Mohanty_R_HW1_Prob2_2

February 20, 2020

1 Math 521 HW1

1.1 Computing Question 2

Import required python packages

[69]:
```
import scipy as spy
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
from scipy.io import loadmat
import math
import pandas as pd
```

Load face and check dimensions

[10]:
```
face1 = loadmat('./HW1data/face1.mat')
face1=face1['face1']
```

[11]:
```
face2 = loadmat('./HW1data/face2.mat')
face2=face2['face2']
```

[12]:
```
face1.shape,face2.shape
```

[12]:
```
((22080, 21), (22080, 21))
```

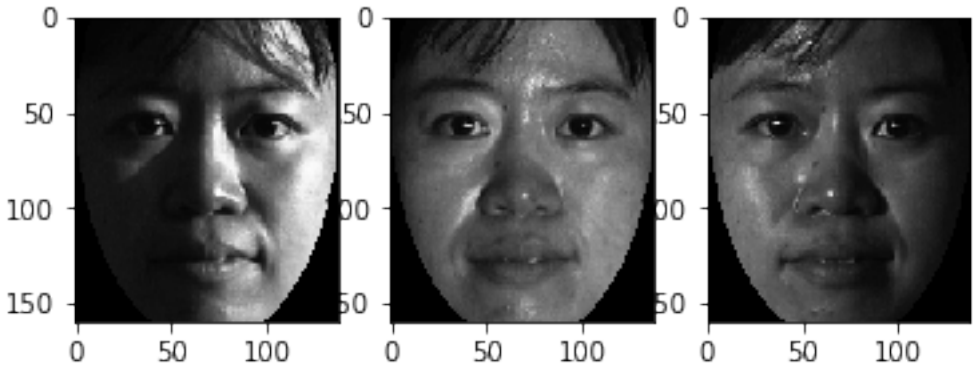
[13]:
```
160*138
```

[13]:
```
22080
```

showing some sample images from face1 data

[14]:
```
fig, ax = plt.subplots(1, 3)
ax[0].imshow(face1[:,0].reshape(138,160).T,cmap='gray')
ax[1].imshow(face1[:,5].reshape(138,160).T,cmap='gray')
ax[2].imshow(face1[:,20].reshape(138,160).T,cmap='gray')
```

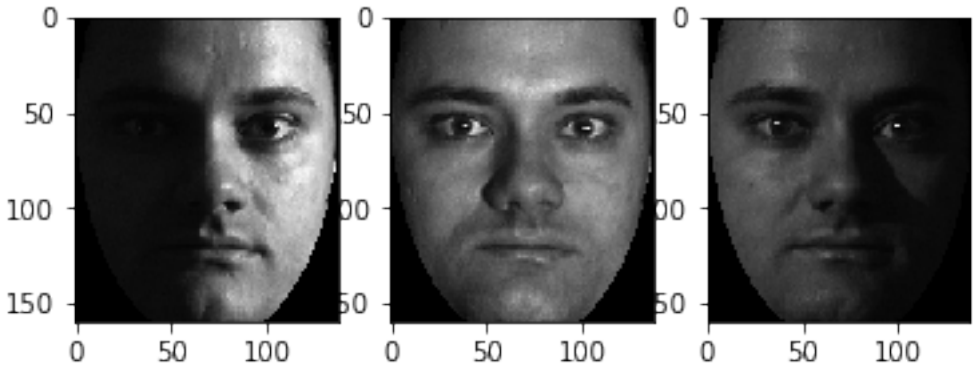
[14]:
```
<matplotlib.image.AxesImage at 0xb15bd1b00>
```



showing some sample images from face1 data

[15]:
```
fig, ax = plt.subplots(1, 3)
ax[0].imshow(face2[:,0].reshape(138,160).T,cmap='gray')
ax[1].imshow(face2[:,5].reshape(138,160).T,cmap='gray')
ax[2].imshow(face2[:,20].reshape(138,160).T,cmap='gray')
```

[15]:
```
<matplotlib.image.AxesImage at 0xb15ce6fd0>
```



Find orthnormal matrices from face1 and face2 martices

[16]:
```
Qx=spy.linalg.orth(face1)
```

[17]:
```
Qy=spy.linalg.orth(face2)
```

[18]:
```
Qx.shape
```

[18]:
```
(22080, 21)
```

[19]:
```
face1.shape,Qx.shape
```

[19]:
```
((22080, 21), (22080, 21))
```

[20]:
```
face2.shape,Qy.shape
```

[20]:
```
((22080, 21), (22080, 21))
```

Verify $Q_x^T Q_x = Q_y^T Q_Y = I$

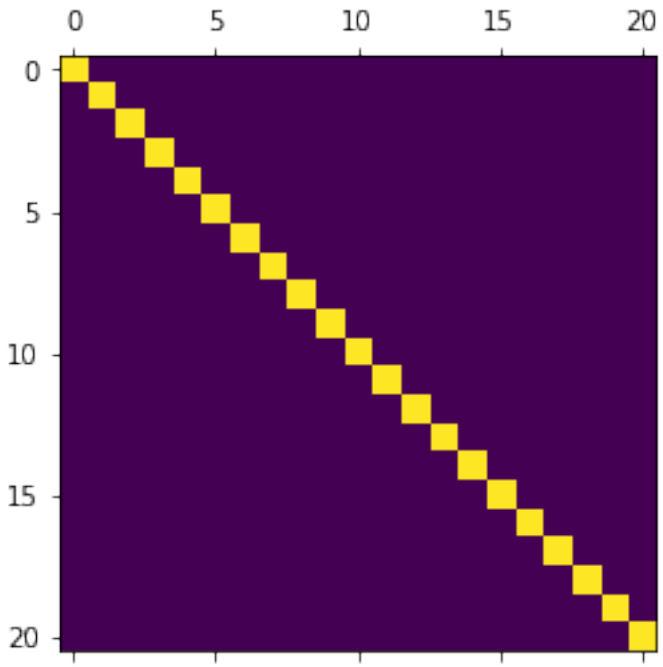
[21]:
```
Ix=np.around(np.matmul(Qx.T,Qx))
```

[22]:
```
Ix.shape
```

[22]:
```
(21, 21)
```

[23]:
```
plt.matshow(Ix)
```

[23]:
```
<matplotlib.image.AxesImage at 0xb15d65c18>
```



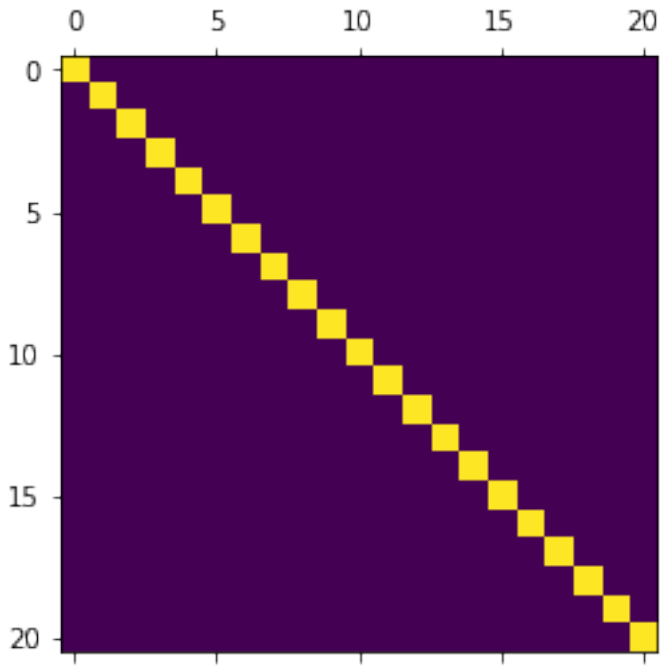
[24]:
```
Iy=np.around(np.matmul(Qy.T,Qy))
```

[25]:
```
Iy.shape
```

[25]:
```
(21, 21)
```

[26]:
```
plt.matshow(Iy)
```

[26]:
```
<matplotlib.image.AxesImage at 0xb19c68eb8>
```



run SVD on $Q_x^T Q_y$

[27]:
```
H,S,Zt=np.linalg.svd(np.matmul(Qx.T,Qy))
```

[28]:
```
H.shape,S.shape,Zt.shape
```

[28]:
```
((21, 21), (21,), (21, 21))
```

[29]:
```
np.linalg.matrix_rank(Qx)
```

[29]:
```
21
```

[30]:
```
np.linalg.matrix_rank(Qy)
```

[30]:
```
21
```

Calculate Y to determine principal angle from *Sin*

[47]:
```
if np.linalg.matrix_rank(Qx)>=np.linalg.matrix_rank(Qy):
    Y=Qy - np.matmul(Qx,np.matmul(Qx.T,Qy))
else:
    Y=Qx - np.matmul(Qy,np.matmul(Qy.T,Qx))
```

[48]:
```
Y.shape
```

[48]:
```
(22080, 21)
```

Run SVD on Y to compute *sin* values of principal angles

[49]:
```
Ssine=np.linalg.svd(Y,compute_uv=False)
```

flip the array to match with corresponding *cos* values

[56]:
```
Ssine_flip=np.flip(Ssine)
```

Compute the principal angles.For very small values of principal angles use *sin*

[76]:
```
principal_angles_deg=[]
principal_angles_rad=[]
for i in range(len(S)):
    if S[i]**2<0.5:
        theta_rad=math.acos(S[i])
        theta=math.degrees(math.acos(S[i]))
    elif Ssine_flip[i]**2<=0.5:
        theta_rad=math.asin(Ssine_flip[i])
        theta=math.degrees(math.asin(Ssine_flip[i]))
    principal_angles_deg.append(theta)
    principal_angles_rad.append(theta_rad)
```

Print the principal angles in degrees and radians

[77]:
```
print("Principal Angles between Face1 and Face 2")
pd.DataFrame({'Principal Angles (in Degrees)': principal_angles_deg, 'Principal␣
 ↪Angles (in Radians)': principal_angles_rad})
```

```
Principal Angles between Face1 and Face 2
```

[77]:

	Principal Angles (in Degrees)	Principal Angles (in Radians)
0	17.339911	0.302639
1	26.865253	0.468887
2	41.931173	0.731837
3	57.759428	1.008092
4	63.938597	1.115939
5	68.908976	1.202689
6	70.336297	1.227600
7	73.998328	1.291514
8	78.548064	1.370922
9	79.414259	1.386040
10	81.515991	1.422722
11	81.897028	1.429373
12	84.119862	1.468169
13	85.170744	1.486510
14	86.614458	1.511707
15	87.001927	1.518470
16	87.581047	1.528578
17	88.425331	1.543313

18	88.692223	1.547971
19	89.225386	1.557277
20	89.895487	1.568972

[ ]: