HOMEWORK 5

CS178

Jeremy Parnell (jmparnel)

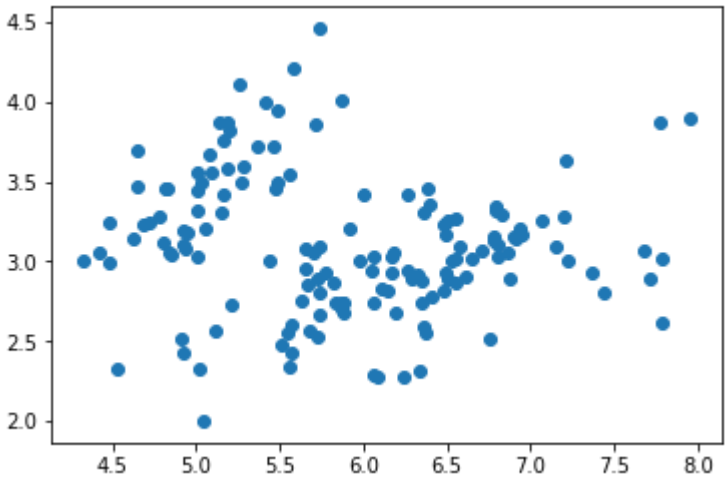
In [1]:
```
from __future__ import division # For python 2.*

import numpy as np
import matplotlib.pyplot as plt
import mltools as ml

# Importing SVD
from scipy.linalg import svd

np.random.seed(0)
%matplotlib inline
```

1 Clustering

Problem 1.1

In [2]:
```
iris = np.genfromtxt("data/iris.txt",delimiter=None)
Y = iris[:,-1]
X = iris[:,0:2]
```

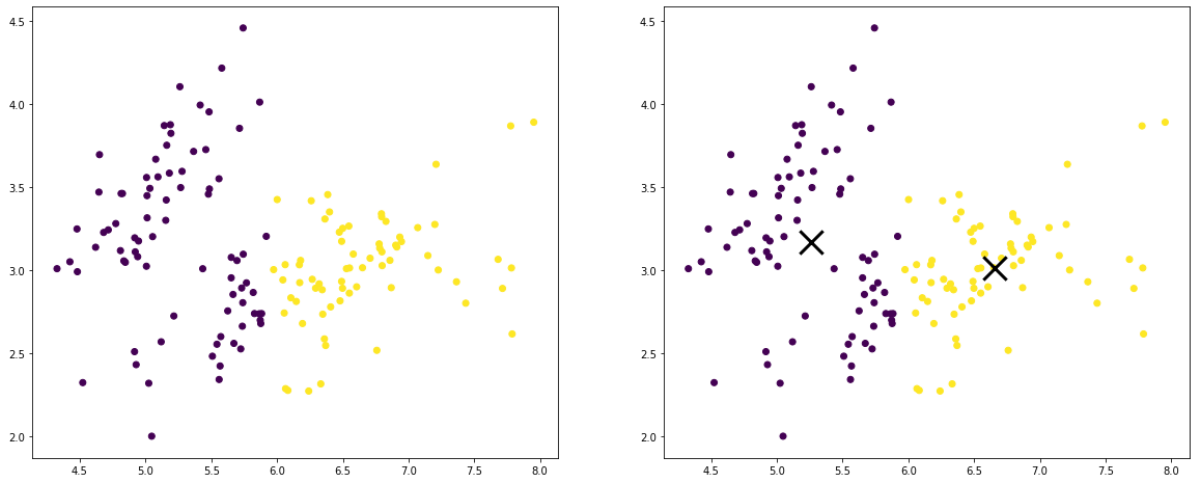
In [3]:
```
plt.scatter(iris[:, 0], iris[:, 1])
plt.show()
```



