1. In this problem, you will follow the steps from the class example taken from Example 2 on page 126 of Duda, Hart, and Stork. We will begin with four data points

$$D = \{x_1, x_2, x_3, x_4\} = \{\begin{pmatrix}0\\1\end{pmatrix}, \begin{pmatrix}2\\0\end{pmatrix}, \begin{pmatrix}1\\1\end{pmatrix} \begin{pmatrix}3\\ *\end{pmatrix}\}$$

Using the initial guess for θ^0 from the Duda-Hart-Stork example, compute the first improved estimate of μ_2.

$$Q(\theta, \theta^0) = \int lnP(x, z|\theta)P(z|x, \theta^{old})$$

$$P(z|x, \theta^{old}) = \frac{P(z, x|\theta^{old})}{P(x|\theta^{old})} = \frac{P(z, x|\theta^{old})}{\int P(z, x|\theta^{old})dz} = \frac{P(x_{(4,1)}, x_{(4,2)}|\theta^{old})}{\int P(x_{(4,1)}, x_{(4,2)}|\theta^{old})dx_{(4,2)}} = \frac{\frac{1}{2\pi\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}^{\frac{1}{2}}} e^{-[\frac{1}{2}(3)^2}}{\int \frac{1}{2\pi\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}^{\frac{1}{2}}} e^{-[\frac{1}{2}(3)^2+}}$$

$$=> \frac{\left[\frac{1}{\sqrt{2\pi}}e^{\frac{-3^2}{2}}\right]\left[\frac{1}{\sqrt{2\pi}}e^{\frac{-x_{(4,2)}^2}{2}}\right]}{\left[\frac{1}{\sqrt{2\pi}}e^{\frac{-3^2}{2}}\right]\left[\int \frac{1}{\sqrt{2\pi}}e^{\frac{-x_{(4,2)}^2}{2}} dx_{(4,2)}\right]} = \left[\frac{1}{\sqrt{2\pi}}e^{\frac{-x_{(4,2)}^2}{2}}\right]$$

$$P(z|x, \theta^{old}) = \frac{1}{\sqrt{2\pi}}e^{\frac{-x_{(4,2)}^2}{2}}$$

$$Q(\theta, \theta^0) = \int_{-\infty}^{\infty} lnP(x, z|\theta)\frac{1}{\sqrt{2\pi}}e^{\frac{-x_{(4,2)}^2}{2}}$$

$$= \sum\nolimits_{\text{k=1}}^{3} lnP(x, z|\theta) + \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi}}e^{\frac{-x_{(4,2)}^2}{2}} dx_{(4,2)} + \int_{-\infty}^{\infty} lnP(\begin{pmatrix}3\\x_{(4,2)}\end{pmatrix}|\theta)\frac{1}{\sqrt{2\pi}}e^{\frac{-x_{(4,2)}^2}{2}} dx_{(4,2)}$$

$$= \sum\nolimits_{\text{k=1}}^{3} lnP(x_k|\theta) + \int_{-\infty}^{\infty} lnP(\begin{pmatrix}3\\x_{(4,2)}\end{pmatrix}|\theta)\frac{1}{\sqrt{2\pi}}e^{\frac{-x_{(4,2)}^2}{2}} dx_{(4,2)}$$

$$= \sum\nolimits_{\text{k=1}}^{3} lnP(x_k|\theta) + \int_{-\infty}^{\infty} ln\left[\frac{1}{2\pi\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}^{\frac{1}{2}}} e^{\left[\frac{-(3-\mu_1)^2}{2} - \frac{-(x_{(4,2)}-\mu_2)^2}{2}\right]}\right]\frac{1}{\sqrt{2\pi}}e^{\frac{-x_{(4,2)}^2}{2}} dx_{(4,2)}$$

$$= \sum\nolimits_{\text{k=1}}^{3} lnP(x_k|\theta) + \int_{-\infty}^{\infty} \left[ln(\frac{1}{2\pi\sigma_1\sigma_2}) - \frac{-(3-\mu_1)^2}{2} - \frac{-(x_{(4,2)}-\mu_2)^2}{2}\right]\frac{1}{\sqrt{2\pi}}e^{\frac{-x_{(4,2)}^2}{2}} dx_{(4,2)}$$

$$\ldots = \sum\nolimits_{\text{k=1}}^{3} lnP(x_k|\theta) + ln(\frac{1}{2\pi\sigma_1\sigma_2}) - \frac{-(3-\mu_1)^2}{2} - \frac{1}{2\pi\sigma_2^2} - \frac{\mu_2^2}{2\pi\sigma_2^2}$$

$$= \left[\sum\nolimits_{\text{k=1}}^{3} ln(\frac{1}{2\pi\sigma_1\sigma_2}) - \left[\frac{-(x_{k1}-\mu_1)^2}{2} + \frac{-(x_{k2}-\mu_2)^2}{2}\right]\right] - \frac{-(3-\mu_1)^2}{2} - \frac{1}{2\pi\sigma_2^2} - \frac{\mu_2^2}{2\pi\sigma_2^2} + ln(\frac{1}{2\pi\sigma_1\sigma}$$

$$= ln(\frac{1}{2\pi\sigma_1\sigma_2}) + ln(\frac{1}{2\pi\sigma_1\sigma_2}) + ln(\frac{1}{2\pi\sigma_1\sigma_2}) + ln(\frac{1}{2\pi\sigma_1\sigma_2}) - \frac{-(x_{(1,1)}-\mu_1)^2}{2} + \frac{-(x_{(1,2)}-\mu_2)^2}{2} - \frac{-(x_{(2}}{}$$

$$- \frac{-(x_{(3,1)}-\mu_1)^2}{2} + \frac{-(x_{(3,3)}-\mu_2)^2}{2} - \frac{-(3-\mu_1)^2}{2} - \frac{1}{2\pi\sigma_2^2} - \frac{\mu_2^2}{2\pi\sigma_2^2}$$

$$= 4ln(\frac{1}{2\pi\sigma_1\sigma_2}) - \frac{-(x_{(1,1)} - \mu_1)^2}{2} + \frac{-(x_{(1,2)} - \mu_2)^2}{2} - \frac{-(x_{(2,1)} - \mu_1)^2}{2} + \frac{-(x_{(2,2)} - \mu_2)^2}{2} - \frac{-(x_{(3,1)} - \mu$$

$$- \frac{1}{2\pi\sigma_2^2} - \frac{\mu_2^2}{2\pi\sigma_2^2}$$

$$= 4ln(\frac{1}{2\pi\sigma_1\sigma_2}) - \frac{-(0 - \mu_1)^2}{2} + \frac{-(1 - \mu_2)^2}{2} - \frac{-(2 - \mu_1)^2}{2} + \frac{-(0 - \mu_2)^2}{2} - \frac{-(1 - \mu_1)^2}{2} + \frac{-(1 - \mu_2)^2}{2}$$

$$= \frac{d}{d\mu_2}\left[4ln(\frac{1}{2\pi\sigma_1\sigma_2}) - \frac{-(0 - \mu_1)^2}{2} + \frac{-(1 - \mu_2)^2}{2} - \frac{-(2 - \mu_1)^2}{2} + \frac{-(0 - \mu_2)^2}{2} - \frac{-(1 - \mu_1)^2}{2} + \frac{-(1 -$$

$$= 0 - 0 - \frac{d}{d\mu_2}\left[\frac{(1 - \mu_2)^2}{2\sigma_2^2}\right] - 0 - \frac{d}{d\mu_2}\left[\frac{(\mu_2)^2}{2\sigma_2^2}\right] - 0 - \frac{d}{d\mu_2}\left[\frac{(1 - \mu_2)^2}{2\sigma_2^2}\right] - 0 - \frac{d}{d\mu_2}\left[\frac{(\mu_2)^2}{2\sigma_2^2}\right]$$

$$\frac{(1 - \mu_2)}{\sigma_2^2} - \frac{(\mu_2)}{\sigma_2^2} + \frac{(1 - \mu_2)}{\sigma_2^2} + \frac{(\mu_2)}{\sigma_2^2} = 0$$

$$\frac{2(1 - \mu_2)}{\sigma_2^2} - \frac{(2\mu_2)}{\sigma_2^2} = 0$$

$$2(1 - \mu_2) - (2\mu_2) = 0$$

$$2 - 2\mu_2) - (2\mu_2) = 0$$

$$4\mu_2 = 2$$

$$=> \mu_2 = \frac{1}{2}$$

idont know why it cuts of here idont know why it cuts of here idont know why it cuts of here idont know why it

2. (Data marginal distribution for Gaussian mixture model) Consider a Gaussian mixture model in which the marginal distribution p(z) for the latent variable z is given by Bishop equation (9.10), and the conditional distribution p(x|z) for the observed variable x is given by Bishop equation (9.11). Show that the marginal distribution p(x), obtained by summing p(z)p(x|z) over all possible values of z, is a Gaussian mixture of the form given in Bishop equation (9.7).

$$P(z) = \prod_{\text{k=1}}^{K} \pi_k^{z_k}$$

$$P(x|z) = \prod_{\text{k=1}}^{K} \pi_k^{z_k} \mathcal{N}(x|\mu_k, \Sigma_k)^{z_k}$$

$$z_k \in \{0, 1\}$$

$$P(x) = \sum_{\text{z}} P(z)P(x|z) = \sum_{\text{k=1}}^{K} (\prod_{\text{k=1}}^{K} \pi_k^{z_k})(\prod_{\text{k=1}}^{K} \mathcal{N}(x|\mu_k, \Sigma_k)^{z_k})$$

$$=> \sum_{\text{z}} \prod_{\text{k=1}}^{K} \left[\pi_k \mathcal{N}(x|\mu_k, \Sigma_k)\right]^{z_k} = \sum_{\text{k=1}}^{K} \left[\pi_k \mathcal{N}(x|\mu_k, \Sigma_k)\right]$$

because of the 1 of k representation for z.

3. Consider a mixture distribution of the form

$$P(x) = \sum_{\mathrm{k}=1}^{K} \pi_k P(x|k)$$

where you may assume the elements of x are discrete. Denote the mean and covariance of p(x|k) by μk and Σk, respectively. Show that the mean and covariance of the mixture distribution are given by Bishop equations (9.49) and (9.50) respectively.

$$E[x] = \sum_{\mathrm{k}=1}^{K} \pi_k E_k[x] = \sum_{\mathrm{k}=1}^{K} \pi_k \mu_k$$
$$cov[x] = E[xx^T] - E[x]E[x]^T$$
$$cov[x] = \sum_{\mathrm{k}=1}^{K} E[xx^T] - E[x]E[x]^T$$
$$cov[x] = \sum_{\mathrm{k}=1}^{K} \left[\Sigma_k + \mu_k \mu_k^T\right] - E[x]E[x]^T$$

4. Consider the joint distribution of latent and observed variables for the Bernoulli distribution obtained by forming the product of p(x|z,μ) (given by Bishop equation (9.52)) and p(z|π) (given by Bishop equation (9.53)). Show that if we marginalize this joint distribution with respect to z, then we obtain Bishop equation (9.47)

$$P(x|z,\mu) = \prod_{\mathrm{k}=1}^{K} P(x|\mu_k)^{z_k}$$

$$P(z|\pi) = \prod_{\mathrm{k}=1}^{K} \pi_k^{z_k}$$

$$\sum_z P(x|\mu,\pi) = \prod_{\mathrm{k}=1}^{K} \pi_k^{z_k} \prod_{\mathrm{k}=1}^{K} P(x|\mu_k)^{z_k}$$
$$= \sum_z \prod_{\mathrm{k}=1}^{K} \left[\pi_k P(x|\mu_k)\right]^{z_k} = \sum_{k=0}^{K} \left[\pi_k P(x|\mu_k)\right]$$

because of the 1 of k representation for z.

