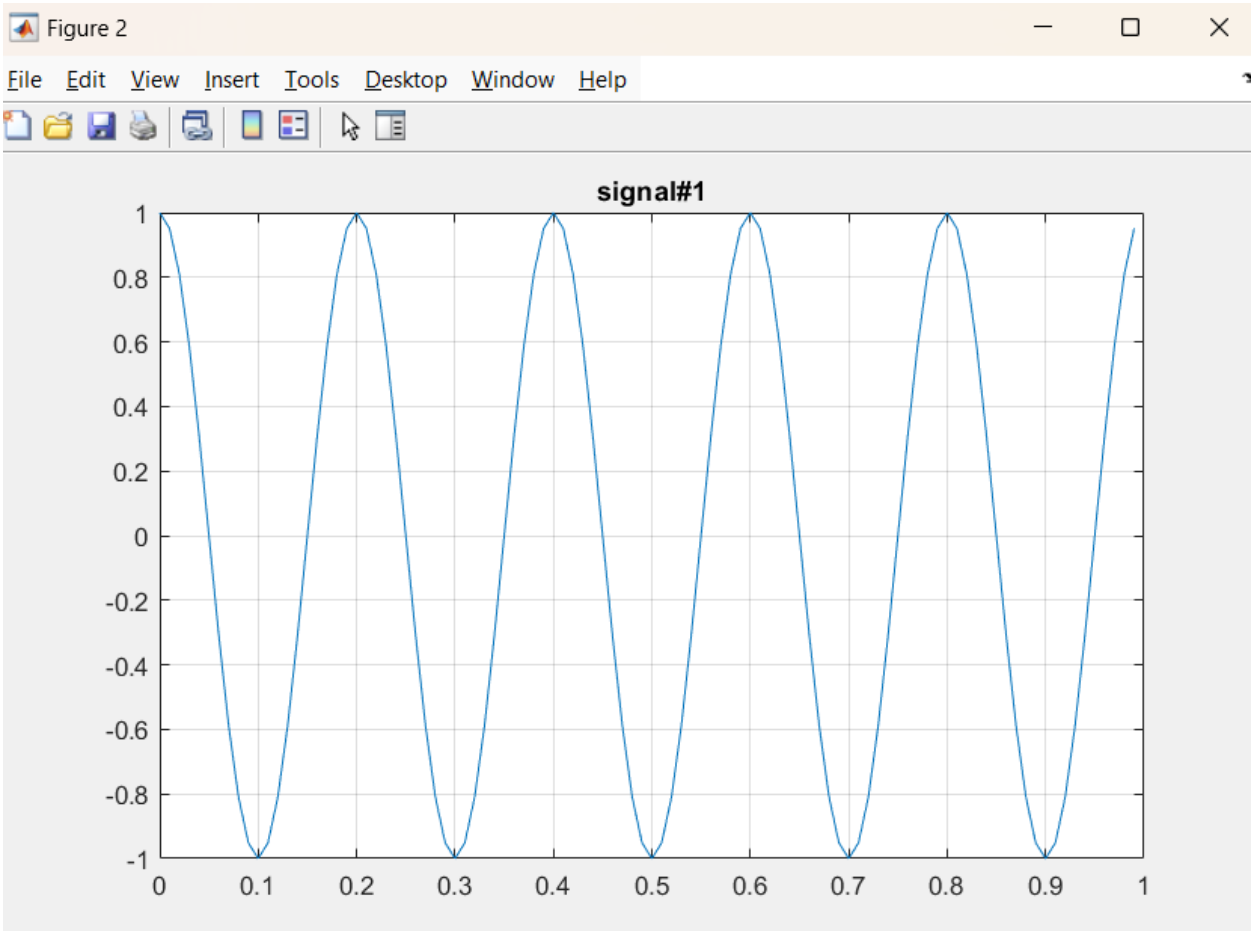
پاسخ تمرین ششم درس سیگنال ها و سیستم ها

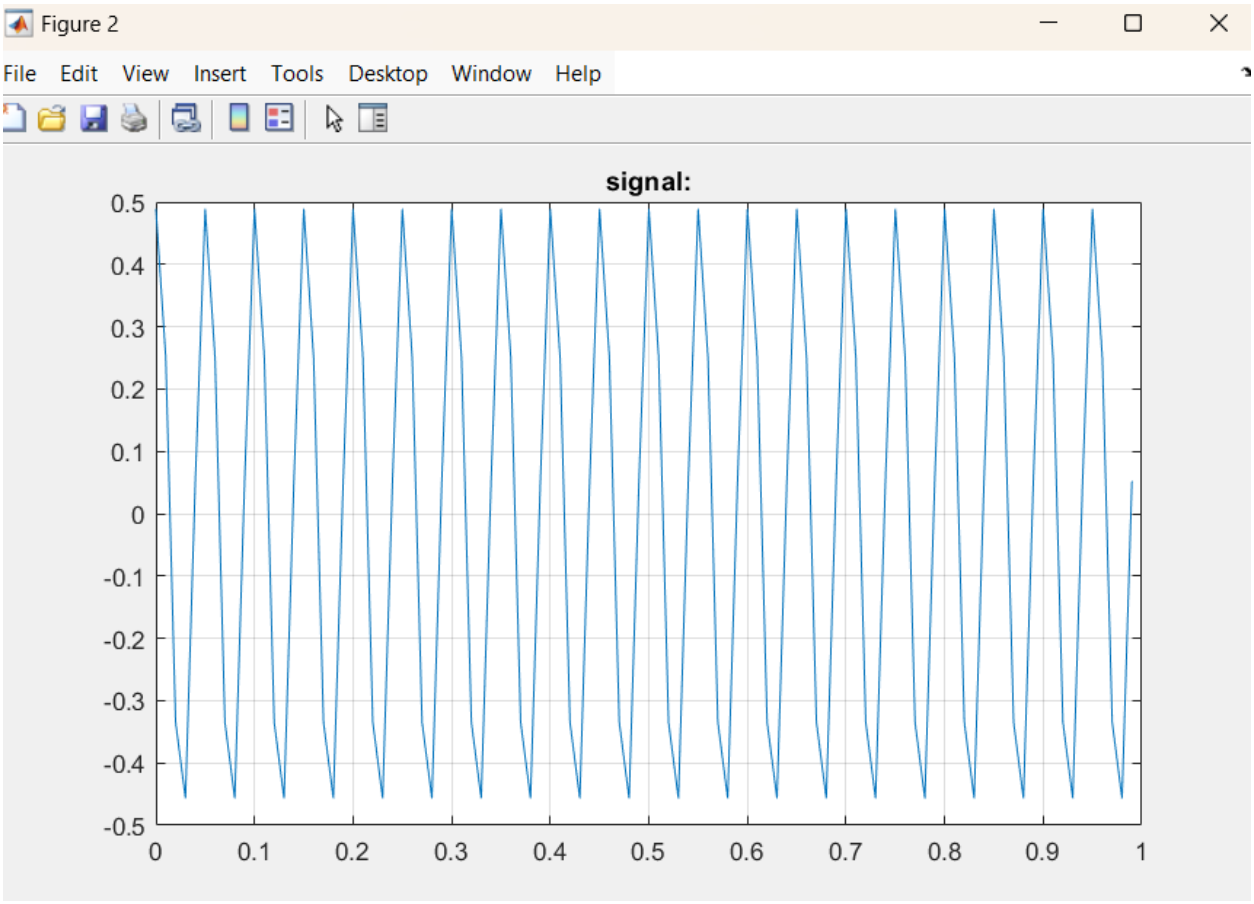
آرمین شباهنگ ۸۱۰۱۰۰۱۷۰

راضیه دوره گرد ۸۱۰۱۹۹۴۱۹

1-1.



1-2.

