随书附光盘一张

测量平差程序设计

宋力杰　编著

国防工业出版社
National Defense Industry Press

测量平差程序设计

宋力杰　编著

国防工业出版社

·北京·

内 容 简 介

本书以C语言作为编程语言，详细讨论了各种平差方法的程序设计原理、编程思路、编程技巧，给出了完整的程序代码和应用算例。平差方法包括参数平差、条件平差、具有条件的参数平差、具有参数的条件平差、参数加权平差、序贯平差、最小二乘配置、卡尔曼滤波、抗差估计、相关抗差估计、粗差探测、半参数估计、岭估计等。重点介绍了水准网、水平网、GPS向量网等网平差程序设计，网平差程序可进行最小二乘平差、粗差探测、抗差估计、自由网平差、拟稳平差等。

本书可作为高等院校测绘类本科“测量平差程序设计”课程的教学用书和“误差理论与测量平差基础”课程的辅助教学用书，也可供相关学科的研究生、科研人员和工程技术人员参考。

图书在版编目(CIP)数据

测量平差程序设计／宋力杰编著．—北京：国防工业出版社，2009.1

ISBN 978-7-118-06057-7

Ⅰ.测… Ⅱ.宋… Ⅲ.测量平差－C语言－程序设计

Ⅳ.P207 TP312

中国版本图书馆CIP数据核字(2008)第181743号

※

国防工业出版社出版发行

(北京市海淀区紫竹院南路23号 邮政编码100048)

腾飞印务有限公司印刷

新华书店经售

*

开本 787×1092 1/16 **印张** 22¼ **字数** 500千字

2009年1月第1版第1次印刷 **印数** 1—4000册 **定价** 40.00元(含光盘)

(本书如有印装错误，我社负责调换)

国防书店：(010)68428422 发行邮购：(010)68414474

发行传真：(010)68411535 发行业务：(010)68472764

前　言

电子计算机的应用，使测量平差作业模式发生了翻天覆地的变化，把平差计算者从繁重的、枯燥乏味的数字计算中彻底解放了出来。例如，一个500余点的二等三角网，在数十年前需要几十人几个月的工作才能完成平差计算，如今，同等规模的三角网平差，放在普通的计算机上，只需几分钟便可完成全部计算。这种巨变，离不开高速的电子计算机，更离不开高效的测量平差程序。

目前，测量平差程序设计能力已成为测量领域的科研人员和中高级技术人员必须具备的基本能力，很多高等院校的测量专业都开设了测量平差程序设计课程。本书的目标是为测量平差程序设计课程提供一本教材，为希望快速掌握测量平差程序设计的工程技术人员和研究人员提供一本参考书。

本书以C语言作为编程语言，详细讨论了测量工程中常用平差程序的设计原理、编程思路、编程技巧，给出了完整的程序代码和应用算例。全书共由以下7章组成。

第1章为平差辅助函数。将矩阵输出、角度值格式转换、矩阵求逆、对称矩阵下标计算、权逆阵传播、权倒数计算、等价权函数值计算等测量平差程序中常用的一些算法设计为一个个单独的函数，以便后面各章节的平差程序直接调用。

第2章为概率计算。将测量平差中常用的正态分布、χ^2分布、t分布、F分布等分布函数值计算和反函数值计算设计为一个C++类。

第3章为经典平差程序设计。包括参数平差、相关参数平差、条件平差、相关条件平差、具有条件的参数平差、具有参数的条件平差等。

第4章为近代平差程序设计。包括参数加权平差、序贯平差、最小二乘配置、卡尔曼滤波、抗差估计、相关抗差估计、粗差探测、半参数估计、岭估计等。

第3、4章的共同特点是，每一种平差方法由一个单独的函数实现，函数的功能单一，程序结构相对简单。欲调用这些函数进行某项具体的平差计算，平差的函数模型和随机模型都必须是已知的。

第5章为水准网平差程序设计。将水准网平差程序设计为一个C++类，成员函数可实现水准网的最小二乘平差、自由网平差、拟稳平差、粗差探测、闭合差计算等各种算法。

第6章为水平网平差程序设计。将水平网平差程序设计为一个C++类，成员函数可实现水平网的最小二乘平差、自由网平差、拟稳平差、粗差探测等各种算法。实

用中的三角网、测边网、导线网等都可用水平网平差程序进行平差。

第7章为GPS向量网平差程序设计。将GPS向量网平差程序设计为一个C++类，成员函数可实现GPS网的最小二乘平差、自由网平差、拟稳平差、粗差探测、抗差估计等各种算法。

第5、6、7章的共同特点是，平差程序可完成从输入原始数据到输出最后平差成果的全部计算，程序的功能丰富、结构复杂。各章充分考虑水准网、水平网、GPS网的自身特点，对程序的数据存储、算法实现等进行了多方面的优化设计。

在内容的编排上，各章首先介绍程序采用的数学模型，再讲解编程思路，然后给出程序的源代码，最后给出程序的应用实例。书中所有程序和算例均在计算机上Microsoft Visual C++6.0环境中成功编译、连接、运行过。

限于作者的能力，本书中错误之处在所难免，恳请读者批评指正。

作 者

2008年11月

目　录

第 1 章　平差辅助函数

本章标题中“函数”一词，指的是 C 语言程序中的函数。

测量平差程序中，不同的平差方法可能会用到一些相同的算法或操作，例如，正定矩阵求逆、矩阵输出、错误信息显示等，为了提高代码的利用率，减少本书的篇幅，本章将这些共同的算法或操作设计为若干个独立的函数，以供其它程序调用。

1.1　提示信息显示

应用程序一般都需要通过计算机界面动态地显示一些提示性信息，例如，正定矩阵求逆计算遇到被求逆的矩阵是非正定矩阵，程序就无法继续运行，应该通过用户界面显示“矩阵求逆失败”的提示信息，然后终止程序运行。类似这种首先显示出错信息，然后退出运行的操作，在本书的平差程序中经常出现，为此本节设计了 MyBreak 函数，专门负责在程序界面中显示提示信息。其它程序需要显示提示信息时可调用 MyBreak 函数。

1. MyBreak 函数的源代码

```
////////////////////////////////////////////////////////////////////////////
//    显示提示信息
#include <stdarg.h>
#include <stdio.h>
void MyBreak(char * fmt, ...)
{
    char buffer[256];
    va_list argptr;
    va_start(argptr, fmt);
    vsprintf(buffer, fmt, argptr);
    va_end(argptr);

#ifdef VC_EXTRALEAN
    AfxMessageBox(buffer);
#else
    printf(buffer);
    getchar();
#endif // VC_EXTRALEAN

}
```

2. 程序说明

(1)MyBreak 函数的形式参数中,fmt 是格式字符串,"…"表示参数个数可变。这种参数设置与 C 语言的库函数 printf 的参数设置类似。

(2)函数体内前五行的作用,是将需要显示的信息组合成字符串,放在 buffer 数组中。

(3)程序中有一组"#ifdef …#else…#endif"语句,是选择性编译语句,如果存在 VC_EXTRALEAN 宏定义,说明应用程序是 Windows 程序,调用 AfxMessageBox 函数将 buffer 数组的字符串显示在对话框中,否则调用 printf 函数将 buffer 数组的字符串写在计算机的屏幕上。本书的程序是在 Microsoft Visual C++6.0 环境中编辑、编译、连接的,因为在 VC 编程环境中,创建 Windows 应用程序的工程时,VC 会自动创建 VC_EXTRALEAN 宏定义,而创建控制台应用程序的工程时没有 VC_EXTRALEAN 宏定义,所以可用是否存在 VC_EXTRALEAN 宏定义来判断应用程序是 Windows 程序还是控制台程序,这种判断只适合在 VC 编程环境中编译程序。

MyBreak 函数既可作为 Windows 应用程序的一部分,也可作为控制台应用程序的一部分。

(4)MyBreak 函数可以像 printf 函数一样被调用。例如,当程序执行下列语句

```
MyBreak("文件打不开:%s","Data.txt");
```

时,应用程序若为 Windows 程序,则在对话框中显示"文件打不开:Data.txt";应用程序若是控制台程序,则在计算机的屏幕上显示"文件打不开:Data.txt"。

本书程序多处调用 MyBreak 函数,本节不再单独举例。

1.2 对称矩阵的下标计算

平差程序经常会处理对称矩阵,例如观测值的权矩阵、权逆阵、法方程系数矩阵等都是对称矩阵,为了节省存储空间和避免重复计算,本书采用仅存下三角矩阵(含主对角线元素)的存储方案存储对称矩阵,亦即仅将对称矩阵的下三角元素按顺序存到数组中。对称矩阵的下标计算,就是当对称矩阵仅存下三角矩阵时根据矩阵元素的行号和列号确定相应矩阵元素在数组中的存储位置。

