تمرین کامپیوتری اول

📡 سیگنال ها و سیستم ها 📡

مجید صادقی نژاد - ۸۱۰۱۰۱۴۵۹

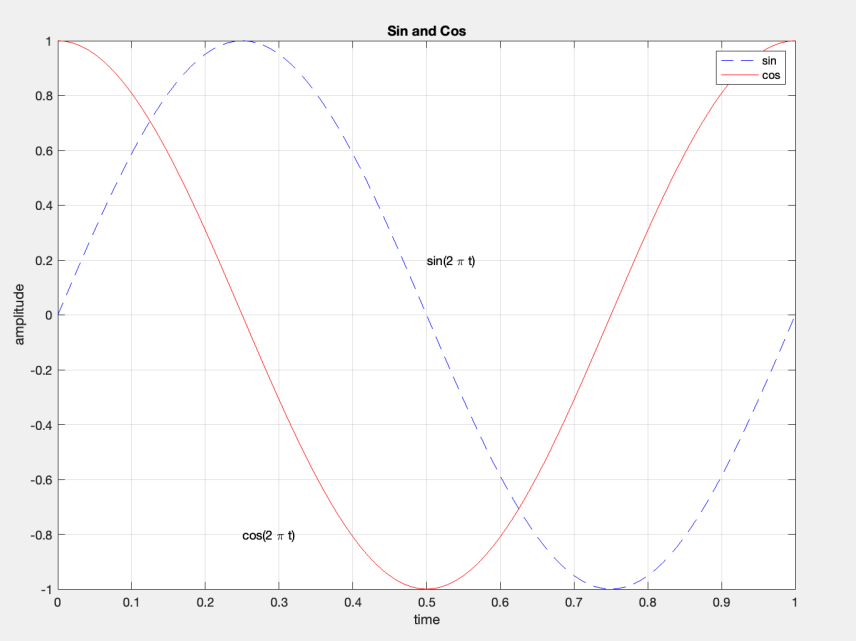
پاییز ۱۴۰۳

استاد: دکتر اخوان

بخش اول 🎓🎓

- تمرین ۱-۱

کد در بخش اول مقادیر t و دو تابع سینوس و کسینوس را بر حسب t تعریف می کند، سپس دو تابع را در یک صفحه می کشد در قطعه کد بعدی در دو نقطه از صفحه ی کشیده شده دو نوشته را نمایش می دهد و در نهایت به پلات های کشیده شده عنوان، راهنما، گرید (خط کشی) و لیبل محور های افقی و عمودی اضافه می کند.



در صورت حذف عبارت hold on فقط گراف دوم در صفحه کشیده می شود و گراف اول دیگر در صفحه وجود ندارد یعنی اگر بخواهیم تمام گراف ها در یک صفحه رسم شوند باید از دستور hold on استفاده کنیم.

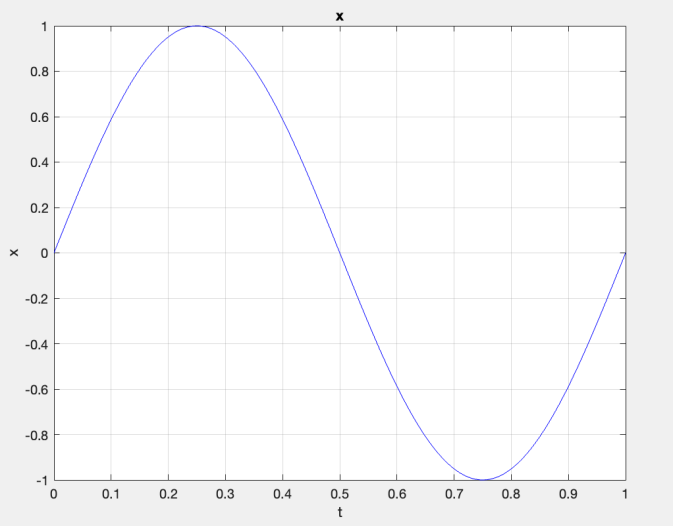
- تمرین ۱-۲

```m
clc;clear;close all

t= 0:0.01:1 ;
z1= sin(2*pi*t);
z2= cos(2*pi*t);

figure;

subplot(1,2,1);
plot(t, z1,'--b');
text(0.5, 0.2, 'sin(2 \pi t)');
title ('Sin');
legend('sin');
xlabel('time');
ylabel('amplitude');
grid on

subplot(1,2,2);
plot(t, z2, 'r');
text(0.25, -0.8, 'cos(2 \pi t)');
title ('Cos');
legend('cos');
xlabel('time');
ylabel('amplitude');
grid on
```

بخش دوم 📏 📐

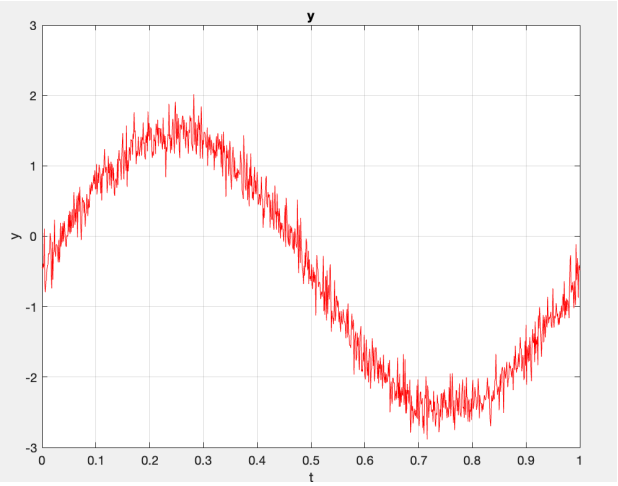
- تمرین ۲-۱

```m
figure('Name','x','NumberTitle','off');
plot(t,x,'b');
title ('x');
xlabel('t');
ylabel('x');
grid on
```



- تمرین ۲-۲

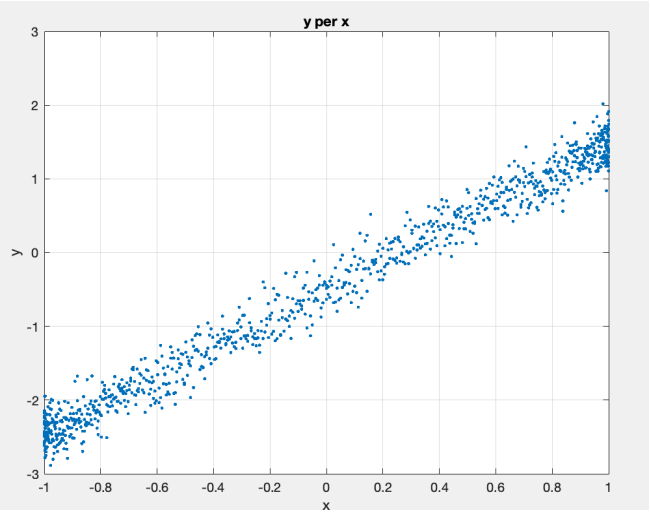
```
figure('Name','y','NumberTitle','off');
plot(t,y,'r');
title ('y');
xlabel('t');
ylabel('y');
grid on
```



- تمرین ۲-۳

با توجه به رابطه ی بین y , x و شکل رسم شده ی آن ها می توان دید که رابطه آن های تقریبا یک رابطه ی خطی است بنابراین شیب این خط برابر با α و عرض از مبدا آن برابر با β می باشد.

```
figure('Name','y per x','NumberTitle','off');
plot(x,y,'.');
title ('y per x');
xlabel('x');
ylabel('y');
grid on
```



- تمرین ۲-۴

از روش حداقل مربعات مانده استفاده می کنیم:

$$f = \sum_t (\alpha x(t) + \beta - y(t))^2 = minimum$$

$$\frac{df}{d\alpha} = 0 \Rightarrow \alpha \sum_t x^2(t) + \beta \sum_t x(t) = \sum_t x(t)y(t)$$

$$\frac{df}{d\beta} = 0 \Rightarrow \alpha \sum_t x(t) + N\beta = \sum_t y(t)$$

$$\Rightarrow \beta = \frac{\sum_t y(t) \times \sum_t x^2(t) - \sum_t x(t) \times \sum_t x(t)y(t)}{N \times \sum_t x^2(t) - \sum_t x(t) \times \sum_t x(t)} \Rightarrow \alpha = \frac{N \times \sum_t x(t)y(t) - \sum_t x(t) \times \sum_t y(t)}{N \times \sum_t x^2(t) - \sum_t x(t) \times \sum_t x(t)}$$

