

# 基于Python平差程序

---

# python实现条件平差

设条件方程式为

$$BV + W = 0 \quad (3-3-1)$$

式中： $V$  为  $n$  维观测值改正数向量； $B$  为  $r \times n$  阶系数矩阵； $W$  为  $r$  维条件方程自由项向量。

由最小二乘原理，条件平差的联系数法方程为

$$BP^{-1}B^TK + W = 0 \quad (3-3-2)$$

式中： $P$  为观测值的权矩阵，并设观测值独立， $P$  为对角阵； $K$  为联系数向量。联系数向量的解

$$K = -(BP^{-1}B^T)W \quad (3-3-3)$$

改正数向量  $V$  的计算公式为

$$V = P^{-1}B^TK \quad (3-3-4)$$

单位权中误差公式为

$$\mu = \pm \sqrt{\frac{V^TPV}{r}} \quad (3-3-5)$$

观测值平差值的权逆阵为

# python实现条件平差

$$Q_{\hat{L}} = P^{-1} - P^{-1}B^T(BP^{-1}B^T)^{-1}BP^{-1} \quad (3-3-6)$$

条件平差函数,是在  $B$ 、 $W$ 、 $P$  已知的情况下计算  $V$ 、 $Q_{\hat{L}}$  和  $\mu$ 。根据上述公式,条件平差的计算步骤如下:

- (1) 计算联系数法方程的系数矩阵  $BP^{-1}B^T$ ;
- (2) 求  $BP^{-1}B^T$  的逆矩阵  $(BP^{-1}B^T)^{-1}$ ;
- (3) 由式(3-3-3)计算联系数向量  $K$ ;
- (4) 由式(3-3-4)计算改正数向量  $V$ ;
- (5) 由式(3-3-5)计算单位权中误差  $\mu$ ;
- (6) 由式(3-3-6)计算观测值平差值的权逆阵。

# python实现条件平差

设条件方程  $\mathbf{BV} + \mathbf{W} = 0$  的系数矩阵和自由项向量分别为

$$\mathbf{B} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 \\ 1 & 0 & 1 & -1 & 0 & -1 \end{bmatrix} \quad \mathbf{W} = \begin{bmatrix} -6 \\ 9 \\ 6 \end{bmatrix}$$

观测值相互独立,观测值的权分别为 1,1,2,1,2,2。试按条件平差法求改正数向量  $\mathbf{V}$ ,观测值平差值的中误差、函数  $F = \hat{L}_1 + \hat{L}_3 - \hat{L}_4$  的中误差以及单位权中误差。

# python实现条件平差

```
6 import numpy as np
7 from numpy import linalg
8
9 def ConditionAdjust(r,B,W,P):
10     """
11     n: 观测值总数;
12     r: 条件方程个数;
13     B: 条件方程系数矩阵, 数组长度为r*n, 已知;
14     W: 条件方程自由项向量, 数组长度为r, 数组内容已知;
15     P: 一观测权数组, 只有权矩阵的对角线元素数组长度为n, 已知;
16     Q: 观测值的平差值之权逆阵Q, 待计算;
17     V: 观测值改正数向量, 数组长度为n, 待计算;
18     函数返回值-若计算成功, 返回单位权中误差μ;
19     """
20
21     N = np.dot(np.dot(B,linalg.inv(P)),B.T)
22     K = -np.dot(linalg.inv(N),W)
23     V = np.dot(np.dot(linalg.inv(P),B.T),K)
24
25     Q = linalg.inv(P) - np.dot(np.dot(np.dot(linalg.inv(P),B.T),linalg.inv(N),
26     μ = np.sqrt(np.dot(np.dot(V.T,P),V)/r)
27
28     return(V,Q,μ)
29
30 if __name__ == "__main__":
31
32     # 观测值总数、条件方程个数
33     n,r = (6,3)
34     # 误差方程
35     BW = np.array([[1,1,0,0,0,0,-6],
36                     [0,0,0,0,1,-1,9],
37                     [1,0,1,-1,0,-1,6]],dtype=np.float64)
```

```
38
39     B = BW[:, :-1]
40     W = BW[:, -1]
41
42     # 权
43     P = np.diag([1,1,2,1,2,2])
44
45     # 条件平差
46     V,Q,μ = ConditionAdjust(r,B,W,P)
47
48     # 输出结果
49     print("条件方程: \n",BW)
50     print("观测值权重P: \n",P)
51     print("观测值改正数V: \n",np.round(V,3))
52     print("单位权中误差: ",np.round(μ,3))
53     print("参数平差值的逆权重:\n",np.round(Q,3))
```