5. Show that if we maximize the expected complete-data log likelihood function (Bishop equation (9.55)) for a mixture of Bernoulli distributions with respect to µk, we obtain the M step equation shown in Bishop equation (9.59).

$$E_z[lnP(x,z|\mu,\pi)] = \sum_{n=1}^{N}\sum_{k=1}^{K}\gamma(z_{nk})\Big[ln\pi_k + \sum_{i=1}^{D}\big[x_{ni}ln\mu_{ki} + (1-x_{ni}ln(1-\mu_{ki}\big]\Big]$$

$$\frac{\partial}{\partial\mu_k}\Bigg[\sum_{n=1}^{N}\sum_{k=1}^{K}\gamma(z_{nk})\Big[ln\pi_k + \sum_{i=1}^{D}\big[x_{ni}ln\mu_{ki} + (1-x_{ni}ln(1-\mu_{ki}))\big]\Big]\Bigg]$$

$$= \sum_{n=1}^{N}\gamma(z_{nk})\left[\frac{x_{ni}}{\mu_{ki}} - \frac{1-x_{ni}}{1-\mu_{ki}}\right]$$

$$\sum_{n=1}^{N}\gamma(z_{nk})\left[\frac{x_{ni}}{\mu_{ki}} - \frac{1-x_{ni}}{1-\mu_{ki}}\right] = 0$$

$$\mu_{ki} = \frac{\sum_{n=1}^{N}\gamma(z_{nk})(x_{ni})}{\sum_{n=1}^{N}\gamma(z_{nk})}$$

$$\sum_{n=1}^{N}\gamma(z_{nk}) = N_k$$

$$\bar{x}_k = \frac{1}{N_k}\sum_{n=1}^{N}\gamma(z_{nk})(x_n)$$

6. If v is an eigenvector of the matrix AA' (A multiplied by it's transpose) with eigenvalue λ, show that A'v is an eigenvector of A'A. Remember to obey the rules for matrix mutiplication.

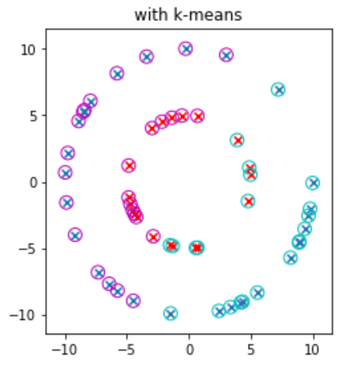
$$(AA')v = \lambda v$$
$$A'(AA')v = A'\lambda v$$
$$(A'A)(A')v = A'\lambda v$$
$$=> (A'A)(A')v = \lambda(A'v)$$
$$(A'v)\ is\ an\ eignen\ vector\ of\ AA'$$

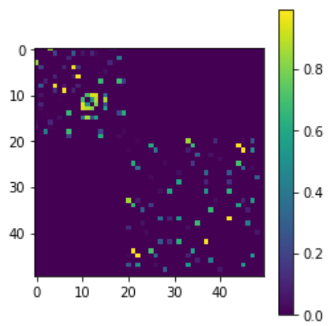
In [3]:
```
# [Vsort,Dsort] = eigsort(V,D)
#
# Sorts a matrix eigenvectors and a matrix of eigenvalues in order
# of eigenvalue size, largest eigenvalue first and smallest eigenvalue
# last.
#
# Example usage:
# [V,D] = eig(A);
# [Vnew,Dnew] = eigsort(V,D);
#
# Tim Marks 2002
import numpy as np

def eigsort(V, D):
    eigvals = np.diag(D)

    # Sort the eigenvalues from largest to smallest. Store the sorted
    # eigenvalues in the column vector eigvec.
    lohival = np.sort(eigvals, axis=0)
    lohiindex = np.argsort(eigvals, axis=0)
    eigvec = np.flipud(lohival)
    index = np.flipud(lohiindex)
    Dsort = np.diag(eigvec)

    # Sort eigenvectors to correspond to the ordered eigenvalues. Store
 sorted
    # eigenvectors as columns of the matrix vsort.
    M = len(eigvec)
    Vsort = np.zeros((M, M))
    for i in range(M):
        Vsort[:,i] = V[:,index[i]]

    return (Vsort, Dsort)
```

