

1. 实验1

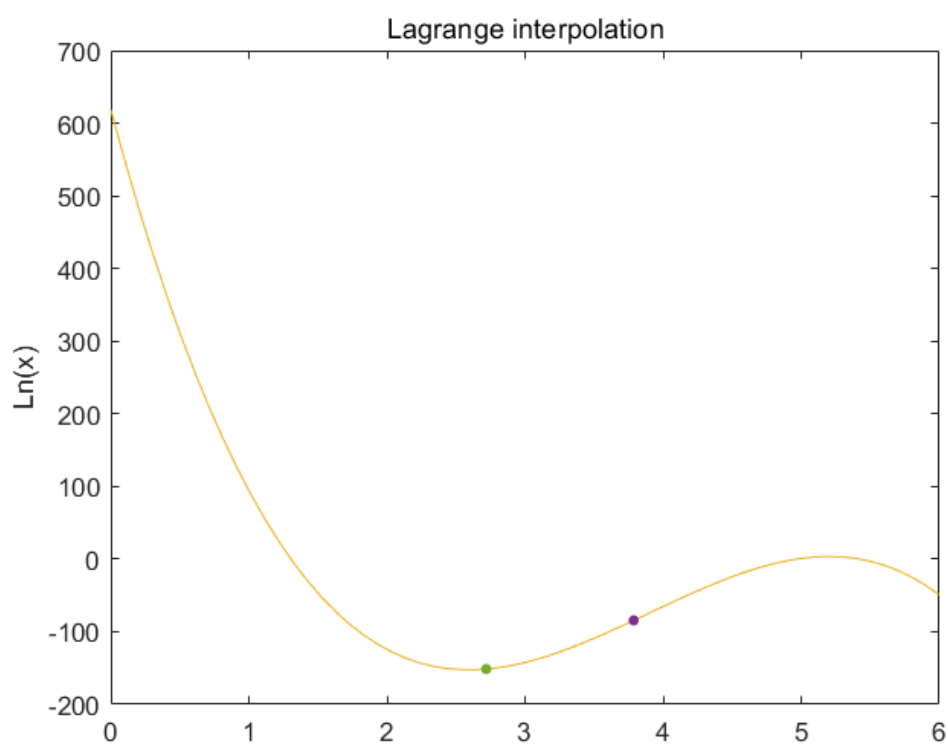
1.1. Lagrange插值

1. Lagrange 插值公式:

$$Ln(x) = 17.612 * x^3 + 205.97 * x^2 - 713.08 * x + 618.77;$$

2. 绘制曲线及计算坐标:

$$x_i = [3.79, 2.72]; \quad y_i = [-84.39, -151.44];$$



3. 参数截图:

名称 ▲	值
a	[5.1625,5.0180,1.2966,1.2875]
ans	<i>1x2 sym</i>
b	[3.5451,1.4296,2.0536,4.4987]
i	4
k	4
l	<i>4x1 sym</i>
L	<i>1x1 sym</i>
L1	<i>4x4 double</i>
m	4
V	[-7.6120,87.3642,-297.6731,255.6836]
x	<i>1x60 double</i>
X	[5.1625,5.0180,1.2966,1.2875]
x0	[3.7868,2.7190]
Y	[3.5451,1.4296,2.0536,4.4987]
y0	[-84.3891,-151.4350]

4. 代码附录:

```
%exp1.1 2020/11/7 zgz
rand('seed',1851960);
%points
a = 1+5*rand(1,4);
b = 1+5*rand(1,4);

%construct coordinates
x = linspace(0,6,60);
y = lagrange(a,b,x);

%draw
plot(x,y);
ylabel('Ln(x)');
title('Lagrange interpolation');
hold on

%draw additional points
x0 = 2 + 3*rand(1,2);
y0 = lagrange(a,b,x0);
for i = 1:2
    plot(x0(1,i),y0(1,i),'.','MarkerSize',14);
end
```

```
function y=lagrange(x0,y0,x)
```

%此函数用于形成拉格朗日插值公式;

```
n = length(x0);
m = length(x);
y = zeros(1,m);
for i=1:m
    z=x(i);s=0;
    for k=1:n
        L=1;
        for j=1:n
            if j~=k
```

```

        L=L*(z-x0(j))/(x0(k)-x0(j));
    end
end
s=s+L*y0(k);
end
y(i)=s;
end

```

```

%%%% 求拉格朗日多项式及基函数  %%%%
%%%%%                               %%%%
%%%%%           Liu Deping          %%%%
%%%%%           2020.06.14          %%%%
%输入的量:n+1个节点(x_i,y_i)(i = 1,2, ..., n+1)横坐标向量X, 纵坐标向量Y
%输出的量: n次拉格朗日插值多项式L和基函数l
X=input('请输入横坐标向量X:\nX='); %输入的数据为一维数组, 例如: [1,3,4,5] (下同);
Y=input('请输入纵坐标向量Y:\nY=');
m = length(X);
L = ones(m,m);
for k = 1 : m
    v = 1;
    for i = 1 : m
        if k ~= i
            v = conv(v,poly(X(i))) / (X(k) - X(i));
        end
    end
    L1(k, :) = v;
    l(k, :) = poly2sym(v);
end
fprintf('基函数为: \n');
for k=1:m
    fprintf('q%d(x)=%s\n',k,l(k));
end
L = Y * l;
fprintf('拉格朗日多项式为:\nP(x)=%s\n',L);

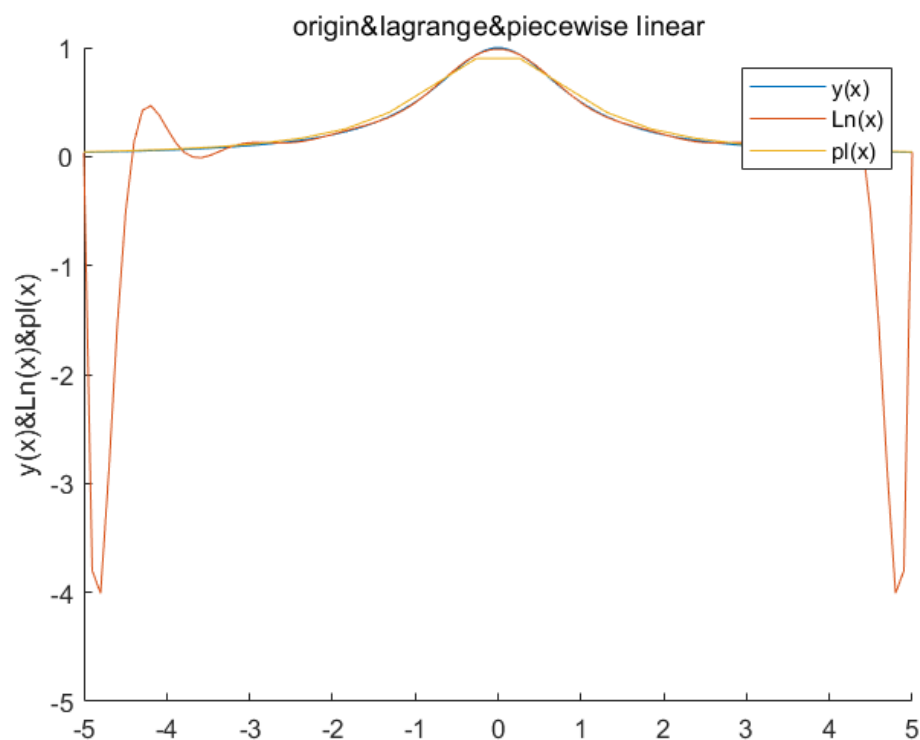
```

1.2. 观察Runge现象及分段线性插值

1. Lagrange 插值公式:

$$\begin{aligned}
 L_n(x) = & -3.202e^{-23} * x^{17} + 4.093e^{-8} * x^{16} + 1.437e^{-19} * x^{15} - 3.472e^{-6} * x^{14} \\
 & + 4.289e^{-19} * x^{13} + 0.0001186 * x^{12} - 3.003e^{-16} * x^{11} - 0.002114 * x^{10} - 3.541e^{-15} * x^9 \\
 & + 0.02131 * x^8 - 1.227e^{-14} * x^7 - 0.1241 * x^6 - 1.573e^{-14} * x^5 + 0.4157 * x^4 \\
 & - 4.154e^{-16} * x^3 - 0.803 * x^2 - 9.729e^{-16} * x + 0.9868
 \end{aligned}$$

2. 绘制曲线, 发现在横坐标两端出现龙格现象, 即发生过拟合; 分段线性插值 (pl) 绘图如下:



3. 分析:

一般情况下，多项式的次数越多，需要的数据就越多，而预测也就越准确；插值次数越高，插值结果越偏离原函数。为了解决 Runge 现象，可采取分段线性插值，使得用分段线性函数进行拟合，避免了过拟合，但是函数失去了平滑性，即出现了不可导点。

4. 参数截图:

名称 ▲	值
a	[27.0389,15.8748,15.8476,22.6203,16.2739]
ans	1x1 sym
b	[59.9833,51.6518,23.1771,9.7016,6.8645]
i	18
k	18
l	18x1 sym
L	1x1 sym
L1	18x18 double
m	18
n	18
p	[1.3795,-56.3655,576.4023]
V	1x18 double
x	1x100 double
X	1x18 double
x0	1x18 double
x3	1x20 double
y	1x25 double
Y	1x18 double
y0	1x18 double
y1	1x100 double
y2	1x100 double
y3	1x20 double

5. 代码附录:

```

%exp1.2 2020/11/7 zgz
rand('seed',1851960);
%construct coordinates
n = randi([8,20],1,1);
x0 = linspace(-5,5,18);
y0 = 1./(1+x0.^2);
x = linspace(-5,5,100);

%original
y1 = 1./(1+x.^2);

%lagrange
y2 = lagrange(x0,y0,x);

%p1
x3 = linspace(-5,5,20);
y3 = p1(20,-5,5);

%draw
plot(x,y1);
hold on
plot(x,y2);
hold on
plot(x3,y3);
ylabel('y(x)&Ln(x)&p1(x)');
legend('y(x)', 'Ln(x)', 'p1(x)');
title('origin&lagrange&piecewise linear');

%draw additional points
hold on

```

```

function Sn=p1(L,b1,b2)
%当在区间内取i个等距节点时对应的小区间的中点值Si并绘制出图形
%b1代表左边界，b2代表右边界
%L可以是一个数组，也可以是一个数字

%插值绘图
n=length(L);
for i=1:n
    s=L(i);
    L1=linspace(b1,b2,s+1);
    for j=2:s+1
        x(j-1)=(L1(j-1)+L1(j))/2; %寻找两端点中点值
        Sn(j-1) = (((x(j-1)-L1(j))/(L1(j-1)-L1(j)))/(1+L1(j-1)^2))+
        (((x(j-1)-L1(j-1))/(L1(j)-L1(j-1)))/(1+L1(j)^2)) %中点值函数值
    end
end
hold on
end

```

```

%%%% 求拉格朗日多项式及基函数 %%%
%%%% Liu Deping %%%
%%%% 2020.06.14 %%%
%输入的量:n+1个节点(x_i,y_i)(i = 1,2, ..., n+1)横坐标向量X，纵坐标向量Y
%输出的量: n次拉格朗日插值多项式L和基函数l

```

```

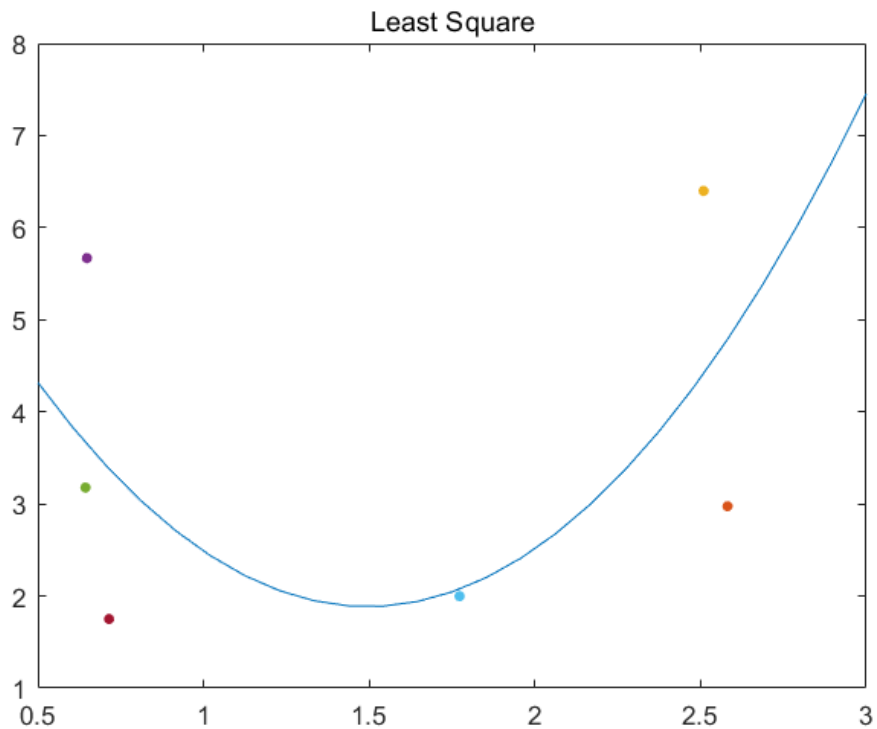
X=input('请输入横坐标向量X:\nX='); %输入的数据为一维数组，例如：[1,3,4,5]（下同）；
Y=input('请输入纵坐标向量Y:\nY=');
m = length(X);
L = ones(m,m);
for k = 1 : m
    V = 1;
    for i = 1 : m
        if k ~= i
            V = conv(V,poly(X(i))) / (X(k) - X(i));
        end
    end
    L1(k, :) = V;
    l(k, :) = poly2sym(V);
end
fprintf('基函数为: \n');
for k=1:m
    fprintf('q%d(x)=%s\n',k,l(k));
end
L = Y * l;
fprintf('拉格朗日多项式为:\nP(x)=%s\n',L);

```

2. 实验2

2.1. 最小二乘法拟合（用polyfit函数）

1. 绘制曲线和数据点：



2. 参数截图：

名称 ▲	值
a	[2.5812,2.5090,0.6483,0.6438,1.7725,0.7148]
b	[2.9751,6.3982,5.6692,3.1777,1.9986,1.7503]
i	6
p	[2.4593,-7.3530,7.3806]
x	1x25 double
y	1x25 double

3. 代码附录:

```
%exp2.1 2020/11/7 zgz
rand('seed',1851960);
%points
a = 0.5+2.5*rand(1,6);
b = 1.5+7*rand(1,6);

%fit
p = polyfit(a,b,2);

%construct coordinates
x = linspace(0.5,3,25);
y = polyval(p,x);

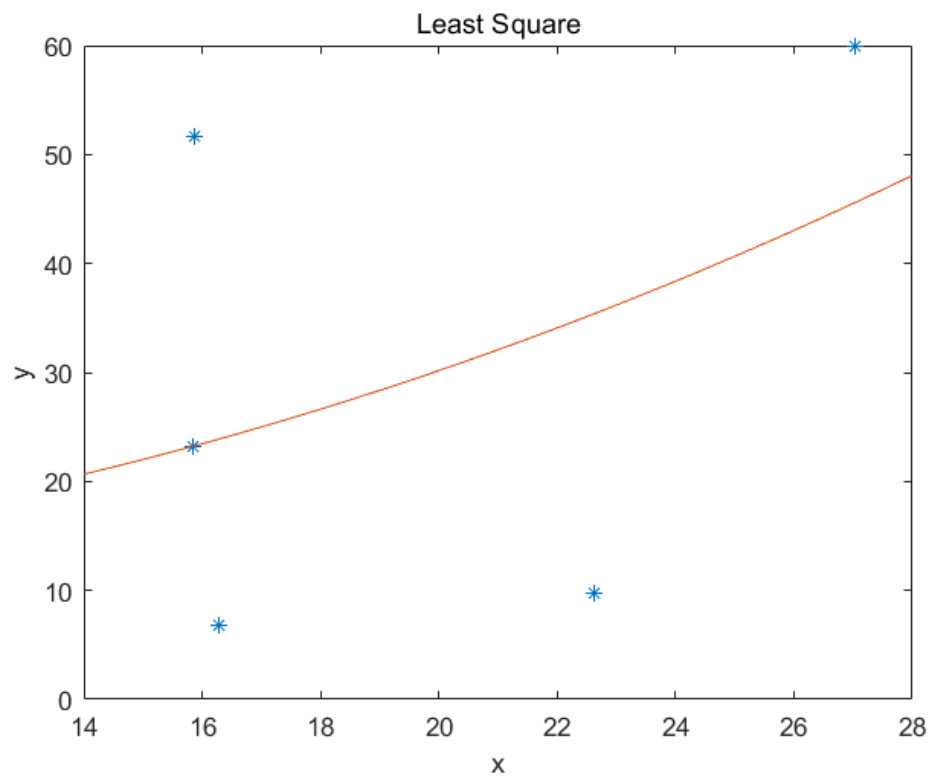
%draw
plot(x,y);
ylabel('');
title('Least Square');
hold on

%draw additional points
for i = 1:6
    plot(a(i),b(i),'.','MarkerSize',14);
    hold on
end
```

2.2. 最小二乘法拟合（用矩形方法求解）

1. 拟合结果:

$$y = 11.53 + 0.047x^2;$$



2. 参数截图:

名称 ▲	值
a	[27.0389,15.8748,15.8476,22.6203,16.2739]
ans	0.0466
b	[59.9833,51.6518,23.1771,9.7016,6.8645]
fun	<i>1x1 inline</i>
i	21
n	[11.5290,0.0466]
n0	[1,1]
s	<i>1x1 sym</i>
t	<i>1x1 sym</i>
x	<i>1x100 double</i>
y	<i>22x1 double</i>
z	<i>22x1 double</i>

3. 代码附录:

```
%exp2_2 2020/11/7 zgz
rand('seed',1851960);
%init
a = 18*rand(1)+15*rand(1,5);
b = 19*rand(1)+80*rand(1,5);
x = linspace(14,28,100);

fun = inline('n(1)+n(2)*x.^2','n','x');
n0 = [1,1];
n=nlinfit(a,b,fun,n0);
plot(a,b,'*');
hold on
plot(x,n(1)+n(2)*x.^2);
```


2.3. Newton-Cotes系列数值求积公式

1. 比较精度 (小区间的宽度取 $w = 0.01$) :

矩形求积法误差: $loss_rec = 0.0012$;

梯形求积法误差: $loss_tra = 7.4162e^{-6}$;

因此梯形求积法误差精度较高;

2. 参数截图:

名称 ▲	值
a	9
b	13
f	@(t)exp(-0.5*t).*(t+pi/b)
i	6
loss_rec	0.0012
loss_tra	7.4162e-06
p	[2.4593,-7.3530,7.3806]
s	4.4833
s_rec	4.4845
s_tra	4.4833
x	1x25 double
y	1x25 double

3. 代码附录:

```
%exp2_3 2020/11/7 zgz
rand('seed',1851960);
%init
a = randi(10,1,1);
b = randi([5 15],1,1);

%calculate sum
% y = exp(-0.5*t).*(t+pi/b);
s_rec = rec_sum(0,a*pi,b);%rec
s_tra = tra_sum(0,a*pi,b);%tra

%calculate loss
f = @(t) exp(-0.5*t).*(t+pi/b);
s = integral(f,0,a*pi);
loss_rec = abs(s-s_rec);
loss_tra = abs(s-s_tra);
```

```
function rec_sum = rec_sum(floor,ceiling,b)
%矩形法求定积分

%init
s = 0;
i = 1;
w = 0.01; %矩形的宽度为0.01
n = (ceiling-floor)/w; %份数
t = floor;
y = exp(-0.5*t).*(t+pi/b);
```

```

while i < n
    s = s+w*y;
    t = t+w;
    y = exp(-0.5*t)*(t+pi/b);
    i = i+1;
end

rec_sum = s;

end

```

```

function tra_sum = tra_sum(floor,ceiling,b)
%梯形法求定积分

%init
s = 0;
i = 1;
w = 0.01; %梯形的宽度为0.01
n = (ceiling-floor)/w; %份数
t = floor;
y1 = exp(-0.5*t).*(t+pi/b);
y2 = exp(-0.5*(t+w)).*((t+w)+pi/b);

while i < n
    s = s+w*(y1+y2)/2;
    t = t+w;
    y1 = y2;
    y2 = exp(-0.5*(t+w)).*((t+w)+pi/b);
    i = i+1;
end

tra_sum = s;





end

```

2.4. Romberg求积公式

1. 积分值: 3.57174

2. 参数截图:

名称 ▲	值
 a	2.0812
 b	2.4108
 fun	@(x)x^b
 s	3.5717

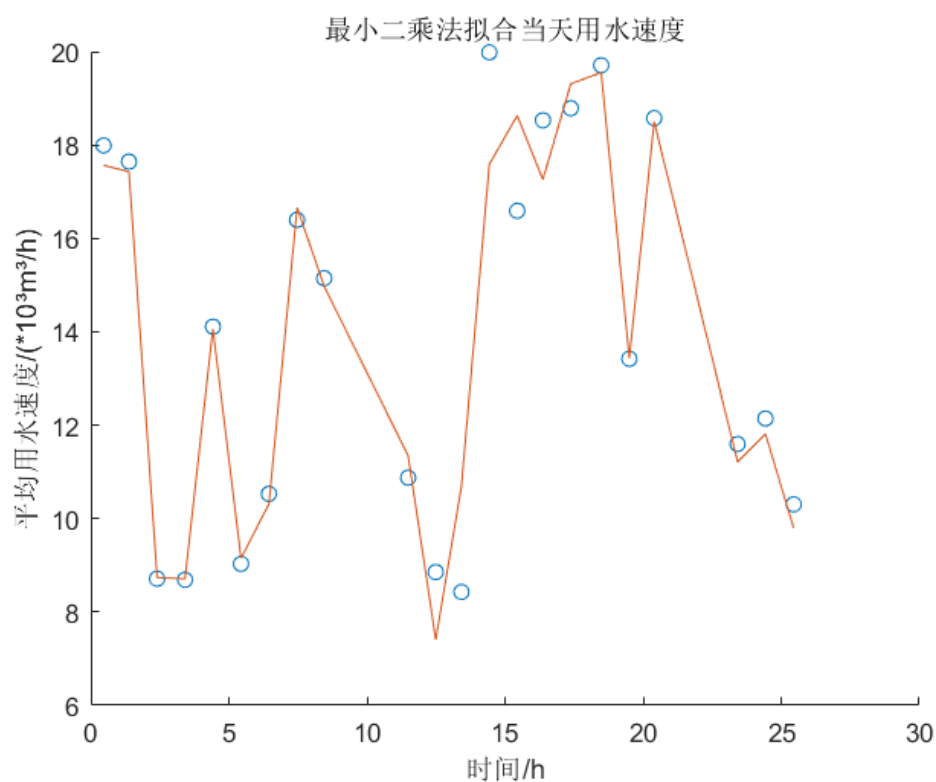
3. 代码附录:

```
%exp2_4 2020/11/7 zgz
rand('seed',1851960);
%init
a = 2.5*rand(1,1);
b = 3*rand(1,1);

%calculate sum
fun = @(x) x^b;
[s,~,~] = romberg(fun,0,a,1e-5);
```

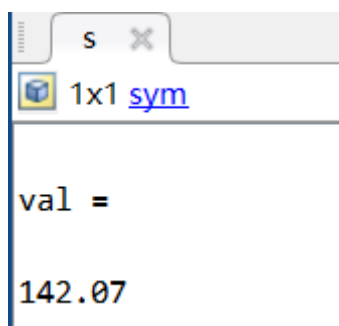
2.5. 估计某地居民的用水速度和每天用水量

1. 拟合该小区当天用水速度的图像如下：



总用水量估计： $142.07 \times 10^3 m^3$ 。

2. 参数设置：



x x y x			x x y x			a x		
22x1 double			22x1 double			21x1 double		
	1			1			1	
1	0.4600		1	17.9900		1	-8.3607e-...	
2	1.3800		2	17.6432		2	2.1003e-13	
3	2.4000		3	8.7119		3	-2.4519e-...	
4	3.4100		4	8.6901		4	1.7659e-09	
5	4.4200		5	14.1082		5	-8.7849e-...	
6	5.4400		6	9.0311		6	3.2029e-06	
7	6.4500		7	10.5287		7	-8.8612e-...	
8	7.4700		8	16.3969		8	0.0019	
9	8.4500		9	15.1472		9	-0.0320	
10	11.4900		10	10.8760		10	0.4261	
11	12.4900		11	8.8547		11	-4.4933	
12	13.4200		12	8.4291		12	37.4582	
13	14.4300		13	19.9841		13	-245.2985	
14	15.4400		14	16.5906		14	1.2485e+...	
15	16.3700		15	18.5284		15	-4.8607e...	
16	17.3800		16	18.7862		16	1.4148e+...	
17	18.4800		17	19.7069		17	-2.9789e...	
18	19.5000		18	13.4205		18	4.3194e+...	
19	20.4000		19	18.5751		19	-3.9843e...	
20	23.4200		20	11.5966		20	2.0215e+...	
21	24.4300		21	12.1434		21	-3.9715e...	

名称 ▲	值
a	21x1 double
ans	1x1 sym
b	2.4108
fun	@(x)x^b
i	21
s	1x1 sym
t	1x1 sym
x	22x1 double
y	22x1 double
z	22x1 double

3. 代码附录:

```
%exp2_5 2020/11/7 zgz
rand('seed',1851960);
%init
x =
[0.46,1.38,2.40,3.41,4.42,5.44,6.45,7.47,8.45,11.49,12.49,13.42,14.43,15
.44,16.37,17.38,18.48,19.50,20.40,23.42,24.43,25.45]';
y = 8+12*rand(22,1);
a = polyfit(x,y,20);
z = polyval(a,x);
```

```
syms t

%cal int
s = 0;
for i = 1:21
    s = s+int(a(1,i)*t^(21-i),t,0,24);
end
s = vpa(s,5);

scatter(x, y);
hold on;
plot(x, z);
hold on;
```