# python实现条件平差

结果:

```
1  条件方程:
2  [[ 1.  1.  0.  0.  0.  0. -6.]
3   [ 0.  0.  0.  0.  1. -1.  9.]
4   [ 1.  0.  1. -1.  0. -1.  6.]]
5  观测值权重P:
6  [[1 0 0 0 0 0]
7   [0 1 0 0 0 0]
8   [0 0 2 0 0 0]
9   [0 0 0 1 0 0]
10  [0 0 0 0 2 0]
11  [0 0 0 0 0 2]]
12 观测值改正数v:
13  [ 2.  4. -1.  2. -4.  5.]
14 单位权中误差: 6.0
15 参数平差值的逆权重:
16  [[ 0.389 -0.389 -0.111  0.222  0.056  0.056]
17   [-0.389  0.389  0.111 -0.222 -0.056 -0.056]
18   [-0.111  0.111  0.389  0.222  0.056  0.056]
19   [ 0.222 -0.222  0.222  0.556 -0.111 -0.111]
20   [ 0.056 -0.056  0.056 -0.111  0.222  0.222]
21   [ 0.056 -0.056  0.056 -0.111  0.222  0.222]]
```

# python实现间接平差

设误差方程式为

$$V = A\hat{X} - L \quad (3-1-1)$$

式中:  $V$  为  $n$  维残差向量;  $A$  为  $n \times t$  阶系数矩阵;  $\hat{X}$  是  $t$  维参数向量;  $L$  是  $n$  维观测值向量。

参数的最小二乘解为

$$\hat{X} = (A^T P A)^{-1} A^T P L \quad (3-1-2)$$

其中  $P$  为观测值的权矩阵, 设观测值相互独立,  $P$  为对角阵。设

$$N = A^T P A, U = A^T P L$$

有

$$\hat{X} = N^{-1} U$$

$\hat{X}$  的权逆阵为

$$Q_{\hat{X}} = (A^T P A)^{-1} \quad (3-1-3)$$

单位权中误差公式为

$$\mu = \pm \sqrt{\frac{V^T P V}{n - t}} \quad (3-1-4)$$

参数平差函数的功能, 是根据已知的  $A$ 、 $P$ 、 $L$  计算  $\hat{X}$ 、 $V$ 、 $Q_{\hat{X}}$  及  $\mu$ 。根据上述公式, 参数平差及精度估计的计算步骤如下:

- (1) 由  $A$ 、 $P$ 、 $L$  计算  $A^T P A$ 、 $A^T P L$ ;
- (2) 求  $A^T P A$  的逆矩阵  $(A^T P A)^{-1}$ ;
- (3) 由式(3-1-2)计算参数平差值  $\hat{X}$ ;
- (4) 将  $\hat{X}$  代入式(3-1-1)计算残差  $V$ ;
- (5) 由式(3-1-4)计算单位权中误差  $\mu$ 。

# python实现间接平差

实例：

设误差方程为：

$$\begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ -1 & 0 & 0 \\ -1 & 1 & 0 \\ 0 & 1 & -1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} - \begin{bmatrix} 0 \\ -6 \\ 0 \\ 3 \\ 0 \\ -9 \end{bmatrix}$$

各观测值相互独立，观测权分别为1、1、2、1、2、2，试按参数平差法计算参数的平差值、观测值残差、单位权中误差。

# python实现间接平差

```
10 def Adjust_example(n,t,A,L,P):
11     """
12     n 观测值个数;
13     t 参数个数;
14     A 误差方程系数矩阵数组长度为 n*t, 已知;
15     L 观测值向量, 数组长度为n, 已知;
16     P 观测值权数组, 只有权矩阵的对角线元素数组长度为n, 已知
17     X 参数平差值向量, 数组长度为t, 待计算
18     N 参数平差值的权逆阵Qx;
19     V 观测值残差向量数组长度为n, 待计算;
20     函数返回值-若计算成功, 返回单位权中误差μ。
21     """
22
23     N = np.dot(np.dot(A.T,P),A)
24     U = np.dot(np.dot(A.T,P),L)
25     # print(N)
26     # print(U)
27     X = np.dot(linalg.inv(N),U)
28     V = np.dot(A,X) - L
29     μ = np.sqrt(np.dot(np.dot(V.T,P),V)/(n-t))
30     Qx = linalg.inv(N)
31
32     return (X,V,μ,Qx)
33
```

```
34 if __name__ == "__main__":
35
36     # 观测值总数和参数总数
37     n,t = (6,3)
38     # 误差方程
39     AL = np.array([[1,0,0,0],
40                    [-1,0,0,-6],
41                    [-1,1,0,0],
42                    [0,1,-1,3],
43                    [0,0,1,0],
44                    [0,0,1,-9]],dtype=np.float)
45
46     A = AL[:, :-1]
47     L = AL[:, -1]
48     # 定权
49     P = np.diag([1,1,2,1,2,2])
50
51     X,V,μ,Qx = Adjust_example(n,t,A,L,P)
52
53     print("误差方程:\n",AL)
54     print("参数平差值X:\n",X)
55     print("最小二乘残差V:\n",V)
56     print("单位权中误差: ",μ)
57     print("参数平差值的权逆阵:\n",Qx)
```

# python实现间接平差

结果:

```
1  误差方程:
2  [[ 1.  0.  0.  0.]
3   [-1.  0.  0. -6.]
4   [-1.  1.  0.  0.]
5   [ 0.  1. -1.  3.]
6   [ 0.  0.  1.  0.]
7   [ 0.  0.  1. -9.]]
8  参数平差值x:
9  [ 2.  1. -4.]
10 最小二乘残差v:
11 [ 2.  4. -1.  2. -4.  5.]
12 单位权中误差:  6.0
13 参数平差值的权逆阵:
14 [[0.38888889 0.27777778 0.05555556]
15  [0.27777778 0.55555556 0.11111111]
16  [0.05555556 0.11111111 0.22222222]]
```

# Python网络爬虫

## 爬取、存储、生成词云

---

# 一、环境搭建

```
pip install requests
pip install lxml
pip install bs4
pip install wordcloud
pip install jieba
pip install cv2
```

| 库名        | 作用                                              |
|-----------|-------------------------------------------------|
| requests  | 访问网页                                            |
| lxml      | 网页解析器                                           |
| bs4       | 使用 <i>BeautifulSoup</i> 的接口将网页字符串生成一个对象，用来提取数据。 |
| wordcloud | 词云库                                             |
| jieba     | 分词，中文引用库                                        |
| cv2       | opencv选取图片背景                                    |

## 二、网络爬取数据以txt格式保存数据

### (一) 爬取入门

```
1 # -*- coding:UTF-8 -*-
2 import requests
3 try:
4     target = 'https://baidu.com/'
5     req = requests.get(url=target)
6     print(req.text)
7 except:
8     print("爬取失败")
```



The screenshot shows a Windows-style console window titled "Console". The command prompt shows the execution of a Python script: `<terminated> _init_.py [D:\mydownload\Python\Python37\python.exe]`. The output of the script is displayed as a single line of HTML meta-information: `<!DOCTYPE html>`  
`<!--STATUS OK--><html> <head><meta http-equiv=content-type content=text/html;charset=utf-8><meta http-equiv=X-UA-Compatible content=IE=Edge><meta content=alway`

### (二) 教程示例

引用网上教程爬取豆瓣网前250部电影名称，并存入txt:

```
霸王别姬
这个杀手不太冷
阿甘正传
美丽人生
泰坦尼克号
千与千寻
辛德勒的名单
盗梦空间
机器人总动员
忠犬八公的故事
三傻大闹宝莱坞
海上钢琴师
放牛班的春天
大话西游之大圣娶亲
龙门客栈
龙猫
星际穿越
教父
熔炉
...

```

### 三、生成词云图<sup>Q</sup>片

读取txt内容，引入中文字体库（宋体）

```
3 from wordcloud import WordCloud
4 import cv2
5 import jieba
6 import matplotlib.pyplot as plt
7
8 with open('test.txt', 'r') as f:
9     text = f.read()
10
11 cut_text = " ".join(jieba.cut(text))
12
13 color_mask = cv2.imread('buck.jpg')
14
15 cloud = WordCloud(
16     # 设置字体，指定路径去加载
17     font_path="C:\\Windows\\Fonts\\simstc.ttf",
18     # font_path指定路径去加载font，simstc.ttf
19     # 设置背景色
20     background_color='white',
21     # 词云形状
22     mask=color_mask,
23     # 允许最大词汇
24     max_words=2000,
25     # 最大字体大小
26     max_font_size=40
27 )
```

