הדס עטיה: 312214109

יובל גבע: 315509174

Assignment 1

1.a.

$||A||_1$=max {(1+2+5+5),(2+4+4+0),(3+4+1+3),(4+8+5+7)}=24

$X =_{def} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$ => $||X||_1$=1

Ax=$\begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 4 & -4 & 8 \\ -5 & 4 & 1 & 5 \\ 5 & 0 & -3 & -7 \end{bmatrix} * \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 4 \\ 8 \\ 5 \\ 7 \end{bmatrix}$ => $||Ax||_1$=4+8+5+7=24

⇨ $\frac{||Ax||}{||x||} = 24 = ||A||_1$

הסבר אינטואיטיבי: AX מבודד את וקטור העמודה הגדול ביותר בA והנורמה שלו היא 1.

$$||A||_\infty = \max\{(1+2+3+4),(2+4+4+8),(5+4+1+5),(5+0+3+7)\} = 18$$

$X =_{def} \begin{bmatrix} 1 \\ 1 \\ -1 \\ 1 \end{bmatrix}$ => $||X||_\infty$=1

Ax=$\begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 4 & -4 & 8 \\ -5 & 4 & 1 & 5 \\ 5 & 0 & -3 & -7 \end{bmatrix} * \begin{bmatrix} 1 \\ 1 \\ -1 \\ 1 \end{bmatrix} = \begin{bmatrix} 10 \\ 18 \\ 15 \\ 15 \end{bmatrix}$ => $||Ax||_\infty$=18

⇨ $\frac{||Ax||}{||x||} = 18 = ||A||_\infty$

הסבר אינטואיטיבי: סכום השורה הכי גדולה בA היא השורה ה-2. לכן AX מותאם לחישוב סכום שורה זו בערך מוחלט, והנורמה שלו היא 1.

1. b.

```
A = np.array([[1, 2, 3, 4], [2, 4, -4, 8], [-5, 4, 1, 5], [5, 0, -3, -7]])
At = A.transpose()
At_A = np.matmul(At, A)
w, v = LA.eig(At_A)
vector_index = np.argmax(w)
x = v[:, vector_index]
Ax = np.matmul(A, x)
norm = LA.norm(Ax, ord=2) / LA.norm(x, ord=2)
```

```
norm:  13.858100376465325
vector x: [-0.29618621  0.35616716  0.06730298  0.88367923]
```

2.a1

E(c)=Min$_c$ $\{((c-x1)^2+(c-x2)^2+(c-x3)^2\}$

=>$\frac{\partial E(c)}{\partial c} = 2(c-x1)+2(c-x2)+2(c-x3) = 0$ => $2(3c) = 2(x1+x2+x3)$

=> $c = \frac{x1+x2+x3}{3}$

2.a2

Min$_c$ $\{\max\{|c-x1|,|c-x2|,|c-x3|\}\}$ =Min$_c$ $\{\max\{|c-x1|,|c-x3|\}\}$

=>$|c-x1| = |c-x3|$

=> $c = \frac{x1+x3}{2}$

הסבר:

אם x3 קרוב יותר ל-c אז המרחק המקסימלי הוא בין x1 ל-c.

אם x1 קרוב יותר ל-c אז המרחק המקסימלי הוא בין x3 ל-c.

לכן נקטין את המרחק המקסימלי כאשר נבחר את C להיות הממוצע של שניהם. זהו המרחק המינימלי שיכול להיות .

2. a3

$\mathrm{Min}_c\ \{|c-x1|+|c-x2|+|c-x3|\} = \mathrm{Min}c\ \{c-x1+|c-x2|+x3-c\} =$
$\mathrm{Min}\ c\ \{x3-x1+|c-x2|\}$

$=\mathrm{Min}\ c\ \{|c-x2|\}$

$\Rightarrow c = x2$

הסבר:

נרצה להקטין את סך המרחקים לc.

המרחק המינימלי יהיה כאשר c נמצאת בטווח שבין הוא בין x1- x3.

2. b.

$b=\begin{bmatrix}6\\1\\5\\2\end{bmatrix},\quad A=\begin{bmatrix}2 & 1 & 2\\1 & -2 & 1\\1 & 2 & 3\\1 & 1 & 1\end{bmatrix}$

$A^tA = LL^t$

נבחין כי (A^tA) הפיכה לכן ניתן להשתמש במשוואה הנורמלית:

$\hat{x} \approx (A^tA)^{-1}A^tb = (LL^t)^{-1}A^tb = (L^T)^{-1}L^{-1}A^tb$

$$\boldsymbol{\hat{x} = (L^T)^{-1}L^{-1}A^tb}$$

```python
from numpy import matmul as mul
from numpy.linalg import inv
import numpy as np
from functools import reduce

A = np.array([[2,1,2], [1,-2,1], [1,2,3], [1,1,1]])
b = np.array([[6], [1], [5], [2]])

At = A.transpose()
At_A = mul(At, A)
L = np.linalg.cholesky(At_A)
Lt = L.transpose()

x =reduce(mul, [inv(Lt), inv(L), At, b])

print("x:", x)
print("Ax:", mul(A,x))
```

```
 X least squares via Cholesky factorization:
  [[1.7]
  [0.6]
  [0.7]]
 r:
  [[-6.00000000e-01]
  [ 2.00000000e-01]
  [-3.55271368e-15]
  [ 1.00000000e+00]]
```

2. c.

$Ax = b$,

QR factorization: $A = QR$

$\hat{x} = R^{-1}Q^T b$ (עמוד 81 בספר)

```
# QR
Q, R = np.linalg.qr(A)
x_qr = reduce(mul, [inv(R), Q.transpose(), b])
r = mul(A,x_qr) - b
print("least squares via QR factorization:\n",x_qr)
print("r:\n", r)
```

```
least squares via QR factorization:
 [[1.7]
 [0.6]
 [0.7]]
r:
 [[-6.0000000e-01]
 [ 2.0000000e-01]
 [-8.8817842e-16]
 [ 1.0000000e+00]]
```

SVD : $A = U\Sigma V^T$

$\Sigma V^T \hat{x} = U^T b$ (עמוד 87)

$\hat{x} = (\Sigma V^T)^{-1} U^T b = (V^T)^{-1} \Sigma^{-1} U^T b = V \Sigma^{-1} U^T b$

```
# SVD
u, s, vt = np.linalg.svd(A, full_matrices=False)
s = np.diag(s)
x_svd = reduce(mul, [vt.transpose(), inv(s), u.transpose(), b])
r = mul(A,x_svd) - b
print("least squares via SVD\n",x_svd)
print("r:\n", r)
```

```
least squares via SVD
 [[1.7]
 [0.6]
 [0.7]]
r:
 [[-6.00000000e-01]
 [ 2.00000000e-01]
 [-1.77635684e-15]
 [ 1.00000000e+00]]
```

2. d.

$$Ax = b \text{ , W} = \begin{bmatrix} 1000 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$\hat{x} = (A^T W A)^{-1} A^T W b$ (עמוד 73)

```
# weighted least squares
At = A.transpose()
w = np.array([1000,1, 1, 1])
w = np.diag(w)
At_w_A = reduce(mul, [At, w, A])
x_weighted = reduce(mul, [inv(At_w_A), At, w, b])
r = mul(A, x_weighted) - b

print("X weighted least squares\n", x_weighted)
print(r)
print(r[0][0])
print(abs(r[0][0]) < 1/1000 )
```

```
X weighted least squares
 [[2.17244031]
 [0.69218347]
 [0.48106425]]
[[-8.07412820e-04]
 [ 2.69137606e-01]
 [ 2.34479103e-13]
 [ 1.34568803e+00]]
-0.0008074128195900698
True
```

3. a.

```python
def gramSchmidtQR(a):
    m, n = a.shape
    # initiation
    r = np.zeros(shape=(n, n))
    q = np.zeros(shape=(m, n))
    a1 = a[:, 0]
    r[0][0] = LA.norm(a1, ord=2)
    q[:, 0] = a1 / r[0][0]

    for i in range(1, n):
        ai = a[:, i]
        q[:, i] = ai
        for j in range(0, i):
            qj = q[:, j]
            r[j][i] = np.matmul(qj.transpose(), ai)
            q[:, i] = q[:, i] - r[j][i]*qj
        r[i][i] = LA.norm(q[:, i], ord=2)
        q[:, i] = q[:, i]/r[i][i]
    return q, r
```

```python
def modifiedGramSchmidtQR(a):
    m, n = a.shape
    # initiation
    r = np.zeros(shape=(n, n))
    q = np.zeros(shape=(m, n))
    a1 = a[:, 0]
    r[0][0] = LA.norm(a1, ord=2)
    q[:, 0] = a1 / r[0][0]

    for i in range(1, n):
        ai = a[:, i]
        q[:, i] = ai
        for j in range(0, i):
            qj = q[:, j]
            qi = q[:, i]
            r[j][i] = np.matmul(qj.transpose(), qi)
            q[:, i] = q[:, i] - r[j][i]*qj
        r[i][i] = LA.norm(q[:, i], ord=2)
        q[:, i] = q[:, i]/r[i][i]
    return q, r
```

3. b.

Gram Schmidt, epsilon = 1

```
Gram Schmidt: ,epsilon = 1
Q:
[[ 0.70710678  0.40824829  0.28867513]
 [ 0.70710678 -0.40824829 -0.28867513]
 [ 0.          0.81649658 -0.28867513]
 [ 0.          0.          0.8660254 ]]
R:
[[1.41421356 0.70710678 0.70710678]
 [0.         1.22474487 0.40824829]
 [0.         0.         1.15470054]]
```

Modified Gram Schmidt, epsilon = 1

```
Modified Gram Schmidt:,epsilon = 1
Q:
[[ 0.70710678  0.40824829  0.28867513]
 [ 0.70710678 -0.40824829 -0.28867513]
 [ 0.          0.81649658 -0.28867513]
 [ 0.          0.          0.8660254 ]]
R:
[[1.41421356 0.70710678 0.70710678]
 [0.         1.22474487 0.40824829]
 [0.         0.         1.15470054]]
```

Gram Schmidt, epsilon = 1e-10

```
Gram Schmidt: ,epsilon = 1e-10
Q:
[[ 1.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 1.00000000e-10 -7.07106781e-01 -7.07106781e-01]
 [ 0.00000000e+00  7.07106781e-01  0.00000000e+00]
 [ 0.00000000e+00  0.00000000e+00  7.07106781e-01]]
R:
[[1.00000000e+00 1.00000000e+00 1.00000000e+00]
 [0.00000000e+00 1.41421356e-10 0.00000000e+00]
 [0.00000000e+00 0.00000000e+00 1.41421356e-10]]
```

Modified Gram Schmidt, epsilon = 1e-10

```
Modified Gram Schmidt:,epsilon = 1e-10
Q:
[[ 1.00000000e+00  0.00000000e+00  0.00000000e+00]
 [ 1.00000000e-10 -7.07106781e-01 -4.08248290e-01]
 [ 0.00000000e+00  7.07106781e-01 -4.08248290e-01]
 [ 0.00000000e+00  0.00000000e+00  8.16496581e-01]]
R:
[[1.00000000e+00 1.00000000e+00 1.00000000e+00]
 [0.00000000e+00 1.41421356e-10 7.07106781e-11]
 [0.00000000e+00 0.00000000e+00 1.22474487e-10]]
```

3. c.

	Gram Schmidt	Modified Gram Schmidt
$\varepsilon = 1$	1.7320508075688774	1.7320508075688772
$\varepsilon = 1e-10$	1.870828693386971	1.7320508075688776

פירוק QR למטריצת A יוצר את Q כך שהיא orthogonal matrix ($Q^TQ = I$),
המשמעות של $\|Q^TQ - I\|_F$ זהו המרחק בין התוצר של Q^TQ ממטריצת היחידה.

ניתן לראות כי באלגוריתם Modified Gram Schmidt ערכי נורמת frobenius עבור $\|Q^TQ - I\|_F$
נמוכים יותר יחסית לאלגוריתם Gram Schmidt.
ולכן ניתן לראות כי אכן Modified Gram Schmidt הוא אלגוריתם מדיוק יותר יחסית לGram Schmidt .

4.

$$V=\begin{bmatrix} v11 & \cdots & v1n \\ \vdots & \ddots & \vdots \\ vn1 & \cdots & vnn \end{bmatrix} €R^{n*n} = \begin{bmatrix} -v1- \\ \vdots \\ -vn- \end{bmatrix} \quad \Sigma^{-1} = \left[\begin{array}{ccc|ccc} \frac{1}{\Theta 1} & \cdots & 0 & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & \vdots & .. & \vdots \\ 0 & \cdots & \frac{1}{\Theta n} & 0 & .. & 0 \end{array}\right] €R^{m*n}, m \geq n$$

$$U = \begin{bmatrix} v11 & \cdots & v1n \\ \vdots & \ddots & \vdots \\ vm1 & \cdots & vmn \end{bmatrix} €R^{m*n} => U^t = \begin{bmatrix} v11 & \cdots & vm1 \\ \vdots & \ddots & \vdots \\ v1n & \cdots & vmn \end{bmatrix} = [u1^t \quad .. \quad um^t]$$

4. b.

$$\hat{x} = (A^tA)^{-1}A^tb = V\Sigma^{-1}U^tb=$$

$$= \begin{bmatrix} -v1- \\ \vdots \\ -vn- \end{bmatrix} \left[\begin{array}{ccc|ccc} \frac{1}{\Theta 1} & \cdots & 0 & 0 & \dots & 0 \\ \vdots & \ddots & \vdots & \vdots & .. & \vdots \\ 0 & \cdots & \frac{1}{\Theta n} & 0 & .. & 0 \end{array}\right] [u1^t \quad .. \quad um^t] \begin{bmatrix} b1 \\ \vdots \\ bn \end{bmatrix} =$$

$$= \sum_{i=1}^{\min(m,n)} \frac{vi}{\Theta i} * [u1^t \quad .. \quad um^t] \begin{bmatrix} b1 \\ \vdots \\ bn \end{bmatrix} = \sum_{i=1}^{\min(m,n)} \frac{vi}{\Theta i} * ui^t * \begin{bmatrix} b1 \\ \vdots \\ bn \end{bmatrix} =$$

$$= \sum_{i=1}^{\min(m,n)} \frac{vi}{\Theta i} * ui^t * b = \sum_{i=1}^{\min(m,n)} \frac{1}{\Theta i} * (ui^t * b)vi$$

$$=> \hat{\mathbf{x}} = \sum_{\mathbf{i=1}}^{\mathbf{\min(m,n)}} \frac{\mathbf{1}}{\mathbf{\Theta i}} * (\mathbf{ui^t} * \mathbf{b})\mathbf{vi}$$

4. c.

$$b=\begin{bmatrix} b1 \\ \vdots \\ bn \end{bmatrix} = \sum_{i=1}^{m} ai * ui \quad ,$$

full SVD => $\{u1, .., um\} = \text{span } R^m \quad , \Theta 1 \geq \cdots \geq \Theta n \geq 0 \ , V^t V = I,$

$U^t U = I \Rightarrow \ ui^t * ui = 1, \forall i : 1 \leq i \leq m$

$$\hat{x} = (A^t A)^{-1} A^t b = V \Sigma^{-1} U^t b =$$

$$= \begin{bmatrix} -v1- \\ \vdots \\ -vn- \end{bmatrix} \begin{bmatrix} \frac{1}{\Theta 1} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \frac{1}{\Theta n} \end{bmatrix} \begin{bmatrix} 0 & \ldots & 0 \\ \vdots & .. & \vdots \\ 0 & .. & 0 \end{bmatrix} [u1^t \quad .. \quad um^t] \begin{bmatrix} b1 \\ \vdots \\ bn \end{bmatrix} =$$

$$= \sum_{i=1}^{\min(m,n)} \frac{vi}{\Theta i} * [u1^t \quad .. \quad um^t] \begin{bmatrix} b1 \\ \vdots \\ bn \end{bmatrix} = \sum_{i=1}^{\min(m,n)} \frac{vi}{\Theta i} * ui^t * \begin{bmatrix} b1 \\ \vdots \\ bn \end{bmatrix} =$$

$$= \sum_{i=1}^{\min(m,n)} \frac{vi}{\Theta i} * ui^t * \ b = \sum_{i=1}^{\min(m,n)} \frac{vi}{\Theta i} * ui^t * \sum_{i=1}^{m} ai * ui \quad =$$

$$\sum_{i=1}^{\min(m,n)} \frac{vi}{\Theta i} * ui^t * ui * ai = \sum_{i=1}^{\min(m,n)} \frac{vi * ai}{\Theta i}$$

$$\Rightarrow \ \boldsymbol{\hat{X}} = \sum_{\mathbf{i=1}}^{\mathbf{min\,(m,n)}} \frac{\mathbf{vi} * \mathbf{ai}}{\boldsymbol{\Theta}\mathbf{i}}$$

4. d.

$$\breve{A} =_{\text{def}} \begin{bmatrix} A \\ - \\ \sqrt{\lambda} \quad 0 \\ 0 \quad \sqrt{\lambda} \end{bmatrix}, \breve{y} =_{\text{def}} \begin{bmatrix} b \\ - \\ 0 \\ 0 \end{bmatrix}$$

E(x)=$||Ax-b||_2 + \lambda||x||_2 = ||\breve{A}x - \breve{y}||_2 =$

$$=\min\{(\breve{A}x - \breve{y})^t(\breve{A}x - \breve{y})\} = \min\{(\breve{A}x)^t(\breve{A}x) - \breve{y}^t\breve{A}x - \breve{A}x\breve{y} + \breve{y}^t\breve{y}\} =$$

$$\frac{\partial E(x)}{\partial x} = 2(\breve{A}^t\breve{A}x - \breve{A}^t\breve{y}) = 0 \Rightarrow \breve{A}^t\breve{A}x = \breve{A}^t\breve{y} \Rightarrow$$

$$\begin{bmatrix} A^t| & \sqrt{\lambda} & 0 \\ & 0 & \sqrt{\lambda} \end{bmatrix} \begin{bmatrix} A \\ - \\ \sqrt{\lambda} \quad 0 \\ 0 \quad \sqrt{\lambda} \end{bmatrix} x = \begin{bmatrix} A^t| & \sqrt{\lambda} & 0 \\ & 0 & \sqrt{\lambda} \end{bmatrix} \begin{bmatrix} b \\ - \\ 0 \\ 0 \end{bmatrix}$$

$$\Rightarrow (\mathbf{A^tA + \lambda I})\mathbf{x} = \mathbf{A^tb}$$

proof $(A^tA + \lambda I)$ PD:

$$A = \begin{pmatrix} a11 & \cdots & a1n \\ \vdots & \ddots & \vdots \\ an1 & \cdots & ann \end{pmatrix}, x = \begin{bmatrix} x1 \\ \vdots \\ xn \end{bmatrix} \Rightarrow x^t = [x1 \quad \cdots \quad xn]$$

$$\Rightarrow Ax = \begin{pmatrix} a11 & \cdots & a1n \\ \vdots & \ddots & \vdots \\ an1 & \cdots & ann \end{pmatrix} \begin{bmatrix} x1 \\ \vdots \\ xn \end{bmatrix} = \begin{pmatrix} \sum_1^n a1i * xi \\ \vdots \\ \sum_1^n ani * xi \end{pmatrix}$$

