EIE 3280 Assignment 4

Tianhao Shi (120090472)

Oct 2023

Q 4.1

We can implement a Python script to solve this.

First, we can calculate $\bar{r}$

```
import numpy as np
R = np.array([
    [5, 0, 5, 4],
    [0, 1, 1, 4],
    [4, 1, 2, 4],
    [3, 4, 0, 3],
    [1, 5, 3, 0]
])

r_bar = np.mean(R[R>0])
```

$\bar{r}$=3.125

Then, we can calculate $\mathbf{A}, \mathbf{c}$

```
A = np.zeros((len(row), np.sum(R.shape)))
c = np.zeros((len(row), 1))
for idx, (_r, _c) in enumerate(zip(row, col)):
    A[idx, _r] = 1
    A[idx, _c+R.shape[0]] = 1
    c[idx] = R[_r, _c] - r_bar
```

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \end{bmatrix} \qquad \mathbf{c} = \begin{bmatrix} 1.875 \\ 1.875 \\ 0.875 \\ -2.125 \\ -2.125 \\ 0.875 \\ 0.875 \\ -2.125 \\ -1.125 \\ 0.875 \\ -0.125 \\ 0.875 \\ -0.125 \\ -2.125 \\ 1.875 \\ -0.125 \end{bmatrix}$$

After that, we can solve this. Since $\mathbf{A}^T\mathbf{A}$ is singular, we have to approximate the solutions using `np.linalg.lstsq`

```
b = np.linalg.lstsq(A.T @ A, A.T @ c)
```

We pick $\mathbf{b} = \begin{bmatrix} 1.52020202 \\ -1.20707071 \\ -0.38888889 \\ 0.06565657 \\ 0.06565657 \\ -0.19065657 \\ -0.00883838 \\ -0.37247475 \\ 0.62752525 \end{bmatrix}$

```
b = b[0]
b_usr, b_itm = b[:R.shape[0]], b[R.shape[0]:]
R_baseline = np.full_like(R, r_bar, dtype=np.float32)
delta = b_usr + b_itm.T
R_baseline += delta
```

Finally, we can calculate $\hat{R} = \begin{bmatrix} 4.4545455 & 4.6363635 & 4.2727275 & 5.2727275 \\ 1.7272727 & 1.9090909 & 1.5454545 & 2.5454545 \\ 2.5454545 & 2.7272727 & 2.3636363 & 3.3636363 \\ 3.0 & 3.1818182 & 2.8181818 & 3.8181818 \\ 3.0 & 3.1818182 & 2.8181818 & 3.8181818 \end{bmatrix}$

Q 4.3:

(a):

Solving using Python gives us

$$\mathbf{b} = \begin{bmatrix} 1.03571429 \\ 0.21428571 \\ 0.53571429 \end{bmatrix}$$

(c):

$$||\mathbf{Ab}-\mathbf{c}||_2^2 + \lambda||\mathbf{b}||_2^2 = (\mathbf{A}^T\mathbf{A}+\lambda\mathbf{I})\mathbf{b}^T\mathbf{b} - 2\mathbf{b}^T\mathbf{A}^T\mathbf{c} + \mathbf{c}^T\mathbf{c}$$