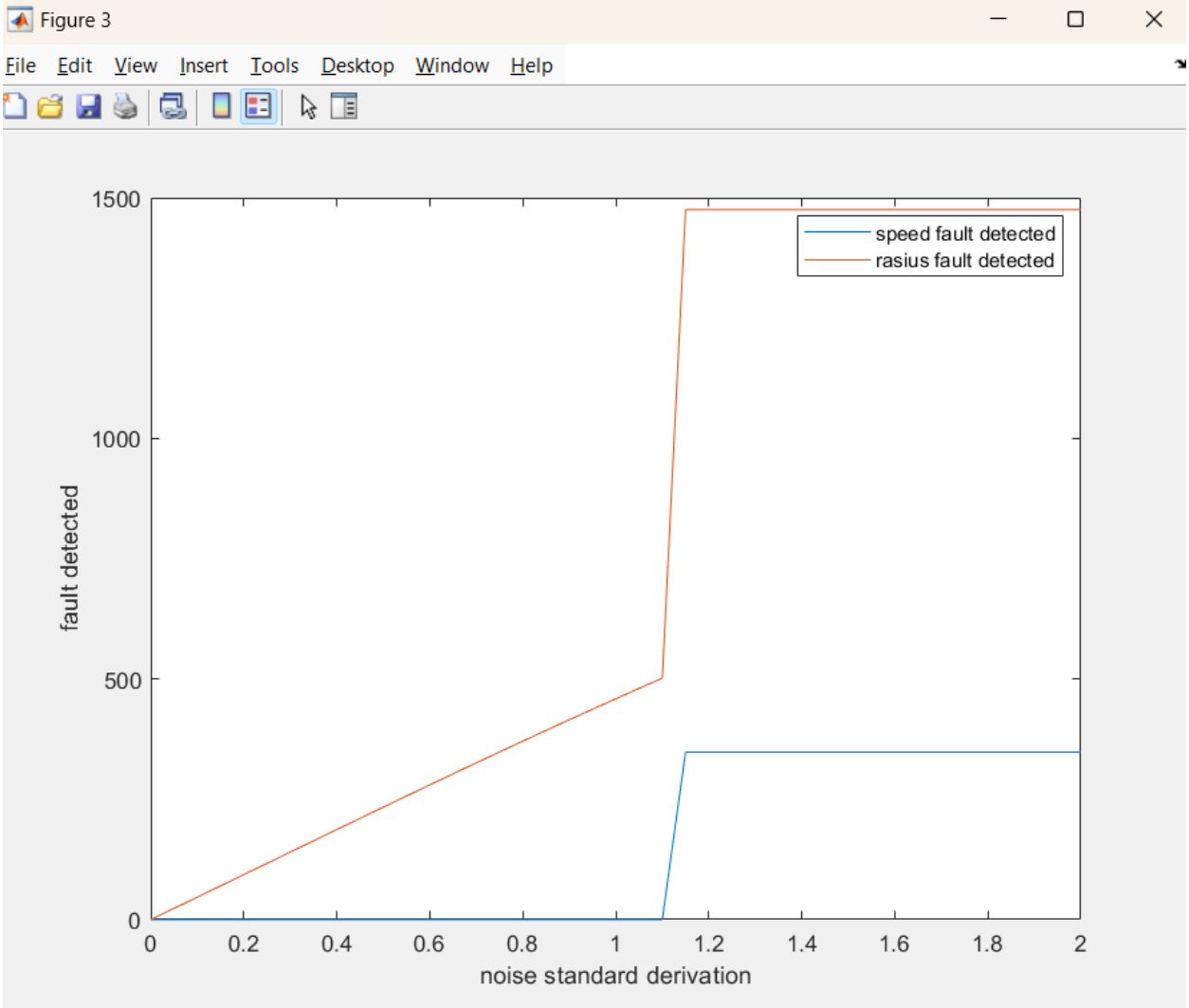


1-3.

```
>> p1_2
estimated Speed(km/h): 180.000000
estimated Radius(km): 250.000000
>>
```

1-4.

همونطور که پلات میزان خطای تشخیص V, R نشون میده، از حدودهای ۱،۱(انحراف معیار نویز ایجاد شده) به بعد، خطای متغیر R شدیدا افزایش پیدا کرده و خیلی حساس تر از V عمل کرده.



1-5.

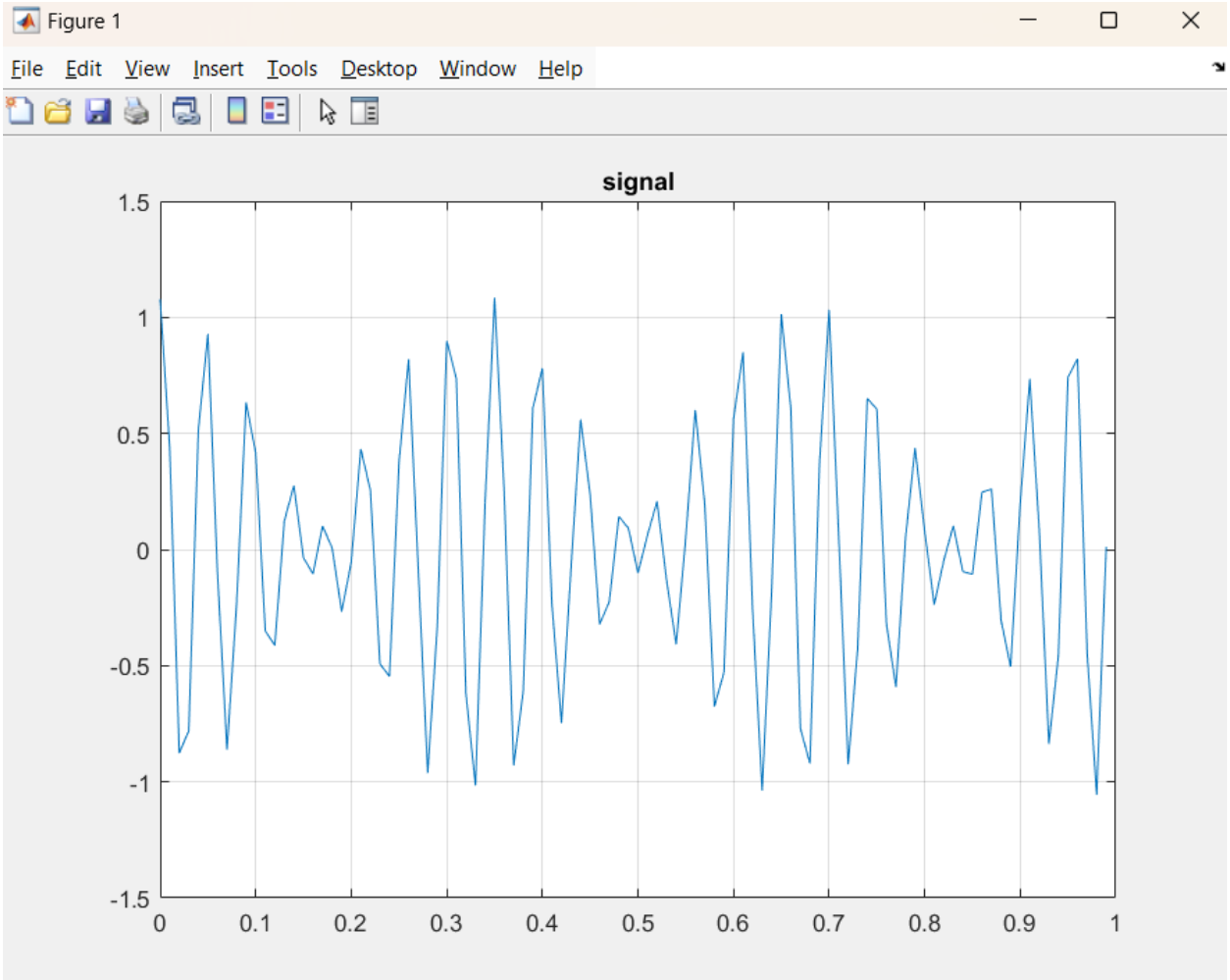
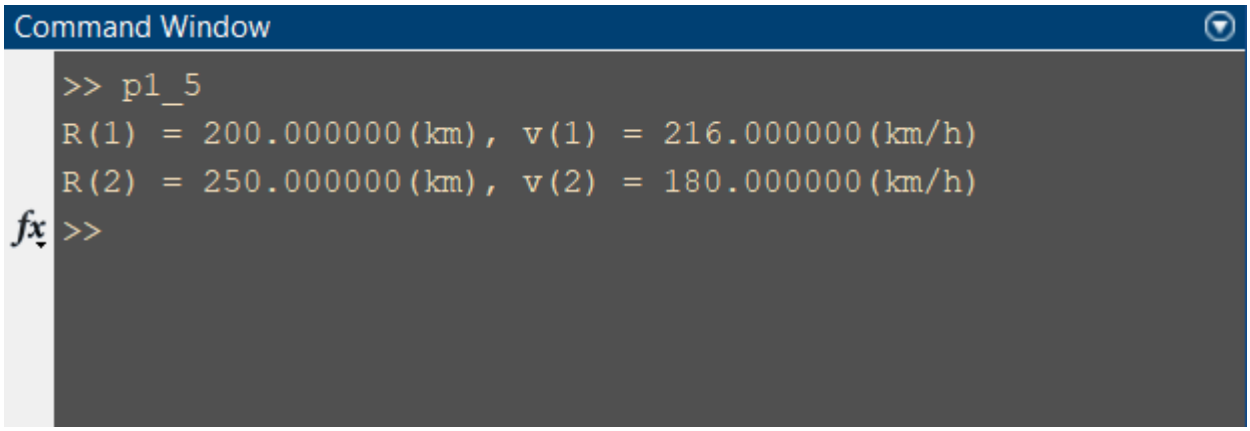


Figure 1
File Edit View Insert Tools Desktop Window Help
signal
1.5
1
0.5
0
-0.5
-1
-1.5
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

