CS189 HW05

Hanze Yao
3033093286

2. Rayleigh Quotient

(a) $R(M, x) = \frac{x'Mx}{x'x}$ Assume $x'x = 1$ then $R(M, x) = x'Mx$

$\because$ M is symmetric $\therefore \underset{x'x=1}{R(M,x)} = x'V\Lambda V^T x$

$\therefore$ Upper bound: when $x = V_{max}$ $V'_{max} V\Lambda V^T V_{max} = \lambda_{max}$

Lower bound: when $x = V_{min}$ $V'_{min} V\Lambda V^T V_{min} = \lambda_{min}$

(b) $\max_{w: \|w\|_2=1} \|Xw\|_2^2 = \max_{w:\|w\|_2=1} w^T X^T X w = \max_w \frac{w^T X^T X w}{w^T w}$

$= \max R(X^T X, w) = \lambda_{max}$ of $X^T X$

(c) $\underset{\lambda}{\arg\min} \|Ax - \lambda x\|_2^2$

$= \underset{\lambda}{\arg\min} (Ax - \lambda x)^T (Ax - \lambda x) = x^T A^T A x - \lambda x^T A^T x - \lambda x^T A x + \lambda^2 x^T x$

$\nabla_\lambda \|Ax - \lambda x\|_2^2 = -x^T A^T x - x^T A x + 2x^T x \lambda = 0$

$2x^T x \lambda = x^T A^T x + x^T A x$

when A is asymmetric,

$2x^T x \lambda = 2x^T A x$

$\lambda = \frac{x^T A x}{x^T x} = R(A, x)$

When x is an eigenvector, λ must be eigenvalue of A.

3. Correlation Coefficient

(a) $$\left|\rho(aX+c,\ bY+d)\right| = \left|\frac{Cov(aX+bY)}{\sqrt{Var(aX)\cdot Var(bY)}}\right| = \left|\frac{ab\ cov(X,Y)}{\sqrt{a^2\cdot b^2\cdot Var(X)\cdot Var(Y)}}\right|$$

$$= \left|\frac{a\cdot b\ Cov(X,Y)}{|a\cdot b|\sqrt{Var(X)\cdot Var(Y)}}\right| \underset{\text{when } a\cdot b\neq 0}{=} \left|\frac{Cov(X,Y)}{\sqrt{Var(X)\cdot Var(Y)}}\right| = \left|\rho(X,Y)\right|$$

(b) See the results, plots and code in Appendix

4. Canonical Correlation Analysis

(a) $\max_{\|u\|=1\ \|v\|=1} \rho(Xu, Yv) = \max_{\|u\|=1\ \|v\|=1} \dfrac{u'X'Yv}{\sqrt{u'X'Xu \cdot v'Y'Yv}}$

$\max_{u,v} \ u' C_{xy} v$

s.t. $u' C_{xx} u = 1 \quad v' C_{yy} v = 1$

where $C_{xy} = X'Y \quad C_{xx} = X'X \quad C_{yy} = Y'Y$

Lagrangian:

$L(u, v; a, b) = 2u'C_{xy}v - a(u'C_{xx}u - 1) - b(v'C_{yy}v - 1)$

$L_u = 2C_{xy}v - 2a\,C_{xx}u = 0 \Rightarrow C_{xy}v = a \cdot C_{xx}u$

$L_v = 2C_{xy}'u - 2b \cdot C_{yy}v = 0 \Rightarrow C_{xy}'u = b \cdot C_{yy}v$

$L_a = u'C_{xx}u - 1 = 0 \Rightarrow u'C_{xx}u = 1$

$L_b = v'C_{yy}v - 1 = 0 \Rightarrow v'C_{yy}v = 1$

$\Downarrow$

$u'C_{xy}v = a\,u'C_{xx}u = a \qquad \Rightarrow \quad C_{xy}v = \lambda C_{xx}u$

$v'C_{xy}'u = b\,v'C_{yy}v = b \qquad \quad C_{xy}'u = \lambda C_{yy}v$

$a = b = \lambda$

$\because C_{xy}' = C_{yx} \quad \therefore \quad C_{xy}v = \lambda C_{xx}u$

$C_{yx}u = \lambda C_{yy}v$

$$\therefore \begin{pmatrix} & C_{xy} \\ C_{yx} & \end{pmatrix}\begin{pmatrix} u \\ v \end{pmatrix} = \lambda \begin{pmatrix} C_{xx} & \\ & C_{yy} \end{pmatrix}\begin{pmatrix} u \\ v \end{pmatrix}$$

(b) Proof:

$$\begin{pmatrix} & C_{xy} \\ C_{yx} & \end{pmatrix}\begin{pmatrix} u \\ v \end{pmatrix} = \lambda \begin{pmatrix} C_{xx} & \\ & C_{yy} \end{pmatrix}\begin{pmatrix} u \\ v \end{pmatrix}$$

$$\begin{pmatrix} & C_{xy} \\ C_{yx} & \end{pmatrix}\begin{pmatrix} u \\ v \end{pmatrix} = \lambda \begin{pmatrix} C_{xx} & \\ & C_{yy} \end{pmatrix}^{\frac{1}{2}}\begin{pmatrix} C_{xx} & \\ & C_{yy} \end{pmatrix}^{\frac{1}{2}}\begin{pmatrix} u \\ v \end{pmatrix}$$

$$\begin{pmatrix} C_{yx} & \\ & C_{yx} \end{pmatrix}^{-\frac{1}{2}}\begin{pmatrix} & C_{xy} \\ C_{yx} & \end{pmatrix}\begin{pmatrix} u \\ v \end{pmatrix} = \lambda \begin{pmatrix} C_{xx} & \\ & C_{yy} \end{pmatrix}^{\frac{1}{2}}\begin{pmatrix} u \\ v \end{pmatrix}$$

$$\begin{pmatrix} C_{xx} & \\ & C_{yy} \end{pmatrix}^{-\frac{1}{2}} \begin{pmatrix} & C_{xy} \\ C_{yx} & \end{pmatrix} \begin{pmatrix} C_{xx} & \\ & C_{yy} \end{pmatrix}^{-\frac{1}{2}} \begin{pmatrix} C_{xx} & \\ & C_{yy} \end{pmatrix}^{\frac{1}{2}} \begin{pmatrix} u \\ v \end{pmatrix} = \lambda \begin{pmatrix} C_{xx} & \\ & C_{yy} \end{pmatrix}^{\frac{1}{2}} \begin{pmatrix} u \\ v \end{pmatrix}$$

After Step 1 $\begin{pmatrix} \tilde{u} \\ \tilde{v} \end{pmatrix} = \begin{pmatrix} C_{xx} & \\ & C_{yy} \end{pmatrix}^{\frac{1}{2}} \begin{pmatrix} u \\ v \end{pmatrix}$

The equation becomes

$$\begin{pmatrix} C_{xx} & \\ & C_{yy} \end{pmatrix}^{-\frac{1}{2}} \begin{pmatrix} & C_{xy} \\ C_{yx} & \end{pmatrix} \begin{pmatrix} C_{xx} & \\ & C_{yy} \end{pmatrix}^{-\frac{1}{2}} \begin{pmatrix} \tilde{u} \\ \tilde{v} \end{pmatrix} = \lambda \begin{pmatrix} \tilde{u} \\ \tilde{v} \end{pmatrix}$$

$$\begin{pmatrix} & C_{xx}^{-\frac{1}{2}} C_{xy} C_{yy}^{-\frac{1}{2}} \\ C_{xy}^{-\frac{1}{2}} C_{yx} C_{xx}^{-\frac{1}{2}} & \end{pmatrix} \begin{pmatrix} \tilde{u} \\ \tilde{v} \end{pmatrix} = \lambda \begin{pmatrix} \tilde{u} \\ \tilde{v} \end{pmatrix}$$