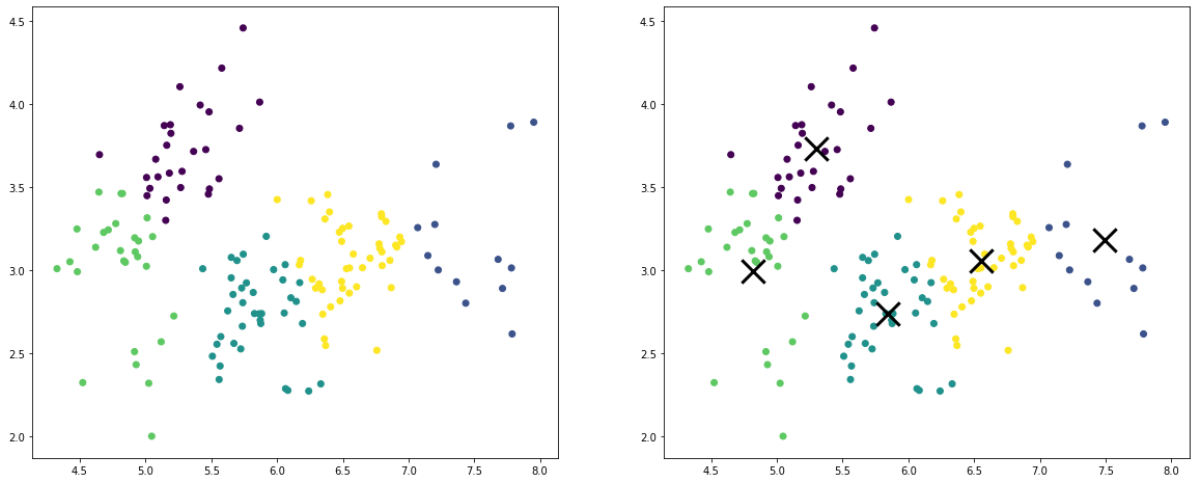
I believe there are <u>THREE</u> clusters

Problem 1.2

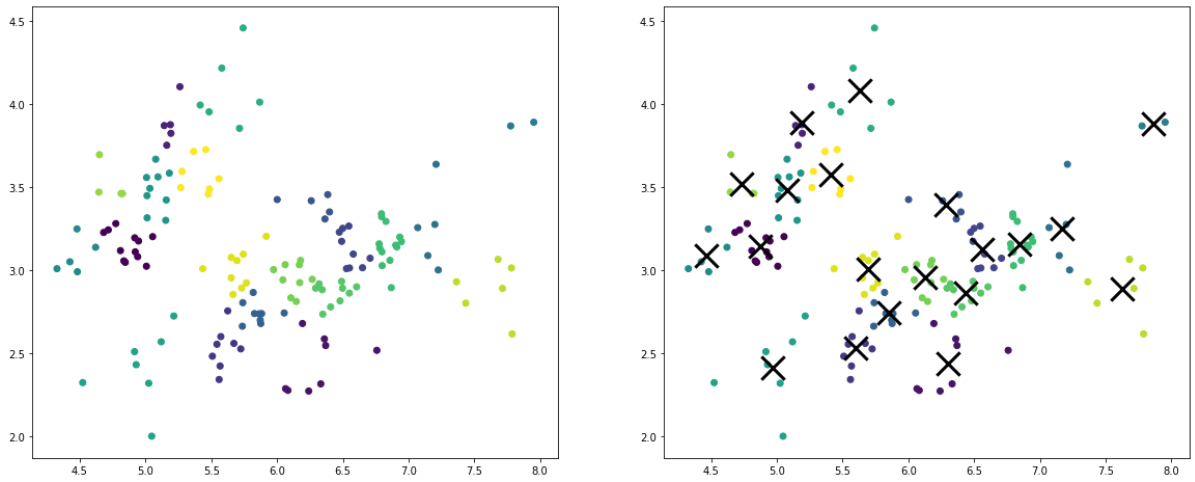
In [4]:
```
seeds = [1,5,10,25,50]
fig, ax = plt.subplots(1, 2, figsize=(20, 8))
ssd = 999

for s in seeds:
    np.random.seed(s)
    temp_Z, temp_mu, temp_ssd = ml.cluster.kmeans(X, K=2, init='k++', max_iter
=100)
    if(temp_ssd < ssd):
        Z = temp_Z
        mu = temp_mu
        ssd = temp_ssd

ax[0].scatter(iris[:, 0], iris[:, 1], c=Z)
ax[1].scatter(iris[:, 0], iris[:, 1], c=Z) # Plotting the data
ax[1].scatter(mu[:, 0], mu[:, 1], s=500, marker='x', facecolor='black', lw=3)
plt.show()
```



