作业8 数据建模

1. 用下列数据拟函数, 求出a, b的值, 并作出拟合效果图。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| x | 0 | 0.2 | 0.4 | 0.6 | 0.8 | 1 |
| y | 4.0 | 4.5 | 5.0 | 6.0 | 6.8 | 7.7 |

**代码：**

*% 数据*

x **=** [0, 0.2, 0.4, 0.6, 0.8, 1];

y **=** [4, 4.5, 5, 6, 6.8, 7.7];

*% 定义模型函数*

fun **=** **@**(p,x) 3 **+** p(1)**\***x **+** exp(p(2)**\***x);

*% 初始猜测值*

p0 **=** [1, 1];

*% 使用 lsqcurvefit 进行拟合*

[p,resnorm,residual,exitflag,output] **=** lsqcurvefit(fun, p0, x, y);

*% 打印结果*

disp(['a = ', num2str(p(1))]);

disp(['b = ', num2str(p(2))]);

*% 作出拟合效果图*

xmodel **=** linspace(min(x), max(x), 100);

ymodel **=** fun(p, xmodel);

figure;

plot(x, y, '\*'); *% 数据点*

hold on;

plot(xmodel, ymodel, 'b-'); *% 拟合曲线*

xlabel('x'); *% x轴标签*

ylabel('y'); *% y轴标签*

title('非线性最小二乘拟合'); *% 标题*

legend('数据', '拟合曲线', 'Location', 'best'); *% 图例*

hold off;

**输出：**

找到局部最小值。优化已完成，因为梯度大小小于最优性容差的值。<停止条件详细信息>

a = 1.302

b = 1.2379

2. 一矿脉有13个相邻样本点，人为地设定一原点，现测得各样本点对原点的距离x与样本点处某种金属含量y的一组数据如下，画出散点图观察合适的函数模型，如二次曲线、双曲线、对数曲线等.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| x | 2 | 3 | 4 | 5 | 7 | 8 | 10 |
| y | 106.42 | 109.20 | 109.58 | 109.50 | 110.00 | 109.93 | 110.49 |
| x | 11 | 14 | 15 | 15 | 18 | 19 |  |
| y | 110.59 | 110.60 | 110.90 | 110.76 | 111.00 | 111.20 |  |

**代码：**

*% 选择最优拟合曲线*

RSSs **=** [RSS1, RSS2, RSS\_log, RSS\_hyp];

[min\_RSS, idx] **=** min(RSSs);

fprintf('最优拟合曲线为\n');

**switch** idx

**case** 1

disp(str1);

str1 **=** [str1,' (最优)'];

**case** 2

disp(str2);

str2 **=** [str2,' (最优)'];

**case** 3

disp(str\_log);

str\_log **=** [str\_log,' (最优)'];

**case** 4

disp(str\_hyp);

str\_hyp **=** [str\_hyp,' (最优)'];

**end**

xlabel('样点与原点的距离');

ylabel('金属含量');