After step 2 : $(\tilde{u}, \tilde{v}, \lambda) = \text{svd}\left(C_{xx}^{-\frac{1}{2}} C_{xy} C_{yy}^{-\frac{1}{2}} \right)$

$$\tilde{u} \lambda \tilde{v}' = C_{xx}^{-\frac{1}{2}} C_{xy} C_{yy}^{-\frac{1}{2}}$$

$$\begin{pmatrix} & \tilde{u}\lambda\tilde{v}' \\ \tilde{v}\lambda\tilde{u}' & \end{pmatrix} \begin{pmatrix} \tilde{u} \\ \tilde{v} \end{pmatrix} = \begin{pmatrix} \tilde{u}\lambda\tilde{v}'\tilde{v} \\ \tilde{v}\lambda\tilde{u}'\tilde{u} \end{pmatrix} = \begin{pmatrix} \lambda\tilde{u} \\ \lambda\tilde{v} \end{pmatrix} = \lambda \begin{pmatrix} \tilde{u} \\ \tilde{v} \end{pmatrix}$$

$\therefore$ We get $\tilde{u}, \tilde{v}$

To get back to u and v, we do step 3.

$$\begin{pmatrix} \tilde{u} \\ \tilde{v} \end{pmatrix} = \begin{pmatrix} C_{xx} & \\ & C_{yy} \end{pmatrix}^{\frac{1}{2}} \begin{pmatrix} u \\ v \end{pmatrix}$$

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} C_{xx} & \\ & C_{yy} \end{pmatrix}^{-\frac{1}{2}} \begin{pmatrix} \tilde{u} \\ \tilde{v} \end{pmatrix}$$

(c) $\rho(Xu, Yv) = \rho\left(X C_{xx}^{-\frac{1}{2}} \cdot C_{xx}^{\frac{1}{2}} u, Y C_{yy}^{-\frac{1}{2}} C_{yy}^{\frac{1}{2}} v\right) = \rho(\tilde{X}\tilde{u}, \tilde{Y}\tilde{v})$

$$\frac{u' C_{xy} v}{\sqrt{u' C_{xx} u \cdot v' C_{yy} v}}$$

$$C_{\tilde{X}\tilde{Y}} = \tilde{X}'\tilde{Y} = C_{xx}^{-\frac{1}{2}} C_{xy} C_{yy}^{-\frac{1}{2}}$$

$$C_{\tilde{X}\tilde{X}} = \tilde{X}'\tilde{X} = (X C_{xx}^{-\frac{1}{2}})' X C_{xx}^{-\frac{1}{2}} = C_{xx}^{-\frac{1}{2}} X'X C_{xx}^{-\frac{1}{2}}$$
$$= C_{xx}^{-\frac{1}{2}} C_{xx} C_{xx}^{-\frac{1}{2}} = I$$

$$C_{\tilde{Y}\tilde{Y}} = \tilde{Y}'\tilde{Y} = (Y C_{yy}^{-\frac{1}{2}})' Y C_{yy}^{-\frac{1}{2}} = C_{yy}^{-\frac{1}{2}} Y'Y C_{yy}^{-\frac{1}{2}} = C_{yy}^{-\frac{1}{2}} C_{yy} C_{yy}^{-\frac{1}{2}}$$
$$= I$$

$\therefore \tilde{X}$ and $\tilde{Y}$ are whitened in their own spaces

(d) $\rho(Xu, Yv) = \rho(\tilde{X}\tilde{u}, \tilde{Y}\tilde{v}) = \dfrac{\tilde{u}' C_{\tilde{X}\tilde{Y}} \tilde{v}}{\sqrt{\tilde{u}'\tilde{u} \cdot \tilde{v}'\tilde{v}}}$

$$= \frac{\tilde{u}' C_{XX}^{-\frac{1}{2}} C_{XY} C_{YY}^{-\frac{1}{2}} \tilde{v}}{\sqrt{\tilde{u}'\tilde{u} \cdot \tilde{v}'\tilde{v}}}$$