In [5]:
```
seeds = [1,5,10,25,50]
fig, ax = plt.subplots(1, 2, figsize=(20, 8))
ssd = 999

for s in seeds:
    np.random.seed(s)
    temp_Z, temp_mu, temp_ssd = ml.cluster.kmeans(X, K=5, init='k++', max_iter
=100)
    if(temp_ssd < ssd):
        Z = temp_Z
        mu = temp_mu
        ssd = temp_ssd

ax[0].scatter(iris[:, 0], iris[:, 1], c=Z)
ax[1].scatter(iris[:, 0], iris[:, 1], c=Z) # Plotting the data
ax[1].scatter(mu[:, 0], mu[:, 1], s=500, marker='x', facecolor='black', lw=3)
plt.show()
```



In [6]:
```
seeds = [1,5,10,25,50]
fig, ax = plt.subplots(1, 2, figsize=(20, 8))
ssd = 999

for s in seeds:
    np.random.seed(s)
    temp_Z, temp_mu, temp_ssd = ml.cluster.kmeans(X, K=20, init='k++', max_ite
r=100)
    if(temp_ssd < ssd):
        Z = temp_Z
        mu = temp_mu
        ssd = temp_ssd

ax[0].scatter(iris[:, 0], iris[:, 1], c=Z)
ax[1].scatter(iris[:, 0], iris[:, 1], c=Z) # Plotting the data
ax[1].scatter(mu[:, 0], mu[:, 1], s=500, marker='x', facecolor='black', lw=3)
plt.show()
```

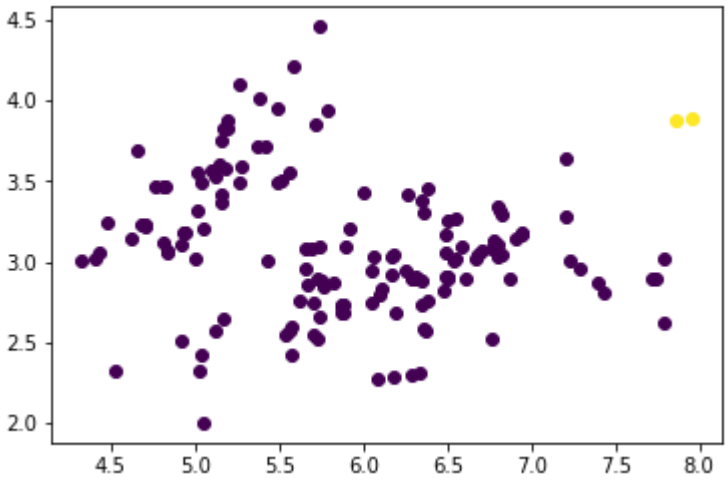


Problem 1.3

Single Linkage

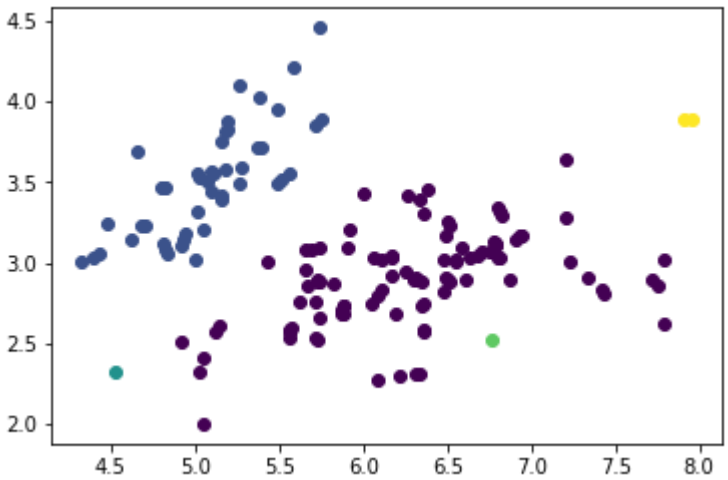
In [7]:
```
Z, _ = ml.cluster.agglomerative(X, K = 2, method = "min")
print("K = 2")
ml.plotClassify2D(None,X,Z)
```

```
K = 2
```