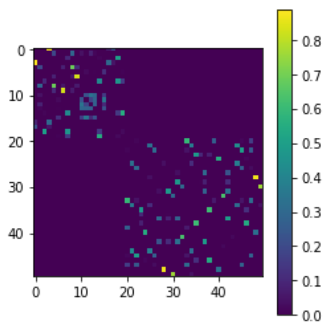
In [5]:
```
## refs
##[1]   Shi, J., and J. Malik (1997) "Normalized Cuts and Image Segmenta
tion",
##      in Proc. of IEEE Conf. on Comp. Vision and Pattern Recognition,
##      Puerto Rico.

##[2]   Kannan, R., S. Vempala, and A. Vetta (2000) "On Clusterings - Go
od, Bad ##      and Spectral", Tech. Report, CS Dept., Yale University.

## This code is from ref [3]
##[3]   Ng, A. Y., M. I. Jordan, and Y. Weiss (2001) "On Spectral Cluste
ring:
##      Analysis and an algorithm", in Advances in Neural Information Pr
ocessing##      Systems 14.


##[4]   Weiss, Y. (1999) "Segmentation using eigenvectors: a unifying vi
ew",
##      Tech. Rep., CS. Dept., UC Berkeley.
import numpy as np
import matplotlib.pyplot as plt
from scipy.io import loadmat
from scipy.cluster.vq import kmeans2

######### For question 9 uncomment this code to make your own random dat
a of the same form
r1 = 5
r2 = 10
set1 = np.random.random((1, 30))*2*np.pi
set2 = np.random.random((1, 30))*2*np.pi
d1 = np.concatenate((r1*np.cos(set1), r1*np.sin(set1)), axis=0)
d2 = np.concatenate((r2*np.cos(set2), r2*np.sin(set2)), axis=0)
######################

############## For question 9 comment out the next 3 lines
allpts = loadmat('HW3.mat')['allpts']
d1 = allpts[0:20,:].T
d2 = allpts[20:50,:].T
####################

plt.subplots_adjust(left=0, right=2, bottom=0, top=2, wspace=0.5, hspace
=0.5)

plt1 = plt.subplot(2, 3, 1)
plt1.scatter(d1[0,:], d1[1,:], c='r', marker='x')
plt1.scatter(d2[0,:], d2[1,:], marker='x')

cluster1 = d1.T
cluster2 = d2.T

allpts = np.concatenate((cluster1, cluster2), axis=0)
goto = len(allpts)

## compute A (step 1)
####  experiment with sigsq in question 8
sigsq = .9
```