1-6

ابتدا تبدیل فوریه ی سیگنال رو میگیریم و بعد با کمک تابع findpeaks ، نقاط بحرانی طیف رو به دست آورده و مرتب میکنیم و بعد که peak هارو گرفتیم، پارامترهای مورد نظر رو فقط برای دوتا peak اول تخمین میزنیم:



1-7

اگر سرعت ها یکی باشه عملا سیگنال ها با هم ادغام میشن و امکان تمیز دادن اونها وجود نخواهد داشت. اختلاف سرعت باید حتما بزرگتر مساوی sampling rate یا همون fs باشه که مطمئن باشیم این اختلاف سرعت خودش رو نشون میده:

S: sampling rate

B: bandwidth

$$\Delta f > \frac{B}{S}$$

همچنین بر اساس قاعده Nyquist سمپلینگ ریت باید حداقل دو برابر اختلاف فرکانس دوتا سیگنال باشه.

1-8

بله، اگر سرعت های سیگنال ها حداقل اختلاف رو باهم داشته باشن، حتی با فاصله های برابر هم سیگنال ها قابل تمیز خواهند بود.

1-9

باز از تبدیل فوریه استفاده میکنیم که سیگنال رو وارد فضای فرکانسی کنیم و توی اون حالت frequency component های سیگنال اصلی قابل استخراج خواهند بود. در ادامه باز از تابع findpeaks استفاده میکنیم که peak هارو به دست بیاریم و از روی تعداد اونها میشه به تعداد سیگنال های سازنده ی سیگنال دریافتی پی ببریم. اگر محدوده ی فرکانسی سیگنال های اصلی رو میدونیم هم میتونیم از فیلتر هایی استفاده کنیم که اون فرکانس های خاص رو عبور میدن و عملا اثر نویز رو هم کم میکنیم.
بعد از استخراج frequency component ها عملیات تخمین رو روی هر کدومشون انجام میدیم.

۱-۲

بخشی از کد قسمت اول که تک به تک نت ها را به ملودی اضافه می کند.

```
10 -    rest = 0.025;
11
12 -    no_periods = 8*4;
13 -    ttotalend = no_periods*tend;
14 -    t = tstart:ts:ttotalend-ts;
15
16 -    audio = zeros(1, no_periods*tend*fs);
17
18 -    audio = audio + make_audio(fG, 2, "full", "full");
19
20 -    audio = audio + make_audio(fFSharp, 3, "full", "full");
21 -    audio = audio + make_audio(fFSharp, 7, "full", "empty");
22 -    audio = audio + make_audio(fFSharp, 11, "full", "full");
23 -    audio = audio + make_audio(fFSharp, 15, "full", "empty");
24 -    audio = audio + make_audio(fFSharp, 19, "full", "full");
25 -    audio = audio + make_audio(fFSharp, 23, "full", "full");
26 -    audio = audio + make_audio(fFSharp, 27, "full", "empty");
27 -    audio = audio + make_audio(fFSharp, 28, "full", "full");
28 -    audio = audio + make_audio(fFSharp, 29, "full", "empty");
29 -    audio = audio + make_audio(fFSharp, 30, "full", "full");
30 -    audio = audio + make_audio(fFSharp, 31, "full", "full");
31
32 -    audio = audio + make_audio(fE, 5, "empty", "full");
33 -    audio = audio + make_audio(fE, 6, "full", "empty");
34 -    audio = audio + make_audio(fE, 8, "full", "full");
35 -    audio = audio + make_audio(fE, 10, "full", "full");
36 -    audio = audio + make_audio(fE, 12, "full", "full");
```

کد سازنده ملودی های هر فریم ۰٫۵ ثانیه ای:

```
    no_rest_samples = rest*fs;

    no_periods = 8*4;

    f = frequency;

    audio = zeros(1, no_periods*tend*fs);
    if (first_half == "full") && (second_half == "full")
        t_full = no_samples*(range_time-1)+1:no_samples*(range_time) ...
            - no_rest_samples;
        t = tstart:ts:tend-ts-rest;
        audio(1, t_full) = sin(2*pi*frequency*t);
        result = audio;
    elseif (first_half == "full") && (second_half == "empty")
        t_full = no_samples*(range_time-1)+1:no_samples*(range_time ...
            - 0.5)-no_rest_samples;
        t = tstart:ts:tend/2-ts-rest;
        audio(1, t_full) = sin(2*pi*frequency*t);
        result = audio;
    elseif (first_half == "empty") && (second_half == "full")
        t_full = no_samples*(range_time-0.5)+1:no_samples*(range_time) ...
            - no_rest_samples;
        t = tstart:ts:tend/2-ts-rest;
        audio(1, t_full) = sin(2*pi*frequency*t);
        result = audio;
    end
end
```