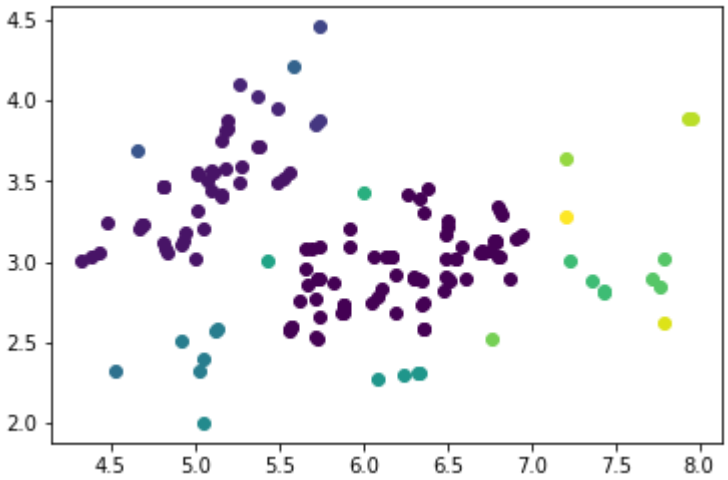
In [8]:
```
Z, _ = ml.cluster.agglomerative(X, K = 5, method = "min")
print("K = 5")
ml.plotClassify2D(None,X,Z)
```

```
K = 5
```



In [9]:
```
Z, _ = ml.cluster.agglomerative(X, K = 20, method = "min")
print("K = 20")
ml.plotClassify2D(None,X,Z)
plt.show()
```

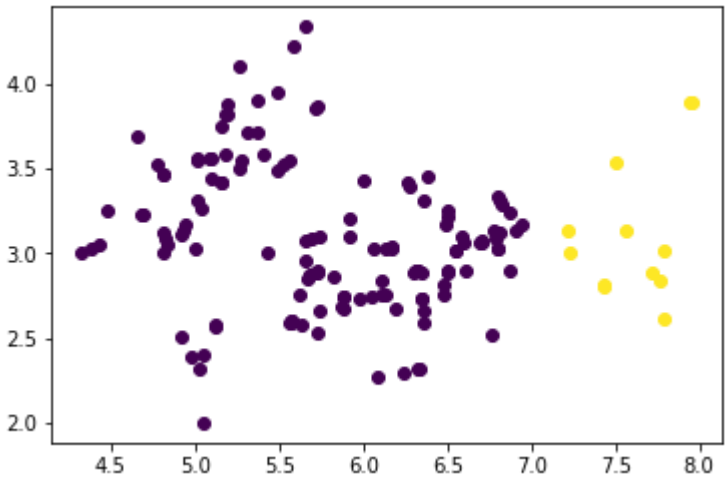
```
K = 20
```



Complete Linkage

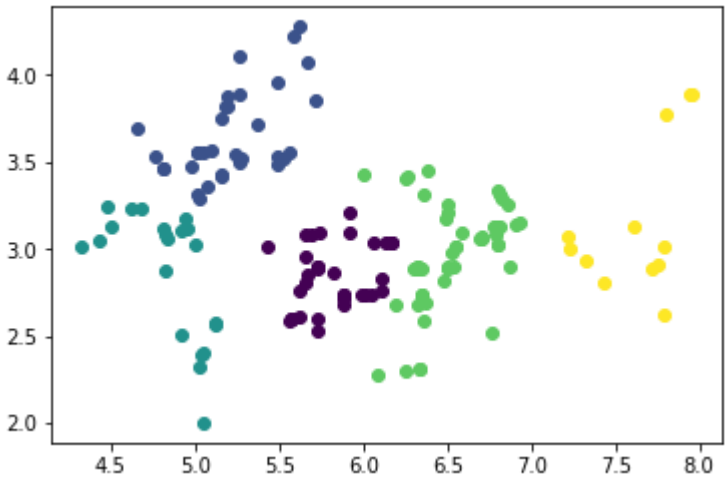
In [10]:
```
Z, _ = ml.cluster.agglomerative(X, K = 2, method = "max")
print("K = 2")
ml.plotClassify2D(None,X,Z)
```

```
K = 2
```



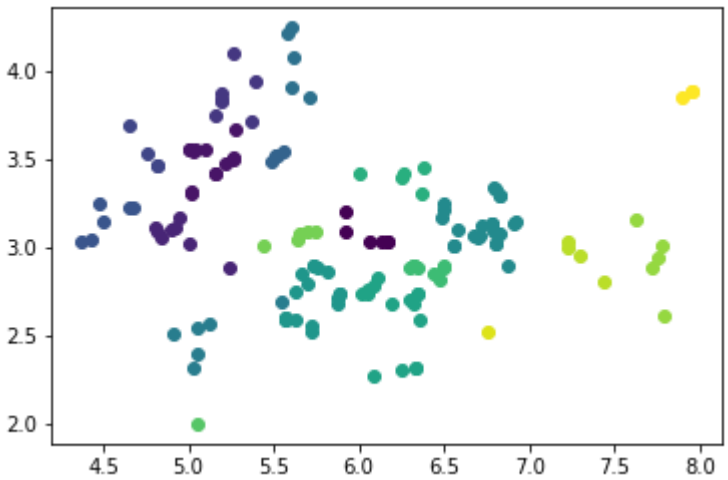
In [11]:
```
Z, _ = ml.cluster.agglomerative(X, K = 5, method = "max")
print("K = 5")
ml.plotClassify2D(None,X,Z)
```

```
K = 5
```



In [12]:
```
Z, _ = ml.cluster.agglomerative(X, K = 20, method = "max")
print("K = 20")
ml.plotClassify2D(None,X,Z)
```

```
K = 20
```



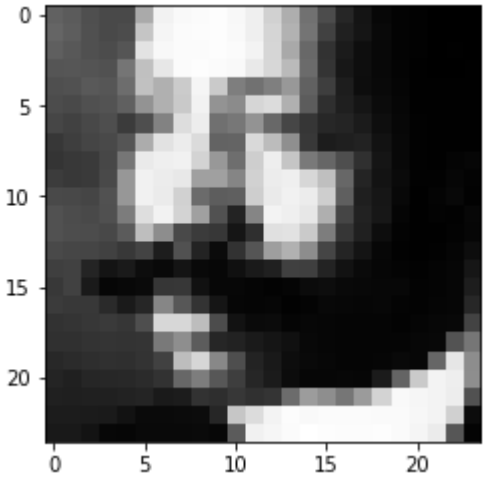
Problem 1.4

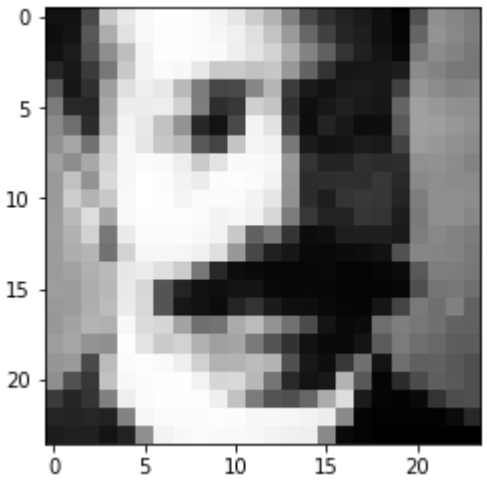
One difference I do notice with the agglomerative clustering is that the sizes of the clusters don't seem as even as with the k-means clustering. In k-means, the clusters seem to be close in relative size to one another. The sizes of the clusters seem to vary, largely, in agglomerative clustering. However, the locations of all the points seem very similar in both the k-means and agglomerative clustering - not quite the same, but relatively close. Thus, in terms of space on the graph, the locations of the clusters look similar to one another when comparing the results of both methods.

