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

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 $f(x_i^*, y_j^*), p(x_i^*, y_j^*)$ $\quad (i = 1, 2, \cdots, 8; j = 1, 2, \cdots, 5)$ 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

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

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:$\{x_i, y_j, f(x_i, y_j)\}$ $\quad (i = 0, 1, 2, \cdots, 10; j = 0, 1, 2, \cdots, 20)$;

   (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:$\{x_i^*, y_j^*, f(x_i^*, y_j^*), p(x_i^*, y_j^*)\}$ $\quad (i = 1, 2, \cdots, 8; j = 1, 2, \cdots, 5)$。

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

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 $\sigma \leq 10^{-7}$, 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 $x_i^* = 0.1i, y_j^* = 0.5{+}0.2j \quad (i = 1, 2, \cdots, 8; j = 1, 2, \cdots, 5)$, 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。

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[1]:

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

具体算法如下:

Algorithm 1 Newton's method

```
Set x^(0) ∈ D and k = 0
while k<M do
    Compute F(x^(k)) and F'(x^(k))
    Compute Δx^(k) = −[F'(x^(k))]^(−1) F(x^(k))
    if ‖Δx^(k)‖/‖x^(k+1)‖ ≤ ε then
        x* = x^(k)
        Break
    end if
    x^(k+1) = x^(k) + Δx^(k)
    k=k+1
end while
```

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

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

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

[1]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:

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

$$u_i = j\tau \qquad (j = 0, 1, \cdots, 5)^{2}$$

对于给定的 (t, u), 如果 (t, u) 满足：

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

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

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

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

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

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

则相应地选取 $j = 1$ 或 $j = 4$

最后得到插值多项式为

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

其中，

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

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

[2] 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:

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

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

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

成立，则称相应曲面 $p^*(x,y)$ 为在曲面族 (??) 中按最小二乘原则确定的对于数据 (2.3) 的拟合曲面。

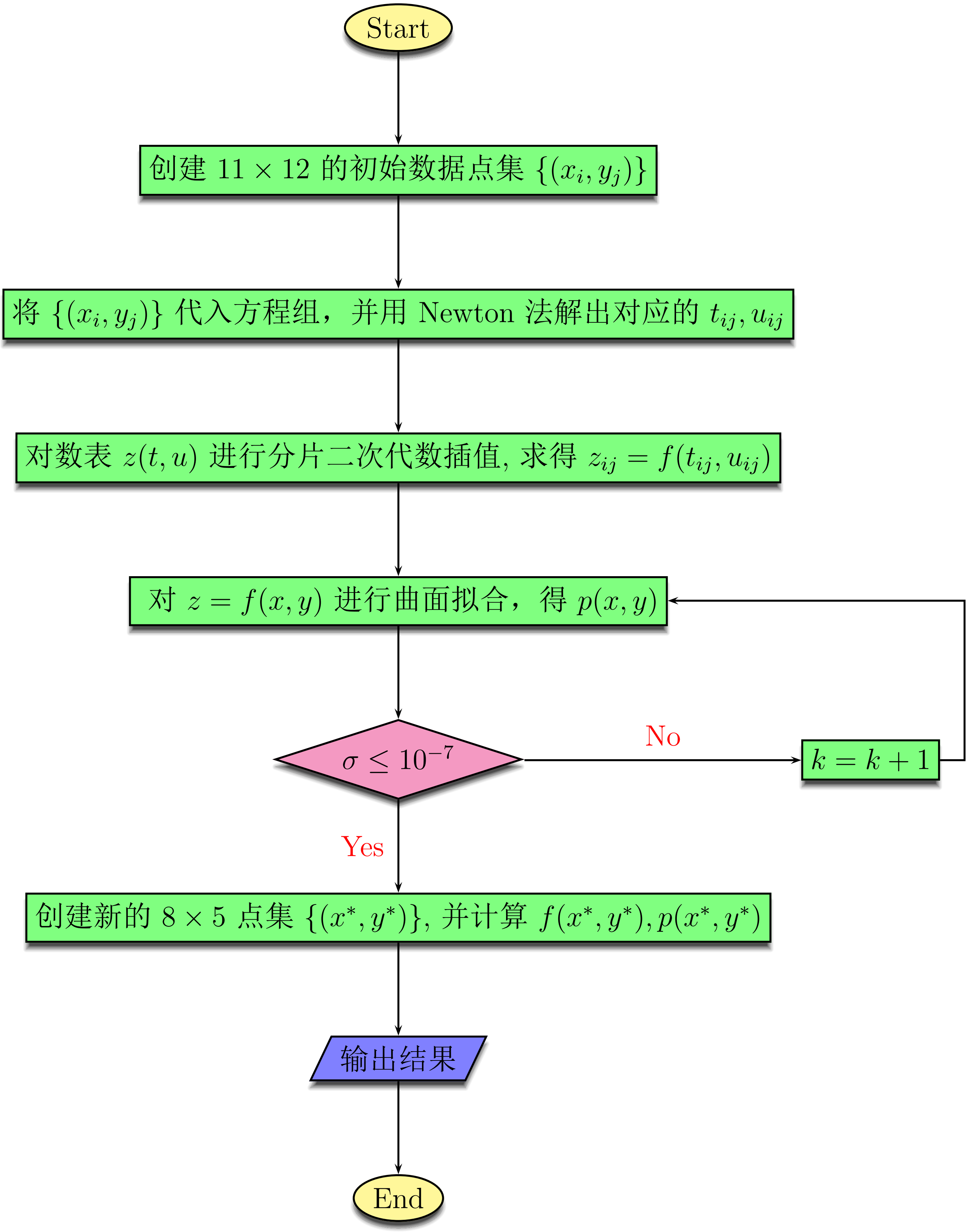
可证得[3]，拟合曲面的系数矩阵为

$$\boldsymbol{C}=(\boldsymbol{B}^T\boldsymbol{B})^{-1}\boldsymbol{B}^T\boldsymbol{U}\boldsymbol{G}(\boldsymbol{G}^T\boldsymbol{G})^{-1}\tag{2.5}$$