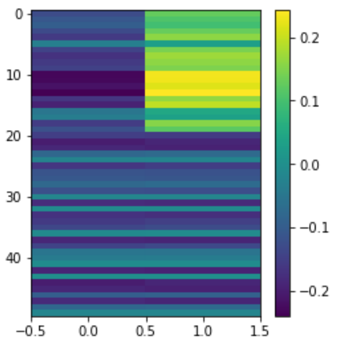
```
Aisq = allpts[:,[0]]**2+allpts[:,[1]]**2
Dotprod = np.dot(allpts, allpts.T)
distmat = -np.tile(Aisq.T, (goto, 1)) - np.tile(Aisq, (1, goto)) + 2*Dot
prod
Afast = np.exp(distmat/(2*sigsq))
A = Afast - np.diag(np.diag(Afast))
plt.subplot(2, 3, 2)
plt.imshow(A)
plt.colorbar()

## step 2
D = np.diag(np.sum(A.T, axis=0))
L = (np.diag(np.diag(np.power(D, -.5, where=D!=0))).dot(
    A)).dot( # NumPy can't handle D**-.5 very well :(
    np.diag(np.diag(np.power(D, -.5, where=D!=0))))
plt.subplot(2, 3, 3)
plt.imshow(L)
plt.colorbar()

## step 3
di, X = np.linalg.eig(L)
Xsort, Dsort = eigsort(X, np.diag(di))
Xuse = Xsort[:,0:2]
plt.subplot(2, 3, 4)
plt.imshow(Xuse, aspect='auto')
plt.colorbar()

## normalize X to get Y (step 4)
Xsq = Xuse*Xuse
divmat = np.tile(np.sqrt(np.sum(Xsq.T, axis=0).reshape((1, -1)).T), (1,
2))
Y = Xuse/divmat
plt.subplot(2, 3, 5)
plt.imshow(Y, aspect='auto')
plt.colorbar()

## step 5/6
centroids, labels = kmeans2(Y[:,[1]], 2, minit='points')
plt.subplot(2, 3, 6)
plt.scatter(allpts[labels==0, 0], allpts[labels==0, 1], s=100, marker=
'o', facecolors='none', edgecolors='c')
plt.scatter(allpts[labels==1, 0], allpts[labels==1, 1], s=100, marker=
'o', facecolors='none', edgecolors='m')
plt.title('with spectral clustering')

## for comparison run kmeans on original data
centroids2, labels2 = kmeans2(allpts, 2, minit='points')
plt1.scatter(allpts[labels2==0, 0], allpts[labels2==0, 1], s=100, marker
='o', facecolors='none', edgecolors='c')
plt1.scatter(allpts[labels2==1, 0], allpts[labels2==1, 1], s=100, marker
='o', facecolors='none', edgecolors='m')
plt1.set_title('with k-means')

plt.show()
```

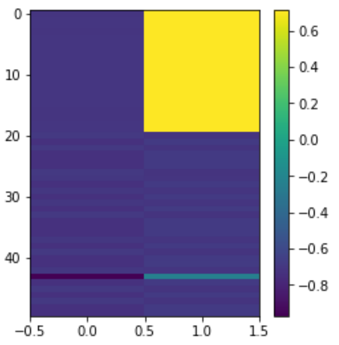
```
[[0.00000000e+00 1.16546837e-21 1.34541170e-08 ... 1.16838029e-10
  4.88504263e-54 2.30904366e-40]
 [1.16546837e-21 0.00000000e+00 1.53649511e-23 ... 7.66082850e-37
  1.81665939e-18 5.88079870e-39]
 [1.34541170e-08 1.53649511e-23 0.00000000e+00 ... 3.44313189e-36
  1.45566862e-32 2.24968618e-13]
 ...
 [1.16838029e-10 7.66082850e-37 3.44313189e-36 ... 0.00000000e+00
  3.32569113e-95 4.60106578e-89]
 [4.88504263e-54 1.81665939e-18 1.45566862e-32 ... 3.32569113e-95
  0.00000000e+00 2.66041834e-18]
 [2.30904366e-40 5.88079870e-39 2.24968618e-13 ... 4.60106578e-89
  2.66041834e-18 0.00000000e+00]]
```