2 EigenFaces

In [13]:
```
X = np.genfromtxt("data/faces.txt", delimiter=None) # load face dataset
print("TEST")
for i in range(2):
    img = np.reshape(X[i,:],(24,24)) # convert vectorized data to 24x24 image
 patches
    plt.imshow( img.T , cmap="gray")
    plt.show()
```

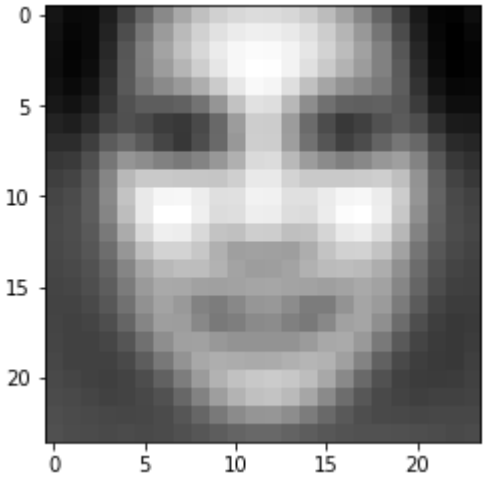
```
TEST
```





Problem 2.1

In [14]:
```
mu = np.mean(X,axis=0)
XO = X - mu
img = np.reshape(mu,(24,24)) # convert vectorized data to 24x24 image patches
plt.imshow( img.T , cmap="gray")
plt.show()
```



Problem 2.2

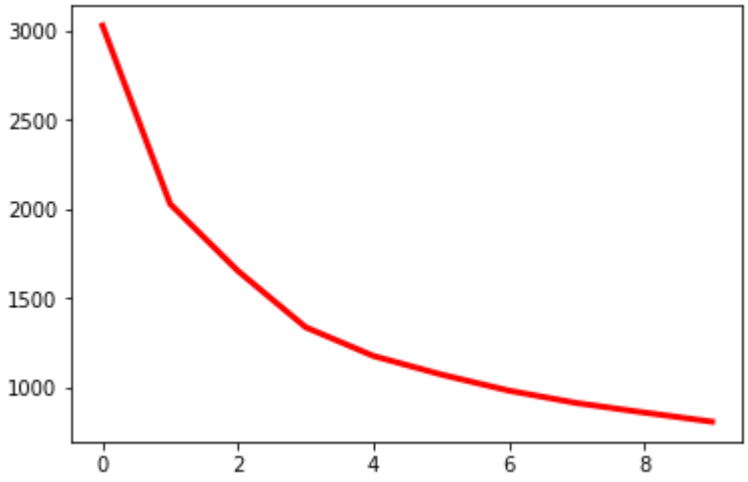
In [15]:
```
U, S, V = svd(XO, full_matrices=False)
W = U.dot( np.diag(S) )
```

In [16]:
```
print("W: " + str(W.shape))
print("V: " + str(V.shape))
```

```
W: (4916L, 576L)
V: (576L, 576L)
```

