Lev Golod, 3032125890, CS 289, Hw 3, Question 3

Question 3, Part A Mean vector:

```
[ 3.13707195  5.77974369]
```

Question 3, Part B Cov Matrix

```
[[ 9.06433217  4.89854908]
 [ 4.89854908  6.00322331]]
```

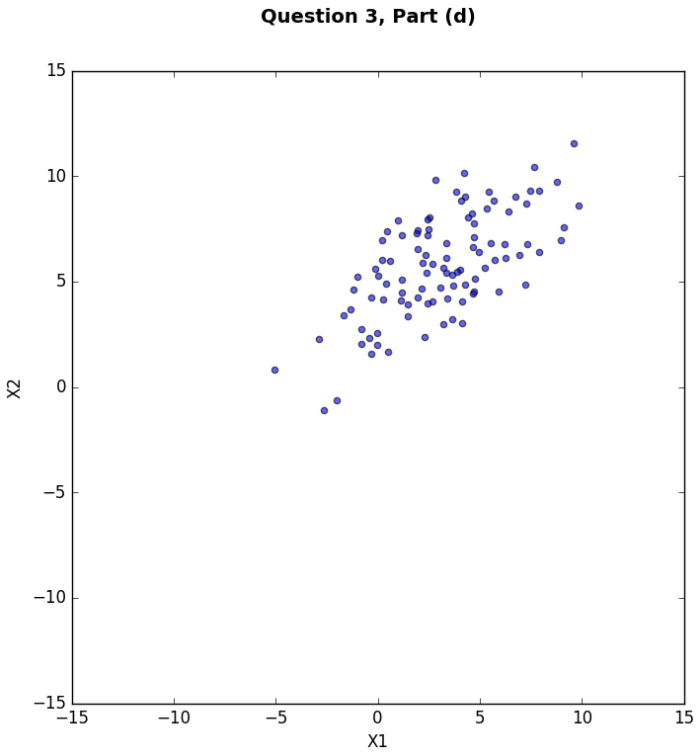
Question 3, Part C Spectral Decomp

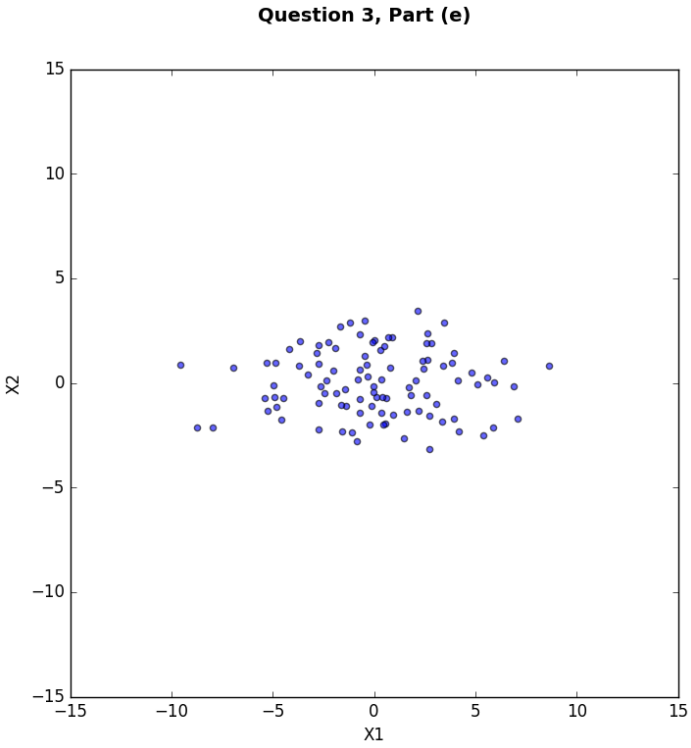
Eigenvalues

```
[ 12.66587087   2.40168461]
```

