MA201_ASSIGNMENT 5

Name: Dipean Dasgupta Date:3/11/2022

STD ID: 202151188

1. Three different Gaussian random variables, i.e., $X1, X2$ and $X3$ with 0 mean and 1 variance., Compute the covariance matrix of $X1$, $X2$ and $X3$.
   Covariance matrix (CV) $CV = \begin{bmatrix} cov(X1,X1) & cov(X1,X2) & cov(X1,X3) \\ cov(X2,X1) & cov(X2,X2) & cov(X2,X3) \\ cov(X3,X1) & cov(X3,X2) & cov(X3,X3) \end{bmatrix}$. Here, $cov(X,Y) = E[XY] - E[X]E[Y]$. Similarly compute correlational matrix.
2. Verify the properties of the covariance matrix.
   1. Symmetric,i.e.,$C_X = C_X^T$.
   2. its eigenvalues are greater than equal to zero
   3. It is positive semi-definite, i.e., for any real valued vector a,$a^T C_X a \geq 0$
3. Generate covariance matrix of correlated data. Take face images as the data. }Show that data and noise are uncorrelated. Take Image files as your data and standard gaussian noise. (Face data is attached in zip file)

Solutions:

```
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm
import matplotlib.image as mpimg
```

Three different Gaussian random variables, i.e., X1, X2 and X3 with 0 mean and 1 variance.

```
M = 2000
X1 =np.random.normal(0,1,M)
X2 =np.random.normal(100,1,M)
X3 = np.random.normal(10,1,M)
X = np.transpose(np.reshape(np.array([X1,X2,X3]),[3,M]))
```

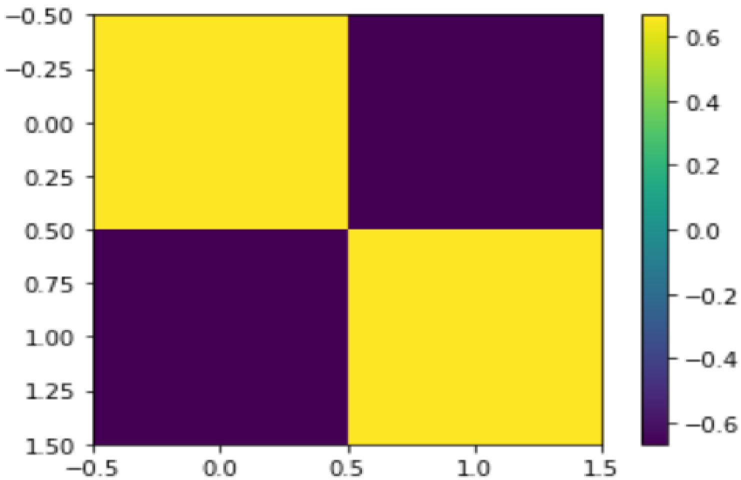
Compute the covariance matrix of $X1$, $X2$ and $X3$. Covariance matrix (CV)

```
def covmat(data_mat):
    [m,n]=np.shape(data_mat)
    CV = np.zeros(shape=(n,n))
    for i in range(n):
        for j in range(n):
            CV[i,j]= np.mean(np.multiply(X[:,i],X[:,j]))-
np.multiply(np.mean(X[:,i]),np.mean(X[:,j]))
    return CV
```

```
#X = np.array([[2.5,2.4],[.5,.7],[2.2,2.9],[1.9,2.2],[3.1,3]])
X = np.array([[0,2],[1,1],[2,0]])

CV = covmat(X)
print(CV)
fig=plt.figure()
plt.imshow(CV)
plt.colorbar()
```

```
[[ 0.66666667 -0.66666667]
 [-0.66666667  0.66666667]]
<matplotlib.colorbar.Colorbar at 0x7f7d35a876d0>
```


```
[[ 0.66666667 -0.66666667]
 [-0.66666667  0.66666667]]
<matplotlib.colorbar.Colorbar at 0x7f7d35a876d0>
```

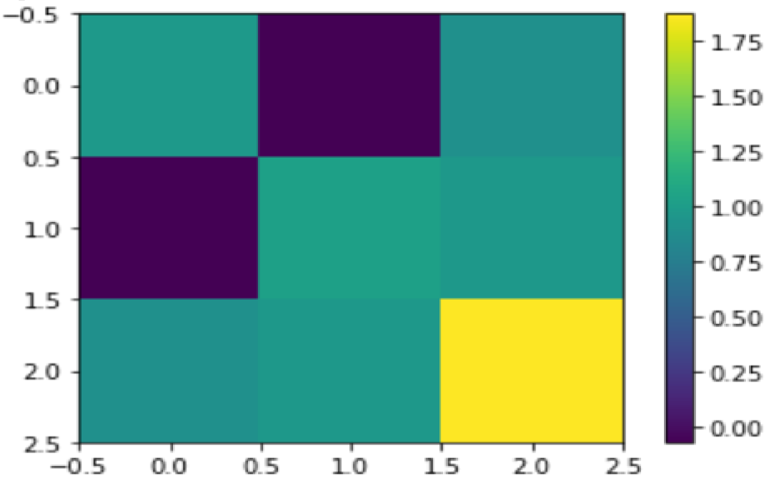

```
np.cov(X.T,bias=True)
array([[ 0.66666667, -0.66666667], [-0.66666667, 0.66666667]])

np.var(np.array([0,1,2]))
0.6666666666666666
```

Generating three random variables as $X1$, $X2$ and $X1+X2$. Compute the covariance matrix

```
X1 =np.random.normal(0,1,M)
X2 =np.random.normal(0,1,M)
X3 = X1+X2
X = np.transpose(np.reshape(np.array([X1,X2,X3]),[3,M]))
CV = covmat(X)
print(CV)
fig=plt.figure()
plt.imshow(CV)
```

```
plt.colorbar()
```

```
[[ 0.97630072 -0.06816225  0.90813847]
 [-0.06816225  1.03335961  0.96519736]
 [ 0.90813847  0.96519736  1.87333583]]
<matplotlib.colorbar.Colorbar at 0x7fdc7dbed6d8>
```

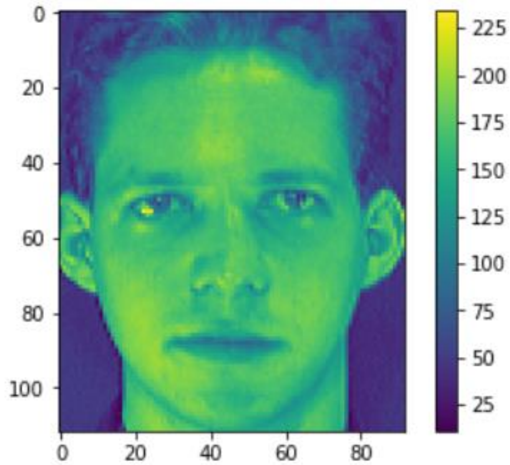


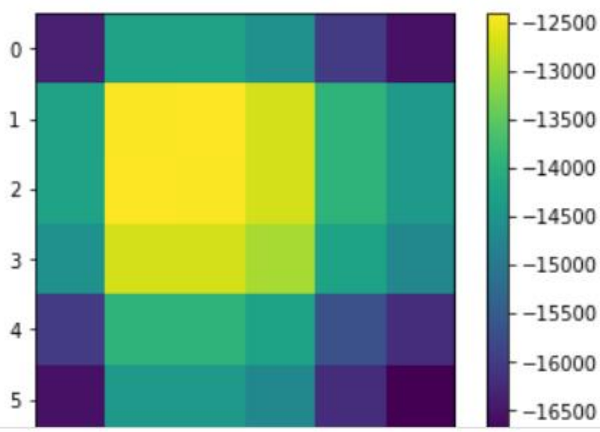
Generating covariance matrix of correlated data. Taking face images as the data.

```
X1 = mpimg.imread('f1.pgm')
plt.imshow(X1)
plt.colorbar()
[m,n]=np.shape(X1)
X1 = X1.flatten()
X2 = mpimg.imread('f2.pgm').flatten()
X3 = mpimg.imread('f3.pgm').flatten()
X4 = mpimg.imread('f4.pgm').flatten()
X5 = mpimg.imread('f5.pgm').flatten()
X6 = mpimg.imread('f6.pgm').flatten()

X = np.transpose(np.reshape(np.array([X1,X2,X3,X4,X5,X6]),[6,m*n]))
CV = covmat(X)
print(CV)
fig=plt.figure()
plt.imshow(CV)
plt.colorbar()
```

```
[[-16354.60012522 -14246.51604614 -14245.30647319 -14557.76122766
  -15987.5130235  -16533.85277879]
 [-14246.51604614 -12438.46267815 -12414.67478852 -12688.2016277
  -13937.5356198  -14412.66278925]
 [-14245.30647319 -12414.67478852 -12432.98480126 -12686.65473454
  -13933.34908939 -14409.73408491]
 [-14557.76122766 -12688.2016277  -12686.65473454 -12986.96458123
  -14241.61305505 -14728.08677508]
```

```
 [-15987.5130235  -13937.5356198  -13933.34908939 -14241.61305505
  -15669.1158713  -16174.92234759]
 [-16533.85277879 -14412.66278925 -14409.73408491 -14728.08677508
  -16174.92234759 -16748.9134148 ]]
<matplotlib.colorbar.Colorbar at 0x7fdc7da78ef0>
```

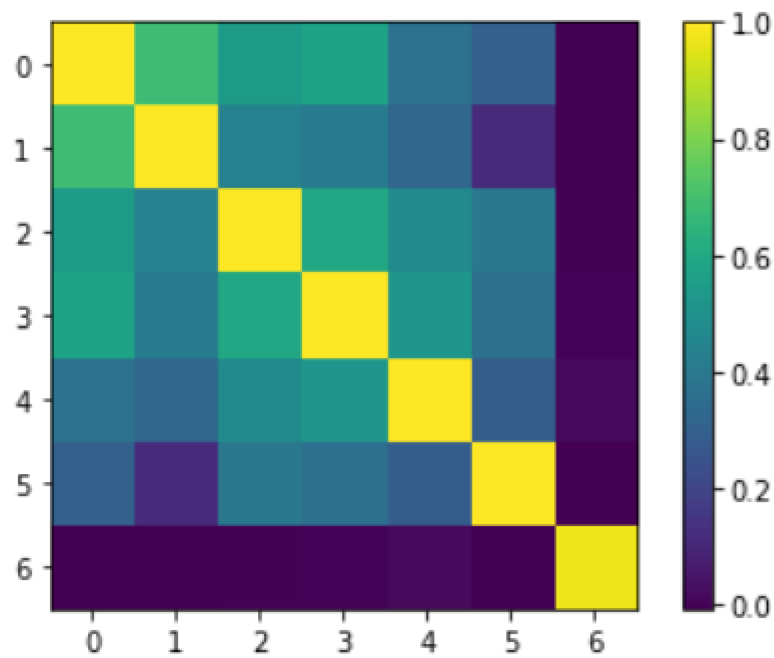




Showing that data and noise are uncorrelated.

```
X1 = (X1-np.mean(X1))/np.std(X1)
X2 = (X2-np.mean(X2))/np.std(X2)
X3 = (X3-np.mean(X3))/np.std(X3)
X4 = (X4-np.mean(X4))/np.std(X4)
X5 = (X5-np.mean(X5))/np.std(X5)
X6 = (X6-np.mean(X6))/np.std(X6)
m2=np.size(X1)

XN = np.random.normal(0,1,m2)
X = np.transpose(np.reshape(np.array([X1,X2,X3,X4,X5,X6,XN]),[7,m*n]))
CV = covmat(X)
print(CV)
fig=plt.figure()
```

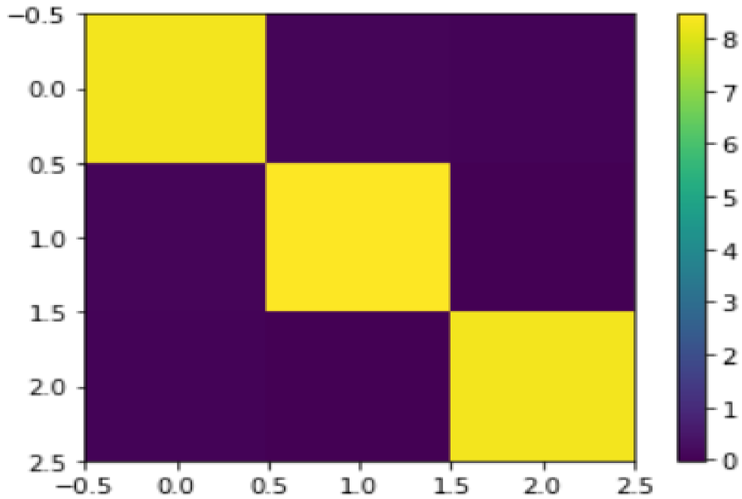
```
plt.imshow(CV)
plt.colorbar()
[[ 1.00000000e+00  6.86364063e-01  5.43393694e-01  5.76192802e-01
   3.72836320e-01  3.00766740e-01 -6.02690863e-03]
 [ 6.86364063e-01  1.00000000e+00  4.36424961e-01  4.08870995e-01
   3.26988068e-01  1.18513726e-01 -4.89635869e-03]
 [ 5.43393694e-01  4.36424961e-01  1.00000000e+00  5.98566612e-01
   4.73639113e-01  3.97086330e-01  4.08574266e-04]
 [ 5.76192802e-01  4.08870995e-01  5.98566612e-01  1.00000000e+00
   5.21458256e-01  3.60153830e-01  2.77972855e-03]
 [ 3.72836320e-01  3.26988068e-01  4.73639113e-01  5.21458256e-01
   1.00000000e+00  2.89964477e-01  2.11497280e-02]
 [ 3.00766740e-01  1.18513726e-01  3.97086330e-01  3.60153830e-01
   2.89964477e-01  1.00000000e+00 -5.08956923e-03]
 [-6.02690863e-03 -4.89635869e-03  4.08574266e-04  2.77972855e-03
   2.11497280e-02 -5.08956923e-03  9.76184233e-01]]
<matplotlib.colorbar.Colorbar at 0x7fdc7d9c7898>
```



Considering $X1$,$X2$ and $X3$ uniform random variables:

```
M = 2000
X1 =np.random.uniform(0,10,M)
X2 =np.random.uniform(5,15,M)
X3 = np.random.uniform(10,20,M)
X = np.transpose(np.reshape(np.array([X1,X2,X3]),[3,M]))
CV = covmat(X)
print(CV)
fig=plt.figure()
plt.imshow(CV)
```

```
plt.colorbar()

[[ 8.30369847  0.09571767  0.06347771]
 [ 0.09571767  8.45631545 -0.01710792]
 [ 0.06347771 -0.01710792  8.29650242]]
<matplotlib.colorbar.Colorbar at 0x7fdc7d95fd30>
```



Verifying the properties of the covariance matrix.

a) Symmetric,i.e.,$C_X = C_X^T$.

```
print(CV- np.transpose(CV))
output:
      [[0. 0. 0.]
      [0. 0. 0.]
      [0. 0. 0.]]
```

b) Its eigenvalues are greater than equal to zero

```
from numpy import linalg as LA
[E,U]=LA.eig(CV)
print(U)
output:
[[-0.44813336 -0.73264155  0.51226249]
 [-0.89168562  0.32541693 -0.31464359]
 [-0.06382208  0.59777938  0.79911611]]
```

c) It is positive semi-definite, i.e., for any real valued vector a, $a^T C_X a \geq 0$

```
a = np.random.rand(np.size(CV,0),1)
print(np.matmul(np.matmul(np.transpose(a),CV),a))

output:
[[2.6879199]]
```

5. Compute correlation coefficient matrix.

```
def corr(data_mat):
    [m,n]=np.shape(data_mat)
    X = data_mat-np.mean(data_mat,axis=0)
    CV = covmat(X)
    CR = CV/np.prod(np.std(data_mat,axis=0))
    return CR
CR = corr(X)
print(CR)
fig=plt.figure()
plt.imshow(CR)
plt.colorbar()
```

```
[[ 0.34403108  0.00396569  0.00262995]
 [ 0.00396569  0.35035416 -0.0007088 ]
 [ 0.00262995 -0.0007088   0.34373294]]
<matplotlib.colorbar.Colorbar at 0x7fdc7d8825c0>
```