组　　号	Z08
实验题目	插值与拟合
队员姓名	1. 宋昭琰 2. 徐意 3. 李宗琳 4. 向飞宇

目　录

1 实验一：对直升飞机外形的插值

问题描述：根据直升飞机机翼外形曲线上的部分坐标求解节点处的导数值和积分值，测量数据如下：

x	0.52	8.0	17.95	28.65	50.65	104.6
y	5.28794	13.8400	20.2000	24.9000	31.1000	36.5000
x	156.6	260.7	364.4	468.0	507.0	520.0
y	36.6000	31.000	20.9000	7.80000	1.50000	0.200000

1.1 利用插值求解机翼的节点导数及整体积分

由于规定了两端的一阶导数，这里需要用到一元差值的高级设置，即 csape 函数. 用 csape(x, y,[1 1], [1.86548 -0.046115]) 来表示左右一阶导数值分别为规定的 1.86548 和 -0.046115. 接着，使用 fnplt 画出插值图形如下：

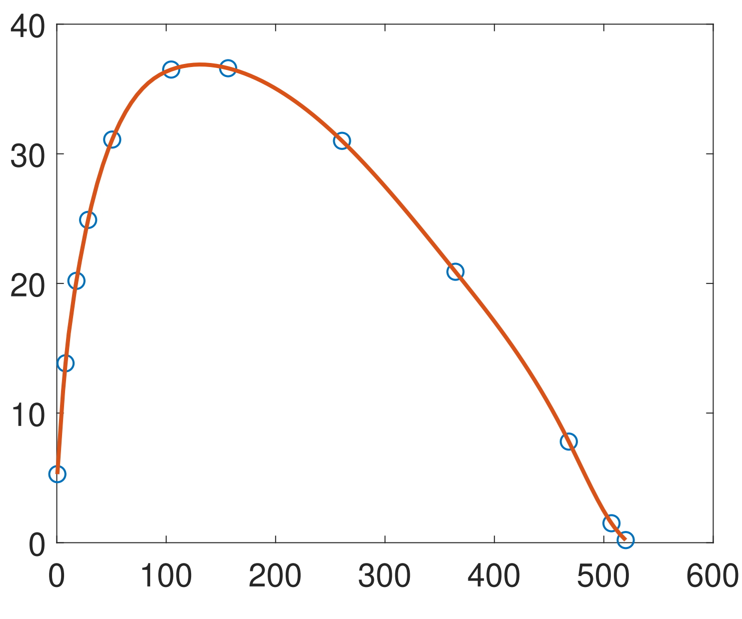

图 1 利用 csape 对机翼数据点插值

可以看出，差值曲线拟合很好，机翼体现出流线型.

接着，用 fnder 函数求出各节点处的导数，再用 fnder(pp, 2) 求出二阶导，再用 fnval(ppdi, x) 来算出每个点的导数值. 所得各节点导数值为：

x	0.52	8.0	17.95	28.65	50.65	104.6
y'	1.8655	0.7437	0.5329	0.3682	0.2088	0.0293
x	156.6	260.7	364.4	468.0	507.0	520.0
y'	-0.0212	-0.0815	-0.1064	-0.1642	-0.1353	-0.0461

表 1　各节点的导数值

各节点二阶导数值为:

x	0.52	8.0	17.95	28.65	50.65	104.6
y''	-0.2793	-0.0206	-0.0217	-0.0091	-0.0054	-0.0012
x	156.6	260.7	364.4	468.0	507.0	520.0
y''	-0.0007	-0.0004	-0.0000	-0.0011	0.0026	0.0112

表 2　各节点的二阶导数值

利用 MatLab 中的 fnint 函数能够样条函数的不定积分，利用 Newton-Lebneiz 公式可以得出定积分的值，所以可以使用 fnval(fnint(pp),[0.52 520]) 算出原函数在 0.52 和 520 时对应的函数值，再利用 diff 函数求出它们的差，即为 0.52 到 520 上的积分. 积分值为 $1.2905 \cdot 10^4$.