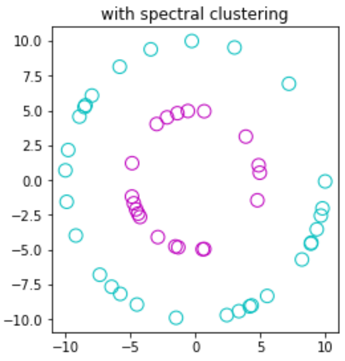






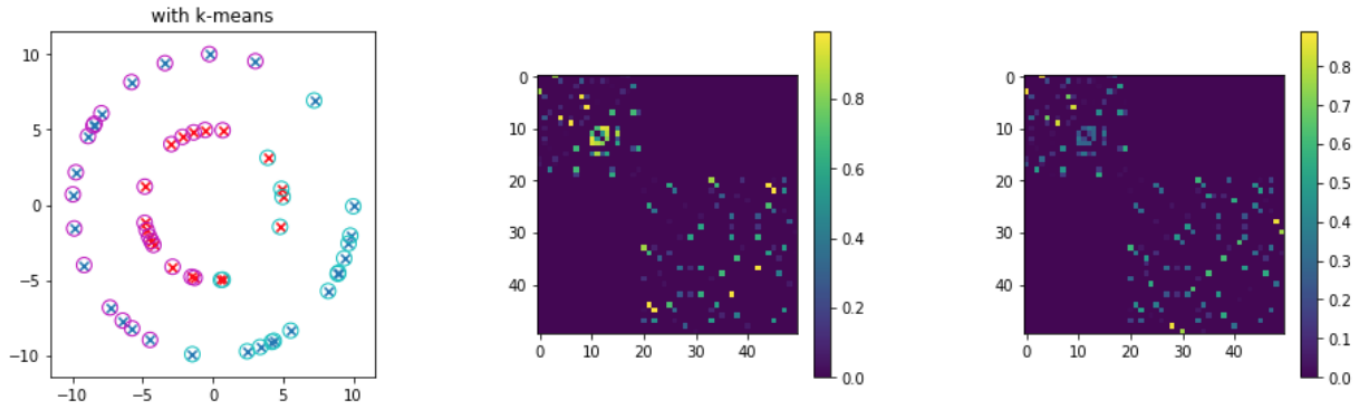




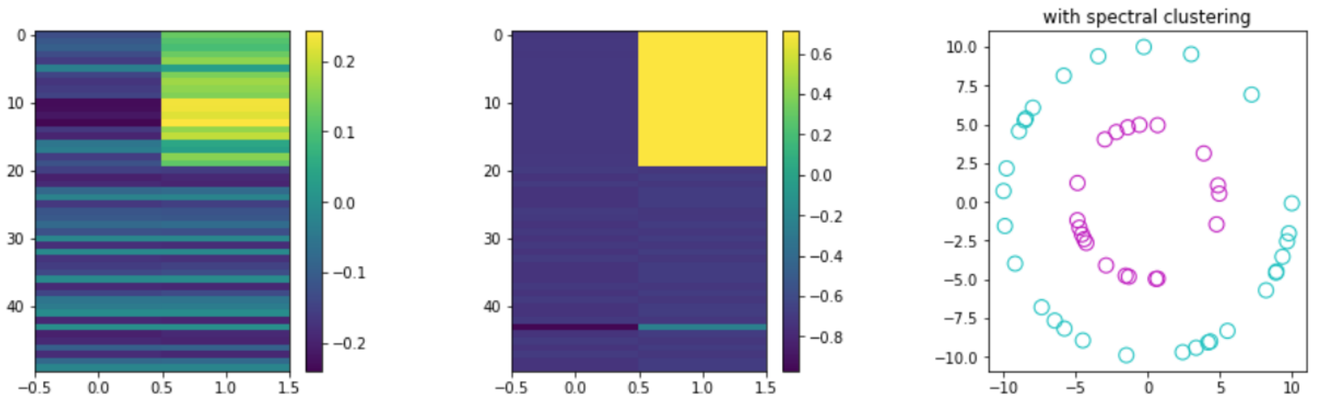


7. Describe each sub-figure in the plot and say what it shows. (e.g. Were the data clustered with spectral clustering as you would wish? How does the k-means clustering compare? What do the rest of the subplots show?

The subfigures represent the Laplacian graph. Laplacian graph shows the noise assosicated with the data. The graphs with the finner granularity are masked by noise.



for the subgraphs above which correspond to k-means, the figures show finner granularity which means the data must be masked by noise. This is not a good thing. You can see from the k-means clustering graph that this isnt an optimal cluster as the inner ring should be one color and the outter should be a different color but in this case they are mixed.