1. 对称矩阵存储

设有 $n \times n$ 阶对称矩阵

$$\boldsymbol{A}=\begin{bmatrix} a_{0,0} & a_{0,1} & \cdots & a_{0,n-1} \\ a_{1,0} & a_{1,1} & \cdots & a_{1,n-1} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n-1,0} & a_{n-1,1} & \cdots & a_{n-1,n-1} \end{bmatrix} \tag{1-2-1}$$

在程序中用一维数组 a 保存该矩阵,仅存下三角矩阵元素,数组的长度为 $n \times (n+1)/2$,数组可定义为

```
double a[n*(n+1)]/2]; //n 为常量
```

或

```
double *a=new double[n*(n+1)/2];
```

我们约定,矩阵元素在一维数组 a 中按行的先后顺序依次排列,每行的元素从第 0 列到对

角线元素为止，即

$$a=[a_{0,0}\quad a_{1,0}\quad a_{1,1}\quad a_{2,0}\quad \cdots\quad a_{n-1,0}\quad a_{n-1,1}\cdots\quad a_{n-1,n-1}] \tag{1-2-2}$$

2. a_{ij}在数组 a 中的位置

按式(1-2-2)存储对称矩阵后，当 a_{ij}属于下三角矩阵时，有

$$a_{ij}=a[i\times(i+1)/2+j]\quad i\geqslant j$$

当 a_{ij}属于上三角矩阵时，数组 a 中并没有 a_{ij}，但因 $a_{ij}=a_{ji}$，故可取

$$a_{ij}=a[j\times(j+1)/2+i]\quad (i<j)$$

由此可得 a_{ij}在数组 a 中的位置计算公式：

$$f(i,j)=\begin{cases}i\times(i+1)/2+j(i\geqslant j)\\ j\times(j+1)/2+i(i<j)\end{cases} \tag{1-2-3}$$

为了向数组 a 存取数据，必须首先根据式(1-2-3)确定矩阵元素 a_{ij}在数组 a 的位置。

3. 下标计算函数

为方便起见，本节将式(1-2-3)的计算设计为一个专门的函数，函数命名为 ij，称为下标计算函数，该函数有两个参数，分别是矩阵元素的行号与列号，相当于式(1-2-3)中的 i,j，函数的返回值是按式(1-2-3)计算的 $f(i,j)$。函数的源代码如下：

```
////////////////////////////////////////////////////////////////
//          对称矩阵下标计算函数
int ij(int i,int j)
{
    return (i>=j)?i*(i+1)/2+j :j*(j+1)/2+i;
}
```

例：设 $\boldsymbol{P}$ 是 $n\times n$ 阶权矩阵，在程序中用数组 P 保存 $\boldsymbol{P}$ 的下三角元素，将 $\boldsymbol{P}$ 的对角线元素全部赋值为 1，非对角线元素全部赋值为 0，程序段如下：

```
for(int i=0; i<n; i++)
{
    for(int j=0; j<=i; j++)
    {
        if(i==j)P[ij(i,j)]=1.0;
        else    P[ij(i,j)]=0.0;
    }
}
```

1.3 数组(矩阵)输出

平差程序经常需要输出数组(矩阵)，特别是向文件中输出数组，为此本节设计了数组输出、对称矩阵输出、方程组输出三个函数。这三个函数都是向文件输出数据，输出时用 FILE 型指针变量实现对文件的写操作。

1.3.1 数组输出

设有 double 型数组 A，数组长度为 size，函数 PrintM 将数组 A 的元素写入指定文件。

1. 函数原型

```
void PrintM(FILE * fp,double A[],int size, int t,char * fmt,
            char * title=NULL,bool IsLabel=true);
```

fp——文件指针；

A——待输出的 double 型数组；

size——数组的长度，亦即输出元素的总个数；

t——每行元素的个数，亦即每 t 个数据进行一次换行；

fmt——输出格式（例如"%lf"、"%e"等）；

title——标题字符串地址，缺省值为 NULL，即无标题；

IsLabel——其值等于 true 时每行前添加行号，等于 false 时不加行号，缺省值为 true。

2. 函数源代码

```
////////////////////////////////////////////////////////////////////////
//    向文件输出数组
void PrintM(FILE * fp,double A[],int size, int t,char * fmt,char * title,bool IsLabel)
{
    if(title)fprintf(fp,"\n %s: ",title);
    int j=0;
    for(int i=0;i<size;i++)
    {
        if(i%t==0)
        {
            j++;
            if(IsLabel)fprintf(fp,"\n%3d ",j);
            else fprintf(fp,"\n ");
        }

        fprintf(fp,fmt,A[i]);
    }
    fprintf(fp,"\n");
}
```

3. 例题

设有 double 型数组 XY，数组长度等于 10，内容为 5 个点的 x、y 坐标，将数组 XY 的内容输出到“result.txt”文件，每行输出一个点的 x、y 坐标，输出格式为" %8.3lf "，前面添加行号，标题为“坐标”。程序代码如下：

```
////////////////////////////////////////////////////////////////////////
void main()
{
    double xy[]={1.0,2.0,3.0,4.0,5.5,6.6,7.7,8.8,9.9,0.0};
    FILE * fp=fopen("result.txt","w");
    if(fp==NULL)
    {
```

```
        MyBreak("结果输出文件打不开!"); //MyBreak 函数见 1.1 节
        return ;
    }
    PrintM(fp,A,10,2,"%8.3lf ","坐标",true);
    fclose(fp);
}
```

程序运行结果("result.txt"文件内容):

```
坐标:
 1      1.000      2.000
 2      3.000      4.000
 3      5.500      6.600
 4      7.700      8.800
 5      9.900      0.000
```

1.3.2 对称矩阵输出

设有 $n \times n$ 阶对称矩阵,程序中仅存矩阵下三角元素,矩阵元素为 double 型,函数 PrintM2 将矩阵输出至指定文件。

1. 函数原型

```
void  PrintM2(FILE * fp, double M[], int n,int t,char *fmt,
                    char * title = NULL,bool IsLabel = true);
```

fp——文件指针;

M——待输出的数组,数组长度 n×(n+1)/2,亦即输出元素的总个数;

n ——矩阵的阶数;

t——格式控制变量(当一行的数据个数大于 t 时,每 t 个数据进行一次换行);

fmt——输出格式(例如"%lf"、"%e"等);

title——标题字符串地址,缺省值为 NULL,即无标题;

IsLabel——其值等于 true 时每行前添加行号,等于 false 时不加行号,缺省值为 true。

2. 函数源代码

```
////////////////////////////////////////////////////////////////////////
//  向文件输出对称矩阵(下三角元素)
void  PrintM2(FILE * fp, double M[], int n, int t,char *fmt,char * title,bool IsLabel)
{
    if(title)fprintf(fp,"\n %s: ",title);

    int index = 0;
    for(int i = 0;i < n;i++)
    {
        if(IsLabel)fprintf(fp,"\n%3d ",i+1);
        else fprintf(fp,"\n     ");
```

```
        for(int j=0;j<=i;j++)
        {
            if(j>0 && j%t==0)fprintf(fp,"\n      ");
            fprintf(fp,fmt,M[index++]);
        }
    }
    fprintf(fp,"\n");
}
```

3. 例题

设有对称矩阵

$$A=\begin{bmatrix} 1.0 & 0.0 & -1.0 \\ 0.0 & 2.0 & 4.0 \\ -1.0 & 4.0 & 5.0 \end{bmatrix}$$

程序中仅存下三角矩阵,用 PrimtM2 函数将数组 A 输出到“result. txt”文件,每行前面添加行号,输出格式为" %8.3lf "。程序代码如下:

```
////////////////////////////////////////////////////////////////////////////
//   对称矩阵输出算例
void main()
{
    double A[]={1.0,0.0,2.0,-1.0,4.0,5.0};
    FILE *fp=fopen("result.txt","w");
    if(fp==NULL)
    {
        MyBreak ("结果输出文件打不开!"); //MyBreak 函数见 1.1 节
        return ;
    }
    PrintM2(fp,A,3,3,"%8.3lf ","A 矩阵",true);
    fclose(fp);
}
```

运行算例程序,“result. txt”文件内容:

```
A 矩阵:
1      1.000
2      0.000      2.000
3     -1.000      4.000      5.000
```