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

$$U=\begin{bmatrix}z_{0,0}&z_{0,1}&\cdots&z_{0,20}\\z_{1,0}&z_{1,1}&\cdots&z_{1,20}\\z_{2,0}&z_{2,1}&\cdots&z_{1,20}\\\vdots&\vdots&\ddots&\vdots\\z_{10,0}&z_{10,1}&\cdots&z_{10,20}\end{bmatrix}$$

[3]在讨论中，我将给出一种与教材不一样的证明方法


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

问题求解 3

1. 初始化定义 A 矩阵
2. 采用 $Householder$ 变化将 A 矩阵拟上三角化得到矩阵 A^{n-1}
3. 采用 QR 分解将矩阵 A^{n-1} 分解为矩阵 Q 与 R，并打印 RQ
4. 用带双步位移的 QR 方法求解所有特征值
5. 用 Gauss 消去法求解 $(A - \lambda I)X = 0$, 得到对应的特征向量
6. 输出

4 源程序

4.1 C 语言版大作业

```
#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);
    printf("\nQ:\n");
    output(Q);
    memset(B,0,sizeof(B));
    MatirixM_M(a,Q);
```

```
    printf("\nR*Q:\n");
    output(B);
    def();
    DQR(a);
    return 0;
}
```

5 计算结果

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)

$\lambda_7 = 1.590313458807e + 000$

Eigenvector=(0.062377, -0.011231, -0.252846, -0.130988, -0.381985, 0.815575, -0.123377, -0.067721, 0.271945, 0.100282)

$\lambda_8 = -2.336865932238e + 000$+ i*8.934379210213e-001

$\lambda_9 = -2.336865932238e + 000$- i*8.934379210213e-001

$\lambda_{10} = 3.389613438816e + 000$

Eigenvector=(-0.104872, -0.217677, -0.474694, -0.259384, -0.304665, -0.259452, 0.086866, 0.405258, 0.509628, 0.239515)

讨论 6

1. 在对 $n \times n$ 实矩阵 $\boldsymbol{A}$ 作 QR 分解或双步位移分解时不需要形成具体的矩阵 $\boldsymbol{H}_i$，这样可以避免矩阵与矩阵相乘，大大减少了计算量。

2. 在 QR 分解中，R 矩阵可以直接储存在 A 的上三角部分，而在 M_k 的 QR 分解中，不需要再生成 $\boldsymbol{B}, \boldsymbol{C}$ 矩阵，直接在 $\boldsymbol{M}, \boldsymbol{A}$ 矩阵中迭代即可，以节约储存空间。

3.QR 方法适用于计算一般实矩阵的全部特征值，但对于大型实矩阵，则收敛速度不够用了，这时我们为了加速收敛，可以引入位移量, 通常, 位移越离特征值越近, 收敛速度就越快，如果位移 σ 与某个特征值非常接近, 则 $\boldsymbol{A}_{n,n}^{(k)} - \sigma$ 就非常接近于 0. 这说明 $\boldsymbol{A}_{n,n}^{(k)}$ 通常会首先收敛到 $\boldsymbol{A}$ 的一个特征值. 所以令 $\sigma = \boldsymbol{A}_{n,n}^{(k)}$ 是一个不错的选择. 但是, 如果这个特征值是复数, 这种位移选取方法就可能失效.

假设 $\sigma \in \mathbb{C}$ 是 $\boldsymbol{A}$ 的某个复特征值 λ 的一个很好的近似, 则其共轭 $\overline{\sigma}$ 也应该是 $\overline{\lambda}$ 的一个很好的近似. 因此我们可以考虑**双位移策略**, 即先以 σ 为位移迭代一次, 然后再以 $\overline{\sigma}$ 为位移迭代一次, 如此不断交替进行迭代. 这样就有

$$\begin{aligned} A_1 - \sigma I &= Q_1 R_1, \\ A_2 &= R_1 Q_1 + \sigma I, \\ A_2 - \overline{\sigma} I &= Q_2 R_2, \\ A_3 &= R_2 Q_2 + \overline{\sigma} I. \end{aligned}$$

容易验证

$$A_3 = Q_2^T A_2 Q_2 = Q_2^* Q_1^* A_1 Q_1 Q_2 = Q^* A_1 Q,$$

其中 $Q = Q_1 Q_2$.

我们注意到 σ 可能是复的, 所以 Q_1 和 Q_2 都可能是复矩阵. 但我们却可以选取适当的 Q_1 和 Q_2, 使的 $Q = Q_1 Q_2$ 是实正交矩阵从而 $A_3 = Q^T A_1 Q$ 也是实矩阵. 因此我们无需计算 A_2, 而是直接由 A_1 计算出 A_3.

4. 最后，在用 Gauss 消去法求解 $(A - \lambda I)X = 0$, 求得到对应的特征向量时, 需要将 $X[n]$ 预设为 1，因为由于线性方程组 $AX = b$ 中的 b 向量全为 0，所以解的自由度为 1，需要自己先将 $X[n]$ 的值定下来，才能迭代出剩下的解，不然求出的的解全为 0.