1.2 实验一的代码

```
x = [0.52 8.0 17.95 28.65 50.65 104.6 156.6 260.7 364.4 468.0 507.0
↪  520.0];
y = [5.28794 13.8400 20.2000 24.9000 31.1000 36.5000 36.6000 31.000
↪  20.9000 7.8000 1.50000 0.200000];
% spline using csape
plot(x, y, 'o', 'MarkerSize', 8, 'linewidth', 1);
hold on;
pp = csape(x, y,[1 1],[1.86548 -0.046115]);
fnplt(pp);
```

```
set(gca, 'fontsize', 16);
hold off;
% get deviVative
ppd1 = fnder(pp);
ppd2 = fnder(pp, 2);
nodepd1 = fnval(ppd1, x);
nodepd2 = fnval(ppd2, x);
% get integral
ppi = diff(fnval(fnint(pp),[0.52 520]));

nodepd1, nodepd2, ppi
```

2 实验二：对 Hougen-Watson 模型的系数求解

问题描述：根据数据求解 Hougen-Watson 模型的参数，模型如下：

$$rate = \frac{\beta_1 x_2 + \beta_2 x_3}{1 + \beta_3 x_1 + \beta_4 x_2 + \beta_5 x_3}$$

测量数据为：

x_1	x_2	x_3	$rate$
470	300	10	8.5500
285	80	10	3.7900
470	300	120	4.8200
470	80	120	0.0200
470	80	10	2.7500
100	190	10	14.3900
100	80	65	2.5400

x_1	x_2	x_3	$rate$
470	190	65	4.3500
100	300	54	13.0000
100	300	120	8.5000
100	80	120	0.0500
285	300	10	11.3200
285	190	120	3.1300

2.1 非线性拟合的数学原理

对于曲线拟合问题，我们实际上是采用最小二乘法的思路进行解决的. 我们先取定待定系数迭代的初始值（向量），将已知的自变量数据带入待定系数为初始值的函数，得

到一组函数值，求这组函数值和实际数据函数值的差的平方和. 通过不断迭代，求出使平方和最小的待定系数值，得到拟合函数.

2.2 题目分析与求解

这个问题是多维自变量的非线性拟合问题，我们使用 lsqcurvefit 函数来进行拟合. 每一组自变量数据我们使用列向量表示，并将所有数据通过行向量的形式表示出来得到 xdata. 对于待定系数的初值选择，我们观察数据发现，rate 值随 x_1 的值增大而减小，随 x_2 的增大而增大，随 x_3 的增大而减小，故 x_1 的系数 β_3 取正值，β_1 大于 β_4，β_2 小于 β_5，因此，我们取 $[2, 0.5, 1, 0.5, 2]$ 作为迭代初值. 这里要注意到在描述函数 rate 时，对自变量逐行取向量，并且由于是矩阵相除，除号要用点除表示.

为了提高迭代次数和求解精度，我们修改了 lsqcurvefit 的默认参数，使它的函数估计次数（Evaluation）为 10000 次，迭代次数（Iteration）为 10000 次，并设容许误差为 10^{-10}. 这样设置后，再使用 lsqcurvefit 得到结果为:

$$\begin{cases} \beta_1 = 1.2526 \\ \beta_2 = -0.8394 \\ \beta_3 = 0.0628 \\ \beta_4 = 0.0400 \\ \beta_5 = 0.1124 \end{cases}$$

2.3 误差分析

通过结果，代入实验数据，对参数进行验证可得:

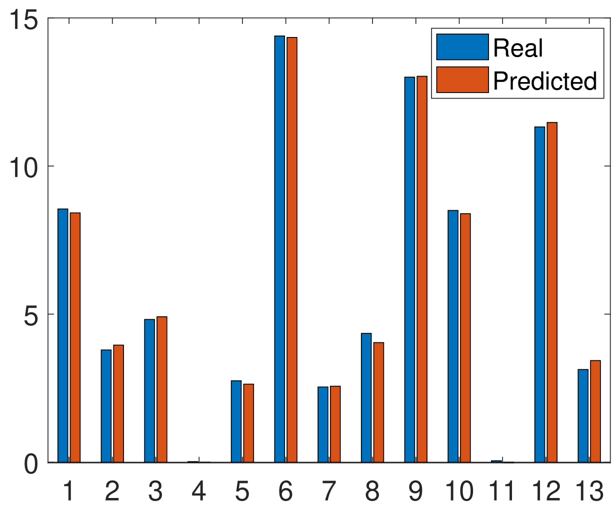

图 2 对参数进行验证

可以发现，真实数据与使用拟合后求解的数据之间的绝对误差比较小，均在 0.35 以内，其中第 8 组和第 13 组数据误差相对于其它数据而言较大. 对于相对误差，由于有部分数据的值接近 0，这些数据的相对误差较大. 整体来看，求解出的参数较好地符合了模型.

2.4 实验二的代码

```
xdata = [470 300 10; 285 80 10; 470 300 120; 470 80 120; 470 80 10; 100
↪ 190 10; 100 80 65; 470 190 65; 100 300 54; 100 300 120; 100 80 120;
↪ 285 300 10; 285 190 120];

rate = [8.5500; 3.7900; 4.8200; 0.0200; 2.7500; 14.3900; 2.5400; 4.3500;
↪ 13.0000; 8.5000; 0.0500; 11.3200; 3.1300];

options = optimset();
options.MaxFunEvals = 10000;
options.MaxIter = 10000;
options.TolX = 1e-10;

f = @(x, xdata) (x(1) * xdata(:, 2) + x(2) * xdata(:, 3)) ./ (1 + x(3) *
↪ xdata(:, 1) + x(4) * xdata(:, 2) + x(5) * xdata(:, 3));
[x, resnorm, residual, exitflag, output] = lsqcurvefit(f, [1 1 1 1 1],
↪ xdata, rate, [], [], options);
bar([rate, f(x, xdata)]);
legend(["Real"; "Predicted"]);
set(gca, 'fontsize', 16);
```

3 实验三：使用插值求解机动车刹车问题

问题描述：对机动车速度与刹车距离的数据进行插值，求解刹车距离 328ft 以内，行驶速度的限制.

测量数据为：

v	20	25	30	35	40	45	50	55	60	65	70	75	80
d	42	56	73.5	91.5	116	142.5	173	209.5	248	292.5	343	401	464