$$x^TA^TAx = (Ax)^T(Ax) = (\sum_1^n a1i * xi \quad \cdots \quad \sum_1^n ani * xi) \begin{pmatrix} \sum_1^n a1i * xi \\ \vdots \\ \sum_1^n ani * xi \end{pmatrix} = \sum_{j=1}^n (\sum_1^n aji * xi)^2 \geq 0$$

$$x^t\lambda Ix = [x1 \quad \cdots \quad xn] \begin{pmatrix} \lambda & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \lambda \end{pmatrix} \begin{bmatrix} x1 \\ \vdots \\ xn \end{bmatrix} = [\lambda x1 \quad \cdots \quad \lambda xn] \begin{bmatrix} x1 \\ \vdots \\ xn \end{bmatrix} = \sum_{i=1}^n \lambda xi^2 > 0$$

$$x^t(A^tA + \lambda I)x = x^tA^tAx + x^t\lambda Ix = (Ax)^tAx + x^t\lambda Ix =$$

$$= \sum_{j=1}^n (\sum_1^n aji * xi)^2 + \sum_{i=1}^n \lambda xi^2 > 0$$

$$\Rightarrow ¥x \neq 0, \lambda > 0: \ \sum_{j=1}^n (\sum_1^n aji * xi)^2 + \sum_{i=1}^n \lambda xi^2 > 0$$

$$\Rightarrow (\mathbf{A^tA + \lambda I})\ \mathbf{PD}$$

4. e.

$$(A^tA + \lambda I) = (u\Sigma v^T)^2 u\Sigma v^T + \lambda I = v\Sigma^T u^T u\Sigma v^T + \lambda I = v\Sigma^T\Sigma v^T + \lambda I$$

$$\left(\Sigma^T\Sigma + \lambda I\right) = \begin{bmatrix} \Theta 1^2 + \lambda & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \Theta n^2 + \lambda \end{bmatrix} => \left(\Sigma^T\Sigma + \lambda I\right)^{-1} = \begin{bmatrix} \frac{1}{\Theta 1^2 + \lambda} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \frac{1}{\Theta n^2 + \lambda} \\ \vdots & \vdots & \vdots \\ 0 & \dots & 0 \end{bmatrix}$$

$$V^tV = U^tU = I => \ ui^t * ui = 1, \forall i : 1 \le i \le m$$

$$(A^tA + \lambda I)\hat{x} = A^tb \ =>$$

$$(v\Sigma^T\Sigma v^T + \lambda I\,)\hat{x} = v\Sigma^T u^T b =>$$

$$(\Sigma^T\Sigma v^T + v^t\lambda I\,)\hat{x} = \Sigma^T u^T b => (\Sigma^T\Sigma + \lambda I\,)v^t x = \Sigma^T u^T b$$

$$(\Sigma^T\Sigma + \lambda I\,)v^t\hat{x} = (\Sigma^T\Sigma + \lambda I\,)^{-1}\Sigma^T u^T b =>$$

$$\hat{x} = v(\Sigma^T\Sigma + \lambda I\,)^{-1}\Sigma^T u^T b =$$

$$= \begin{bmatrix} -v1- \\ \vdots \\ -vn- \end{bmatrix} \begin{bmatrix} \frac{1}{\Theta 1^2 + \lambda} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \frac{1}{\Theta n^2 + \lambda} \\ \vdots & \vdots & \vdots \\ 0 & \dots & 0 \end{bmatrix} \begin{bmatrix} \Theta 1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \Theta p \\ \vdots & \vdots & \vdots \\ 0 & \dots & 0 \end{bmatrix} [u1^t \quad .. \quad um^t] \begin{bmatrix} -b1- \\ \vdots \\ -bn- \end{bmatrix} =$$

$$= \sum_{i=1}^{\min(n,m)} vi \frac{1}{\Theta 1^2 + \lambda} \Theta i * \sum_{i=1}^{m} ui^t\, bi \quad = \sum_{i=1}^{\min(n,m)} vi \frac{1}{\Theta 1^2 + \lambda} \Theta i * \sum_{i=1}^{m} ai * ui^t\, ui =$$

$$= \sum_{i=1}^{\min(n,m)} vi \frac{1}{\Theta 1^2 + \lambda} \Theta i * ai$$

$$=> \ \hat{\mathbf{x}} = \sum_{\mathbf{i=1}}^{\mathbf{min\,(n,m)}} \frac{\boldsymbol{\Theta i}}{\boldsymbol{\Theta i^2 + \lambda}}\ \mathbf{ai} * \mathbf{vi}$$

4. e.

