实验二 最小二乘法（4 课时）

一、实验目的

1. 了解最小二乘拟合的基本原理和方法，注意与插值方法的区别。
2. 掌握最小二乘法。

二、实验要求

1. 掌握用 C 语言作最小二乘多项式拟合的方法。
2. 进一步加深对最小二乘法的理解。

三、实验原理

（一）最小二乘多项式拟合

已知数据对 $(x_{j,}y_j)(j=1,2,\cdots,n)$，求多项式

$$P(x)=\sum_{i=0}^{m}a_ix^i\,(m<n)$$

使得 $\Phi(a_0,a_1,\cdots,a_n)=\sum_{j=1}^{n}\left(\sum_{i=0}^{m}a_ix_j{}^i-y_j\right)^2$ 为最小，这就是一个最小二乘问题。

（二）法方程组为

$$\begin{bmatrix} m & \sum_{i=1}^{m}x_i & \cdots & \sum_{i=1}^{m}x_i^n \\ \sum_{i=1}^{m}x_i & \sum_{i=1}^{m}x_i^2 & \cdots & \sum_{i=1}^{m}x_i^{n+1} \\ \cdots & \cdots & \cdots & \cdots \\ \sum_{i=1}^{m}x_i^n & \sum_{i=1}^{m}x_i^{n+1} & \cdots & \sum_{i=1}^{m}x_i^{2n} \end{bmatrix}\begin{bmatrix} a_0 \\ a_1 \\ \cdots \\ a_n \end{bmatrix}=\begin{bmatrix} \sum_{i=1}^{m}y_i \\ \sum_{i=1}^{m}x_iy_i \\ \cdots \\ \sum_{i=1}^{m}x_i^ny_i \end{bmatrix}$$

（三）最小二乘法计算步骤

用线性函数 $P(x)=a+bx$ 为例，拟合给定数据 $(x_{i,}y_i)(i=1,2,\cdots,m)$。

算法描述：

步骤 1：输入 m 值，及 $(x_{i,}y_i)(i=1,2,\cdots,m)$。

步骤 2：建立法方程组 $A^T AX=AY$。

步骤 3：解法方程组。

步骤 4：输出 $P(x)=a+bx$。

四、实验内容

（一）算法流程图

1. 算法整体流程图（如图 2-1）

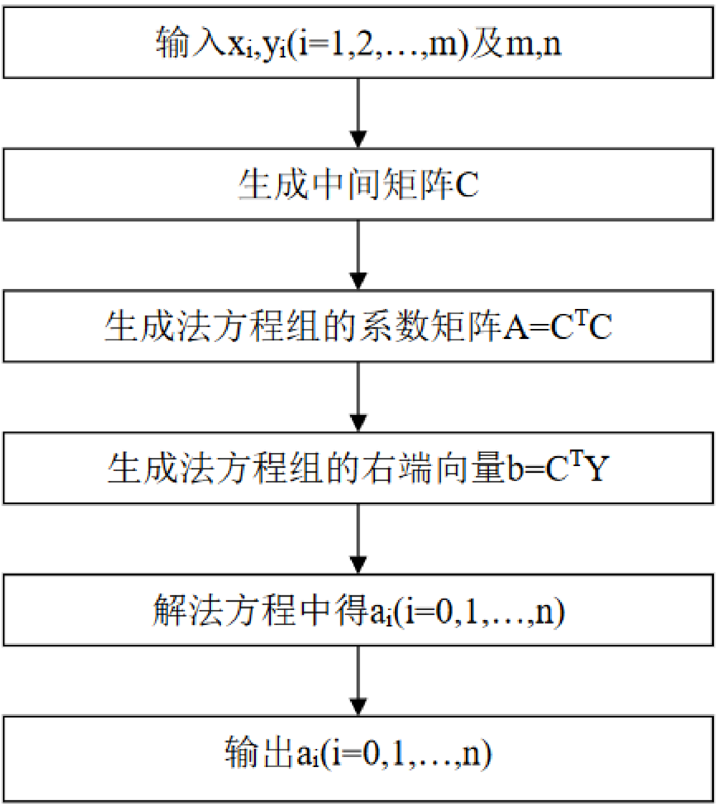

图 2-1 最小二乘法算法整体流程图

2. “生成中间矩阵 C”算法流程图（如图 2-2）

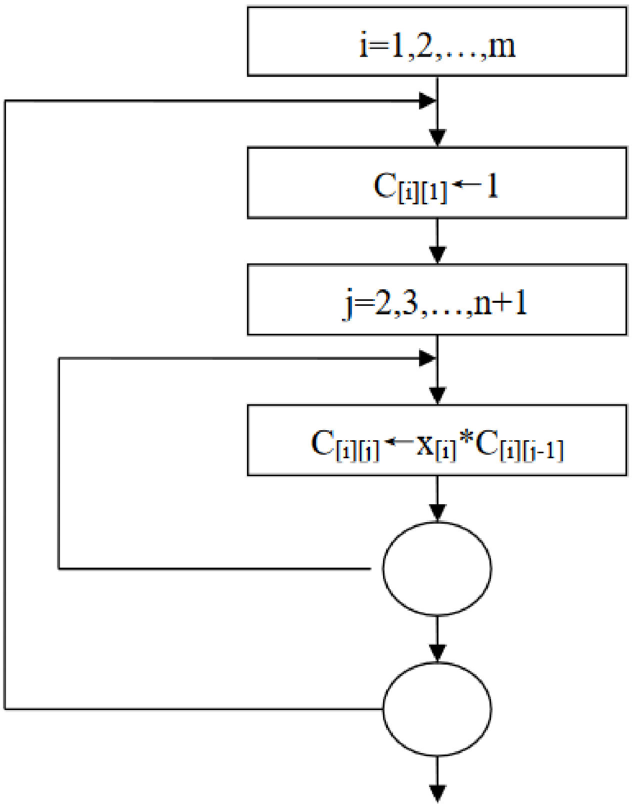

图 2-2 最小二乘法中“生成中间矩阵 C”算法流程图

3.中间矩阵 C 的重要作用

$$
C=\begin{bmatrix}1 & x_1 & x_1^{\ 2} & \dots & x_1^{\ n}\\ 1 & x_2 & x_2^{\ 2} & \dots & x_2^{\ n}\\ \vdots & \vdots & \vdots & \dots & \vdots\\ 1 & x_m & x_m^{\ 2} & \dots & x_m^{\ n}\end{bmatrix},\quad Y=\begin{bmatrix}y_1\\ y_2\\ \vdots\\ y_m\end{bmatrix}
$$

$$
\text{则 } C^T=\begin{bmatrix}1 & 1 & \dots & 1\\ x_1 & x_2 & \dots & x_m\\ x_1^{\ 2} & x_2^{\ 2} & \dots & x_2^{\ 2}\\ \vdots & \vdots & \dots & \vdots\\ x_1^{\ n} & x_2^{\ n} & \dots & x_m^{\ n}\end{bmatrix},
$$

$$
A=C^TC=\begin{bmatrix}m & \sum_{i=1}^{m}x_i & \sum_{i=1}^{m}x_i^{\ 2} & \dots & \sum_{i=1}^{m}x_i^{\ n}\\ \sum_{i=1}^{m}x_i & \sum_{i=1}^{m}x_i^2 & \sum_{i=1}^{m}x_i^3 & \dots & \sum_{i=1}^{m}x_i^{n+1}\\ \vdots & \vdots & \vdots & \dots & \vdots\\ \sum_{i=1}^{m}x_i^n & \sum_{i=1}^{m}x_i^{n+1} & \sum_{i=1}^{m}x_i^{n+2} & \dots & \sum_{i=1}^{m}x_i^{2n}\end{bmatrix},
$$

$$
b=C^TY=\begin{bmatrix}\sum_{i=1}^{m}y_i\\ \sum_{i=1}^{m}x_i y_i\\ \sum_{i=1}^{m}x_i^2 y_i\\ \vdots\\ \sum_{i=1}^{m}x_i^n y_i\end{bmatrix}
$$

（二）编程作业

测得铜导线在温度 Ti （℃)时的电阻 Ri 如下表，求电阻 R 与温度 T 的近似函数关系

i	0	1	2	3	4	5	6
Ti (℃)	19.1	25.0	30.1	36.0	40.0	45.1	50.0
Ri(Ω)	76.30	77.80	79.25	80.80	82.35	83.90	85.10