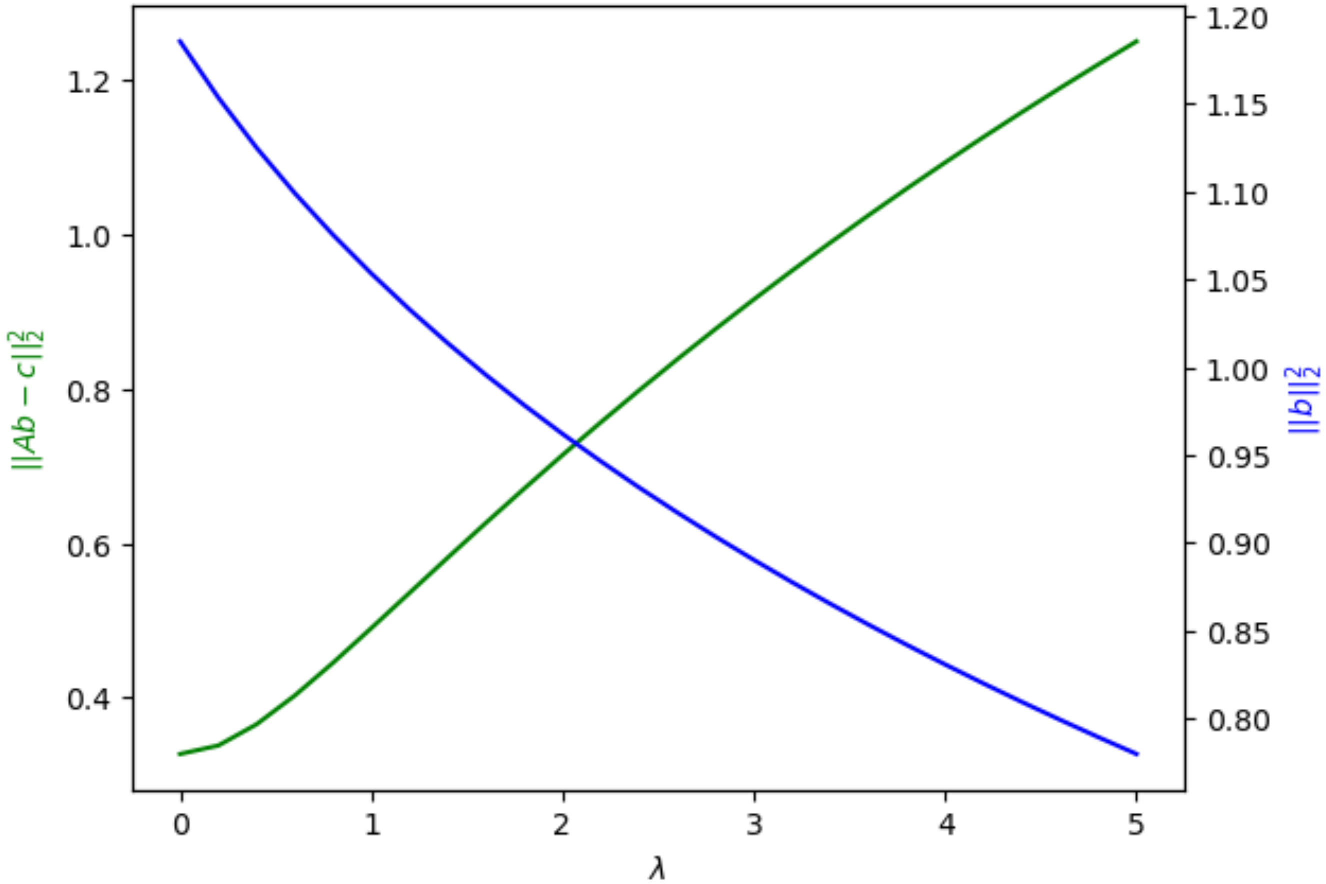
Taking derivative, and setting the derivative to 0

$$2(\mathbf{A}^T\mathbf{A}+\lambda\mathbf{I})\mathbf{b} = 2\mathbf{A}^T\mathbf{c}$$

$$(\mathbf{A}^T\mathbf{A}+\lambda\mathbf{I})\mathbf{b} = \mathbf{A}^T\mathbf{c}$$

```python
A = np.array([[1, 0, 2], [1, 1, 0], [0, 2, 1], [2, 1, 1]])
c = np.array([[2], [1], [1], [3]])
lmds = np.arange(0, 5.2, 0.2)
sq_errs = []
regs = []
for lmd in lmds:
    b = np.linalg.lstsq(A.T @ A + lmd * np.eye(A.shape[1]), A.T @ c)
    sq_err = np.linalg.norm((A @ b[0] - c))
    reg = np.linalg.norm(b[0])
    sq_errs.append(sq_err)
    regs.append(reg)
# plot with twin axis
import matplotlib.pyplot as plt
fig, ax1 = plt.subplots()
ax2 = ax1.twinx()
ax1.plot(lmds, sq_errs, 'g-')
ax2.plot(lmds, regs, 'b-')
ax1.set_xlabel('$\lambda$')
ax1.set_ylabel('$||Ab-c||_2^2$', color='g')
ax2.set_ylabel('$||b||_2^2$', color='b')
plt.show()
```

The result is shown below:



$||Ab - c||_2^2$
$||b||_2^2$
λ
0
1
2
3
4
5
0.4
0.6
0.8
1.0
1.2
0.80
0.85
0.90
0.95
1.00
1.05
1.10
1.15
1.20