ניתן לראות כי בתמונה שבה השתמשו במינימום ריבועים הלא מנורמל-יצא פלט ללא שינוי/לא ברור,
בעוד שבתמונה שבה השתמשו במינימום ריבועים מנורמל התמונה התמקדה/לא השתנתה.

הסיבה לכך היא שאם קיימים ב-A ערכים סינגולריים קטנים-בשיטה הראשונה הם יהיו בעלי השפעה רבה על הרעש, ובשיטה השנייה כמעט ולא ישפיעו.

במינימום ריבועים לא מנורמל נקבל את הפלט: $\sum_{i=1}^{\min(m,n)} \frac{vi*ai}{\Theta i}$
נבחין כי הערכים הסינגולריים הקטנים של A משפיעים מאד על טישטוש התמונה באופן הבא

$$\lim_{\Theta i \to 0} \sum_{i=1}^{\min(m,n)} \frac{vi * ai}{\Theta i} = \infty * \ vi * ai \to \infty * \ noice$$

ככל שיש יותר ערכים סינגולריים קטנים כך נקבל תמונה רועשת ומטושטת יותר.

במינימום ריבועים המנורמל נקבל את הפלט: $\sum_{i=1}^{\min(n,m)} \frac{\Theta i}{\Theta 1^2+\lambda} \ ai * vi$

ובהנחה כי $\Theta i \ll \lambda$

$$\lim_{\Theta i \to 0} \sum_{i=1}^{\min(n,m)} \frac{\Theta i}{\Theta i^2 + \lambda} \ ai * vi = \lim_{\Theta i \to 0} \sum_{i=1}^{\min(n,m)} \frac{1}{\Theta i + \frac{\lambda}{\Theta i}} \ ai * vi \to 0 * \ noice$$

זאת אומרת שהערכים הסינגולריים הקטנים כמעט ולא משפיעים על הרעש בתמונה.

הערכים הסינגולריים הגדולים נשארו פחות או יותר בעלי אותה השפעה.

בשיטה זו נקבל את התמונה בצורה הרבה יותר ברורה.

5. a.

על מנת למצוא את מטריצה A עלינו למצוא פתרונות ל-4 נעלמים - fx,fy,y0,x0.

לשם כך המספר המינימלי של משוואות שעלינו למצוא הוא - 4.

משום שמכל סט i של וקטורים ניתן להרכיב 2 משוואות, המס המינימלי של סטים שאנו צריכים כדי למצוא פתרון לבעיה הוא 2. כן ניתן יהיה להגדיר:

$$\begin{bmatrix} u1 \\ v1 \\ u2 \\ v2 \end{bmatrix} = \begin{bmatrix} x1 & z1 & 0 & 0 \\ 0 & 0 & y1 & z1 \\ x2 & z1 & 0 & 0 \\ 0 & 0 & y2 & z2 \end{bmatrix} \begin{bmatrix} fx \\ x0 \\ fy \\ yo \end{bmatrix}$$

ישנן 4 משוואות ליניאריות ב"ת ו-4 נעלמים, זאת אומרת שקיים פתרון יחיד למערכת המשוואות.

5. b.

בהנחה שישנם n>2 סטים של זוגות וקטורים- ישנם אינסוף פתרונות.
נחפש את הפתרון הטוב ביותר על ידי פתרון באמצעות מינימום ריבועים.

