数值分析第三次大作业

张晋

学号：15091060

最后更新于：May 29, 2017

目录 | 0

题目 1

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 \ 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 $D = \{(x, y)|0 \le x \le 0.8, 0.5 \le y \le 1.5\}$ 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：

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

其中，$x_i=0.08i, y_j=0.5+0.05j$

2. 计算 $f(x_i^*, y_j^*), p(x_i^*, y_j^*) \qquad (i=1,2,\cdots,8; j=1,2,\cdots,5)$ 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 $\boldsymbol{x}^{(k)}$ 满足以下精度

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

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:$\{x_i, y_j, f(x_i, y_j)\} \qquad (i=0,1,2,\cdots,10; j=0,1,2,\cdots,20)$;
   (c) 选择过程的 k, σ 值;
   (d) 达到精度要求时的 k 和 σ 值以及 $p(x,y)$ 中的系数 $c_{rs}(r=0,1,\cdots,k; s=0,1,\cdots,k)$;
   (e) 数表:$\{x_i^*, y_j^*, f(x_i^*, y_j^*), p(x_i^*, y_j^*)\} \qquad (i=1,2,\cdots,8; j=1,2,\cdots,5)$。
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。

算法设计方案 | 2

2.1 方案综述

1. 将 $x_i = 0.08i, y_j = 0.5 + 0.05j \quad (i = 0, 1, \cdots, 10; j = 0, 1, \cdots, 20)$ 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 $\sigma \le 10^{-7}$, 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 $x_i^* = 0.1i, y_j^* = 0.5+0.2j \quad (i = 1, 2, \cdots, 8; j = 1, 2, \cdots, 5)$, 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。[1]

[1]算法流程图见第9页。

2.2 Newton 迭代法

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

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

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

具体算法如下：

Algorithm 1 Newton's method

Set $\boldsymbol{x}^{(0)} \in D$ and $k = 0$
while k<M **do**
  Compute $\boldsymbol{F}(\boldsymbol{x}^{(k)})$ and $\boldsymbol{F}'(\boldsymbol{x}^{(k)})$
  Compute $\Delta\boldsymbol{x}^{(k)} = -[\boldsymbol{F}'(\boldsymbol{x}^{(k)})]^{-1}\boldsymbol{F}(\boldsymbol{x}^{(k)})$
  if $\|\Delta\boldsymbol{x}^{(k)}\|/\|\boldsymbol{x}^{(k+1)}\| \le \varepsilon$ **then**
    $\boldsymbol{x}^* = \boldsymbol{x}^{(k)}$
    Break
  end if
  $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \Delta\boldsymbol{x}^{(k)}$
  k=k+1
end while

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

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

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

[2]迭代的终止条件为 $\|\Delta\boldsymbol{x}^{(k)}\|/\|\boldsymbol{x}^{(k+1)}\| \le \varepsilon$, 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 $z_{ij} = \hat{z}(t_{ij}, u_{ij}) \qquad (i = 0, 1, 2, \cdots, 10; j = 0, 1, 2, \cdots, 20)$.

因为表1.1为 6×6 的数表，故可设:

$$t_i = ih \qquad (i = 0, 1, \cdots, 5)$$

$$u_i = j\tau \qquad (j = 0, 1, \cdots, 5)$$ [3]

对于给定的 (t, u), 如果 (t, u) 满足：

$$t_i - \frac{h}{2} < t \le t_i + \frac{h}{2}, \qquad 2 \le i \le 3$$ [4]

$$u_j - \frac{\tau}{2} < u \le u_j + \frac{\tau}{2}, \qquad 2 \le j \le 3$$ [5]

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

若 t 满足：

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

或

$$t > t_3 + \frac{h}{2}$$

则相应地选取 $i = 1$ 或 $i = 4$

同样的，若 u 满足：

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

或

$$u > u_3 + \frac{\tau}{2}$$

则相应地选取 $j = 1$ 或 $j = 4$

最后得到插值多项式为

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

其中，

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

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

---

[3]其中: $h = 0.2, \tau = 0.4$

[4]计算 i, j 时有个小技巧: 可取 $i = \lfloor \frac{t}{h} + 0.5 \rfloor$

[5]同样的，$j = \lfloor \frac{u}{\tau} + 0.5 \rfloor$

2.4 曲面拟合

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

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

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

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

若参数 $\{c_{rs}^*\}$ 使得

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

在 $\boldsymbol{C}=\boldsymbol{C}^*$ 处取到最小值 $L(\boldsymbol{C}^*)$，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:

$$\boldsymbol{B}=\left[\varphi_r(x_i)\right]_{(m+1)\times(M+1)}$$
$$\boldsymbol{G}=\left[\psi_s(y_j)\right]_{(n+1)\times(N+1)}$$
$$\boldsymbol{U}=\left[u_{ij}\right]_{(m+1)\times(n+1)}$$
$$\boldsymbol{C}=\left[c_{rs}\right]_{(M+1)\times(N+1)}$$

可证得[6]，拟合曲面的系数矩阵为

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

在本实验中:

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

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

$$\boldsymbol{U}=\left[u_{ij}\right]_{(m+1)\times(M+1)}=\left[z_{ij}\right]_{11\times21}=\left[f(x_i,y_j)\right]_{11\times21}$$

[6] 在第五章的讨论中，本文将给出一种与教材不同的证明方法

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

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

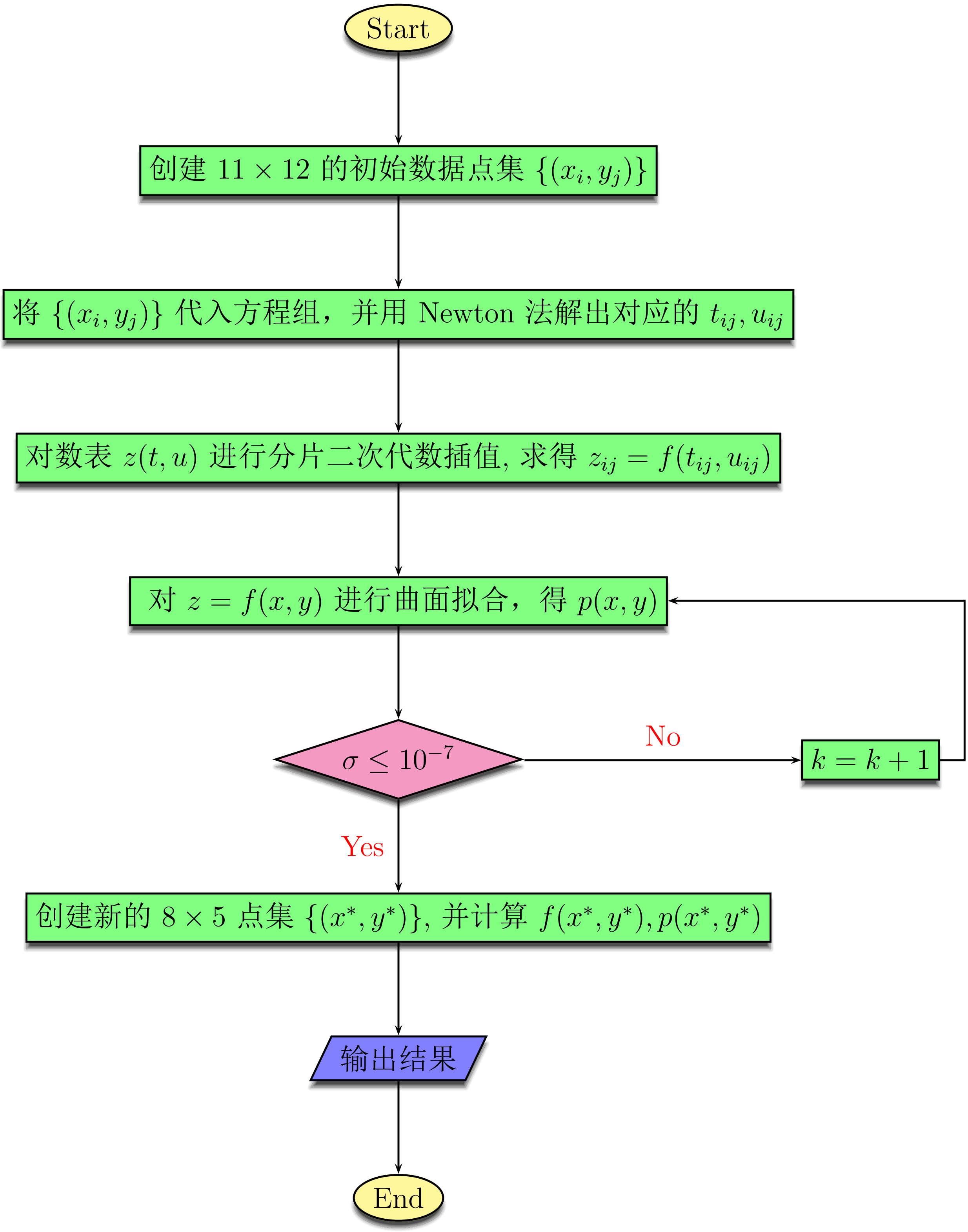
其算法如下：[7]

Algorithm 2 Surface Fitting

Set $k = 0, \sigma = 1$
while $\sigma > \varepsilon$ **do**
  Compute $\boldsymbol{C} = (\boldsymbol{B}^T\boldsymbol{B})^{-1}\boldsymbol{B}^T\boldsymbol{U}\boldsymbol{G}(\boldsymbol{G}^T\boldsymbol{G})^{-1}$
  Compute $\boldsymbol{P} = \boldsymbol{B}\boldsymbol{C}\boldsymbol{G}^T$
  Compute $\sigma = \|\boldsymbol{P} - \boldsymbol{U}\|_E^2$
  k=k+1
  if $k > N$ **then**
    Break
  end if
end while

---

[7] 写成这样的矩阵形式能有更直观的理解，详细请参看第五章


Start
创建 11×12 的初始数据点集 $\{(x_i, y_j)\}$
将 $\{(x_i, y_j)\}$ 代入方程组，并用 Newton 法解出对应的 t_{ij}, u_{ij}
对数表 $z(t, u)$ 进行分片二次代数插值, 求得 $z_{ij} = f(t_{ij}, u_{ij})$
对 $z = f(x, y)$ 进行曲面拟合，得 $p(x, y)$
$\sigma \leq 10^{-7}$
No
$k = k + 1$
Yes
创建新的 8×5 点集 $\{(x^*, y^*)\}$, 并计算 $f(x^*, y^*), p(x^*, y^*)$
输出结果
End

源程序 3

3.1 C 语言版大作业

```c
#include<stdio.h>
#include<math.h>
#include<string.h>
#define N 20
const double eps=1e-12;
double a[N][N],B[N][N],C[N][N];
int n=10;

typedef struct{
//定义复数结构体
    double Re;
    double Im;
}ComplexNumber;


int main(){
    double Q[N][N];
    freopen("Works.in","r",stdin);
    freopen("Works.out","w",stdout);
    def();
    Householder_Triangularization(a);
    printf("A_n-1:\n");
    output(a);
    QR(a,Q);
    printf("\nR:\n");
    output(a);

    return 0;
}
```

计算结果 4

后面为输出结果:

特征值与向量如下:

$\lambda_1 = 9.432879572769e - 001$

Eigenvector=(0.079620, 0.045421, -0.018272, -0.047961, -0.349567, 0.207215, -0.152312, 0.820634, -0.355466, 0.028866)

$\lambda_2 = 6.489488202110e - 001$

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

$\lambda_5 = 4.954990923624e - 002$

Eigenvector=(-0.213768, -0.206774, 0.386829, -0.031112, -0.380939, -0.125174, 0.644716, -0.308201, -0.295977, 0.043723)

$\lambda_6 = -1.493147080915e + 000$

Eigenvector=(-0.561341, 0.778192, 0.014364, -0.277602, 0.003568, -0.002548, -0.022061, -0.011758, -0.013173, 0.035016)

讨论 | 5

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:

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

然后我们将误差函数写成向量相乘的形式:

$$\begin{aligned} \boldsymbol{F}(\boldsymbol{c}) &= \sum_{i=0}^{m} \Big[f(x_i) - \sum_{k=0}^{n} c_k \varphi_k(x_i) \Big]^2 && (5.1) \\ &= (\boldsymbol{y} - \boldsymbol{A}\boldsymbol{c})^T (\boldsymbol{y} - \boldsymbol{A}\boldsymbol{c}) && (5.2) \\ &= \boldsymbol{y}^T \boldsymbol{y} - 2\boldsymbol{c}^T \boldsymbol{A}^T \boldsymbol{y} + C^T \boldsymbol{A}^T \boldsymbol{A}\boldsymbol{c} && (5.3) \end{aligned}$$

然后应用向量求导法则，令其关于 $\boldsymbol{c}$ 的偏导为 0

$$\frac{\partial \boldsymbol{F}}{\partial \boldsymbol{c}} = -2\boldsymbol{A}^T Y + 2\boldsymbol{A}^T \boldsymbol{A}\boldsymbol{c} = 0 \tag{5.4}$$

即可解得所求的 $\boldsymbol{c}$:

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

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:

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

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

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

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 $\boldsymbol{P}$，那么这个矩阵可以拆分成矩阵 $\boldsymbol{B}, \boldsymbol{C}, \boldsymbol{G}^T$ 的乘积。[1]

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

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

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

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

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

关于式 (5.8) 的证明如下:

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

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

[1]在 $k < N$ 的情况下，它都成立

$$
\begin{aligned}
\frac{\partial L}{\partial c_{rs}} &= 2\sum_{i=0}^{m}\sum_{j=0}^{n}\Big[\psi_s(y_j)\varphi_r(x_i)c_{rs}\psi_s(y_j)\varphi_r(x_i) - \psi_s(y_j)u_{ij}\varphi_r(x_i)\Big] && (5.11)\\
&= 2\sum_{i=0}^{m}\sum_{j=0}^{n}\psi_s(y_j)[\varphi_r(x_i)c_{rs}\psi_s(y_j)]\varphi_r(x_i) - 2\sum_{i=0}^{m}\sum_{j=0}^{n}\psi_s(y_j)u_{ij}\varphi_r(x_i) && (5.12)
\end{aligned}
$$

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

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：

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

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。