حال بر اساس رابطه هایی که در بالا به دست آوردیم function مورد نظر را پیاده سازی می کنیم:

```
function [a,B] = p2_4(x,y)
    N = length(x);
    S_xx = sum(x.*x,"all");
    S_y = sum(y,"all");
    S_xy = sum(x.*y,"all");
    S_x = sum(x,"all");

    a = (N*S_xy - S_x*S_y) / (N*S_xx - S_x*S_x);
    B = (S_y*S_xx - S_x*S_xy) / (N*S_xx - S_x*S_x);
end
```

در ابتدا مقادیر α, β سوال را به دست آورده و سپس به تست تابع می پردازیم:

```
load("p2.mat");

[a,B] = p2_4(x,y) ;
disp(['a = ', num2str(a),' , B = ', num2str(B)]);
```

و پاسخ به دست آمده ی نهایی برای مقادیر خواسته شده به شکل زیر است:

```
a = 1.9736 , B = -0.49834
```

برای تست فانکشن نوشته شده مقادیر α, β, x جدید تولید می کنیم و یکبار بدون حضور noise و بار دیگر با حضور آن، فانکشن را تست می کنیم در این تست $\alpha = 2.45 , \beta = -11.5$ می باشد.

```
a_test = 2.45 ;
B_test = -11.5 ;
x_test = rand(1,1001);
noise = randn(1,1001) * 0.5;
y_test = a_test * x_test + B_test + noise;
y_test_2 = a_test * x_test + B_test ;

disp(['a_test = ', num2str(a_test),' , B_test = ', num2str(B_test)]);

[a_ans , B_ans] = p2_4(x_test,y_test);
disp('with noise:')
disp(['a_ans = ', num2str(a_ans),' , B_ans = ', num2str(B_ans)]);

[a_ans_2 , B_ans_2] = p2_4(x_test,y_test_2);
disp('without noise:');
disp(['a_ans_2 = ', num2str(a_ans_2),' , B_ans_2 = ', num2str(B_ans_2)]);
```

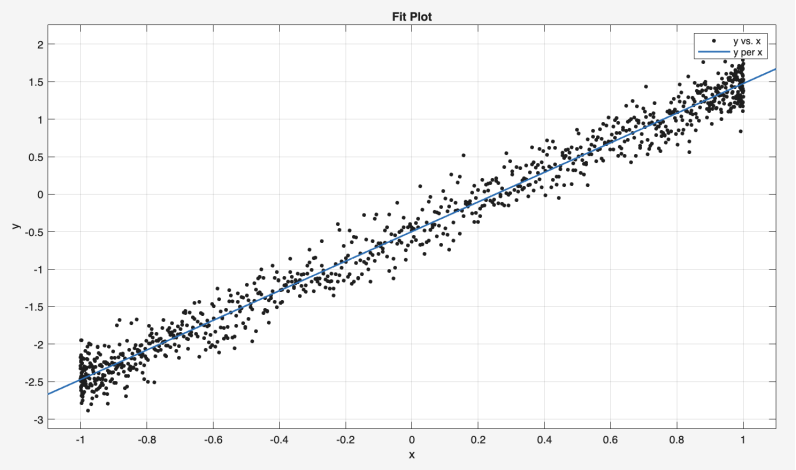
نتیجه ی تست به صورت زیر است:

```
a_test = 2.45 , B_test = -11.5
with noise:
a_ans = 2.4039 , B_ans = -11.5058
without noise:
a_ans_2 = 2.45 , B_ans_2 = -11.5
```

که نشان می دهد فانکشن نوشته شده به خوبی مقادیر خواسته شده را تخمین می زند.