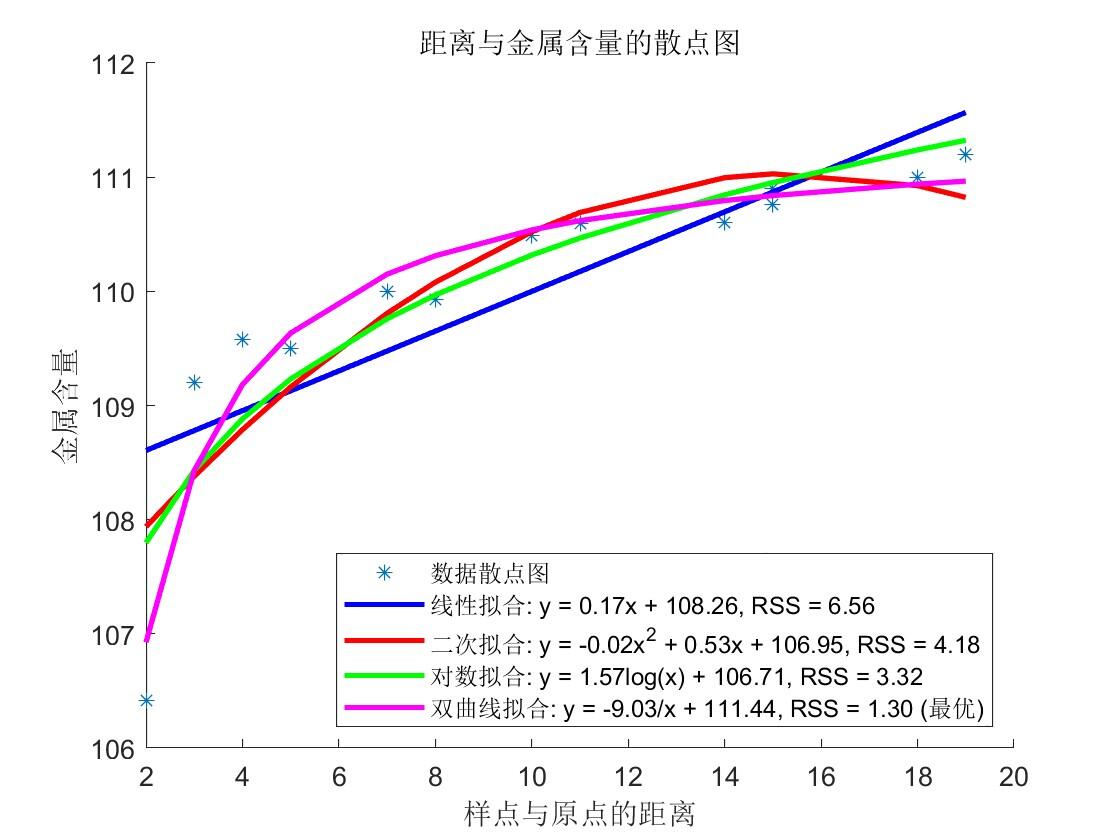
legend({'数据散点图',str1, str2, str\_log, str\_hyp},'Location','southeast');

hold off;