3.1 对问题的求解

首先，我们使用了三次样条插值函数对速度与刹车距离得出一条曲线，拟合的结果如下：

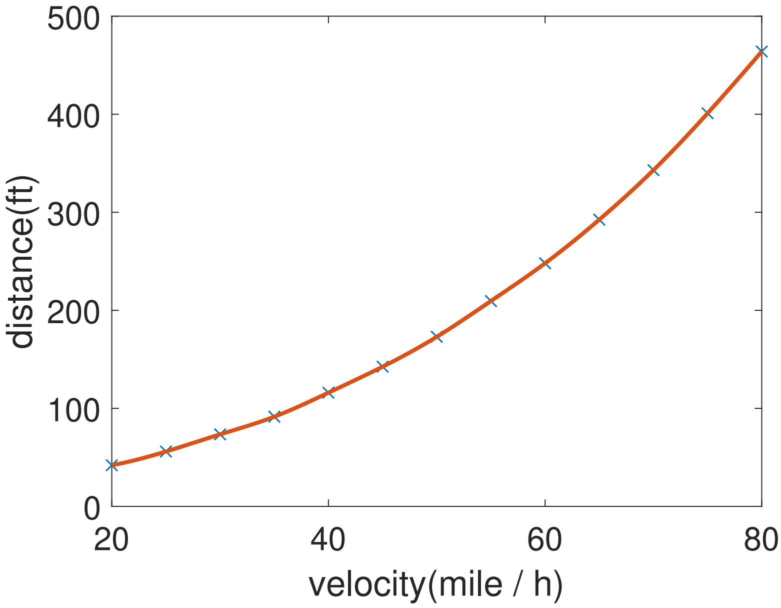

图 3 三次样条插值生成速度与刹车距离的关系

然后，需要在已经拟合的曲线上寻找一个点 v_{max}，使得该点处的刹车距离为 $d = 328\text{ft}$. 从上面插值的结果可以看出，刹车距离关于刹车速度为单调递增，所以找到的解是唯一的. 寻找到这个解有很多方法，此处采用 MatLab 自带的 fsolve 函数求解，求解的步骤如下：

- 使用 fnplt 将插值函数构造为一个匿名函数，便于 fsolve 调用；
- 通过观察插值图像，设定求解初值为 $v = 65\text{mile/h}$.

最终，得到最大的刹车速度 $v_{max} = 68.5860\text{mile/h}$.

3.2 实验三的代码

```
v = [20 25 30 35 40 45 50 55 60 65 70 75 80];
d = [42 56 73.5 91.5 116 142.5 173 209.5 248 292.5 343 401 464];
pp = spline(v, d);
plot(v, d, 'x', 'MarkerSize', 8);
hold on;
fnplt(pp);
hold off;
xlabel('velocity(mile / h)');
```

```
ylabel('distance(ft)');
set(gca, 'fontsize', 16);
x = fsolve(@(x) fnval(pp, x) - 328, 65);
```

4 实验四：利用 DME 信号台数据确定飞机位置

问题描述：利用 5 个 DME 信号台的位置确定飞机的位置，测量数据为：

DME 编号	x 坐标	y 坐标	z 坐标	测量数据	测量误差
1	0	10.0000	0.2000	94.2340	2
2	200.0000	300.0000	0.5000	339.4322	1
3	−300.0000	500.0000	0.2000	482.0996	1.5
4	−400.0000	−200.0000	1.0000	441.8201	3
5	200.0000	−200.0000	−1.0000	383.7252	0.5

4.1 利用回归分析建立模型

设不同信号台的坐标为 (x_i, y_i, z_i), 测量误差为 $\varepsilon_i, i = 1, \cdots, 5$. 我们将飞机到信号台的距离 d_i 表示成信号台的坐标和测量误差的函数 $d(x_i, y_i, z_i, \varepsilon_i)$. 由坐标距离公式易知函数形式为:

$$|d(x_i, y_i, z_i, \varepsilon_i) - \sqrt{(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2}| = \varepsilon_i, \quad i = 1, \cdots, 5$$

从回归的角度看，x,y,z 都是回归方程的系数,

$$|d(x_i, y_i, z_i, \varepsilon_i) - \sqrt{(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2}| = \varepsilon_i \sim N(0, r^2)$$

是每一个方程的残差，但题目中给定每一个的测量误差不相同，这就产生了异方差的情况，为了消除异方差性，由于已知残差的分布，可以将回归方程的方差“统一”, 极小化相对误差平方和以确保无量纲、归一化, 即

$$\left| \frac{d(x_i, y_i, z_i, \varepsilon_i)}{r_i} - \frac{\sqrt{(x - x_i)^2 + (y - y_i)^2 + (z - z_i)^2}}{r_i} \right| = \frac{\varepsilon_i}{r_i}, \quad i = 1, \cdots, 5.$$

这样可以运用 Matlab 有关最小二乘的函数 lsqcurvefit 来解决系数 x, y, z 的求解，从而得到飞机的位置，经下面代码的运行，解得 $(x, y, z) = (-60.0000, 82.0000, 10.0001)$.

以下是用 DME 定向的结果图:

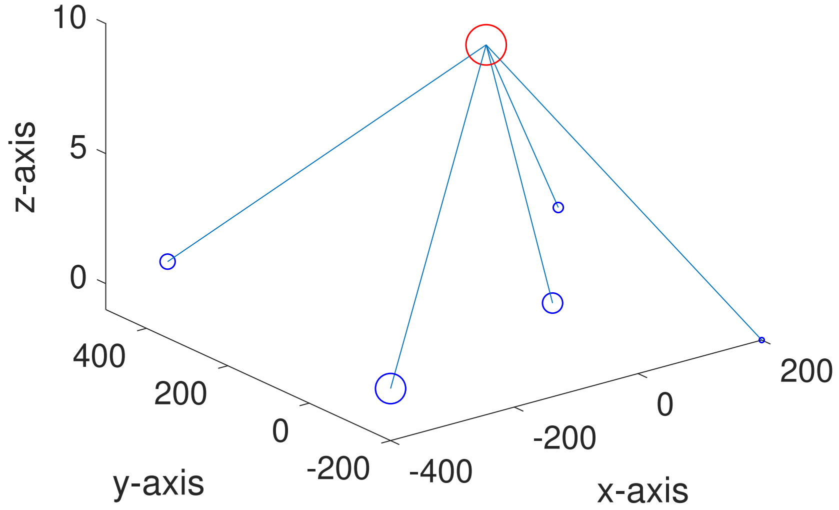

图 4 DME 定向结果

4.2 实验四的代码

```
DME = [0 10.0000 0.2000;
200.0000 300.0000 0.5000;
-300.0000 500.0000 0.2000;
-400.0000 -200.0000 1.0000;
200.0000 -200.0000 -1.0000];
d = [94.2340; 339.4322; 482.0996; 441.8201; 383.7252];
r = [2; 1; 1.5; 3; 0.5];
DME(:,4) = r;
dr = d ./ r;
x0 = [1; 1; 1];
x = lsqcurvefit(@curvefun1, x0, DME, dr);

plot3(DME(1, 1), DME(1, 2), DME(1, 3), 'bo', 'MarkerSize', 2*5);
hold on;
plot3(DME(2, 1), DME(2, 2), DME(2, 3), 'bo', 'MarkerSize', 1*5);
hold on;
plot3(DME(3, 1), DME(3, 2), DME(3, 3), 'bo', 'MarkerSize', 1.5*5);
hold on;
plot3(DME(4, 1), DME(4, 2), DME(4, 3), 'bo', 'MarkerSize', 3*5);
```

```
hold on;
plot3(DME(5, 1), DME(5, 2), DME(5, 3), 'bo', 'MarkerSize', 0.5*5);
hold on;
plot3(x(1), x(2), x(3), 'ro', 'MarkerSize', 20);
hold on;
line([DME(1, 1) x(1)], [DME(1, 2) x(2)], [DME(1, 3) x(3)]);
hold on;
line([DME(2, 1) x(1)], [DME(2, 2) x(2)], [DME(2, 3) x(3)]);
hold on;
line([DME(3, 1) x(1)], [DME(3, 2) x(2)], [DME(3, 3) x(3)]);
hold on;
line([DME(4, 1) x(1)], [DME(4, 2) x(2)], [DME(4, 3) x(3)]);
hold on;
line([DME(5, 1) x(1)], [DME(5, 2) x(2)], [DME(5, 3) x(3)]);
hold off;
xlabel('x-axis');
ylabel('y-axis');
zlabel('z-axis');
set(gca, 'fontsize', 16);

function f=curvefun1(b, x)
    f = sqrt((b(1) - x(:, 1)).^2 + (b(2) - x(:,2)).^2 + (b(3) -
↪   x(:,3)).^2) ./ x(:,4);
end
```