

# 数值分析第三次大作业

张晋

学号：15091060

最后更新于：May 28, 2017

# 目录 | 0

1	题目	2
2	算法设计方案	4
2.1	方案综述	4
2.2	Newton 迭代法	5
2.3	分片双二次插值	6
2.4	曲面拟合	7
3	源程序	10
3.1	C 语言版大作业	10
4	计算结果	11
5	讨论	12
5.1	向量求导在曲线拟合中的应用	12
5.2	矩阵求导在曲面拟合中的应用	13

关于  $x, y, t, u, v, w$  的下列方程组:

$$\begin{cases} 0.5 \cos t + u + v + w - x = 2.67 \\ t + 0.5 \sin u + v + w - y = 1.07 \\ 0.5t + u + \cos v + w - x = 3.74 \\ t + 0.5u + v + \sin w - y = 0.79 \end{cases}$$

以及关于  $z, t, u$  的下列二维数表确定了一个二元函数  $z = f(x, y)$ 。

表 1.1: 二维数表

$z \backslash y$ $t$	0	0.4	0.8	1.2	1.6	2
0	-0.5	-0.34	0.14	0.94	2.06	3.5
0.2	-0.42	-0.5	-0.26	0.3	1.18	2.38
0.4	-0.18	-0.5	-0.5	-0.18	0.46	1.42
0.6	0.22	-0.34	-0.58	-0.5	-0.1	0.62
0.8	0.78	-0.02	-0.5	-0.66	-0.5	-0.02
1	1.5	0.46	-0.26	-0.66	-0.74	-0.5

1. 试用数值方法求出  $f(x, y)$  在区域  $D = \{(x, y) | 0 \leq x \leq 0.8, 0.5 \leq y \leq 1.5\}$  上的一个近似表达式:

$$p(x, y) = \sum_{r=0}^k \sum_{s=0}^k c_{rs} x^r y^s$$

要求  $p(x, y)$  最小的  $k$  值达到以下的精度:

$$\sigma = \sum_{i=0}^{10} \sum_{j=0}^{20} [f(x_i, y_j) - p(x_i, y_j)]^2 \leq 10^{-7}$$

其中,  $x_i = 0.08i, y_j = 0.5 + 0.05j$

2. 计算  $f(x_i^*, y_j^*), p(x_i^*, y_j^*)$  ( $i = 1, 2, \dots, 8; j = 1, 2, \dots, 5$ ) 的值, 以观察  $p(x, y)$  逼近  $f(x, y)$  的效果, 其中  $x_i^* = 0.1i, y_j^* = 0.5 + 0.2j$

### 说明:

1. 用迭代方法求解非线性方程组时, 要求近似解向量  $\mathbf{x}^{(k)}$  满足以下精度

$$\frac{\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\|_{\infty}}{\|\mathbf{x}^{(k)}\|_{\infty}} \leq 10^{-12}$$

2. 作二元插值时, 要使用分片二次代数插值。
3. 要由程序自动确定最小的  $k$  值。
4. 打印以下内容:
  - (a) 全部源程序;
  - (b) 数表:  $\{x_i, y_j, f(x_i, y_j)\}$  ( $i = 0, 1, 2, \dots, 10; j = 0, 1, 2, \dots, 20$ );
  - (c) 选择过程的  $k, \sigma$  值;
  - (d) 达到精度要求时的  $k$  和  $\sigma$  值以及  $p(x, y)$  中的系数  $c_{rs}$  ( $r = 0, 1, \dots, k; s = 0, 1, \dots, k$ );
  - (e) 数表:  $\{x_i^*, y_j^*, f(x_i^*, y_j^*), p(x_i^*, y_j^*)\}$  ( $i = 1, 2, \dots, 8; j = 1, 2, \dots, 5$ )。
5. 采用  $f$  型输出  $x_i, y_j, x_i^*, y_j^*$  的准确值, 其余实型数采用  $e$  型输出并且至少显示 12 位有效数字。

2.1 方案综述	4
2.2 Newton 迭代法	5
2.3 分片双二次插值	6
2.4 曲面拟合	7

## 2.1 方案综述

1. 将  $x_i = 0.08i, y_j = 0.5 + 0.05j$  ( $i = 0, 1, \dots, 10; j = 0, 1, \dots, 20$ ) 代入非线性方程组 (2.1) 中, 用Newton 迭代法解出  $t_{ij}$  和  $u_{ij}$ ;
2. 对数表  $z(t, u)$  进行分片双二次插值, 求得  $z_{ij} = \hat{z}(t_{ij}, u_{ij})$
3. 根据  $z_{ij}$  的值进行曲面拟合, 要求精度  $\sigma \leq 10^{-7}$ , 得拟合函数

$$p(x, y) = \sum_{r=0}^k \sum_{s=0}^k c_{rs} x^r y^s$$

4. 创建新的数据点集  $x_i^* = 0.1i, y_j^* = 0.5 + 0.2j$  ( $i = 1, 2, \dots, 8; j = 1, 2, \dots, 5$ ), 并代入非线性方程组 (2.1) 中, 用Newton 迭代法解出  $t^*$  和  $u^*$ , 再用分片双二次插值计算出  $f(x^*, y^*)$ , 将其与  $p(x^*, y^*)$  输出并观察比较。<sup>1</sup>

---

<sup>1</sup>算法流程图见第9页。

## 2.2 Newton 迭代法

$$\begin{cases} 0.5 \cos t + u + v + w - x = 2.67 \\ t + 0.5 \sin u + v + w - y = 1.07 \\ 0.5t + u + \cos v + w - x = 3.74 \\ t + 0.5u + v + \sin w - y = 0.79 \end{cases} \quad (2.1)$$

对于该非线性方程组来说,  $x, y$  为已知量, 需解出  $t, u, v, w$ 。设  $\mathbf{x} = (t, u, v, w)^T$ , 并设定精度水平  $\varepsilon = 10^{-12}$  和最大迭代次数  $M$  先在  $\mathbf{x}^*$  附近选取  $\mathbf{x}^{(0)} = (t^{(0)}, u^{(0)}, v^{(0)}, w^{(0)})^T$  然后迭代<sup>2</sup>:

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - [\mathbf{F}'(\mathbf{x}^{(k)})]^{-1} \mathbf{F}(\mathbf{x}^{(k)})$$

具体算法如下:

---

### Algorithm 1 Newton's method

---

```

1: Set  $\mathbf{x}^{(0)} \in D$  and  $k = 0$ 
2: while  $k < M$  do
3:   Compute  $\mathbf{F}(\mathbf{x}^{(k)})$  and  $\mathbf{F}'(\mathbf{x}^{(k)})$ 
4:   Compute  $\Delta \mathbf{x}^{(k)} = -[\mathbf{F}'(\mathbf{x}^{(k)})]^{-1} \mathbf{F}(\mathbf{x}^{(k)})$ 
5:   if  $\|\Delta \mathbf{x}^{(k)}\| / \|\mathbf{x}^{(k+1)}\| \leq \varepsilon$  then
6:      $\mathbf{x}^* = \mathbf{x}^{(k)}$ 
7:     Break
8:   end if
9:    $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \Delta \mathbf{x}^{(k)}$ 
10:   $k = k + 1$ 
11: end while

```

---

其中, 基于方程组 (2.1) 的  $\mathbf{F}(\mathbf{x})$  及其雅可比矩阵  $\mathbf{F}'(\mathbf{x})$  分别为:

$$\mathbf{F}(\mathbf{x}) = \begin{bmatrix} 0.5 * \cos(t) + u + v + w - x - 2.67 \\ t + 0.5 * \sin(u) + v + w - y - 1.07 \\ 0.5 * t + u + \cos(v) + w - x - 3.74 \\ t + 0.5 * u + v + \sin(w) - y - 0.79 \end{bmatrix}$$

$$\mathbf{F}'(\mathbf{x}) = \begin{bmatrix} -0.5 * \sin(t) & 1 & 1 & 1 \\ 1 & 0.5 * \cos(u) & 1 & 1 \\ 0.5 & 1 & -\sin(v) & 1 \\ 1 & 0.5 & 1 & \cos(w) \end{bmatrix}$$

---

<sup>2</sup>迭代的终止条件为  $\|\Delta \mathbf{x}^{(k)}\| / \|\mathbf{x}^{(k+1)}\| \leq \varepsilon$ , 若  $k > M$  时仍未达到迭代精度, 则迭代失败。

## 2.3 分片双二次插值

前面我们将  $\{(x_i, y_j)\}$  代入非线性方程组 (2.1) 中, 然后用Newton 迭代法解出了  $t_{ij}$  和  $u_{ij}$ .

在这一节中我们需要根据表1.1对  $z(t, u)$  进行分片双二次插值, 求得  $z_{ij} = \hat{z}(t_{ij}, u_{ij})$  ( $i = 0, 1, 2, \dots, 10; j = 0, 1, 2, \dots, 20$ ).

因为表1.1为  $6 \times 6$  的数表, 故可设:

$$t_i = ih \quad (i = 0, 1, \dots, 5)$$

$$u_j = j\tau \quad (j = 0, 1, \dots, 5)^3$$

对于给定的  $(t, u)$ , 如果  $(t, u)$  满足:

$$t_i - \frac{h}{2} < t \leq t_i + \frac{h}{2}, \quad 2 \leq i \leq 3$$

$$u_j - \frac{\tau}{2} < u \leq u_j + \frac{\tau}{2}, \quad 2 \leq j \leq 3$$

那么应选择  $(t_k, u_r)$  ( $k = i - 1, i, i + 1; r = j - 1, j, j + 1$ ) 为插值节点。

若  $t$  满足:

$$t \leq t_1 + \frac{h}{2}$$

或

$$t > t_4 + \frac{h}{2}$$

则相应地选取  $i = 1$  或  $i = 4$

同样的, 若  $u$  满足:

$$u \leq u_1 + \frac{\tau}{2}$$

或

$$u > u_4 + \frac{\tau}{2}$$

则相应地选取  $j = 1$  或  $j = 4$

最后得到插值多项式为

$$\hat{z}(t, u) = \sum_{k=i-1}^{i+1} \sum_{r=j-1}^{j+1} l_k(t) \tilde{l}_r(u) z(t_k, u_r) \quad (2.2)$$

其中,

$$l_k(t) = \prod_{\substack{m=i-1 \\ m \neq k}}^{i+1} \frac{t - t_m}{t_k - t_m} \quad (k = i - 1, i, i + 1)$$

$$\tilde{l}_r(u) = \prod_{\substack{n=j-1 \\ n \neq r}}^{j+1} \frac{u - u_n}{u_r - u_n} \quad (r = j - 1, j, j + 1)$$

---

<sup>3</sup>其中:  $h = 0.2, \tau = 0.4$

## 2.4 曲面拟合

设在三维坐标系  $Oxyu$  中给定  $(m+1) \times (n+1)$  个点:

$$\mathfrak{D} = \{(x_i, y_j), z_{ij}\} \quad (i = 0, 1, \dots, m; j = 0, 1, \dots, n) \quad (2.3)$$

选定  $M+1$  个  $x$  的函数  $\{\varphi_r(x)\}_{r=0}^M$  和  $N+1$  个  $y$  的函数  $\{\psi_s(y)\}_{s=0}^N$

以函数组  $\{\varphi_r(x)\psi_s(y)\} \quad (r = 0, 1, \dots, M; s = 0, 1, \dots, N)$  为基函数, 构成以  $\{c_{rs}\}$  为参数的曲面族

$$p(x, y) = \sum_{r=0}^M \sum_{s=0}^N c_{rs} \varphi_r(x) \psi_s(y) \quad (2.4)$$

若参数  $\{c_{rs}^*\}$  使得

$$L(\mathbf{C}) = \sum_{i=0}^m \sum_{j=0}^n \left[ \sum_{r=0}^M \sum_{s=0}^N c_{rs} \varphi_r(x_i) \psi_s(y_j) - u_{ij} \right]^2 \quad (2.5)$$