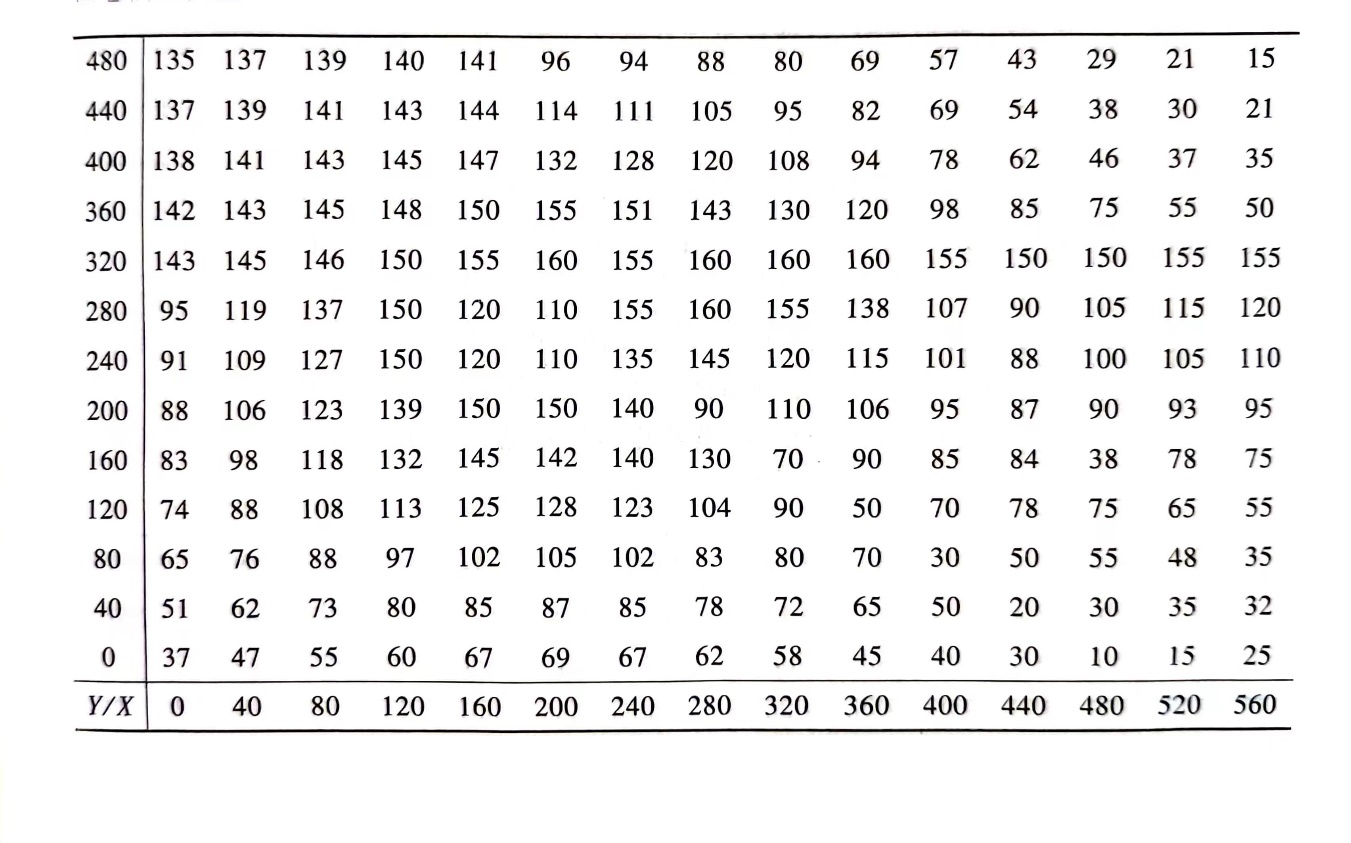
**输出：**

最优拟合曲线为

双曲线拟合: y = -9.03/x + 111.44, RSS = 1.30



3. 下面是一山区海拔高度每400m 的网格数据(单位:10m). 为了作修建道路的成本预算，需要给出每100m的网格数据. 已知山区有一个山峰、一条山谷和一条溪流(其源头约1350m), 画出它们的位置.



**代码：**

X**=**0:40:560;

Y**=**0:40:480;

[x,y]**=**meshgrid(X,Y);

z**=**[37 47 55 60 67 69 67 62 58 45 40 30 10 15 25

51 62 73 80 85 87 85 78 72 65 50 20 30 35 32

65 76 88 97 102 105 102 83 80 70 30 50 55 48 35

74 88 108 113 125 128 123 104 90 50 70 78 75 65 55

83 98 118 132 145 142 140 130 70 90 85 84 38 78 75

88 106 123 139 150 150 140 90 110 106 95 87 90 93 95

91 109 127 150 120 110 135 145 120 115 101 88 100 105 110

95 119 137 150 120 110 155 160 155 138 107 90 105 115 120

143 145 146 150 155 160 155 160 160 160 155 150 150 155 155

142 143 145 148 150 155 151 143 130 120 98 85 75 55 50

138 141 143 145 147 132 128 120 108 94 78 62 46 37 35

137 139 141 143 144 114 111 105 95 82 69 54 38 30 21

135 137 139 140 141 96 94 88 80 69 57 43 29 21 15];

*% mesh(x,y,z)*

*%%*

xi**=**linspace(**-**3,3,300);

xi**=**0:10:560;yi**=**0:10:480;

[XI,YI]**=**meshgrid(xi,yi);

ZI**=**interp2(x,y,z,XI,YI,'spline');

figure *%生成图窗*

set(gcf,'position',[250 300 1000 400])

subplot(1,2,1);

mesh(XI,YI,ZI);

hold on;

*% 箭头矢量的大小*

vector\_size **=** 15;

*% 山峰、山谷和溪流的位置*

peak **=** [192, 332, interp2(XI, YI, ZI, 192, 332)];

valley **=** [176, 268, interp2(XI, YI, ZI, 176, 268)];

stream\_source **=** [240, 216, interp2(XI, YI, ZI, 240, 216)];

*% 调整摄影机位置*

campos([873.7 **-**1632.6 1648]);