1.3.3 线性方程组输出

设有线性方程组 $\boldsymbol{AX}+\boldsymbol{b}=0$,$\boldsymbol{A}$ 是 $n\times t$ 矩阵,$\boldsymbol{b}$ 是 n 维向量。函数 PrintEquation 将此方程组的系数矩阵和常数项向量输出至指定文件。数据在文件中共占 n 行、$t+1$ 列,其中前 t 列是系数阵 $\boldsymbol{A}$,最后一列是常数项向量 $\boldsymbol{b}$。

1. 函数原型

```
void  PrintEquation(FILE * fp, double A[ ], double b[ ],
                    int n, int t, char *fmt, char * title = NULL)
```

fp——文件指针；

A——方程组系数矩阵，数组长度为 n × t；

b——方程组常数项向量，数组长度为 n；

n ——方程的个数；

t——方程未知数个数；

fmt——输出格式（例如"%7.3lf"、"%e"等）；

title——标题字符串地址，缺省值为 NULL，即无标题。

2. 函数源代码

```
////////////////////////////////////////////////////////////////////////
//  向文件输出线性方程组
void  PrintEquation(FILE * fp, double A[ ], double b[ ],
                    int n, int t, char *fmt, char * title)
{
    if(title)fprintf(fp,"\n %s: ",title);
    for(int i=0;i<n;i++)
    {
        fprintf(fp,"\n%3d ",i+1);
        for(int j=0;j<t;j++)
        {
            fprintf(fp,fmt,A[i*t+j]);
        }
        fprintf(fp,fmt,b[i]);
    }
}
```

3. 例题

设有线性方程组

$$\begin{bmatrix} 1 & 0 \\ 2 & -1 \\ 4 & 5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

用 PrintEquation 函数将方程组输出到“result.txt”文件，输出格式为"%8.3lf"，标题为“线性方程组”。

程序代码如下：

```
////////////////////////////////////////////////////////////////////////
//  方程输出算例
void main( )
{
    double A[ ] = {1.0, 0.0,
```

```
                2.0, -1.0,
                4.0,5.0 };
    double b[ ] = { 1.0,2.0,3.0};

    FILE * fp = fopen( "算例\\result. txt" , "w" );
    if( fp = = NULL)
    {
        MyBreak ( "结果输出文件打不开!" );
        return ;
    }
    PrintEquation( fp, A, b, 3, 2, "%8.3lf " , "线性方程组" );
    fclose( fp );
}
```

“result. txt”文件内容：

```
线性方程组：
  1     1.000     0.000     1.000
  2     2.000    -1.000     2.000
  3     4.000     5.000     3.000
```

1.4 对称正定矩阵求逆

各种平差计算都要进行法方程解算，解算过程中要对法方程系数矩阵进行求逆计算，因为法方程系数矩阵一般是对称正定矩阵，所以本节只考虑对称正定矩阵的求逆计算。求逆函数中矩阵的存储也是仅存下三角矩阵元素。

1. 公式与算法

设 $\boldsymbol{A}$ 为 $n \times n$ 阶对称正定矩阵，求 $\boldsymbol{A}$ 的逆矩阵 $\boldsymbol{A}^{-1}$。采用“变量循环重新编号法”，经过 n 次变换，将 $\boldsymbol{A}$ 变换为 $\boldsymbol{A})^{-1}$，计算公式如下：

$$
\begin{aligned}
a'_{n-1,n-1} &= 1/a_{0,0} \\
a'_{n-1,j-1} &= -a_{0,j}/a_{0,0} \quad (j = 1,2,\cdots,n-k-1) \\
a'_{n-1,j-1} &= a_{0,j}/a_{0,0} \quad (j = n-k,n-k+1,\cdots,n-1) \\
a'_{i-1,j-1} &= a_{i,j} - a_{0,i}a_{0,j}/a_{0,0} \quad (i = 1,2,\cdots,n-1;\ j = 1,2,\cdots,n-k-1) \\
a'_{i-1,j-1} &= a_{i,j} + a_{0,i}a_{0,j}/a_{0,0} \quad (i = 1,2,\cdots,n-1;\ j = n-k,n-k+1,\cdots,n-1)
\end{aligned}
$$

式中：$k = 0,1,2,\cdots,n-1$，为循环变换次数。

当 $\boldsymbol{A}$ 为对称正定矩阵时，其逆矩阵 $\boldsymbol{A}^{-1}$ 也是正定矩阵。

2. 函数原型

```
bool inverse( double a[ ], int n);
```

a——函数调用前为待求逆矩阵的元素，调用结束后为逆矩阵元素；

n——矩阵的阶数；