Assume $\|\tilde{u}\|_2 = \|\tilde{v}\|_2 = 1$

~~then~~ $\max\limits_{\|\tilde{u}\|_2 = \|\tilde{v}\|_2 = 1} \rho(\tilde{X}\tilde{u}, \tilde{Y}\tilde{v}) = \max\limits_{\|\tilde{u}\|_2 = \|\tilde{v}\|_2 = 1} \tilde{u}' C_{XX}^{-\frac{1}{2}} C_{XY} C_{YY}^{-\frac{1}{2}} \tilde{v}$

$\because$ step 2 : $(\tilde{u}, \tilde{v}, \lambda) = \mathrm{svd}(C_{XX}^{-\frac{1}{2}} C_{XY} C_{YY}^{-\frac{1}{2}})$

$\therefore \tilde{u} \lambda \tilde{v}' = C_{XX}^{-\frac{1}{2}} C_{XY} C_{YY}^{-\frac{1}{2}}$

$\lambda = \tilde{u}' C_{XX}^{-\frac{1}{2}} C_{XY} C_{YY}^{-\frac{1}{2}} \tilde{v}$

$\therefore = \max\limits_{\|\tilde{u}\|_2 = \|\tilde{v}\|_2 = 1} \lambda = \lambda_1$

(e) The plots and code are shown in Appendix.

Appendix

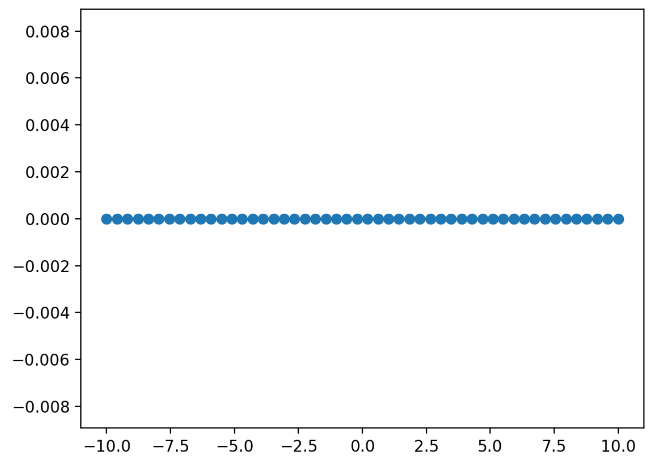
3.

(b)

The code is pointsGenerator.py. The results are shown in the picture below.

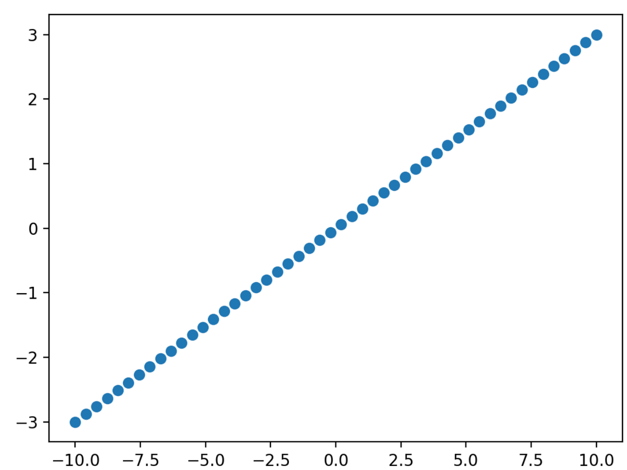
```
N/A
0.9999999999999998
0.9999999999999998
0.9764150185459565
-0.06013660964413438
0.5937835513268563
```