在  $\mathbf{C} = \mathbf{C}^*$  处取到最小值  $L(\mathbf{C}^*)$ , 则称相应曲面  $p^*(x, y)$  为在曲面族 (2.4) 中按最小二乘原则确定的对于数据 (2.3) 的拟合曲面。

设:

$$\mathbf{B} = [\varphi_r(x_i)]_{(m+1) \times (M+1)}$$

$$\mathbf{G} = [\psi_s(y_j)]_{(n+1) \times (N+1)}$$

$$\mathbf{U} = [u_{ij}]_{(m+1) \times (n+1)}$$

$$\mathbf{C} = [c_{rs}]_{(M+1) \times (N+1)}$$

可证得<sup>4</sup>, 拟合曲面的系数矩阵为

$$\mathbf{C} = (\mathbf{B}^T \mathbf{B})^{-1} \mathbf{B}^T \mathbf{U} \mathbf{G} (\mathbf{G}^T \mathbf{G})^{-1} \quad (2.6)$$

在本实验中:

$$\mathbf{B} = \begin{bmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^k \\ 1 & x_1 & x_1^2 & \cdots & x_1^k \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{10}^2 & x_{10}^2 & \cdots & x_{10}^k \end{bmatrix}_{11 \times (k+1)}$$

$$\mathbf{G} = \begin{bmatrix} 1 & y_0 & y_0^2 & \cdots & y_0^k \\ 1 & y_1 & y_1^2 & \cdots & y_1^k \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & y_{20}^2 & y_{20}^2 & \cdots & y_{20}^k \end{bmatrix}_{21 \times (k+1)}$$

$$\mathbf{U} = [u_{ij}]_{(m+1) \times (M+1)} = [z_{ij}]_{11 \times 21} = [f(x_i, y_j)]_{11 \times 21}$$

<sup>4</sup>在第五章的讨论中, 我将给出一种与教材不一样的证明方法



在计算 (2.6) 式时，要求矩阵的逆，此处可采用 Gauss 消元法。  
解出  $c_{rs}^k$  后，可得：

$$p^{(k)}(x, y) = \sum_{r,s=0}^k c_{rs}^k x^r y^s$$

其算法如下：<sup>5</sup>

---

**Algorithm 2** Surface Fitting

---

```

1: Set  $k = 0, \sigma = 1$ 
2: while  $\sigma > \varepsilon$  do
3:   Compute  $\mathbf{C} = (\mathbf{B}^T \mathbf{B})^{-1} \mathbf{B}^T \mathbf{U} \mathbf{G} (\mathbf{G}^T \mathbf{G})^{-1}$ 
4:   Compute  $\mathbf{P} = \mathbf{B} \mathbf{C} \mathbf{G}^T$ 
5:   Compute  $\sigma = \|\mathbf{P} - \mathbf{U}\|_E^2$ 
6:    $k = k + 1$ 
7:   if  $k > N$  then
8:     Break
9:   end if
10: end while

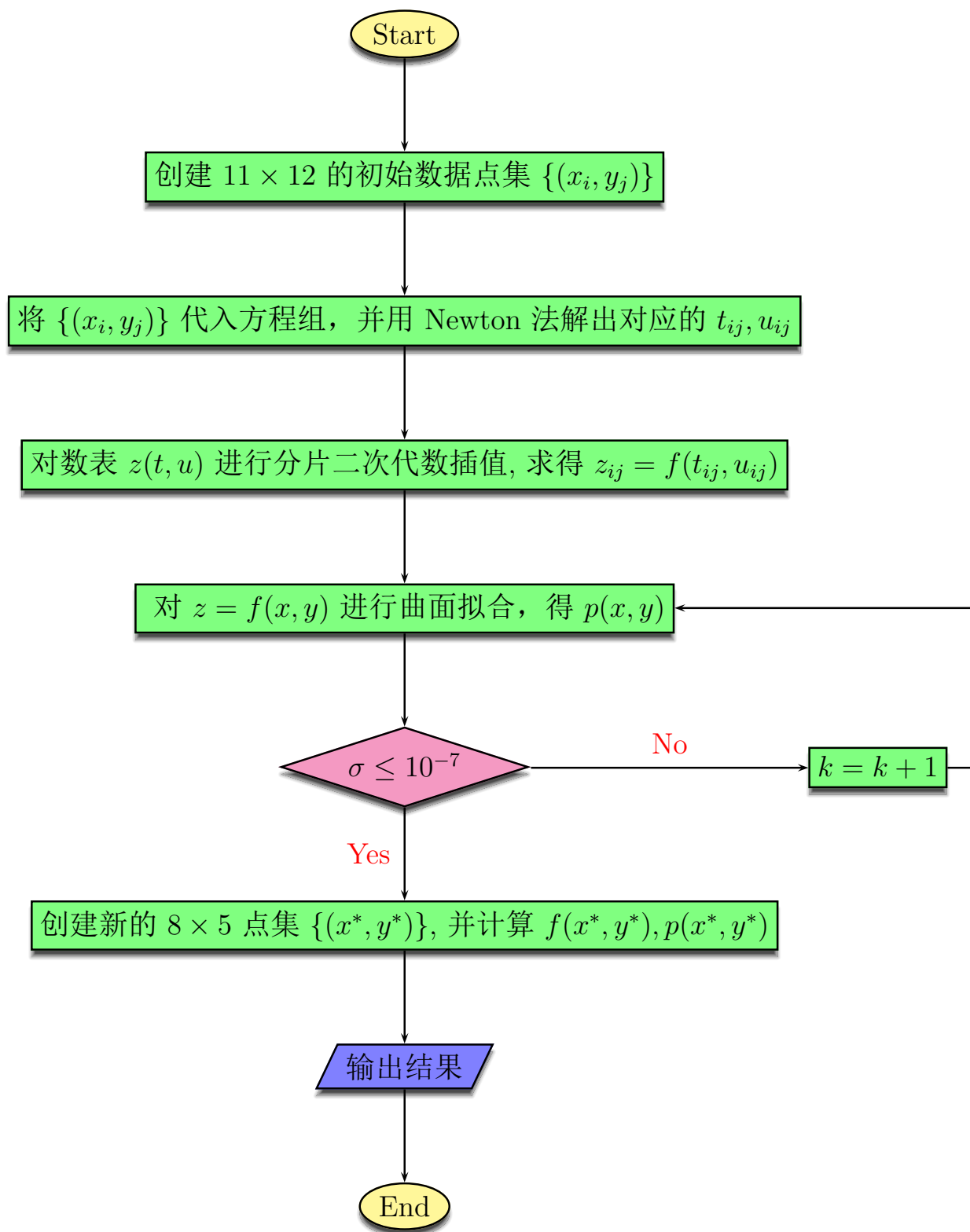
```

---



---

<sup>5</sup>为什么写成这样的矩阵形式将在第五章中说明



3.1 C 语言版大作业	10
--------------	----

### 3.1 C 语言版大作业

```
1 #include<stdio.h>
2 #include<math.h>
3 #include<string.h>
4 #define N 20
5 const double eps=1e-12;
6 double a[N][N],B[N][N],C[N][N];
7 int n=10;
8
9 typedef struct{
10 //定义复数结构体
11     double Re;
12     double Im;
13 }ComplexNumber;
14
15
16 int main(){
17     double Q[N][N];
18     freopen("Works.in","r",stdin);
19     freopen("Works.out","w",stdout);
20     def();
21     Householder_Triangularization(a);
22     printf("A_n-1:\n");
23     output(a);
24     QR(a,Q);
25     printf("\nR:\n");
26     output(a);
27
28     return 0;
29 }
```

后面为输出结果:

特征值与向量如下:

$$\lambda_1 = 9.432879572769e - 001$$

$$\text{Eigenvector} = (0.079620, 0.045421, -0.018272, -0.047961, -0.349567, 0.207215, -0.152312, 0.820634, -0.355466, 0.028866)$$

$$\lambda_2 = 6.489488202110e - 001$$

$$\text{Eigenvector} = (0.108435, 0.071344, 0.382502, -0.047100, -0.717804, 0.181519, -0.226006, 0.388381, 0.289696, 0.024333)$$

$$\lambda_3 = -9.891143464725e - 001 + i*1.084758631513e-001$$

$$\lambda_4 = -9.891143464725e - 001 - i*1.084758631513e-001$$

$$\lambda_5 = 4.954990923624e - 002$$

$$\text{Eigenvector} = (-0.213768, -0.206774, 0.386829, -0.031112, -0.380939, -0.125174, 0.644716, -0.308201, -0.295977, 0.043723)$$

$$\lambda_6 = -1.493147080915e + 000$$

$$\text{Eigenvector} = (-0.561341, 0.778192, 0.014364, -0.277602, 0.003568, -0.002548, -0.022061, -0.011758, -0.013173, 0.035016)$$

5.1 向量求导在曲线拟合中的应用 .....	12
5.2 矩阵求导在曲面拟合中的应用 .....	13

## 5.1 向量求导在曲线拟合中的应用

在曲线拟合中，我发现如果应用向量的求导法则，那么能大大简化证明的过程。

首先给出条件：

$$\begin{aligned}\mathbf{c} &= (c_0, c_1, \dots, c_n)^T \\ \mathbf{y} &= (y_0, y_1, \dots, y_m)^T \\ \mathbf{A} &= \begin{bmatrix} \varphi_0(x_0) & \varphi_1(x_0) & \cdots & \varphi_k(x_0) \\ \varphi_0(x_1) & \varphi_1(x_1) & \cdots & \varphi_k(x_1) \\ \vdots & \vdots & \ddots & \vdots \\ \varphi_0(x_m) & \varphi_1(x_m) & \cdots & \varphi_k(x_m) \end{bmatrix}\end{aligned}$$

然后将误差函数写成向量相乘的形式：

$$\mathbf{F}(\mathbf{c}) = \sum_{i=0}^m \left[ f(x_i) - \sum_{k=0}^n c_k \varphi_k(x_i) \right]^2 \quad (5.1)$$

$$= (\mathbf{y} - \mathbf{A}\mathbf{c})^T (\mathbf{y} - \mathbf{A}\mathbf{c}) \quad (5.2)$$

$$= \mathbf{y}^T \mathbf{y} - 2\mathbf{c}^T \mathbf{A}^T \mathbf{y} + \mathbf{c}^T \mathbf{A}^T \mathbf{A} \mathbf{c} \quad (5.3)$$

然后应用向量求导法则，令其关于  $\mathbf{c}$  的偏导为 0

$$\frac{\partial \mathbf{F}}{\partial \mathbf{c}} = -2\mathbf{A}^T \mathbf{y} + 2\mathbf{A}^T \mathbf{A} \mathbf{c} = 0 \quad (5.4)$$

即可解得所求的  $\mathbf{c}$ ：

$$\boxed{\mathbf{c} = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \mathbf{y}} \quad (5.5)$$

## 5.2 矩阵求导在曲面拟合中的应用

收到之前的启发，我开始思考，在**曲面拟合**中，也能用类似的方法简化流程吗？

在翻阅了一些关于矩阵求导的资料后，我开始了尝试，最后成功得出了相同的结果，这是将曲线拟合从一维推广到二维的情况，只要我们注意观察比较，就能发现其中的相似之处并得到启发，而且如果关于矩阵的掌握熟练的话，那么计算的过程将会更简单自然。

首先受到式 (5.1) 的启发，我们第一步应该尝试着将误差函数转化为矩阵乘积的形式：

$$\mathbf{B} = [\varphi_r(x_i)]_{(m+1) \times (M+1)} = \begin{bmatrix} \varphi_0(x_0) & \varphi_1(x_0) & \cdots & \varphi_M(x_0) \\ \varphi_0(x_1) & \varphi_1(x_1) & \cdots & \varphi_M(x_1) \\ \vdots & \vdots & \ddots & \vdots \\ \varphi_0(x_m) & \varphi_1(x_m) & \cdots & \varphi_M(x_m) \end{bmatrix}$$

$$\mathbf{G} = [\psi_s(y_j)]_{(n+1) \times (N+1)} = \begin{bmatrix} \psi_0(y_0) & \psi_1(y_0) & \cdots & \psi_N(y_0) \\ \psi_0(y_1) & \psi_1(y_1) & \cdots & \psi_N(y_1) \\ \vdots & \vdots & \ddots & \vdots \\ \psi_0(y_n) & \psi_1(y_n) & \cdots & \psi_N(y_n) \end{bmatrix}$$

$$\mathbf{C} = [c_{rs}]_{(M+1) \times (N+1)}$$

我们仔细观察上面的三个矩阵，可以发现：如果将  $c_{rs}\varphi_r(x_i)\psi_s(y_j)$  排成一个  $(m+1) \times (n+1)$  的矩阵  $\mathbf{P}$ ，那么这个矩阵可以拆分成矩阵  $\mathbf{B}, \mathbf{C}, \mathbf{G}^T$  的乘积。<sup>1</sup>

$$[\varphi_r(x_i)c_{rs}\psi_s(y_j)]_{(m+1) \times (n+1)} = \mathbf{P} = \mathbf{BCG}^T \quad (5.6)$$

而式 (2.5) 则刚好为矩阵  $\mathbf{BCG}^T - \mathbf{U}$  的 Euclid—范数的平方，即：

$$L(\mathbf{C}) = \|\mathbf{BCG}^T - \mathbf{U}\|_E^2 \quad (5.7)$$

我们想让  $L(\mathbf{C})$  取最小值，这时应该使  $L(\mathbf{C})$  在  $\mathbf{C}^*$  处的导数  $L(\mathbf{C}^*) = 0$ ，于是这时我们用矩阵的求导法则，对  $L(\mathbf{C})$  求关于  $\mathbf{C}$  的偏导，得：

$$\boxed{\frac{\partial L}{\partial \mathbf{C}} = 2\mathbf{B}^T(\mathbf{BCG}^T - \mathbf{U})\mathbf{G}} \quad (5.8)$$

关于式 (5.8) 的证明如下：

$$L(\mathbf{C}) = \|\mathbf{BCG}^T - \mathbf{U}\|_E^2 \quad (5.9)$$

$$= \sum_{i=0}^m \sum_{j=0}^n \left[ \sum_{r=0}^M \sum_{s=0}^N \varphi_r(x_i)c_{rs}\psi_s(y_j) - u_{ij} \right]^2 \quad (5.10)$$

<sup>1</sup>在  $k < N$  的情况下，它都成立

$$\frac{\partial L}{\partial c_{rs}} = 2 \sum_{i=0}^m \sum_{j=0}^n \left[ \psi_s(y_j) \varphi_r(x_i) c_{rs} \psi_s(y_j) \varphi_r(x_i) - \psi_s(y_j) u_{ij} \varphi_r(x_i) \right] \quad (5.11)$$

$$= 2 \sum_{i=0}^m \sum_{j=0}^n \psi_s(y_j) [\varphi_r(x_i) c_{rs} \psi_s(y_j)] \varphi_r(x_i) - 2 \sum_{i=0}^m \sum_{j=0}^n \psi_s(y_j) u_{ij} \varphi_r(x_i) \quad (5.12)$$

$$\left[ \frac{\partial L}{\partial c_{rs}} \right]_{(M+1) \times (N+1)} = 2 \mathbf{B}^T (\mathbf{B} \mathbf{C} \mathbf{G}^T) \mathbf{G} - 2 \mathbf{B}^T \mathbf{U} \mathbf{G} \quad (5.13)$$

我们可以发现式 (5.12) 到式 (5.13) 的转化和式 (5.6) 是类似的。

当式 (5.8) 为 0 时，解得：

$$\mathbf{C} = (\mathbf{B}^T \mathbf{B})^{-1} \mathbf{B}^T \mathbf{U} \mathbf{G} (\mathbf{G}^T \mathbf{G})^{-1} \quad (5.14)$$

这和式 (2.6) 是相同的，于是我们找到了一种更简洁的方法证明了这个结论。

可见，如果我们能掌握一些关于矩阵求导的知识，在数值分析中对简化运算是有非常大帮助的。