Numerical Method

National Cheng Kung University Department of Engineering Science

Instructor: Chi-Hua Yu

LAB 5

Programming, Due 17:00, Tuesday, March 28th , 2023

注意事項

1. Lab 的繳交期限為星期二**(3/28)17:00 a.m.**。
2. Lab 的分數分配:Lab 分數 100%。
3. 請儘量於 Lab 時段完成練習, 完成後請找助教檢查, 檢查後即可離開。
4. 檔名規定:檔名錯誤將記為 0 分
   i. Lab: 請用 學號**_LabNumber** 為檔名做一個資料夾**(e.g.,N96091350_Lab5)**, 將 **ipynb** 檔放入資料夾, 壓縮後上傳至課程網站**(e.g., N96091350_Lab5.zip)**。
5. **Code** 中需有註解。
6. 未完成者可於下周一 **(4/03) 09:00 a.m.** 前上傳至 Moodle, 惟補交的分數將乘以0.8計, 超過期限後不予補交。
7. Bonus 需於下周一 **(4/03) 09:00 a.m.** 前上傳至 Moodle, 不予補交。
8. 準時繳交者, 請交至「Lab5 準時繳交區」;補交者, 請交至「Lab5 補交區」。

請勿抄襲, 抄襲者與被抄襲者本次作業皆**0**分計算

---

1. **(100%)** Download the template file `gauss_seidel.ipynb`. Write a Python program to solve the equations by using the Gauss–Seidel method.

$$Ax = y$$

$$\begin{bmatrix} a_{1.1} & a_{1.2} & a_{2.1} & a_{2.2} & \cdots & a_{1.n} & \cdots & a_{2.n} \\ \vdots & \vdots & a_{m.1} & a_{m.2} & \ddots & \vdots & \cdots & a_{m.n} \end{bmatrix} \begin{vmatrix} x_1 & x_2 & \vdots & x_n \end{vmatrix} = \begin{vmatrix} y_1 & y_2 & \vdots & y_m \end{vmatrix}$$

We can assume all values of $x^{(0)}$ as 0 and using following the equation to substitute $x^{(1)}$ in the first iteration.

$$x_i = \frac{1}{a_{i,i}} \left[y_i - \sum_{j=1, j\neq 1}^{j=n} a_{i,j} x_j \right]$$

After obtaining $x^{(1)}$, we continue to iterate until the difference between $x^{(k)}$ and $x^{(k-1)}$ is smaller than a predefined threshold ε = 0.0001 or the maximum iterations is reached.

Below is the running example

Sample 1

```
a = np.array([[5, -1, -3], [2, 9, 3], [2, 4, 8]])
y = np.array([14, 5, -8])

Iteration results
 k,     x1,       x2,        x3
 1    2.8000    0.5556    -1.0000
 2    2.3111    0.2667    -1.9778
 3    1.6667    0.7012    -1.7111
 4    1.9136    0.7556    -1.7673
 5    1.8907    0.7194    -1.8562
 6    1.8302    0.7541    -1.8324
```

```
 7   1.8514   0.7596   -1.8346
 8   1.8512   0.7557   -1.8427
 9   1.8455   0.7584   -1.8406
10   1.8473   0.7590   -1.8406
11   1.8474   0.7586   -1.8413
12   1.8469   0.7588   -1.8411
13   1.8471   0.7588   -1.8411
14   1.8471   0.7588   -1.8412
Converged!

Check
my solve: [ 1.84709462  0.75880367 -1.84118893]
np solve: [ 1.84705882  0.75882353 -1.84117647]
```

Sample 2

```
a = np.array([[12, 3, -5, 2], [1, 7, 3, 1], [3, 7, 13, -2], [-2, 2,
5, 20]])
y = np.array([10, 6, 3, 2])

Iteration results
 k,    x1,      x2,       x3
 1   0.8333   0.8571   0.2308   0.1000
 2   0.6985   0.6249   -0.4077   0.0399
 3   0.5006   0.9264   -0.2608   0.2093
 4   0.4582   0.8675   -0.3514   0.1226
 5   0.4496   0.9248   -0.3232   0.1469
 6   0.4430   0.9104   -0.3483   0.1333
 7   0.4384   0.9241   -0.3412   0.1403
 8   0.4368   0.9207   -0.3464   0.1367
 9   0.4360   0.9237   -0.3447   0.1382
10   0.4357   0.9229   -0.3460   0.1374
11   0.4356   0.9235   -0.3456   0.1378
12   0.4355   0.9233   -0.3458   0.1376
13   0.4355   0.9235   -0.3457   0.1377
14   0.4355   0.9234   -0.3458   0.1376
Converged!

Check
my solve: [ 0.43545583  0.92344103 -0.34580485  0.13763416]
np solve: [ 0.43544344  0.92347075 -0.34579489  0.13764599]
```