Dataset 1



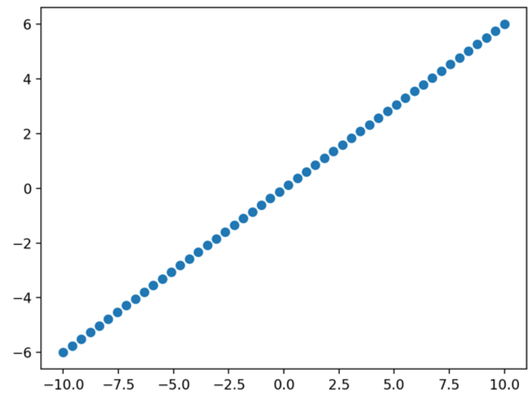
Correlation coefficient: N/A

Dataset 2



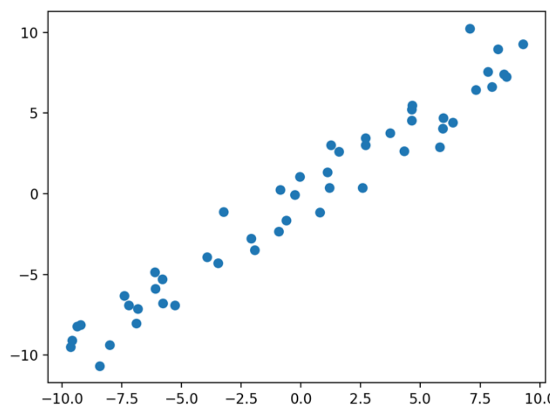
Correlation coefficient: 0.99998

Dataset 3:



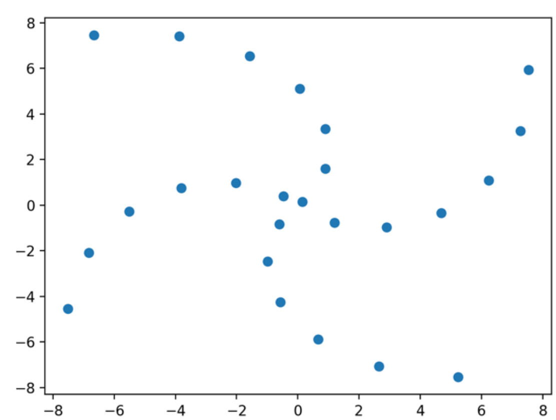
Correlation coefficient: 0.99998

Dataset 4:



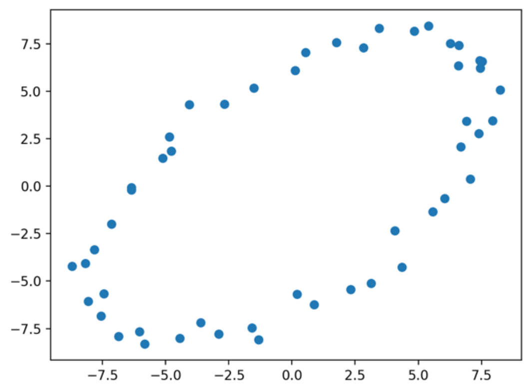
Correlation coefficient: 0.976415

Dataset 5



Correlation coefficient: -0.06013

Dataset 6:

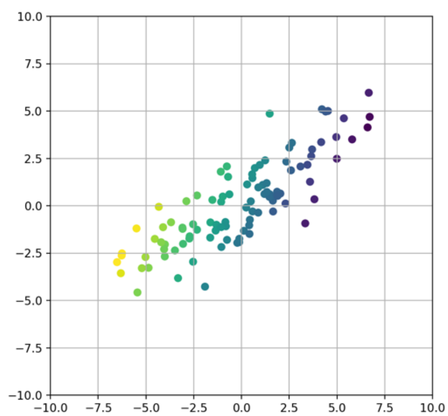


Correlation coefficient: 0.593784

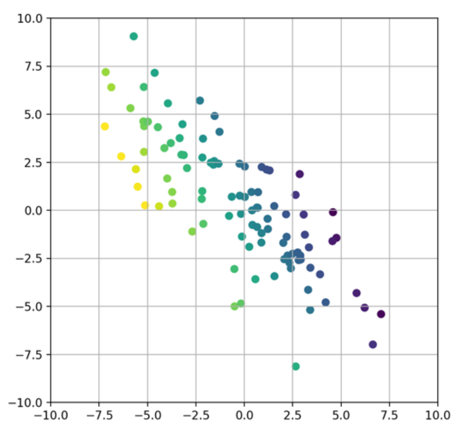
4.