Eigenvectors

```
[[ 0.80567736 -0.59235462]
 [ 0.59235462  0.80567736]]
```





APENDIX: CODE

```
import sys
import os

dir1="/Users/levgolod/GoogleDrive/Berkeley/02 Spring Semester/"
dir2="COMPSCI 289 - Machine Learning/homework/hw03/q3"
os.chdir(dir1+dir2)
```

Lev Golod, 3032125890, CS 289, Hw 3, Question 3

```
import numpy as np
import scipy
import math

import matplotlib
import matplotlib.pyplot as plt
# import plotly.plotly as py

# import matplotlib.cm as cm
# import matplotlib.mlab as mlab


#### Preliminaries
np.random.seed(10101020)
N = 100
X1 = np.random.normal(loc=3,scale=math.pow(9,0.5),size = N)
X2 = 0.5*X1 + np.random.normal(loc=4,scale=math.pow(4,0.5),size = N)
X = np.stack((X1,X2))
X = np.transpose(X)


#### Part A - Compute mean
mean = np.mean(X, axis = 0)
print(mean)


#### Part B - Compute Cov Matrix
cov = np.cov(X, rowvar=False)
print(cov)


#### Part C - Spectral Decomp of Cov Matrix
decomp = np.linalg.eig(cov)
evals = decomp[0]
evecs = decomp[1]
print(evals)
print(evecs)

# Check that we did the decomp correctly
np.dot(cov, evecs[:,:1])
evecs[:,:1] * evals[0]
np.allclose(np.dot(cov, evecs[:,:1]).flatten(),
 evecs[:,:1].flatten() * evals[0])
np.allclose(np.dot(cov, evecs[:,1:]).flatten(),
 evecs[:,1:].flatten() * evals[1])
# (np.absolute(np.dot(cov, evecs[:,:1]).flatten() -
#  evecs[:,:1].flatten() * evals[0]) < 1e-10)
```

Lev Golod, 3032125890, CS 289, Hw 3, Question 3

```
#### Part D - Plot
arrow1 = (evecs[:,:1] * evals[0]).flatten()
arrow2 = (evecs[:,1:] * evals[1]).flatten()

fig = plt.figure(figsize=(8,8))
fig.suptitle('Question 3, Part (d)', fontsize=14, fontweight='bold')
ax = fig.add_subplot(111)
# fig.subplots_adjust(top=0.85)
ax.set_xlabel('X1')
ax.set_ylabel('X2')
ax.axis([-15, 15, -15, 15])
ax.scatter(X1,X2,alpha=0.6)
ax.arrow(mean[0], mean[1], arrow1[0], arrow1[1],
 head_width=0.5, head_length=0.5, fc='k', ec='k')
ax.arrow(mean[0], mean[1], arrow2[0], arrow2[1],
 head_width=0.5, head_length=0.5, fc='k', ec='k')
# plt.show()
plt.savefig('./plot/q3_d.png', bbox_inches='tight')


#### Part E - Center, rotate, plot again
mean.shape = (1, mean.shape[0])
X = X - mean
# np.mean(X, axis = 0)

# X[:5,:]
# X[k:k+1,:]
# np.transpose(np.dot( np.transpose(evecs), np.transpose(X[k:k+1,:]) ))

Xr = np.zeros(X.shape)
for k in range(X.shape[0]):
    Xr[k:k+1,:] = np.transpose(np.dot( np.transpose(evecs),
     np.transpose(X[k:k+1,:]) ))

fig = plt.figure(figsize=(8,8))
fig.suptitle('Question 3, Part (e)', fontsize=14, fontweight='bold')
ax = fig.add_subplot(111)
# fig.subplots_adjust(top=0.85)
ax.set_xlabel('X1')
ax.set_ylabel('X2')
ax.axis([-15, 15, -15, 15])
ax.scatter(Xr[:,:1],Xr[:,1:],alpha=0.6)
# plt.show()
plt.savefig('./plot/q3_e.png', bbox_inches='tight')
```