*% 在地形图上标记山峰、山谷和溪流*

quiver3(peak(1), peak(2), peak(3), vector\_size, vector\_size, vector\_size**+**15, 'r', 'LineWidth', 1.5, "Marker","x");

text(peak(1), peak(2), peak(3) **+** 50, '山峰', 'Color', 'r');

quiver3(valley(1), valley(2), valley(3), **-**vector\_size, **-**vector\_size, vector\_size**+**20, 'Color', [0.9290 0.6940 0.1250],'LineWidth', 1.5, "Marker","x");

text(valley(1)**-**50, valley(2)**-**20, valley(3) **+** 50, '山谷', 'Color', [0.9290 0.6940 0.1250]);

quiver3(stream\_source(1), stream\_source(2), stream\_source(3), vector\_size, vector\_size, vector\_size, 'b', 'LineWidth', 1.5, "Marker","x");

text(stream\_source(1)**+**20, stream\_source(2)**+**10, stream\_source(3) **+** 25, '溪流源头', 'Color', 'b');

title('山区地势图');

text(400, 700, '比例尺: 1:10', 'Color', 'k');

hold off;

*%%*

subplot(1,2,2);

h**=**contour(XI,YI,ZI);

clabel(h)

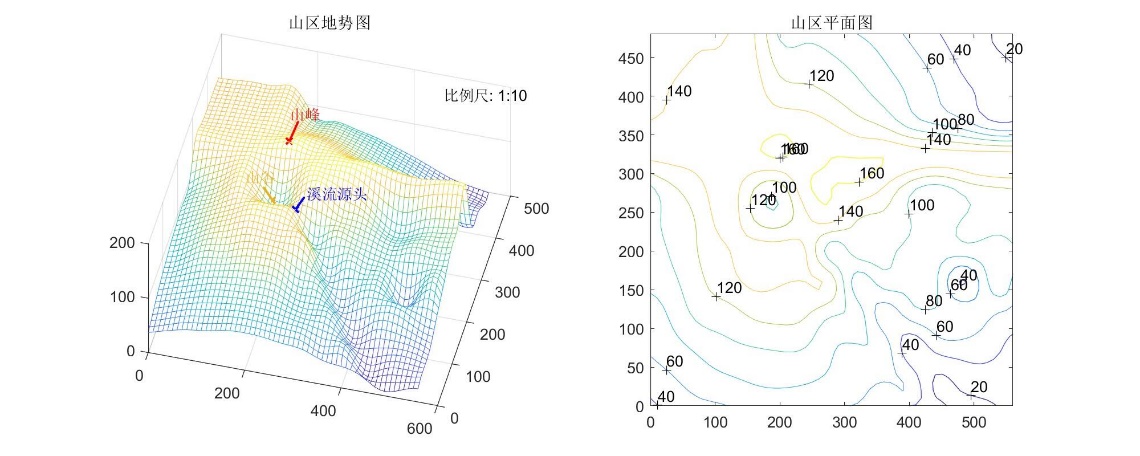
title('山区平面图')

result **=** [XI,YI,ZI];

disp("插值出的每100m的网格数据已存放在result中！")

**输出：**

插值出的每100m的网格数据已存放在result中！



4. 在一丘陵地带测量高程, x和y方向每隔100米测一个点, 得高程如表所示, 试插值一曲面, 确定合适的模型, 并由此找出最高点和该点的高程.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 100 | 200 | 300 | 400 | 500 |
| 100 | 636 | 697 | 624 | 478 | 450 |
| 200 | 698 | 712 | 630 | 478 | 420 |
| 300 | 680 | 674 | 598 | 412 | 400 |
| 400 | 662 | 626 | 552 | 334 | 310 |

**代码：**

*% 初始化x和y的坐标*

x **=** 100:100:500;

y **=** 100:100:400;

*% 初始化高程数据*

Z **=** [636 697 624 478 450;

698 712 630 478 420;

680 674 598 412 400;

662 626 552 334 310];

*% 创建一个更精细的网格,以备插值*

xq **=** 100:10:500;

yq **=** 100:10:400;

