基于Python平差程序

python实现条件平差

设条件方程式为

$$BV + W = 0 \tag{3-3-1}$$

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

$$BP^{-1}B^{T}K + W = 0 \tag{3-3-2}$$

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

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

改正数向量 V 的计算公式为

$$V = P^{-1}B^{T}K \tag{3-3-4}$$

单位权中误差公式为

$$\mu = \pm\sqrt{\frac{V^{T}PV}{r}} \tag{3-3-5}$$

观测值平差值的权逆阵为

python实现条件平差

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

条件平差函数，是在 B、W、P 已知的情况下计算 V、$Q_{\hat{L}}$ 和 μ。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 μ；

(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 $BV+W=0$ 的系数矩阵和自由项向量分别为

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

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 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

```python
import numpy as np
from numpy import linalg

def ConditionAdjust(r,B,W,P):
    """
    n：观测值总数；
    r：条件方程个数；
    B：条件方程系数矩阵，数组长度为r*n，已知；
    W：条件方程自由项向量，数组长度为r，数组内容已知；
    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；
    Q：观测值的平差值之权逆阵Q，待计算；
    V：观测值改正数向量，数组长度为n，待计算；
    函数返回值-若计算成功，返回单位权中误差μ；
    """

    N = np.dot(np.dot(B,linalg.inv(P)),B.T)
    K = -np.dot(linalg.inv(N),W)
    V = np.dot(np.dot(linalg.inv(P),B.T),K)

    Q = linalg.inv(P) - np.dot(np.dot(np.dot(linalg.inv(P),B.T),linalg.inv(N
    μ = np.sqrt(np.dot(np.dot(V.T,P),V)/r)

    return(V,Q,μ)

if __name__  == "__main__":

    # 观测值总数、条件方程个数
    n,r = (6,3)
    # 误差方程
    BW = np.array([[1,1,0,0,0,0,-6],
                   [0,0,0,0,1,-1,9],
                   [1,0,1,-1,0,-1,6]],dtype=np.float64)
    B = BW[:,:-1]
    W = BW[:,-1]

    # 权
    P = np.diag([1,1,2,1,2,2])

    # 条件平差
    V,Q,μ = ConditionAdjust(r,B,W,P)

    # 输出结果
    print("条件方程：\n",BW)
    print("观测值权重P：\n",P)
    print("观测值改正数V：\n",np.round(V,3))
    print("单位权中误差：",np.round(μ,3))
    print("参数平差值的逆权重:\n",np.round(Q,3))
```

python实现条件平差

结果：

```
条件方程：
 [[ 1.  1.  0.  0.  0.  0. -6.]
 [ 0.  0.  0.  0.  1. -1.  9.]
 [ 1.  0.  1. -1.  0. -1.  6.]]
观测值权重P：
 [[1 0 0 0 0 0]
 [0 1 0 0 0 0]
 [0 0 2 0 0 0]
 [0 0 0 1 0 0]
 [0 0 0 0 2 0]
 [0 0 0 0 0 2]]
观测值改正数V：
 [ 2.  4. -1.  2. -4.  5.]
单位权中误差：6.0
参数平差值的逆权重：
 [[ 0.389 -0.389 -0.111  0.222  0.056  0.056]
 [-0.389  0.389  0.111 -0.222 -0.056 -0.056]
 [-0.111  0.111  0.389  0.222  0.056  0.056]
 [ 0.222 -0.222  0.222  0.556 -0.111 -0.111]
 [ 0.056 -0.056  0.056 -0.111  0.222  0.222]
 [ 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

$$V = A\hat{X} - L \tag{3-1-1}$$

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

$$\hat{X} = (A^{\mathrm{T}}PA)^{-1}A^{\mathrm{T}}PL \tag{3-1-2}$$

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

$$N = A^{\mathrm{T}}PA, U = A^{\mathrm{T}}PL$$

有

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

$\hat{X}$ 的权逆阵为

$$Q_{\hat{X}} = (A^{\mathrm{T}}PA)^{-1} \tag{3-1-3}$$

单位权中误差公式为

$$\mu = \pm\sqrt{\frac{V^{\mathrm{T}}PV}{n-t}} \tag{3-1-4}$$

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 μ。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 $A^{\mathrm{T}}PA$、$A^{\mathrm{T}}PL$；

(2)求 $A^{\mathrm{T}}PA$ 的逆矩阵 $(A^{\mathrm{T}}PA)^{-1}$；

(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 μ。

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

```
def Adjust_example(n,t,A,L,P):
    """
    n 观测值个数；
    t 参数个数；
    A 误差方程系数矩阵数组长度为 n*t，已知；
    L 观测值向量，数组长度为n，已知；
    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
    X 参数平差值向量，数组长度为t，待计算
    N 参数平差值的权逆阵Qx；
    V 观测值残差向量数组长度为n，待计算；
    函数返回值–若计算成功，返回单位权中误差μ。
    """

    N = np.dot(np.dot(A.T,P),A)
    U = np.dot(np.dot(A.T,P),L)
    # print(N)
    # print(U)
    X = np.dot(linalg.inv(N),U)
    V = np.dot(A,X) - L
    μ = np.sqrt(np.dot(np.dot(V.T,P),V)/(n-t))
    Qx = linalg.inv(N)

    return (X,V,μ,Qx)

```

```
if __name__ == "__main__":

    # 观测值总数和参数总数
    n,t = (6,3)
    # 误差方程
    AL = np.array([[1,0,0,0],
                   [-1,0,0,-6],
                   [-1,1,0,0],
                   [0,1,-1,3],
                   [0,0,1,0],
                   [0,0,1,-9]],dtype=np.float)

    A = AL[:,:-1]
    L = AL[:,-1]
    # 定权
    P = np.diag([1,1,2,1,2,2])

    X,V,μ,Qx = Adjust_example(n,t,A,L,P)

    print("误差方程:\n",AL)
    print("参数平差值X:\n",X)
    print("最小二乘残差V:\n",V)
    print("单位权中误差: ",μ)
    print("参数平差值的权逆阵:\n",Qx)
```

python实现间接平差

结果：

```
误差方程：
 [[ 1.  0.  0.  0.]
 [-1.  0.  0. -6.]
 [-1.  1.  0.  0.]
 [ 0.  1. -1.  3.]
 [ 0.  0.  1.  0.]
 [ 0.  0.  1. -9.]]
参数平差值X：
 [ 2.  1. -4.]
最小二乘残差V：
 [ 2.  4. -1.  2. -4.  5.]
单位权中误差：  6.0
参数平差值的权逆阵：
 [[0.38888889 0.27777778 0.05555556]
 [0.27777778 0.55555556 0.11111111]
 [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 ***BeautifulSoup*** 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

```python
# -*- coding:UTF-8 -*-
import requests
try:
    target = 'https://baidu.com/'
    req = requests.get(url=target)
    print(req.text)
except:
    print("爬取失败")
```

（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：

三、生成词云图片

读取txt内容，引入中文字体库（宋体）

```
from wordcloud import WordCloud
import cv2
import jieba
import matplotlib.pyplot as plt

with open('test.txt', 'r') as f:
    text = f.read()

cut_text = " ".join(jieba.cut(text))

color_mask = cv2.imread('buck.jpg')

cloud = WordCloud(
    font_path="C:\\Windows\\Fonts\\simsun.ttc",
    background_color='white',
    mask=color_mask,
    max_words=2000,
    max_font_size=40
)
```