【提示】在进行程序实现时，务必注意中间矩阵的作用，以及非奇次线性方程组求解问题！

为了实验的顺利完成，此处给出解非奇次线性方程组的高斯消元法的函数。请认真阅读并理解。

```
float gs(float a[20][20],float b[20],int n )
{int i,j,k,l;
 float s;
  k=1;
      while(k!=n+1)
      {
            if(a[k][k]!=0)
           {
                  for(i=k+1;i<=n+1;i++)
                {
                        a[i][k]=a[i][k]/a[k][k];
                        b[i]=b[i]-a[i][k]*b[k];
                        for(j=k+1;j<=n+1;j++)
                           a[i][j]=a[i][j]-a[i][k]*a[k][j];
                }
           }
            k=k+1;
      }
      for(k=n+1;k>=1;k--)
      {
            s=0;
            for(l=k+1;l<=n+1;l++)
                  s=s+a[k][l]*b[l];
            b[k]=(b[k]-s)/a[k][k];
       }
 return 0;
}
```

实验主程序如下（请加上必要的注释）。

```
#include <bits/stdc++.h>

using namespace std;
int n, m;
double x[10], y[10];

/*
7 1
19.1 76.30
25.0 77.80
30.1 79.25
36.0 80.80
40.0 82.35
45.1 83.90
50.0 85.10
```

```
 */
int main() {
    // 输入
   cin >> m >> n;
    for (int i = 1; i <= m; i++) {
        cin >> x[i] >> y[i];
    }
    // 生成中间矩阵 C
    double C[m + 1][n + 2];
    for (int i = 1; i <= m; i++) {
        C[i][1] = 1;
        for (int j = 2; j <= n + 1; j++) {
            C[i][j] = x[i] * C[i][j - 1];
        }
    }
    // 生成法方程组系数矩阵 A = C^T * C
    double A[n + 2][n + 2];
    for (int i = 1; i <= n + 1; i++) {
        for (int j = 1; j <= n + 1; j++) {
            for (int k = 1; k <= m; k++) {
                A[i][j] += C[k][i] * C[k][j];//C^T * C
            }
        }
    }
    // 生成法方程组右端向量 b = C^T * Y
    double b[n + 2];
    for (int i = 1; i <= n + 1; i++) {
        for (int k = 1; k <= m; k++) {
            b[i] += C[k][i] * y[k];//C^T * Y
        }
    }
    // 使用高斯消元法解非齐次线性方程组 AX=b
   for (int k = 1; k <= n + 1; k++) {
        if (A[k][k] == 0) continue;
        for (int i = k + 1; i <= n + 1; i++) {
            A[i][k] = A[i][k] / A[k][k];
            b[i] = b[i] - A[i][k] * b[k];
            for (int j = k + 1; j <= n + 1; j++)
                A[i][j] = A[i][j] - A[i][k] * A[k][j];
        }
    }
    for (int k = n + 1; k >= 1; k--) {
        double s = 0;
        for (int l = k + 1; l <= n + 1; l++)
```

```
            s = s + A[k][l] * b[l];
        b[k] = (b[k] - s) / A[k][k];
    }
    // 输出结果
    cout << "a = ";
    for (int i = 1; i <= n + 1; i++) {
        cout << b[i] << ", ";
    }
    // 对结果进行验算,计算误差
    cout << "\nR(t) = " << b[1] << " + " << b[2] << "t\n";
    double diff = 0;
    for (int i = 1; i <= m; i++) {
        double r = b[1] + b[2] * x[i];
        double d = abs(r - y[i]) * 100 / y[i];
        cout << "R(" << x[i] << ") = " << r << " ;\t 相对误差为" << d
<< "%\n";
        diff += d;
    }
    cout << "平均误差:" << diff / m << "%\n";
}
```

拟合结果:

```
7 1
19.1 76.30
25.0 77.80
30.1 79.25
36.0 80.80
40.0 82.35
45.1 83.90
50.0 85.10
a = 70.5723, 0.291456,
R(t) = 70.5723 + 0.291456t
R(19.1) = 76.1391 ;  相对误差为0.210905%
R(25) = 77.8587 ;    相对误差为0.075408%
R(30.1) = 79.3451 ;  相对误差为0.119989%
R(36) = 81.0647 ;    相对误差为0.327573%
R(40) = 82.2305 ;    相对误差为0.145111%
R(45.1) = 83.7169 ;  相对误差为0.218206%
R(50) = 85.1451 ;    相对误差为0.0529462%
平均误差:0.164305%

进程已结束，退出代码为 0
```