HW6_P2

November 12, 2018

In [55]:
```
import numpy as np
from matplotlib import pyplot as plt
# load the data
yalefaces = np.loadtxt('yalefaces.csv', delimiter=',')
```

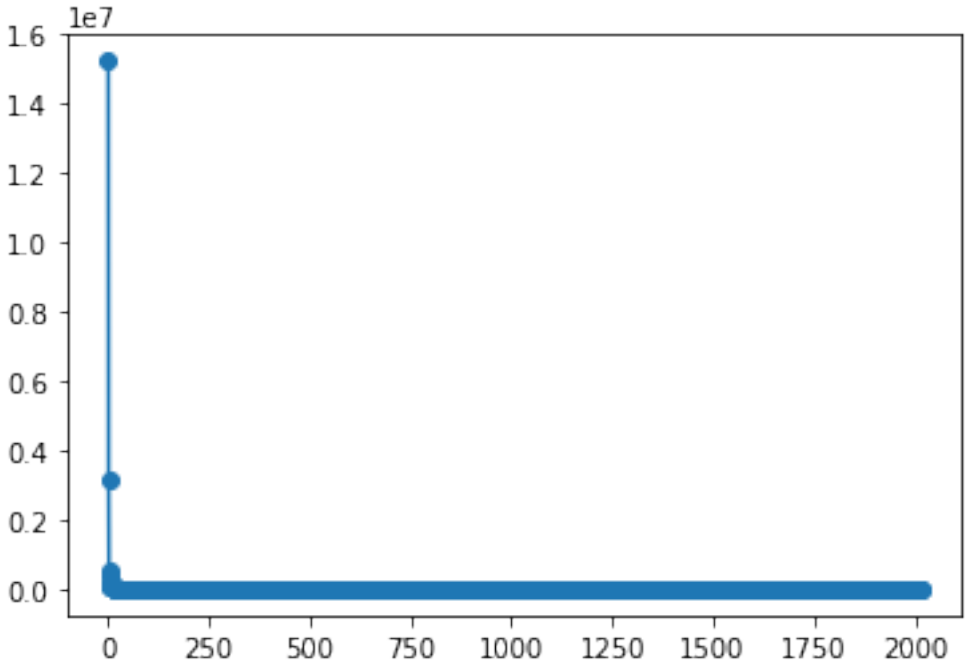
In [56]:
```
# display the first image
plt.imshow(yalefaces[:, 0].reshape((48, 42)), cmap='gray')
plt.axis('off')
plt.show()
```

In [57]:
```
# Problem2.1
```

In [58]:
```
u, s, vh = np.linalg.svd(yalefaces, full_matrices=True)
```

In [59]:
```
# compute covariance matrix and its corresponding eigenvaluexs/eigenvectors
S = 1/(len(yalefaces)) * (yalefaces @ yalefaces.T)
w, v = np.linalg.eig(S)
```

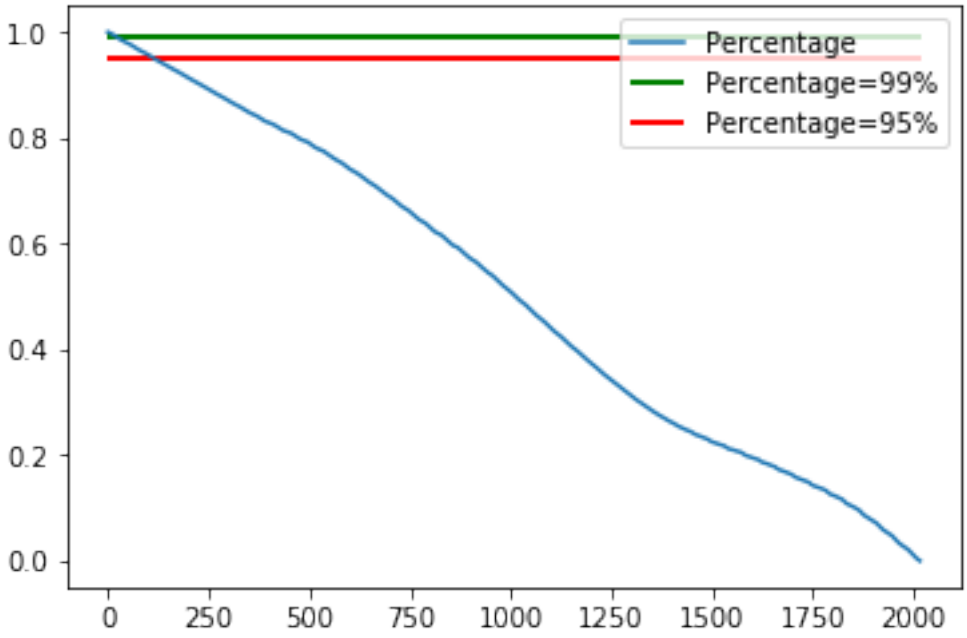
In [65]:
```
plt.plot(w, '-o')
plt.show()
```



In [50]:
```
total_variation = np.trace(S)

percentage = np.ones(len(S))
for i in range(1, len(S)):
    percentage[i] = np.trace(S[:-i, :-i]) / total_variation
```

In [51]:
```
plt.plot(percentage)
plt.hlines(y=0.99, xmin=0, xmax=len(S), linewidth=2, color='g')
plt.hlines(y=0.95, xmin=0, xmax=len(S), linewidth=2, color='r')
plt.legend(('Percentage', 'Percentage=99%', 'Percentage=95%'))
plt.show()
```




```
In [52]: # number of principle components to reprecent 99%
         print([percentage[21], percentage[22], percentage[23]])
         print('number of principle components to reprecent 99%: ', 22)
```

```
[0.9910067281362276, 0.9904885757990138, 0.9899657913738378]
number of principle components to reprecent 99%:  22
```

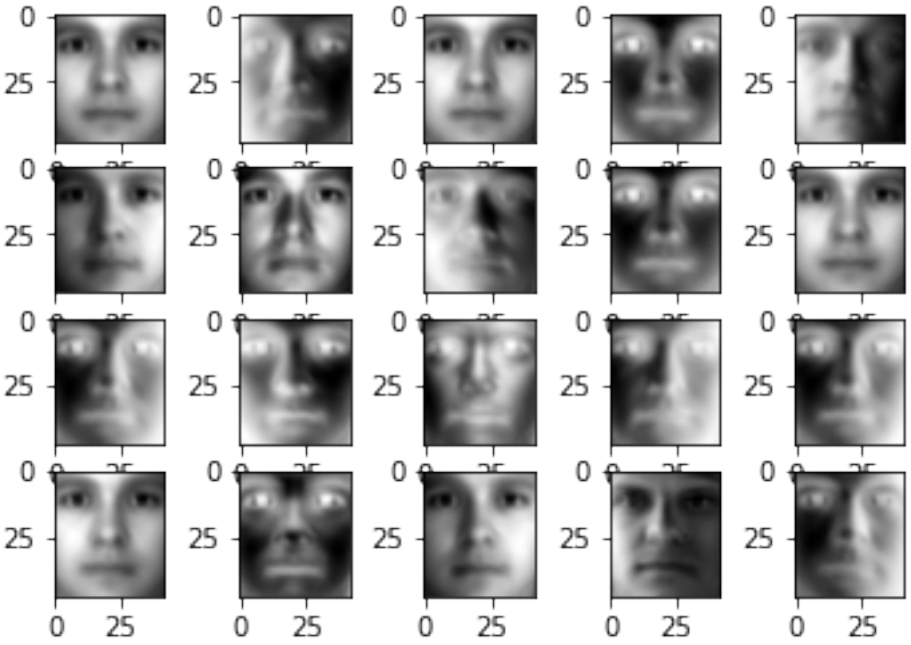
```
In [53]: # number of principle components to reprecent 95%
         print([percentage[114], percentage[115], percentage[116]])
         print('number of principle components to reprecent 95%: ', 115)
```

```
[0.9505043808225179, 0.9500408986834836, 0.9495801575438998]
number of principle components to reprecent 95%:  115
```

```
In [11]: # Problem2.2
```

```
In [40]: sample_mean = np.zeros(len(S))
         for i in range(len(S[0])):
             sample_mean[i] = np.mean(S[:,i])
```

```
In [54]: for i in range(0,20):
             plt.subplot(4, 5, i+1)
             if i == 0:
                 plt.imshow(sample_mean.reshape((48, 42)), cmap='gray')
             else:
                 plt.imshow((S@v[i-1]).reshape((48, 42)), cmap='gray')
         plt.show()
```



In [ ]: