Activity 7

Muhammad Meesum Ali Qazalbash (mq06861)

December 14, 2023

```python
import numpy as np
import math

from matplotlib import pyplot as plt

X = np.array([-1.0, -0.75, -0.50, -0.25, 0.0, 0.25, 0.50, 0.75, 1.0])
Y = np.array([0.97, 1.02, 0.61, 0.63, 0.57, 0.51, 0.44, 0.14, -0.19])

n = len(X)

sum_XY = sum(X * Y)
sum_X = sum(X)
sum_Y = sum(Y)
sum_X2 = sum(np.power(X, 2))

avg_X = sum_X / n
avg_Y = sum_Y / n

Err_X = X - avg_X
Err_Y = Y - avg_Y

sum_err_XY = sum(Err_X * Err_Y)
sum_err_X2 = sum(np.power(Err_X, 2))
sum_err_Y2 = sum(np.power(Err_Y, 2))

# Ordinary Least Square(s)

OLS_b1 = (n * sum_XY - sum_X * sum_Y) / (n * sum_X2 - sum_X**2)
OLS_b0 = avg_Y - OLS_b1 * avg_X

# Principal Component Analysis

Theta_Hat = 0.5 * math.atan((2 * sum_err_XY) / (sum_err_X2 - sum_err_Y2))
PCA_b1 = math.tan(Theta_Hat)
PCA_b0 = avg_Y - PCA_b1 * avg_X

# Plot

plt.scatter(X, Y, color='red', marker='o', label='Data')
plt.plot(X, OLS_b0 + OLS_b1 * X, 'b-', label='OLS')
plt.plot(X, PCA_b0 + PCA_b1 * X, 'g-', label='PCA')
plt.legend()
plt.tight_layout()
plt.savefig('test.png')
plt.show()
```

Listing 1: Code for computing and plotting OCS and PCA lines

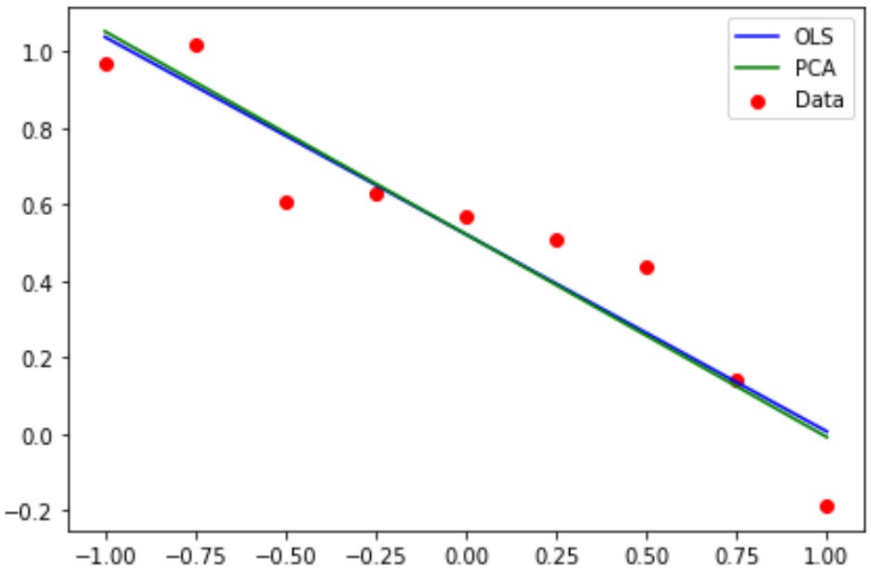

Figure 1: Plot of data points, OCS, and PCA lines

We observe that both the methods, PCA and OLS fit the given data to approximately equal lines. However, we consider the line obtained from PCA as more accurate as it is used to approximate θ, contrary to OLS which estimates $\tan\theta$ instead.