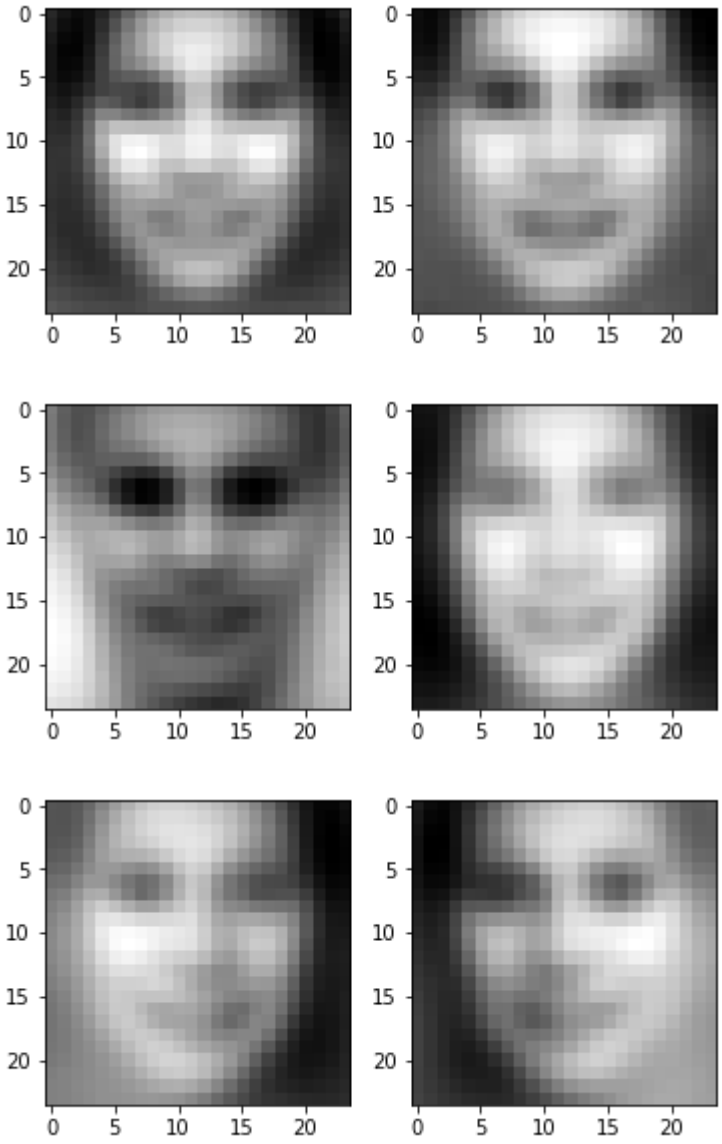
Problem 2.3

In [17]:
```
mse = np.zeros(10)
for k in range(10):
    XhatO = W[:,:k].dot(V[:k,:])
    mse[k] = np.mean((XO - XhatO)**2)
plt.plot(range(10),mse,'r-',linewidth=3)
plt.show()
```



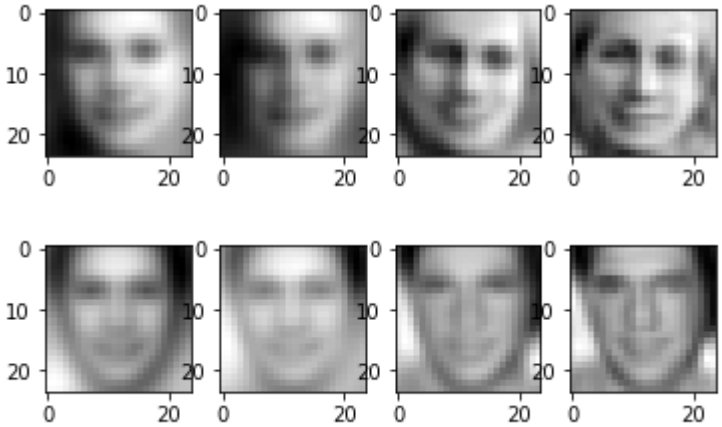
Problem 2.4

In [18]:
```
for j in range(3):
    a = 2*np.median(np.abs(W[:,j]))
    img1 = np.reshape(mu+a*V[j,:],(24,24))
    img2 = np.reshape(mu-a*V[j,:],(24,24))
    f, (ax1,ax2) = plt.subplots(1,2)
    ax1.imshow( img1.T , cmap="gray")
    ax2.imshow( img2.T , cmap="gray")
plt.show()
```