- ## تمرین ۲-۵

با استفاده از curve fitter به شکل و مقادیر زیر می رسیم:



Fit Name: y per x

Polynomial Curve Fit (poly1)

f(x) = p1*x + p2

Coefficients and 95% Confidence Bounds

	Value	Lower	Upper
p1	1.9736	1.9560	1.9911
p2	-0.4983	-0.5107	-0.4859

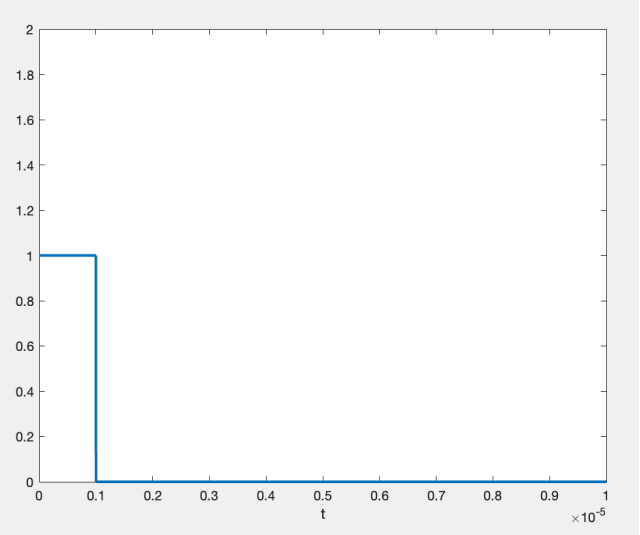
مشاهده می شود مقادیری که این نرم افزار به دست آورد دقیقا با مقادیر به دست آمده توسط فانکشن همخوانی دارد.

بخش سوم 📡 📡

در این تمرین سرعت نور در تمامی بخش ها 3×10^8 در نظر گرفته شده است.

- ## تمرین ۳-۱

تصویر مربوط به سیگنال ارسالی:

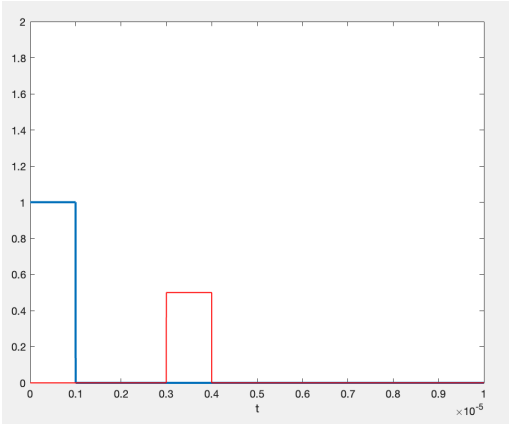


کد مربوط به تولید سیگنال ارسالی:

```
ts = 1e-9; T = 1e-5; tau = 1e-6;
R = 450; C = 3e8; a = 0.5;
t = 0:ts:T;
td = (2*R)/C;
tau_index = round(tau/ts)+1;

sended_signal = zeros(1,length(t));
sended_signal(1:tau_index) = 1;
```

- ## تمرین ۳-۲

تصویر مربوط به سیگنالی دریافتی و ارسالی (سیگنال دریافتی قرمز رنگ است):



و کد مربوط به این سیگنال دریافتی به شکل زیر است:

```
recieved_signal = zeros(1,length(t));
recieved_signal((td/ts)+1:round((td+tau)/ts)+1) = a;
```

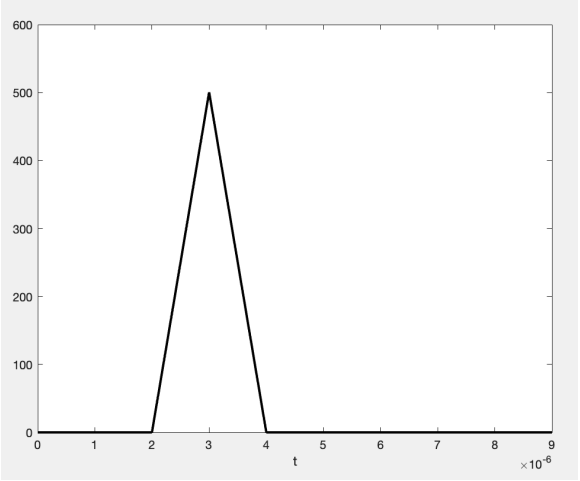
- ## تمرین ۳-۳

با استفاده از روش correlation به وسیله کد زیر، گراف correlation رسم می شود همچنین نقطه ماکسیمم این گراف را می یابیم و به وسیله ی آن زمان که طول کشیده تا سیگنال دریافت شود را به دست می آوریم و سپس بر اساس رابطه ای که داشتیم R را می یابیم.

کد مربوط به بخش یافتن R:

```
sample_signal = ones(1,tau_index);
correlation = zeros(1,((T-tau)/ts)+1);

for k=1:((T-tau)/ts)+1
    correlation(k) = recieved_signal(1,k:k+tau_index-1)*(sample_signal');
end

[~, peak_index]= max(correlation);
td_ans = (peak_index-1)*ts;
R_ans = td_ans*C/2 ;
disp(['the R ,that we found it, is ', num2str(R_ans)]);
```

گراف مربوط به correlation:



و در نهایت جواب محاسبه شده برابر 450m می باشد که همان چیزی است که انتظار داشتیم:

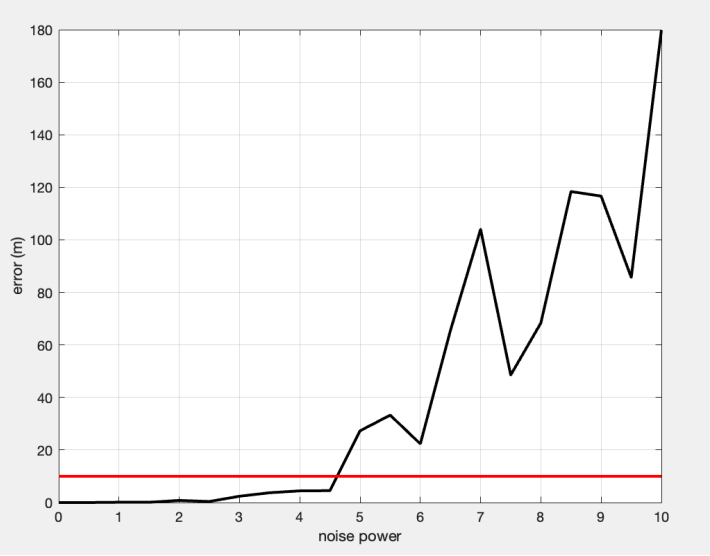
```
the R ,that we found it, is 450
```

- ## تمرین ۳-۴

برای این آزمایش نویز را به وسیله فانکشن randn تولید می کنیم و قدرت نویز (عددی که در مقادیر تصادفی تولید شده ضرب می شود) را از ۰ تا ۱۰ با گام ۰.۵ افزایش می دهیم سپس برای هر قدرت نویز ۱۰۰ بار با اعداد تصادفی جدید مرحله پیدا کردن R را تکرار می کنیم سپس میانگین R های پیدا شده برای آن قدرت نویز مشخص را محاسبه کرده و در نهایت خطای R را برای آن قدرت نویز محاسبه می کنیم کد نوشته شده برای این بخش به صورت زیر است:

```
num_of_exams = 100;
noise_power_ceil = 10;
founded_Rs = zeros(1,length(0:0.5:noise_power_ceil));

for i=0:0.5:noise_power_ceil
    for j=1:num_of_exams
        noisy_signal = recieved_signal + randn(1,length(t)) * i;
        for k=1:((T-tau)/ts)+1
            correlation(k) = noisy_signal(1,k:k+tau_index-1)*(sample_signal');
        end
        [~, peak_index]= max(correlation);
        td_ans = (peak_index-1)*ts;
        founded_Rs(1,i*2+1) = founded_Rs(1,i*2+1) + (td_ans*C/2) ;
    end
    founded_Rs(1,i*2+1) = founded_Rs(1,i*2+1) / num_of_exams ;
end

Rs_error = abs(founded_Rs - R) ;
```

و گراف به دست آمده از این خطا ها به شکل زیر است. محور افقی نشان دهنده ی قدرت نویز، محور عمودی میزان خطا با واحد متر، گراف سیاه رنگ نشان دهنده ی خطا هاست ، از آن جا که گام های ما ۰.۵ است نقاط به دست آمده با خط به هم وصل شده اند و در نهایت خط قرمز رنگ نشان دهنده ی خطای ۱۰ متر است. همانطور که مشاهده می شود تا قدرت نویز ۴.۵ میزان خطا قابل قبول است و برنامه ی ما به درستی کار می کند.



بخش چهارم 🎤 🎤

- ## تمرین ۴-۱

کد مربوطه:

```
clc;clear;close all

[x,fs] = audioread("voice.wav");
disp(['fs = ',num2str(fs)]);
```

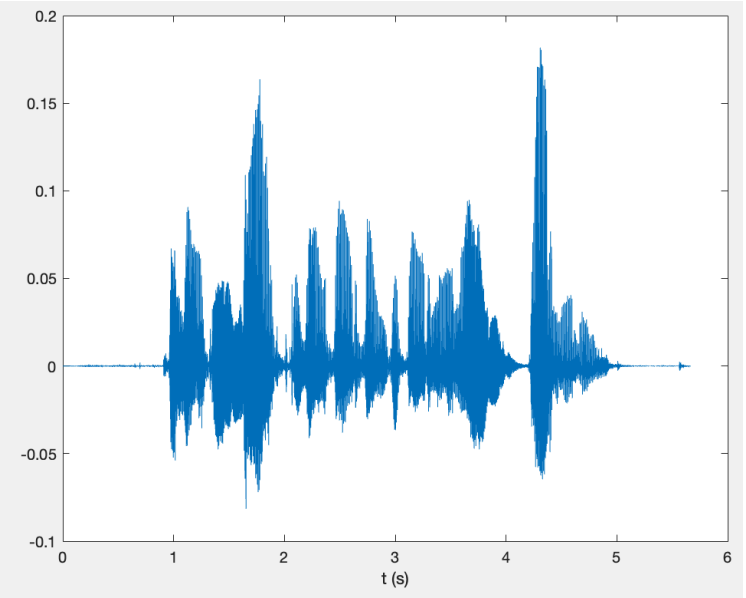
فرکانس نمونه برداری:

```
fs = 44100
```

- ## تمرین ۴-۲

برای لیبل گذاری صحیح باید محور افقی را از صفر تا تعداد نمونه های برداشته شده در نظر بگیرم و برای اینکه مقیاس آن تبدیل به ثانیه شود باید هر عضو از محور افقی ضربدر دوره تناوب نمونه برداری شود (عکس فرکانس):

```
figure;
plot((0:length(x)-1) * (1/fs),x);
xlabel('t (s)');

sound(x,fs);
audiowrite('x.wav',x,fs);
```



- ## تمرین ۴-۳

برای این کار ما نیاز به آن نمونه هایی از صدای اولیه داریم که اندیس آن ها برابر اندیس نمونه های صدای جدید ضربدر سرعت باشد برای مثال برای سرعت نیم ما نیاز به اندیس های ۰،۰.۵،۱،۱.۵ و ... از صدای اولیه داریم در صورتی که اندیس های غیر صحیح وجود خارجی ندارند ! پس باید این اندیس ها را با میانگین گیری خطی از دو نمونه صحیح دو طرف شان به دست آوریم ! کد راه کار بالا در زیر آمده است. (تست این تابع در فایل p4.m آمده است):

```
function new_audio = p4_3(audio_file,speed)
    [audio,fs] = audioread(audio_file);

    if (speed ~= 2 && speed ~= 0.5)
        error('just 2 and 0.5 are accepted for speed');
    end

    new_audio = zeros(length(0:speed:length(audio)-1),1);
    index = 0 ;

    for i=0:speed:length(audio)-1
        c = ceil(i) ; f = floor(i);
        if (c == f)
            new_audio(index+1) = audio(i+1);
        else
            new_audio(index+1) = (c-i) * audio(f+1) + (i-f) * audio(c+1) ;
        end
        index = index + 1 ;
    end

    sound(new_audio,fs);

end
```

- ## تمرین ۴-۴

دقیقا همانند تابع قبل عمل می کند با این تفاوت که هنگام میانگین گیری از میانگین وزن دار استفاده می کند برای مثال اگر به اندیس ۱.۷ نیاز داشتیم وزن اندیس ۱ ام برابر ۰.۳ و وزن اندیس ۲ ام برابر ۰.۷ است. (تست این تابع در فایل p4.m آمده است)

```
function new_audio = p4_4(audio_file,speed)
    [audio,fs] = audioread(audio_file);

    new_audio = zeros(length(0:speed:length(audio)-1),1);
    index = 0 ;

    for i=0:speed:length(audio)-1
        c = ceil(i) ; f = floor(i);
        if (c == f)
            new_audio(index+1) = audio(i+1);
        else
            new_audio(index+1) = (c-i) * audio(f+1) + (i-f) * audio(c+1) ;
        end
        index = index + 1 ;
    end

    sound(new_audio,fs);

end
```