# پروژه شماره یک درس ریاضیات مهندسی آنالیز فوریه

پریا پاسه ورز شماره دانشجویی:810101393

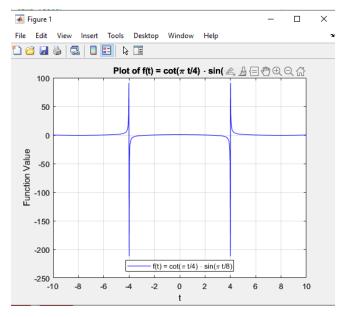
1. آشنایی با Matlab

(1.1

$$\cot \frac{\pi t}{4} \sin \frac{\pi t}{8}$$

#### Code:

```
Introduction_1.m × +
           % Define the time vector
           t = linspace(-10, 10, 1000); % 1000 points from -\pi to \pi
           % Compute the combined function
           combined_function = cot(pi * t / 4) .* sin(pi * t / 8);
   6
   7
           % Plot the combined function
   8
           figure;
           plot(t, combined_function, 'b', 'DisplayName', 'f(t) = cot(\pi t/4) \cdot sin(\pi t/8)');
  9
           xlabel('t');
  10
           ylabel('Function Value');
  11
           title('Plot of f(t) = cot(\pi t/4) \cdot sin(\pi t/8)');
           legend('Location', 'south');
 13
  14
           grid on;
 15
```

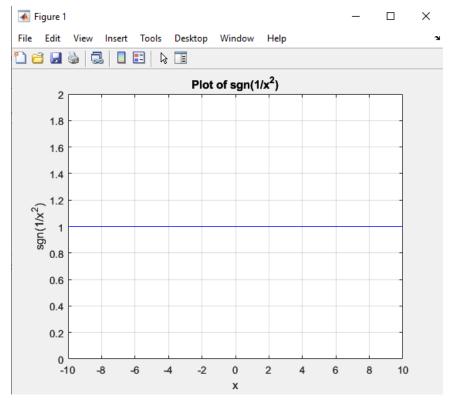


همانطور که در کد نیز قابل مشاهده است، بازه حرکت (-10,10) و گام حرکت، 1000 در نظر گرفته شده است.

```
sgn(\frac{1}{t^2})
```

Code:

```
Introduction_2.m × +
          % Define the function f(x) = 1/x^2
          f = @(x) 1 ./ (x.^2);
          % Define the domain (e.g., from -10 to 10)
          x_values = linspace(-10, 10, 1000);
          % Compute the sign of the function
 8
          sign_function = sign(f(x_values));
 9
10
          % Plot the function
11
          figure;
          plot(x_values, sign_function, 'b', 'DisplayName', 'sgn(1/x^2)');
12
13
          xlabel('x');
14
          ylabel('sgn(1/x^2)');
          title('Plot of sgn(1/x^2)');
15
16
          grid on;
```

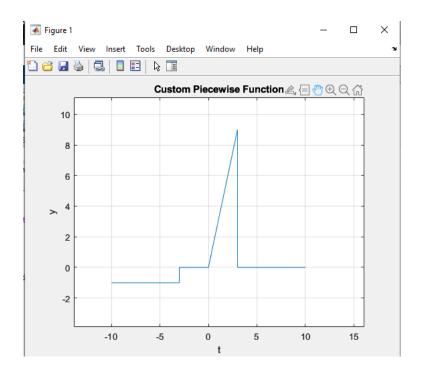


همانطور که در کد نیز قابل مشاهده است، بازه حرکت (-10,10) و گام حرکت، 1000 در نظر گرفته شده است. برای محاسبه تابع sign، از تابع داخلی مثلب استفاده کرده ایم.

```
\begin{cases} -1, & t < -3 \\ 3ramp(t), & -3 < t < 3 \\ e^{-2.5t}, & t > 3 \end{cases}
```

Code:

```
Introduction_3.m * × +
         % Plot the function over the desired interval
 2
          fplot(@piecewise, [-10, 10])
          title('Custom Piecewise Function')
 3
         xlabel('t')
 4
 5
          ylabel('y')
 6
          grid on
 8
          % Define the piecewise function
 9
          function y = piecewise(t)
             if t < -3
10
                 y = -1;
11
              elseif t < 3
12
                 y = 3 * ramp(t);
13
14
              else
15
                 y = exp(-2.5 * t);
16
              end
17
          end
18
          % Define the ramp function
19
20
     口
         function r = ramp(t)
21
             r = max(0, t); % Simple ramp function
22
```



همانطور که در کد نیز قابل مشاهده است، بازه حرکت (-10,10) در نظر گرفته شده است. چون اینجا تابع چند ضابطه ای است، خودمان یک تابع جدید تعریف کردیم تا بتوانیم آن را بسازیم. همچنین برای ساخت تابع (-10,10) نیز یک تابع جداگانه تعریف کرده ایم.

#### 2. سرى فوريه

## 1.2 محاسبه سرى فوريه

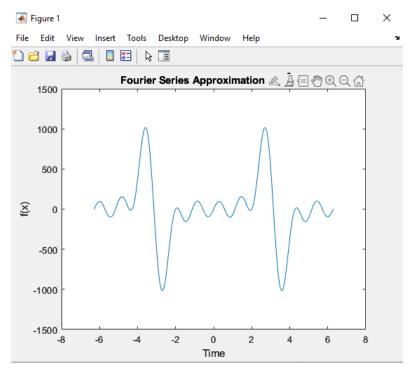
```
exponential_1.m × fourier_series_3.m × +
         function [f, t] = exponential_1(Num, P, alpha, Nshow)
 2
         t = linspace(-2*P, 2*P, 1000);
 4
  5
         a_0 = (1/(P)) * integral(@(x) x.^alpha, -P, P);
  6
         disp(a_0)
  7
         a_n = zeros(Num, 1);
         b_n = zeros(Num, 1);
  8
 9 🛱
         for n = 1:Num
              a_n(n) = (1/P) * integral(@(x) x.^alpha .* cos((pi*n*x)/P), -P, P); \\ b_n(n) = (1/P) * integral(@(x) x.^alpha .* sin((pi*n*x)/P), -P, P); \\ 
 10
 11
 12
 13
         disp('an')
14
         disp(a_n)
 15
         disp('bn')
 16
17
         disp(b_n)
18
 19
         f = (a_0/2);
20 E
         for n = 1:Nshow
21
             f = f + a_n(n)*cos(pi*n*t/P) + b_n(n)*sin(pi*n*t/P);
22
23
```

همانطور که در کد نیز قابل مشاهده است، بازه حرکت (-2p,2p) و گام حرکت، 1000 در نظر گرفته شده است.

## 2.2 محاسبه سرى فوريه يك تابع خاص

```
harmonic_1.m × harmonic_test.m × In_function.m × In_test.m × +
 1 🗆
       function [f, t] = ln_function(Num, P, a, b , Nshow)
 2
 3
       t = linspace(-2*P, 2*P, 1000);
 5
 6
       a0 = (1/(P)) * integral(@(x) x.^b .* log(a.*x), -P, P);
 7
       disp(a0)
 8
       an = zeros(Num, 1);
 9
       bn = zeros(Num, 1);
10 🖹
       for n = 1:Num
11
           an(n) = (1/P) * integral(@(x) x.^b .* log(a.*x) .* cos((pi*n*x)/P), -P, P);
12
           bn(n) = (1/P) * integral(@(x) x.^b .* log(a.*x) .* sin((pi*n*x)/P), -P, P);
13
14
       disp('an')
15
       disp(an)
16
       disp('bn')
17
       disp(bn)
18
19
       f = (a0/2);
20 🖨
       for n = 1:Nshow
21
           f = f + an(n)*cos(pi*n*t/P) + bn(n)*sin(pi*n*t/P);
22
23
24
25
```

```
harmonic_test.m × In_function.m × In_test.m × +
  harmonic_1.m
 1
          Num = 10;
 2
          P = pi;
 3
          a = 100;
          b = 5;
 4
          Nshow = 5;
          % Calculate the Fourier series
 8
          [f, t] = ln_function(Num, P, a, b , Nshow);
 9
10
          %Plot the Fourier series approximation
          plot(t, f);
11
          xlabel('Time');
12
          ylabel('f(x)');
13
          title('Fourier Series Approximation of x^21nx');
14
```



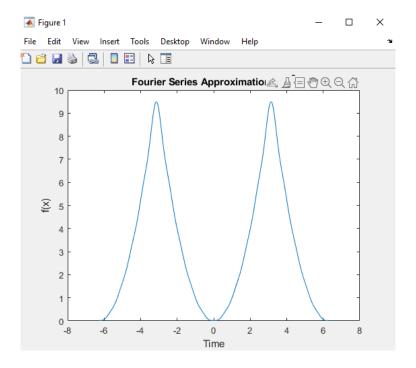
همانطور که در کد نیز قابل مشاهده است، بازه حرکت (2p,2p) و گام حرکت، 1000 در نظر گرفته شده است. مقدار a نیز a نیز a و مقدار a کر نظر گرفته شده است.

## 3.2 رسم سرى فوريه و مقايسه با تابع اصلى

## Num = 10:

Code:

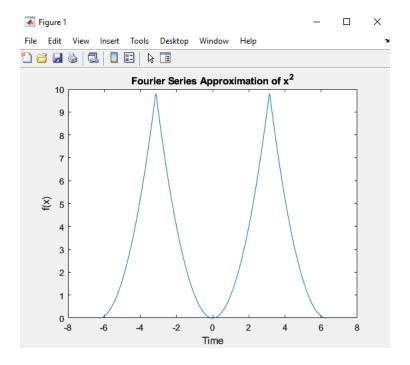
```
Z Editor - E:\Documents\Rizmo\CA\Engineering Mathematics CA1 Fourier Analysis Spring 2024\Codes\2\fourier_series_3.m
    exponential_1.m × fourier_series_3.m × +
             Num = 10;
   2
             P = pi;
             alpha = 2;
   3
             Nshow = 10;
             [f, t] = exponential_1(Num, P, alpha, Nshow);
             plot(t, f);
   8
             xlabel('Time');
ylabel('f(x)');
   9
  10
             title('Fourier Series Approximation of x^2');
  11
```



## Num = 50:

## Code:

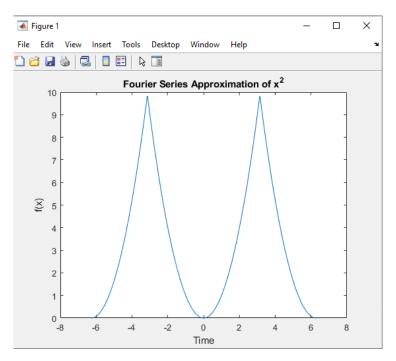
```
Editor - E:\Documents\Rizmo\CA\Engineering Mathematics CA1 Fourier Analysis Spring 2024\Codes\2\fourier_series_3.m
    exponential_1.m × fourier_series_3.m × +
            Num = 50;
   1
   2
            P = pi;
   3
            alpha = 2;
   4
            Nshow = 50;
   5
   6
            [f, t] = exponential_1(Num, P, alpha, Nshow);
   7
   8
            plot(t, f);
   9
            xlabel('Time');
            ylabel('f(x)');
  10
            title('Fourier Series Approximation of x^2');
  11
```



Num = 100:

Code:

Result:



می دانیم هر چه جملاتی که سری فوریه را به از ای آن محاسبه می کنیم بیشتر باشد، حاصل نهایی به مقدار حقیقی تابع نزدیک تر خواهد بود.

در اینجا نیز مشاهده می کنیم که با افزایش مقدار Num، شکل نمودار رسم شده به  $f(x)=x^2$  نزدیک تر شده و نقاطی که هر دوره تناوب شروع می شود، تیزتر شکسته می شود.

## 4.2 محاسبه حد مجموع و تطبيق نتايج

تحليل تئوري:

$$f(x) = a_0 + \sum_{n=1}^{+\infty} a_n \cos(nx) + b_n \sin(nx)$$

$$a_0 = \frac{1}{2\pi} \int_{-\pi}^{\pi} x^2 dx = \frac{1}{3} \pi^2$$

$$a_n = \frac{1}{\pi} \int_{-\pi}^{\pi} x^2 \cos(nx) dx = \frac{1}{\pi} \frac{2(\pi^2 n^2 - 2)\sin(n\pi) + 4\pi n\cos(n\pi)}{n^3} = \frac{4(-1)^n}{n^2}$$

پس: پس: چون تابع  $x^2$  جون تابع  $f(x)=x^2$  تابعی

$$b_n = 0 \quad \forall n \ge 1$$

$$f(x) = \frac{\pi^2}{3} + 4 \sum_{n=1}^{\infty} \frac{(-1)^n}{n^2} \cos(nx)$$

حال اگر قرار دهیم  $x=\pi$  خواهیم داشت:

$$\pi^2 = \frac{\pi^2}{3} + 4\sum_{n=1}^{\infty} \frac{(-1)^n}{n^2} \cos(n\pi)$$
$$2\frac{\pi^2}{3} = 4\sum_{n=1}^{\infty} \frac{(-1)^n}{n^2} (-1)^n$$
$$\sum_{n=1}^{\infty} \frac{1}{n^2} = \frac{\pi^2}{6}$$

تحليل كامپيوترى:

کد را مشابه روبهرو تغییر می دهیم و مقدار t را  $\pi$  می گذاریم تا مقدار محاسبه شده برای سری فوریه را در این نقطه بیابیم:

Code:

```
Editor - E:\Documents\Rizmo\CA\Engineering Mathematics CA1 Fourier Analysis Spring 2024\Codes\2\exponen
 sound_1.m × exponential_1.m × fourier_series_3.m × +
       function [f, t] = exponential_1(Num, P, alpha, Nshow)
 3
 4
5
       a_0 = (1/(2*P)) * integral(@(x) x.^alpha, -P, P);
       disp('a0')
 6
       disp(a_0)
 8
       a_n = zeros(Num, 1);
 9
       b_n = zeros(Num, 1);
10
       for n = 1:Num
          11
12
13
14
       disp('an')
15
       disp(a_n)
16
       disp('bn')
17
18
       disp(b_n)
19
       f = (a_0);
20
21 -
       for n = 1:Nshow
          f = f + a_n(n)*cos(pi*n*t/P) + b_n(n)*sin(pi*n*t/P);
22
23
24
25
       disp('f')
26
       disp(f)
27
```

Result:

a0:3.289

f: 9.8298

حال اگر از رابطه سری فوریه استفاده کنیم، خواهیم داشت:

$$9.8298 = 3.289 + 4 \sum_{n=1}^{\infty} \frac{(-1)^n}{n^2} (-1)^n$$

$$\sum_{n=1}^{\infty} \frac{1}{n^2} = \frac{9.8298 - 3.289}{4} = 1.6352$$

در روش تئوری به دست آوردیم:

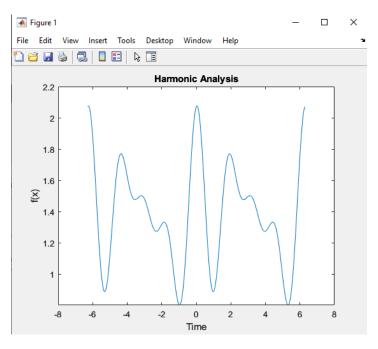
$$\sum_{n=1}^{\infty} \frac{1}{n^2} = \frac{\pi^2}{6} = 1.6449$$

همین طور که مشاهده می کنیم، اندکی خطا در این دو آنالیز وجود دارد، ولی تا حد خوبی مطابقت دارند.

#### 1.5.2 شبیه سازی آنالیز هارمونیک

```
sound_1.m × harmonic_1.m × harmonic_test.m × +
 1 📮
       function [f, t] = harmonic_1(x_values, fx_values, Nshow, P)
 3
       t = linspace(-2*P, 2*P, 1000);
 4
       arr_size = numel(x_values);
 5
       a 0 = 2 * mean(fx values);
 7
 8
       disp('a0')
       disp(a_0)
 10
 11
       a_n = zeros(Nshow, 1);
 12
       b_n = zeros(Nshow, 1);
 13
14 🖹
        for n = 1:Nshow
 15
           sum_fx_cos_nx = 0;
 16
           sum_fx_sin_nx = 0;
17 白
           for i = 1:arr size
 18
            x = x_values(i);
 19
           sum_fx_cos_nx = sum_fx_cos_nx + fx_values(i) * cos(n * x);
 20
 21
           mean_fx_cos_nx = sum_fx_cos_nx / arr_size;
 22
           a_n(n) = 2 * mean_fx_cos_nx;
 23
24 🖃
           for i = 1:arr_size
 25
           x = x_values(i);
           sum_fx_sin_nx = sum_fx_sin_nx + fx_values(i) * sin(n * x);
 26
 27
           mean_fx_sin_nx = sum_fx_sin_nx / arr_size;
 28
 29
           b_n(n) = 2 * mean_fx_sin_nx;
 30
 31
 32
       disp('an')
 33
       disp(a_n)
 34
       disp('bn')
 35
 36
       disp(b_n)
 37
 38
       f = (a_0/2);
 39 🖨
       for n = 1:Nshow
40
           f = f + a_n(n)*cos(n*t) + b_n(n)*sin(n*t);
41
sound_1.m × harmonic_1.m × harmonic_test.m × +
           x_{values} = [0, pi/3, (2*pi)/3, pi, (4*pi)/3, (5*pi)/3, 2*pi];
  2
            fx_values = [1, 1.4, 1.9, 1.7, 1.5, 1.2, 1];
  3
           P = pi;
  4
           Nshow = 4;
  5
  6
            [f, t] = harmonic_1(x_values, fx_values, Nshow, P);
  8
           plot(t, f);
  9
            xlabel('Time');
            ylabel('f(x)');
 10
 11
            title('Harmonic Analysis');
```

همانطور که در کد نیز قابل مشاهده است، بازه حرکت (-2p,2p) و گام حرکت، 1000 در نظر گرفته شده است.



ضرایب در صورت پروژه به اشتباه محاسبه شده اند، مقادیر صحیح:

a0

2.7714

an

-0.0286

0.2000

0.3143

0.2000

bn

0.1485

-0.0495

0.0000

0.0495

#### 3 تبدیل فوریه

## 3.2 بررسى حوزه زمان و فركانس چند تابع

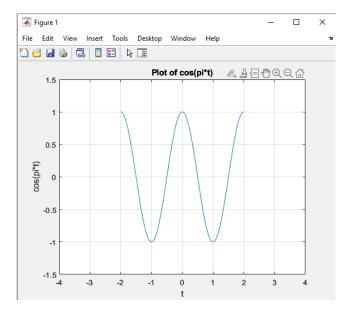
نحوه كار تابع fft و fftshift:

وقتی fft را روی یک تابع اعمال می کنیم، تبدیل فوریه اعمال شده در فرکانس صفر مرکزی نشده است، به همین دلیل نتیجه درستی را نشان نمی دهد. با اعمال کردن fftshift، تابع را شیفت می دهد تا فرکانس صفر مرکز آن قرار گیرد و نمایش صحیحی داشته باشیم

 $f(t) = cos(\pi t)$  بررسی تابع

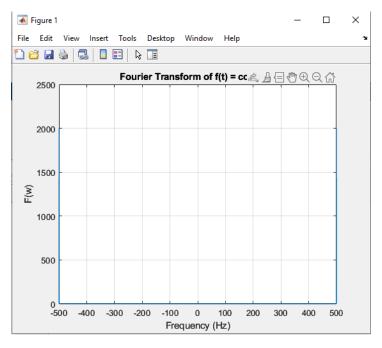
رسم تابع در دو دوره تناوب:

```
fourier_series_3.m × harmonic_1.m × harmonic_test.m × f_tranfrom_1.m × +
             fs = 1000;
            y = cos(pi*t);
t = -2:1/fs:2;
            %Plot cos(pi*t)
            plot(t,y)
xlim([-4 4])
            ylim([-1.5 1.5])
             xlabel('t')
             ylabel('cos(pi*t)')
  11
            title('Plot of cos(pi*t)')
  12
            grid on
  13
            \%\mathrm{U} fft and fftshift to calculate the Fourier transform
  14
            y1 = fft(y);
  15
             f = (-length(y)/2:length(y)/2-1) * fs/length(y);
  16
  17
            %Plot y1
%plot(f, abs(y1))
%xlabel('Frequency (Hz)');
  18 📮
  19
  20
  21
             %ylabel('F(w)');
             %title('Fourier Transform of f(t) = cos(pi*t)');
  24
  25
            y_shifted = fftshift(y1);
  26
  27 📮
            %Plot y_shifted
            % plot(f, abs(y_shifted));
  28
            % xlabel('Frequency (Hz)');
% ylabel('F(w)');
% title('Fourier Transform of f(t) = cos(pi*t)');
  29
  30
  31
  32
            % grid on
```



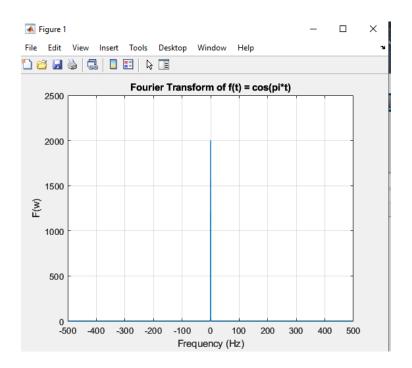
محاسبه تبدیل فوریه با کمک دستور fft:

```
fourier_series_3.m × harmonic_1.m × harmonic_test.m × f_tranfrom_1.m × +
         fs = 1000;
 1
         y = cos(pi*t);
3
         t = -2:1/fs:2;
4
5
         %Plot cos(pi*t)
6
         % plot(t,y)
7
         % xlim([-4 4])
         % ylim([-1.5 1.5])
8
9
         % xlabel('t')
         % ylabel('cos(pi*t)')
10
11
         % title('Plot of cos(pi*t)')
12
         % grid on
13
         %Use fft and fftshift to calculate the Fourier transform
14
15
         y1 = fft(y);
16
         f = (-length(y)/2:length(y)/2-1) * fs/length(y);
17
18
         %Plot y1
19
         plot(f, abs(y1))
         xlabel('Frequency (Hz)');
20
21
         ylabel('F(w)');
         title('Fourier Transform of f(t) = cos(pi*t)');
22
23
         grid on
24
25
         y_shifted = fftshift(y1);
26
         %Plot y_shifted
27
28
         % plot(f, abs(y_shifted));
29
         % xlabel('Frequency (Hz)');
30
         % ylabel('F(w)');
31
         % title('Fourier Transform of f(t) = cos(pi*t)');
32
         % grid on
```



محاسبه تبدیل فوریه با کمک دستور fftshift:

```
fourier_series_3.m × harmonic_1.m × harmonic_test.m × f_tranfrom_1.m × +
             fs = 1000;
             y = cos(pi*t);
   2
             t = -2:1/fs:2;
   5
            %Plot cos(pi*t)
             % plot(t,y)
             % xlim([-4 4])
             % ylim([-1.5 1.5])
            % xlabel('t')
            % ylabel('cos(pi*t)')
  10
            % title('Plot of cos(pi*t)')
 11
            % grid on
 12
 13
 14
             %Use fft and fftshift to calculate the Fourier transform
             y1 = fft(y);
f = (-length(y)/2:length(y)/2-1) * fs/length(y);
 15
 16
 17
           %Plot y1
       18
            % plot(f, abs(y1))
% xlabel('Frequency (Hz)');
 19
 20
            % xlabel( 'F(w)');
% ylabel('F(w)');
% title('Fourier Transform of f(t) = cos(pi*t)');
 21
 22
 23
            % grid on
 24
             y_shifted = fftshift(y1);
 25
 26
            %Plot y_shifted
plot(f, abs(y_shifted));
xlabel('Frequency (Hz)');
 27
 28
 29
 30
             ylabel('F(w)');
             title('Fourier Transform of f(t) = cos(pi*t)');
 31
 32
             grid on
```



محاسبه تبدیل فوریه به صورت دستی:

$$f(t) = Cos(w_0 t)$$

$$F(w) = \int_{-\infty}^{\infty} f(t)e^{-iwt} dt = \int_{-\infty}^{\infty} Cos(w_0 t) e^{-iwt} dt$$

$$Cos(w_0 t) = \frac{e^{-iw_0 t} + e^{iw_0 t}}{2}$$

$$F(w) = \int_{-\infty}^{\infty} (\frac{e^{-iw_0 t} + e^{iw_0 t}}{2})e^{-iwt} dt = \frac{1}{2} \int_{-\infty}^{\infty} e^{-i(w-w_0)t} dt + \frac{1}{2} \int_{-\infty}^{\infty} e^{-i(w+w_0)t} dt$$

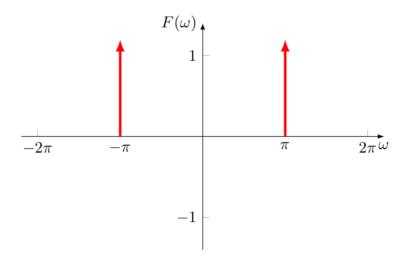
$$\delta(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} e^{-iwt} dt$$

$$\int_{-\infty}^{\infty} e^{-i(w-w_0)t} dt = 2\pi \delta(w - w_0)$$

$$\int_{-\infty}^{\infty} e^{-i(w+w_0)t} dt = 2\pi \delta(w + w_0)$$

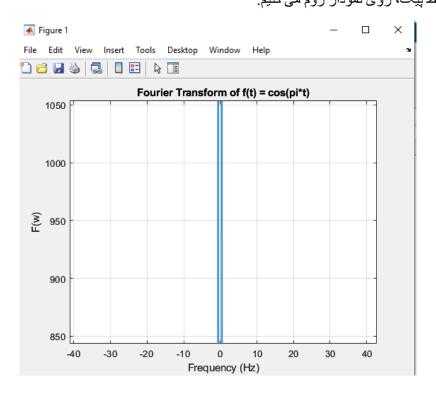
$$F(Cos(w_0 t)) = \pi(\delta(w - w_0) + \delta(w + w_0))$$

$$w_0 = \pi \Rightarrow F(Cos(w_0 t)) = \pi(\delta(w - \pi) + \delta(w + \pi))$$



همانطور که توضیح دادیم، نمایش تبدیل فوریه حاصل از اعمال تابع fft، صحیح نیست، ولی پس از اعمال تابع fftshift، مطابق با محاسبات عملی روی دو نقطه 0.5 و 0.5 پیک می زند.

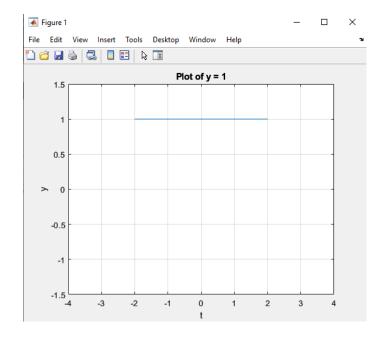
دقت کنید دلیل اینکه روی  $\pi$  و  $\pi$  – پیک نمی زند این است که ما اینجا فرکانس را اندازه می گیریم، نه فرکانس زاویه ای. برای بهتر دیدن نقاط پیک، روی نمودار زوم می کنیم:



```
f(t) = 1 بررسی تابع و بررسی تابع در دو دوره تناوب:
```

Code:

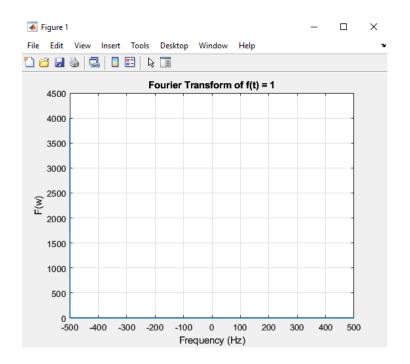
```
fourier_series_3.m × harmonic_1.m × harmonic_test.m × f_tranfrom_1.m × f_tranfrom_2.m ×
            fs = 1000;
  1
  2
            a = 1;
           y = a * ones(size(t));
  3
            t = -2:1/fs:2;
  4
  5
           %Plot y=1
  6
  7
           plot(t,y)
  8
            xlim([-4 4])
  9
           ylim([-1.5 1.5])
           xlabel('t')
 10
 11
           ylabel('y')
 12
            title('Plot of y = 1')
 13
           grid on
 14
 15
            %Use fft and fftshift to calculate the Fourier transform
 16
           y1 = fft(y);
            f = (-length(y)/2:length(y)/2-1) * fs/length(y);
 17
 18
            %Plot y1
 19
 20
            %plot(f, y1)
 21
            %xlabel('Frequency (Hz)');
            %ylabel('F(w)');
 22
 23
           %title('Fourier Transform of f(t) = 1');
 24
           %grid on
 25
           y_shifted = fftshift(y1);
 26
 27
 28
           %Plot y_shifted
           % plot(f, abs(y_shifted));
% xlabel('Frequency (Hz)');
 29
 30
  31
            % ylabel('F(w)');
 32
           % title('Fourier Transform of f(t) = 1');
           % grid on
 33
```



## محاسبه تبدیل فوریه با کمک دستور fft:

Code:

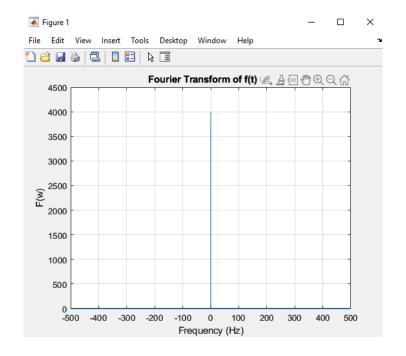
```
fourier_series_3.m × harmonic_1.m × harmonic_test.m × f_tranfrom_1.m × f_tr
  1
           fs = 1000;
  2
           a = 1;
y = a * ones(size(t));
  3
  4
           t = -2:1/fs:2;
  5
  6
           %Plot y=1
      曱
  7
           % plot(t,y)
  8
           % xlim([-4 4])
           % ylim([-1.5 1.5])
  9
 10
           % xlabel('t')
 11
           % ylabel('y')
           % title('Plot of y = 1')
 12
 13
           % grid on
 14
 15
           %Use fft and fftshift to calculate the Fourier transform
 16
           y1 = fft(y);
 17
           f = (-length(y)/2:length(y)/2-1) * fs/length(y);
 18
 19
           %Plot y1
 20
           plot(f, y1)
 21
           xlabel('Frequency (Hz)');
 22
           ylabel('F(w)');
 23
           title('Fourier Transform of f(t) = 1');
 24
           grid on
 25
 26
           y_shifted = fftshift(y1);
 27
      曱
           %Plot y_shifted
 28
 29
           % plot(f, abs(y_shifted));
 30
           % xlabel('Frequency (Hz)');
 31
           % ylabel('F(w)');
 32
           % title('Fourier Transform of f(t) = 1');
           % grid on
 33
```



## محاسبه تبدیل فوریه با کمک دستور fftshift:

Code:

```
fourier_series_3.m × harmonic_1.m × harmonic_test.m × f_tranfrom_1.m ×
          fs = 1000;
 1
 2
          a = 1;
 3
         y = a * ones(size(t));
         t = -2:1/fs:2;
 4
 5
 6
     巨
         %Plot y=1
 7
          % plot(t,y)
          % xlim([-4 4])
         % ylim([-1.5 1.5])
9
10
         % xlabel('t')
11
          % ylabel('y')
         % title('Plot of y = 1')
12
13
         % grid on
14
15
         %Use fft and fftshift to calculate the Fourier transform
16
         y1 = fft(y);
          f = (-length(y)/2:length(y)/2-1) * fs/length(y);
17
18
    早
         %Plot y1
19
20
         % plot(f, y1)
         % xlabel('Frequency (Hz)');
21
         % ylabel('F(w)');
22
23
          % title('Fourier Transform of f(t) = 1');
24
         % grid on
25
26
         y_shifted = fftshift(y1);
27
28
         %Plot y_shifted
         plot(f, abs(y_shifted));
xlabel('Frequency (Hz)');
29
30
         ylabel('F(w)');
31
32
          title('Fourier Transform of f(t) = 1');
33
          grid on
```



محاسبه تبدیل فوریه به صورت دستی:

$$f(t) = A$$

$$F(w) = \int_{-\infty}^{\infty} f(t)e^{-iwt} dt = \int_{-\infty}^{\infty} A e^{-iwt} dt$$

$$\delta(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} e^{-iwt} dt$$

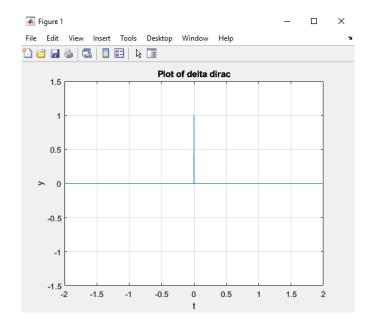
$$\int_{-\infty}^{\infty} A e^{-iwt} dt = A \int_{-\infty}^{\infty} e^{-iwt} dt = 2\pi A \delta(w)$$

$$F(A) = 2\pi A \delta(w)$$

$$A = 1 \Rightarrow F(1) = 2\pi \delta(w)$$

همانطور که توضیح دادیم، نمایش تبدیل فوریه حاصل از اعمال تابع fft، صحیح نیست، ولی پس از اعمال تابع fftshift، مطابق با محاسبات عملی روی نقطه صفر پیک می زند. Code:

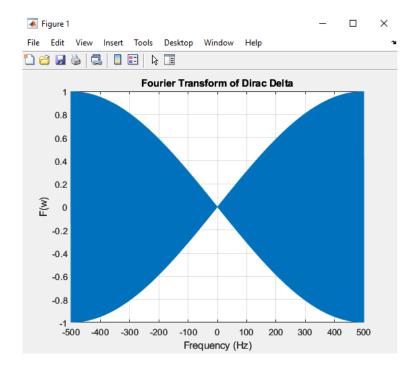
```
fourier_series_3.m × harmonic_1.m × harmonic_test.m × f_tranfrom_1.m ×
           fs = 1000;
 1
           t = -2:1/fs:2;
 2
          y = double(t==0);
4
5
           %Plot y=delta(t)
           plot(t,y)
          xlim([-2 2])
ylim([-1.5 1.5])
8
9
           xlabel('t')
          ylabel('y')
title('Plot of delta dirac')
10
11
12
           grid on
13
           %Use fft and fftshift to calculate the Fourier transform
14
15
           y1 = fft(y);
16
           f = (-length(y)/2:length(y)/2-1) * fs/length(y);
17
           %Plot y1
18
          %plot(f, y1)
%xlabel('Frequency (Hz)');
19
20
           %ylabel('F(w)');
21
22
           %title('Fourier Transform of Dirac Delta');
23
           %grid on
24
           y_shifted = fftshift(y1);
25
26
27
          %Plot y_shifted
          % plot(f, abs(y_shifted));
% xlabel('Frequency (Hz)');
% ylabel('F(w)');
28
29
30
31
           % ylim([-5 5])
32
           % title('Fourier Transform of Dirac Delta');
33
          % grid on
```



## محاسبه تبدیل فوریه با کمک دستور fft:

Code:

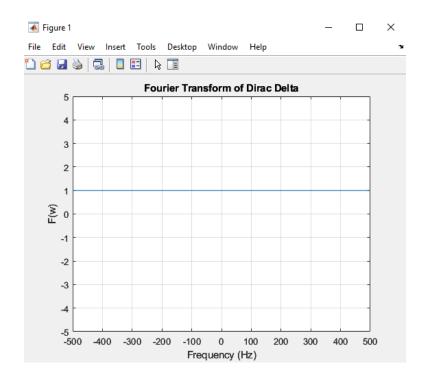
```
fourier_series_3.m × harmonic_1.m × harmonic_test.m × f_tranfrom_1.m × f_tran
          fs = 1000;
          t = -2:1/fs:2;
          y = double(t==0);
 3
 4
 5
          %Plot y=delta(t)
          % plot(t,y)
          % xlim([-2 2])
% ylim([-1.5 1.5])
 8
 9
          % xlabel('t')
          % ylabel('y')
% title('Plot of delta dirac')
10
11
12
          % grid on
13
          %Use fft and fftshift to calculate the Fourier transform
14
15
          y1 = fft(y);
16
          f = (-length(y)/2:length(y)/2-1) * fs/length(y);
17
          %Plot y1
18
          plot(f, y1)
xlabel('Frequency (Hz)');
19
20
21
          ylabel('F(w)');
22
          title('Fourier Transform of Dirac Delta');
23
          grid on
24
25
          y_shifted = fftshift(y1);
26
          Plot y_shifted
27
          % plot(f, abs(y_shifted));
28
          % xlabel('Frequency (Hz)');
29
          % ylabel('F(w)');
% ylim([-5 5])
30
31
32
          % title('Fourier Transform of Dirac Delta');
33
          % grid on
```



## محاسبه تبدیل فوریه با کمک دستور fftshift:

Code:

```
fourier_series_3.m × harmonic_1.m × harmonic_test.m × f_tranfrom_1.m × f_tranfr
  1
            fs = 1000;
  2
            t = -2:1/fs:2;
            y = double(t==0);
  3
  4
  5
           %Plot y=delta(t)
  6
            % plot(t,y)
  7
            % xlim([-2 2])
            % ylim([-1.5 1.5])
  8
  9
            % xlabel('t')
            % ylabel('y')
 10
            % title('Plot of delta dirac')
 11
 12
            % grid on
 13
            %Use fft and fftshift to calculate the Fourier transform
 14
            y1 = fft(y);
 15
 16
            f = (-length(y)/2:length(y)/2-1) * fs/length(y);
 17
 18
       巨
            %Plot y1
            %plot(f, y1)
%xlabel('Frequency (Hz)');
 19
 20
 21
            %ylabel('F(w)');
 22
            %title('Fourier Transform of Dirac Delta');
 23
            %grid on
 24
 25
            y_shifted = fftshift(y1);
 26
            %Plot y_shifted
plot(f, abs(y_shifted));
xlabel('Frequency (Hz)');
 27
 28
 29
 30
            ylabel('F(w)');
 31
            ylim([-5 5])
            title('Fourier Transform of Dirac Delta');
 32
 33
            grid on
 3/1
```



محاسبه تبدیل فوریه به صورت دستی:

$$f(t) = \delta(t)$$

$$F(w) = \int_{-\infty}^{\infty} f(t)e^{-iwt} dt = \int_{-\infty}^{\infty} \delta(t) e^{-iwt} dt = \int_{-\infty}^{\infty} \delta(t) e^{-iw(0)} dt = 1$$

$$F(\delta(t)) = 1$$

همانطور که توضیح دادیم، نمایش تبدیل فوریه حاصل از اعمال تابع fft، صحیح نیست، ولی پس از اعمال تابع fftshift، مطابق با محاسبات عملی مقدار ثابت یک نمایش داده می شود.

## 3.3 موسيقى

فایل audio داده شده را با کمک sound و sound باز می کنیم:

Code:

```
sound_1.m x +

file_path = 'ABITW.mp3';

[y, Fs] = audioread(file_path);

sound(y, Fs);
```

گزارش فرکانس نمونه برداری:

Code:

```
sound_1.m x sound_2.m x +

file_path = 'ABITW.mp3';

[y, Fs] = audioread(file_path);

printf('Sampling frequency : %d Hz', Fs);
```

Result:

Sampling frequency: 44100 Hz

## Nyquist-Shannon sampling theorem

قضیه نمونهبرداری نایکوئیست-شانون بیان میکند که یک سیگنال را میتوان از روی سیگنال نمونهبرداری شده به طور دقیق بازسازی کرد، اگر فرکانس نمونهبرداری بزرگتر از دو برابر بالاترین مولفه فرکانسی سیگنال باشد. در عمل، غالباً فرکانس نمونهبرداری را بزرگتر از دو برابر پهنای باند لازم در نظر میگیرند.

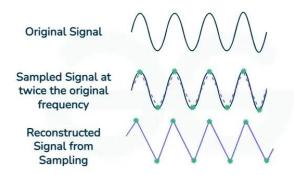
این کار به ما اطمینان می دهد که به از ای هر واحد زمانی، تعداد کافی نمونه برداری انجام شده است تا تمام جزئیات موج اصلی استخراج شود، بدون آنکه aliasing رخ دهد.

Aliasing اثری ناخواسته در سیگنال است که باعث کاهش کیفیت آن می شود. منظور از آن، تغییر کردن سیگنال اصلی پس از بازیابی سیگنال نمونه بر داری شده است. نمونه ای از aliasing دندانه دار شدن (پیکسلی شدن) تصویر دیجیتالی است.

بازه شنوایی انسان بین 20 Hz تا 20000 Hz است. بنابراین با توجه به قضیه نمونهبرداری نایکوئیست-شانون، برای جلوگیری از از دست رفتن اطلاعات، sample rate باید بزرگ تر از بیشترین مقدار ممکن برای فرکانس باشد. پس sampling frequency باید عددی بزرگتر از 40 KHz باشد.

دلیل اینکه sampling frequency را مقدار 44.1 KHz قرار می دهیم، نه 44 KHz این است که ممکن است خطایی در دستگاه ضبط صدا یا پخش صدا وجود داشته باشد و فرکانس با اندکی خطا اندازه گیری شود. این buffer یا فضای اضافی از بروز مشکلات در این شرایط جلوگیری می کند.

نحوه نمونه برداری و ساخت signal جدید:



فرکانس نمونه بر داری را دو بر ابر حالت اولیه کر ده و نتیجه را با کمک دستور audiowrite نخیره می کنیم:

Code:

```
sound_1.m x sound_2.m x sound_3.m x +

input_file = 'ABITW.mp3';

output_file = 'ABITW2.wav';

[y, Fs] = audioread(input_file);

new_Fs = 2 * Fs;

y_resample = resample(y, new_Fs, Fs);

audiowrite(output_file, y_resample, new_Fs);
```

فایل audioread را گزارش می دهیم: sound و audioread را گزارش می دهیم:

Code:

Result:

Sampling frequency: 88200 Hz

فر کانس نمونه بر داری را نصف حالت اولیه کرده و نتیجه را با کمک دستور audiowrite ذخیره می کنیم:

Code:

فایل audio جدید را با کمک audioread و sound باز می کنیم و sampling frequency را گزارش می دهیم:

Code:

Result:

Sampling frequency: 22050 Hz

همانطور که مشاهده می کنیم، حجم فایل ABITW2، تقریبا دو برابر حجم فایل ABITW است و حجم فایل ABITW3، تقریبا نصف حجم فایل ABITW3 است.

هرچه sampling frequency بیشتر باشد، تعداد نمونه های انتخاب شده در هر واحد زمانی بیشتر است، پس فضای مورد نیاز برای ذخیره سازی آنها نیز بیشتر خواهد بود، و بر عکس.

اما توجه کنید که در هر دو حالت، طول فایل یکسان باقی می ماند.