The code is in CCA.py

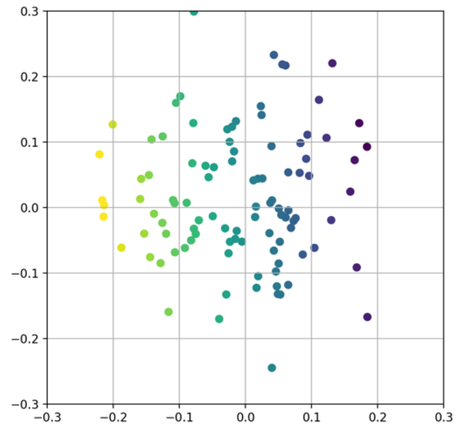
X



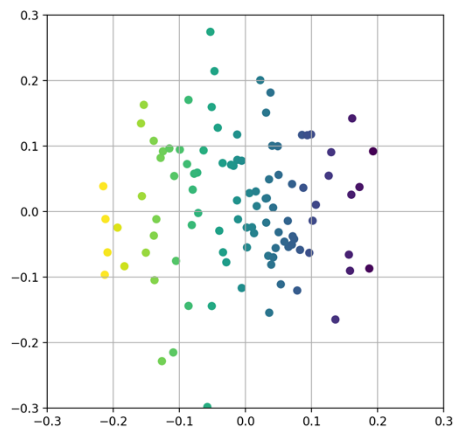
Y



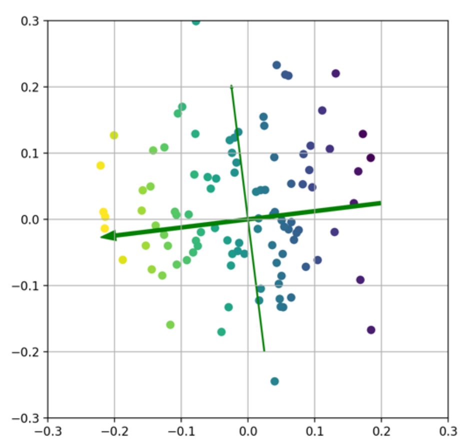
Whitened X



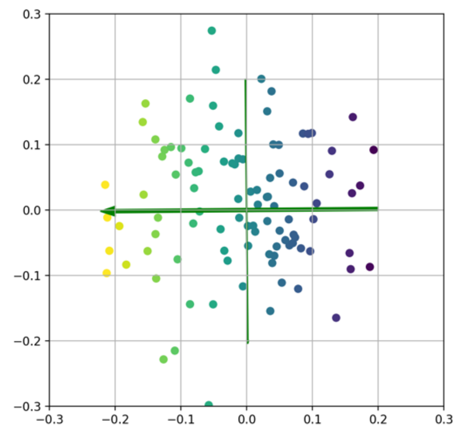
Whitened Y



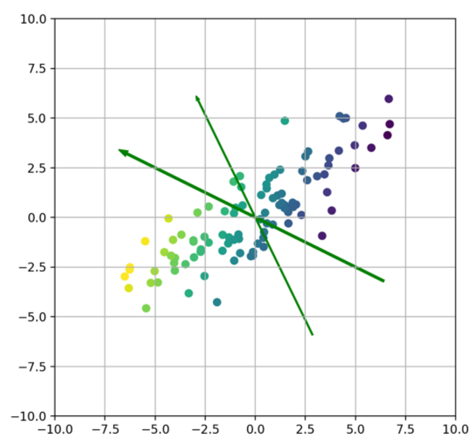
CCA on whitened X



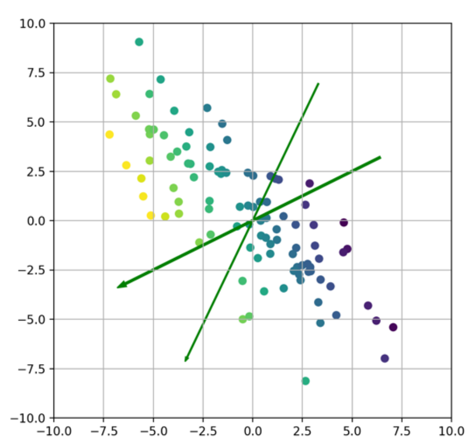
CCA on whitened Y



CCA on X



CCA on Y



```
import numpy as np
import matplotlib.pyplot as plt

#################
# Generate Data #
#################

num_points = 50
x = np.linspace(-10,10,num_points)

#Dataset 1
X_1 = np.vstack((x,np.zeros(num_points))).T
#Dataset 2
X_2 = np.vstack((x,0.3*x)).T
#Dataset 3
X_3 = np.vstack((x,0.6*x)).T
#Dataset 4
X_4 = np.vstack((x,x)).T + np.random.randn(num_points,2)
#Dataset 5
x_abs = abs(x)
X_5 = np.vstack((x_abs*np.cos(4*x_abs),x_abs*np.sin(4*x_abs))).T
#Dataset 6
t = np.linspace(0,359,num_points) * np.pi/180
X_6 = np.vstack((10*np.cos(t),5*np.sin(t))).T
cs = np.cos(-np.pi/4)
ss = np.sin(-np.pi/4)
X_6 = X_6 @ np.asarray([[cs,-ss],[ss,cs]])
X_6 = X_6 + np.random.randn(num_points,2) * 0.5

#Correlation Coefficient calculation and Dataset plot function
def CorrCoeff(data):
    x = data[:, 0].T
    y = data[:, 1].T
    mean_x = np.mean(x)
    mean_y = np.mean(y)
    var_x = np.var(x)
    var_y = np.var(y)
    plt.scatter(x, y)
    plt.show()
    if var_x == 0 or var_y == 0:
        return "N/A"
    return
     np.sum((x-mean_x)*(y-mean_y))/np.sqrt(np.sum((x-mean_x)**2)*np
     .sum((y-mean_y)**2))

#Dataset 1 result
x_1 = CorrCoeff(X_1)
print(x_1)