*% 对高程数据进行插值*

Zq **=** interp2(x,y,Z,xq,yq','cubic**'**);

*% 找到最高点的坐标和高程*

[maxZ, idx] **=** max(Zq(:));

[xInd, yInd] **=** ind2sub(size(Zq), idx);

maxX **=** xq(xInd);

maxY **=** yq(yInd);

*% 输出结果*

fprintf('最高点的坐标为 (%d, %d)，高程为 %f\n', maxX, maxY, maxZ);

**输出：**

最高点的坐标为 (190, 170)，高程为 718.061800

5. 有如下数据：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| X | 1 | 1.1 | 1.2 | 1.3 | 1.4 |
| y | 1.00000 | 1.23368 | 1.55271 | 1.99372 | 2.61170 |

利用三种插值方法对其进行插值。

**代码：**

*% 定义X和Y的数据*

X **=** [1, 1.1, 1.2, 1.3, 1.4];

Y **=** [1, 1.2337, 1.5527, 1.9937, 2.6117];

*% 定义一个新的X值，用于插值*

Xq **=** 1:0.01:1.4;

*% 线性插值*

Yq\_linear **=** interp1(X, Y, Xq, 'linear');

*%disp('线性插值结果:');*

*%disp([Xq; Yq\_linear]);*

*% 多项式插值*

P **=** polyfit(X, Y, length(X) **-** 1);

Yq\_poly **=** polyval(P, Xq);

*%disp('多项式插值结果:');*

*%disp([Xq; Yq\_poly]);*

*% 三次样条插值*

Yq\_spline **=** interp1(X, Y, Xq, 'spline');

disp('三次样条插值结果:');

disp([Xq; Yq\_spline]);

**输出：**

三次样条插值结果:

列 1 至 12

1.0000 1.0100 1.0200 1.0300 1.0400 1.0500 1.0600 1.0700 1.0800 1.0900 1.1000 1.1100

1.0000 1.0204 1.0415 1.0631 1.0853 1.1082 1.1318 1.1561 1.1812 1.2070 1.2337 1.2612

列 13 至 24

1.1200 1.1300 1.1400 1.1500 1.1600 1.1700 1.1800 1.1900 1.2000 1.2100 1.2200 1.2300

1.2896 1.3190 1.3493 1.3805 1.4128 1.4461 1.4805 1.5160 1.5527 1.5905 1.6296 1.6700

列 25 至 36

1.2400 1.2500 1.2600 1.2700 1.2800 1.2900 1.3000 1.3100 1.3200 1.3300 1.3400 1.3500

1.7117 1.7548 1.7994 1.8455 1.8932 1.9426 1.9937 2.0466 2.1012 2.1578 2.2163 2.2769

列 37 至 41

1.3600 1.3700 1.3800 1.3900 1.4000

2.3394 2.4042 2.4711 2.5402 2.6117

6. 假定某地某天的气温变化记录数据见下表，误差不超过0.5℃，试找出其这一天的气温变化规律。

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 时刻/h | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| 温度/℃ | 15 | 14 | 14 | 14 | 14 | 15 | 16 | 18 | 20 |
| 时刻/h | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |  |
| 温度/℃ | 22 | 23 | 25 | 28 | 31 | 32 | 31 | 29 |  |
| 时刻/h | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 |  |
| 温度/℃ | 27 | 25 | 24 | 22 | 20 | 18 | 17 | 16 |  |

对一天的温度进行分析，采用多项式拟合的数学方法，建立温度和时刻的模型。

**答：**

对一天的温度进行分析，采用多项式拟合的数学方法，建立温度和时刻的模型。

先绘制散点图，温度的变化呈现出一个周期性的模式，因此使用一个三角函数来拟合可能会更合适。我们可以通过一个非线性最小二乘法的方法（lsqcurvefit函数）来进行拟合。

假设我们使用以下的函数形式来拟合数据：

**代码：**

*% 使用lsqcurvefit进行非线性最小二乘拟合*

p **=** lsqcurvefit(fitfun, p0, time, temperature);

*% 显示拟合的参数*

disp(['振幅 A = ', num2str(p(1))])

disp(['相位偏移 φ = ', num2str(p(2))])

disp(['平均温度 B = ', num2str(p(3))])

str\_ **=** sprintf("$$T(t)=%6.2f\\cdot\\sin(\\frac{2\\pi }{24}+%6.2f)+%6.2f$$",p(1),p(2),p(3));

*% 生成拟合的温度数据*

fit\_temperature **=** fitfun(p, time);

*% 绘制原始数据和拟合数据*

figure

plot(time, temperature, 'o') *% 原始数据*

hold on

plot(time, fit\_temperature, '-') *% 拟合数据*

xlabel('时间 (小时)')

ylabel('温度 (℃)')

legend('原始数据', '拟合数据', 'Location', 'Best')

title('温度与时间的关系')

text(10, 16, str\_,'Interpreter','latex')

hold off

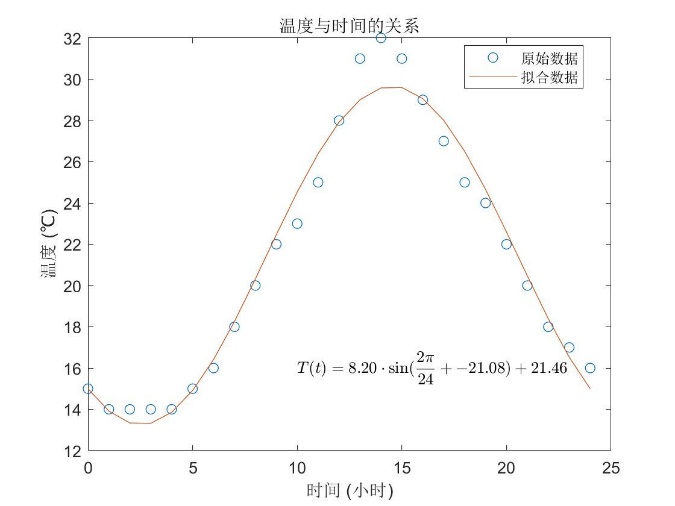
**输出：**

找到局部最小值。优化已完成，因为梯度大小小于最优性容差的值。<停止条件详细信息>

振幅 A = 8.1988

相位偏移 φ = -21.0832

平均温度 B = 21.4585



7. 在某化学反应中, 已知生成物的浓度与时间有关。今测得一组数据如表所示, 求出拟合曲线，分别用二、三、六次多项式，, 拟合。

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 时间t/min | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| 浓度y/ | 4.00 | 6.40 | 8.00 | 8.80 | 9.22 | 9.50 | 9.70 | 9.86 |
| 时间t/min | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
| 浓度y/ | 10.00 | 10.20 | 10.32 | 10.32 | 10.50 | 10.55 | 10.58 | 10.60 |

**代码：**

t **=** [1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16];

y **=** [4 6.4 8 8.8 9.22 9.5 9.7 9.86 10 10.2 10.32 10.32 10.5 10.55 10.58 10.6];

*% 二次拟合*

p2 **=** polyfit(t, y, 2);

y2 **=** polyval(p2, t);

*% 三次拟合*

p3 **=** polyfit(t, y, 3);

y3 **=** polyval(p3, t);

*% 六次拟合*

p6 **=** polyfit(t, y, 6);

y6 **=** polyval(p6, t);

*% 定义损失函数*

loss\_func1 **=** **@**(param) sum((y **-** (t**.**/(param(1)**\***t **+** param(2))))**.^**2);

loss\_func2 **=** **@**(param) sum((y **-** param(1)**\***exp(param(2)**.**/t))**.^**2);

*% 使用fminsearch找到最小化损失函数的参数*

param1 **=** fminsearch(loss\_func1, [1, 1]);

param2 **=** fminsearch(loss\_func2, [1, **-**1]);

*% 计算拟合的y值*

y1 **=** t**.**/(param1(1)**\***t **+** param1(2));

y2 **=** param2(1)**\***exp(param2(2)**.**/t);

*% 画出原始数据*

plot(t, y, 'ko'); hold on;

*% 画出拟合曲线*

plot(t, y2, 'r-');

plot(t, y3, 'g-');

plot(t, y6, 'b-');

plot(t, y1, 'c-');

plot(t, y2, 'm-');

*% 添加图例*

legend('Original Data', '2nd Order Fit', '3rd Order Fit', '6th Order Fit', 'phi1 Fit', 'phi2 Fit',"Location","east");

*% 关闭hold*

hold off;

*% 输出二次拟合的函数*

fprintf('二次拟合函数：y = %f \* t^2 + %f \* t + %f\n', p2(1), p2(2), p2(3));

*% 输出三次拟合的函数*

fprintf('三次拟合函数：y = %f \* t^3 + %f \* t^2 + %f \* t + %f\n', p3(1), p3(2), p3(3), p3(4));

*% 输出六次拟合的函数*

fprintf('六次拟合函数：y = %f \* t^6 + %f \* t^5 + %f \* t^4 + %f \* t^3 + %f \* t^2 + %f \* t + %f\n', p6(1), p6(2), p6(3), p6(4), p6(5), p6(6), p6(7));

*% 输出非线性拟合的函数*

fprintf('phi1拟合函数：y = t / (%f \* t + %f)\n', param1(1), param1(2));

fprintf('phi2拟合函数：y = %f \* exp(%f / t)\n', param2(1), param2(2));

**输出：**

二次拟合函数：y = -0.044308 \* t^2 + 1.062259 \* t + 4.398018

三次拟合函数：y = 0.006344 \* t^3 + -0.206092 \* t^2 + 2.196015 \* t + 2.553681

六次拟合函数：y = -0.000004 \* t^6 + 0.000282 \* t^5 + -0.008502 \* t^4 + 0.131659 \* t^3 + -1.110576 \* t^2 + 5.018710 \* t + -0.051721

phi1拟合函数：y = t / (0.084419 \* t + 0.135891)

phi2拟合函数：y = 11.344570 \* exp(-1.070079 / t)

8. 已知当温度为T=[700,720,740,760,780]时，过热蒸汽体积的变化为V= [0.0977, 0.1218, 0.1406, 0.1551, 0.1664], 分别采用线性插值和三次样条插值求解T= 750, 770时的体积变化，并在一个图形界面中画出线性插值函数和三次样条插值函数的图形。

**代码：**

*% 已知数据*

T **=** [700,720,740,760,780];

V **=** [0.0977, 0.1218, 0.1406, 0.1551, 0.1664];

*% 插值求解的温度点*

T\_query **=** [750, 770];

*% 线性插值*

V\_linear **=** interp1(T, V, T\_query, 'linear');

*% 三次样条插值*

V\_spline **=** interp1(T, V, T\_query, 'spline');

*% 输出插值结果*

fprintf('在T=750时，线性插值、三次样条插值得到的体积变化为：%.4f, %.4f\n', V\_linear(1),V\_spline(1));

fprintf('在T=770时，线性插值、三次样条插值得到的体积变化为：%.4f, %.4f\n', V\_linear(2), V\_spline(2));

*% 生成插值函数的图形*

T\_fine **=** linspace(min(T), max(T), 500); *% 细分的温度区间，用于画图*

V\_linear\_fine **=** interp1(T, V, T\_fine, 'linear'); *% 细分区间的线性插值结果*

V\_spline\_fine **=** interp1(T, V, T\_fine, 'spline'); *% 细分区间的三次样条插值结果*

figure;

hold on; *% 保持当前图形*

plot(T, V, 'ko', 'MarkerSize', 10); *% 原始数据点*

plot(T\_fine, V\_linear\_fine, 'r-'); *% 线性插值函数*

plot(T\_fine, V\_spline\_fine, 'b-'); *% 三次样条插值函数*

legend('原始数据', '线性插值', '三次样条插值','Location','east'); *% 图例*

xlabel('T'); ylabel('V');

title('线性插值与三次样条插值的比较');

grid on;

hold off;

**输出：**

在T=750时，线性插值、三次样条插值得到的体积变化为：0.1478, 0.1483

在T=770时，线性插值、三次样条插值得到的体积变化为：0.1608, 0.1611