נגדיר:

$$b =_{def} \begin{bmatrix} u1 \\ v1 \\ \vdots \\ un \\ vn \end{bmatrix},\ A=_{def} \begin{bmatrix} x1 & z1 & 0 & 0 \\ 0 & 0 & y1 & z1 \\ \vdots & \vdots & \vdots & \vdots \\ xn & zn & 0 & 0 \\ 0 & 0 & yn & zn \end{bmatrix},\ x_{def} = \begin{bmatrix} fx \\ x0 \\ fy \\ yo \end{bmatrix}$$

$E(fx,x0,fy,y0)=Min\ ||Ax-b||_2^2 =$

$$min \sum_{i=1}^{n}((fx * xi + x0 * zi - ui)^2 + (\mathrm{fy} * \mathrm{yi} + \mathrm{y0} * \mathrm{zi} - \mathrm{vi})^2)$$

1. $\frac{\partial E(fx,x0,fy,y0)}{\partial fx} = \sum_{i=1}^{n} 2(\mathrm{fx} * \mathrm{xi} + \mathrm{x0} * \mathrm{zi} - \mathrm{ui}) * \mathrm{xi} = 0$

$=> \ \mathrm{fx} * \sum_{i=1}^{n}(\mathrm{xi}^2) + \mathrm{x0} * \sum_{i=1}^{n}(\mathrm{zi} * \mathrm{xi}) = \sum_{i=1}^{n}(\mathrm{xi} * \mathrm{ui})$

2. $\frac{\partial E(fx,x0,fy,y0)}{\partial x0} = \sum_{i=1}^{n} 2(\mathrm{fx} * \mathrm{xi} + \mathrm{x0} * \mathrm{zi} - \mathrm{ui}) * \mathrm{zi} = 0$

$=> \ \mathrm{fx} * \sum_{i=1}^{n}(\mathrm{xi} * \mathrm{zi}) + \mathrm{x0} * \sum_{i=1}^{n}(\mathrm{zi}^2) = \sum_{i=1}^{n}(\mathrm{zi} * \mathrm{ui})$

3. $\frac{\partial E(fx,x0,fy,y0)}{\partial fy} = \sum_{i=1}^{n} 2(\mathrm{fy} * \mathrm{yi} + \mathrm{y0} * \mathrm{zi} - \mathrm{vi}) * \mathrm{yi} = 0$

$=> \ \mathrm{fy} * \sum_{i=1}^{n}(\mathrm{yi}^2) + \mathrm{y0} * \sum_{i=1}^{n}(\mathrm{zi} * \mathrm{yi}) = \sum_{i=1}^{n}(\mathrm{yi} * \mathrm{vi})$

4. $\frac{\partial E(fx,x0,fy,y0)}{\partial y0} = \sum_{i=1}^{n} 2(\mathrm{fy} * \mathrm{yi} + \mathrm{y0} * \mathrm{zi} - \mathrm{vi}) * \mathrm{zi} = 0$

$=> \ \mathrm{fy} * \sum_{i=1}^{n}(\mathrm{yi} * \mathrm{zi}) + \mathrm{y0} * \sum_{i=1}^{n}(\mathrm{zi}^2) = \sum_{i=1}^{n}(\mathrm{zi} * \mathrm{vi})$

נקבל 4 משואות ליניאריות עם 4 נעלמים:

$$\begin{bmatrix} \sum_{i=1}^{n}(xi^2) & \sum_{i=1}^{n}(zi * xi) & 0 & 0 \\ \sum_{i=1}^{n}(xi * zi) & \sum_{i=1}^{n}(zi^2) & 0 & 0 \\ 0 & 0 & \sum_{i=1}^{n}(yi^2) & \sum_{i=1}^{n}(zi * yi) \\ 0 & 0 & \sum_{i=1}^{n}(yi * zi) & \sum_{i=1}^{n}(zi^2) \end{bmatrix} * \begin{bmatrix} fx \\ x0 \\ fy \\ yo \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n}(xi * ui) \\ \sum_{i=1}^{n}(zi * ui) \\ \sum_{i=1}^{n}(yi * vi) \\ \sum_{i=1}^{n}(zi * vi) \end{bmatrix}$$

נפתור את מערכת המשוואות ונקבל את וקטור הפתרון.