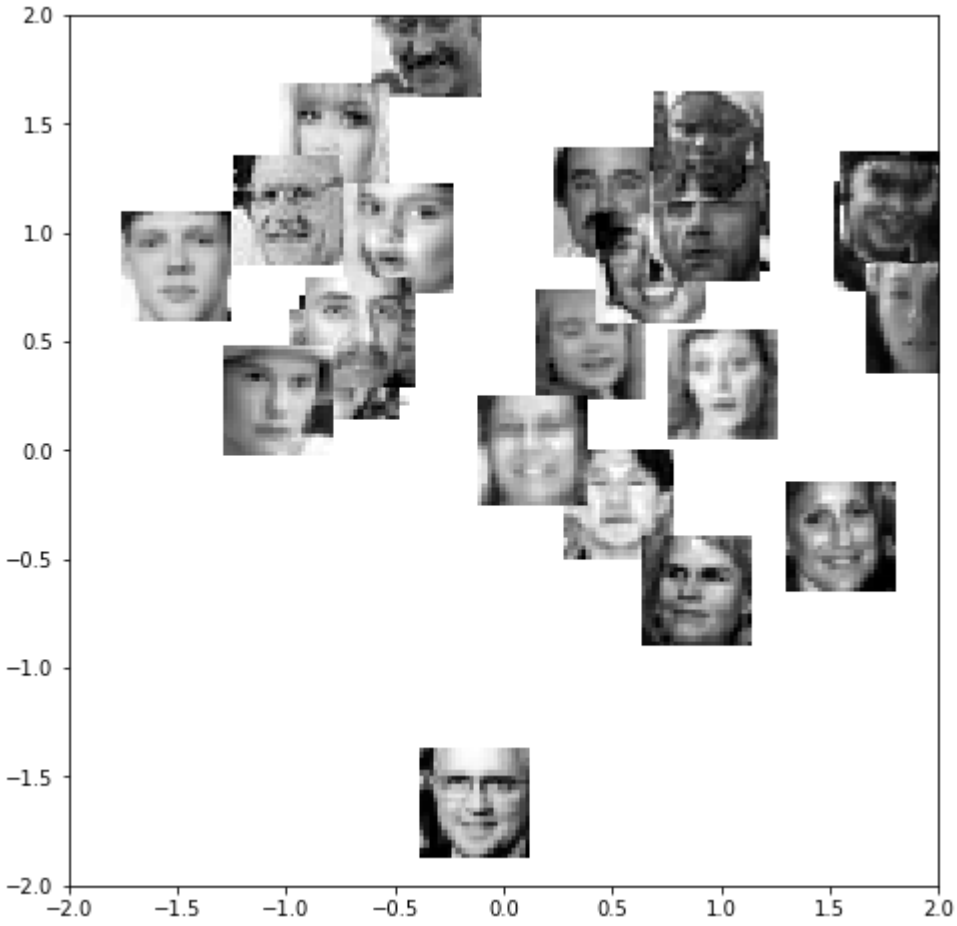
Problem 2.5

In [19]:
```
kvals = np.array([5,10,50,100])

for i in range(2):
    i = np.random.randint(X.shape[1])
    a = X[i,:]
    a = np.reshape(a,(24,24))
    f,ax = plt.subplots(1,4)
    ax[0].imshow(a.T, cmap="gray")
    for b,c in enumerate(kvals):
        a = mu + W[i,0:c].dot(V[0:c,:])
        a = np.reshape(a,(24,24))
        ax[b].imshow(a.T, cmap="gray")
plt.show()
```



Problem 2.6

In [20]:
```
import mltools.transforms
f, ax = plt.subplots(1, 1, figsize=(12, 8))
idx = np.random.choice(X.shape[0], 25, replace=False)
coord,params = ml.transforms.rescale( W[:,0:2] ) # normalize scale of "W" locations

for i in idx:
    # compute where to place image (scaled W values) & size
    loc = (coord[i,0],coord[i,0]+0.5, coord[i,1],coord[i,1]+0.5)
    img = np.reshape( X[i,:], (24,24) ) # reshape to square
    ax.imshow( img.T , cmap="gray", extent=loc)
ax.axis([-2, 2, -2, 2]) # set axis to reasonable visual scale
plt.show()
```



Statement of Collaboration

All of the work that is done and that is respresented on this homework was done by me and me alone. If I ever needed help or if I was ever stuck on a certain part of this assignment, I would look to Piazza for questions or for guidance in order to help me achieve completion. I did not work on this homework with anyone else. I did meet up with Chad Lei and Sergey Kochetov on campus to discuss about notes and about different concepts that we learned in class that pertained to the homework (such as the Eigen-Faces lecture), as well as to discuss some of the functionality of the code already-provided to us in the homework. However, no personal code was shared during our meetings. We only discussed code that was either given in the homework, or functions that were included within plyplot library that were used for plotting.