#Dataset 2 result
x_2 = CorrCoeff(X_2)
print(x_2)

#Dataset 3 result
x_3 = CorrCoeff(X_3)
```

```
print(x_3)

#Dataset 4 result
x_4 = CorrCoeff(X_4)
print(x_4)

#Dataset 5 result
x_5 = CorrCoeff(X_5)
print(x_5)

#Dataset 6 result
x_6 = CorrCoeff(X_6)
print(x_6)
```

```python
import numpy as np
import matplotlib.pyplot as plt
from scipy.linalg import fractional_matrix_power

#################
# Generate Data #
#################

num_points = 100

sigH = 3.5
sigNx = 2
sigNy = 3

Hx = np.sort(np.random.randn(num_points,1)) * sigH
Hy = Hx

Nx = np.random.randn(num_points,1) * sigNx
Ny = np.random.randn(num_points,1) * sigNy

t = np.asarray([[1,0.5],[0.5,1]])
X = np.hstack((Hx,Nx)) @ t
t = np.asarray([[1,-0.5],[-0.5,1]])
Y = np.hstack((Hy,Ny)) @ t

c_xx = X.T@X
c_yy = Y.T@Y
c_xy = X.T@Y
X_w = X@(fractional_matrix_power(c_xx, -0.5))
Y_w = Y@(fractional_matrix_power(c_yy, -0.5))
u_t, s, v_t_t = np.linalg.svd(fractional_matrix_power(c_xx,
 -0.5)@c_xy@fractional_matrix_power(c_yy, -0.5))
v_t = v_t_t.T
u = fractional_matrix_power(c_xx, -0.5)@u_t
v = fractional_matrix_power(c_yy, -0.5)@v_t
u_t1 = u_t[:,0]
u_t2 = u_t[:,1]
v_t1 = v_t[:,0]
v_t2 = v_t[:,1]
u1 = u[:,0]
u2 = u[:,1]
v1 = v[:,0]
v2 = v[:,1]
#Original X
plt.figure(figsize=(6,6))
plt.scatter(X[:,0], X[:,1], c=(X@(u1.T)).T)
plt.xlim(-10, 10)
plt.ylim(-10, 10)
plt.grid()
plt.show()
#Original Y
plt.figure(figsize=(6,6))
plt.scatter(Y[:,0], Y[:,1], c=(Y@(v1.T)).T)
plt.xlim(-10, 10)
plt.ylim(-10, 10)
```

```
plt.grid()
plt.show()
#Whitened X
plt.figure(figsize=(6,6))
plt.scatter(X_w[:,0], X_w[:,1], c=(X_w@(u_t1.T)).T)
plt.xlim(-0.3, 0.3)
plt.ylim(-0.3, 0.3)
plt.grid()
plt.show()
#Whitened Y
plt.figure(figsize=(6,6))
plt.scatter(Y_w[:,0], Y_w[:,1], c=(Y_w@(v_t1.T)).T)
plt.xlim(-0.3, 0.3)
plt.ylim(-0.3, 0.3)
plt.grid()
plt.show()
#CCA on whitened X
plt.figure(figsize=(6,6))
plt.scatter(X_w[:,0], X_w[:,1], c=(X_w@(u_t1.T)).T)
plt.xlim(-0.3, 0.3)
plt.ylim(-0.3, 0.3)
plt.grid()
plt.arrow(-u_t1[0]*0.2,-u_t1[1]*0.2,u_t1[0]*0.4,u_t1[1]*0.4, width=0.005,
 color='g')
plt.arrow(-u_t2[0]*0.2,-u_t2[1]*0.2,u_t2[0]*0.4,u_t2[1]*0.4, width=0.001,
 color='g')
plt.show()
#CCA on whitened Y
plt.figure(figsize=(6,6))
plt.scatter(Y_w[:,0], Y_w[:,1], c=(Y_w@(v_t1.T)).T)
plt.xlim(-0.3, 0.3)
plt.ylim(-0.3, 0.3)
plt.grid()
plt.arrow(-v_t1[0]*0.2,-v_t1[1]*0.2,v_t1[0]*0.4,v_t1[1]*0.4, width=0.005,
 color='g')
plt.arrow(-v_t2[0]*0.2,-v_t2[1]*0.2,v_t2[0]*0.4,v_t2[1]*0.4, width=0.001,
 color='g')
plt.show()
#CCA on X
plt.figure(figsize=(6,6))
plt.scatter(X[:,0], X[:,1], c=(X@(u1.T)).T)
plt.xlim(-10, 10)
plt.ylim(-10, 10)
plt.grid()
plt.arrow(-u1[0]*150,-u1[1]*150,u1[0]*300,u1[1]*300, width=0.1, color='g')
plt.arrow(-u2[0]*80,-u2[1]*80,u2[0]*160,u2[1]*160, width=0.05, color='g')
plt.show()
#CCA on Y
plt.figure(figsize=(6,6))
plt.scatter(Y[:,0], Y[:,1], c=(Y@(v1.T)).T)
plt.xlim(-10, 10)
plt.ylim(-10, 10)
plt.grid()
plt.arrow(-v1[0]*150,-v1[1]*150,v1[0]*300,v1[1]*300, width=0.1, color='g')
plt.arrow(-v2[0]*160,-v2[1]*160,v2[0]*320,v2[1]*320, width=0.05, color='g')
```

```
plt.show()
```