۲-۲

ملودی جدید تولید شده

```
fs = 44100;
durations = [0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5,...
    0.5, 0.5, 0.5, 1];
% Notes: E4, E4, F4, G4, G4, F4, E4, D4, C4, C4, D4, E4, E4, D4, D4
frequencies = [329.63, 329.63, 349.23, 392.00, 392.00, 349.23, 329.63, ..
    293.66, 261.63, 261.63, 293.66, 329.63, 329.63, 293.66, 293.66];

melody = [];
for i = 1:length(frequencies)
    t = 0:1/fs:durations(i)-1/fs;
    note = sin(2 * pi * frequencies(i) * t);
    melody = [melody, note];
end

note_gap = zeros(1, fs * 0.05); % 50 ms silence
melody_with_gaps = [];
for i = 1:length(frequencies)
    t = 0:1/fs:durations(i)-1/fs;
    note = sin(2 * pi * frequencies(i) * t);
    melody_with_gaps = [melody_with_gaps, note, note_gap];
end

melody_with_gaps = melody_with_gaps / max(abs(melody_with_gaps));

audiowrite('OdeToJoyExcerpt.wav', melody_with_gaps, fs);

disp('Melody saved as OdeToJoyExcerpt.wav');
```

۳-۲

در اینجا هم همانند بخش اول این تمرین از تابع fft و تبدیل فوریه برای شناسایی فرکانس های استفاده شده در هر اسلایس زمانی (فریم) بهره گرفته شده است. به گونه ای که با توجه به این که در هر فریم تنها یک فرکانس از فرکانس های چهارگانه مورد استفاده قرار گرفته، بنابراین قطعا یکی از چهار فرکانس در خروجی تابع تبدیل فوریه وجود خواهد داشت که به کمک آن می‌توان حتی دوباره ملودی را بازسازی کرد.

```
 7
 8 -    noteNames = {'G', 'F#', 'E', 'D',};
 9 -    frequencies = [783.99, 739.99, 659.25, 587.33];
10 -    noteMap = containers.Map(frequencies, noteNames);
11
12 -    rest = 0.025;
13 -    numRestNodes = rest * fs;
14
15      % Analyze each frame
16 -    for i = 1:numFrames
17 -        frame = audio((i-1)*frameSize + 1 : i*frameSize);
18
19 -        windowedFrame = frame .* hamming(frameSize);
20
21 -        fftResult = fft(windowedFrame);
22 -        magnitude = abs(fftResult(1:floor(frameSize/2)));
23 -        freqAxis = (0:floor(frameSize/2)-1) * fs / frameSize;
24
25 -        [~, peakIndex] = max(magnitude);
26 -        dominantFreq = freqAxis(peakIndex);
27
28 -        [~, closestIndex] = min(abs(frequencies - dominantFreq));
29 -        detectedNote = noteMap(frequencies(closestIndex));
30
31 -        notes = [notes; {(i-1) * frameDuration, i * frameDuration, ...
32                  detectedNote}];
33 -    end
```

نتیجه به دست آمده

Command Window

[9.2500]	[9.5000]	'F#'
[9.5000]	[9.7500]	'E'
[9.7500]	[10]	'E'
[10]	[10.2500]	'D'
[10.2500]	[10.5000]	'D'
[10.5000]	[10.7500]	'E'
[10.7500]	[11]	'D'
[11]	[11.2500]	'F#'
[11.2500]	[11.5000]	'F#'
[11.5000]	[11.7500]	'E'
[11.7500]	[12]	'E'
[12]	[12.2500]	'D'
[12.2500]	[12.5000]	'D'
[12.5000]	[12.7500]	'E'
[12.7500]	[13]	'E'
[13]	[13.2500]	'F#'
[13.2500]	[13.5000]	'E'
[13.5000]	[13.7500]	'F#'
[13.7500]	[14]	'F#'
[14]	[14.2500]	'F#'
[14.2500]	[14.5000]	'E'
[14.5000]	[14.7500]	'F#'
[14.7500]	[15]	'F#'
[15]	[15.2500]	'F#'
[15.2500]	[15.5000]	'F#'
[15.5000]	[15.7500]	'D'
[15.7500]	[16]	'D'