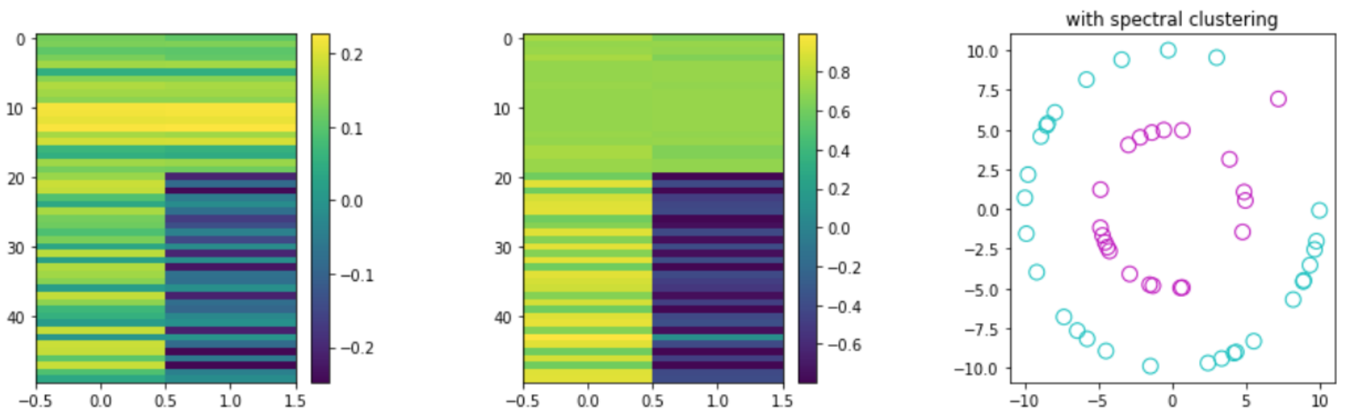
for the subgraphs above which correspond to spectial clustering, the figures show far less finner granularity than the k-mean figures which means the data has been transformed so it wont be effected as much by the noise. specital clustering uses the top eigen vectors of a matrix derived from the distance between points. As a result specital clustering will find tight clusters as it normalizes A (to form L) and X, with these modifications. The preformance of the specital clustering is better compared to a normal k-means clustering algorithm.

8. Now try changing sigsq to different values. What does sigsq represent? How does this effect the different stages? What happens when sigsq is too small? What happens when sigsq is too large?

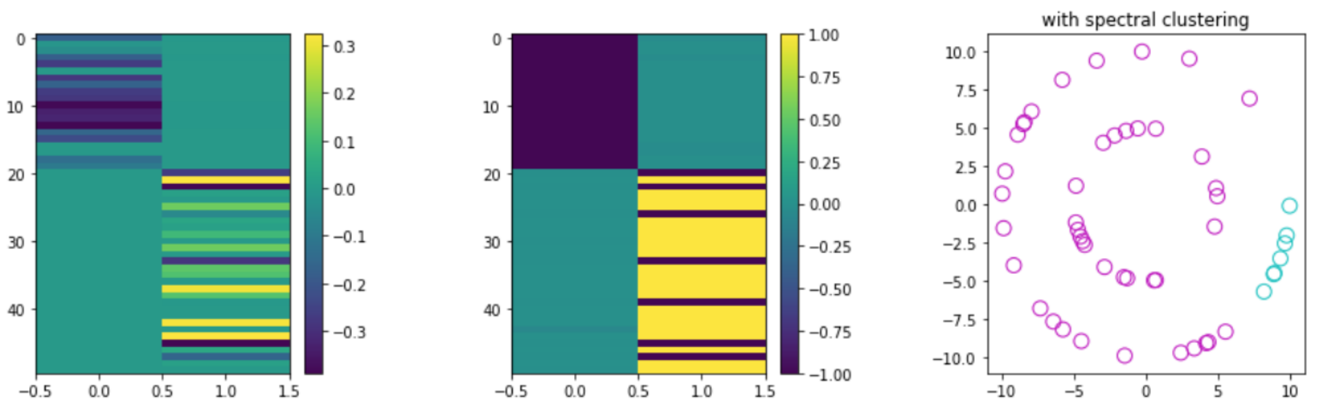
When sigsq is (.2,1.3] the clustering preforms correctly as the inner ring is classified as one color (ex. red) and the outer ring is classified as another color (ex. blue).

If sigsq >= 1.3 the clustering starts to become less optimal as the colors start to mix between the inner and outter ring

When sigsq = 1.4 the outer ring starts to get classified as blue



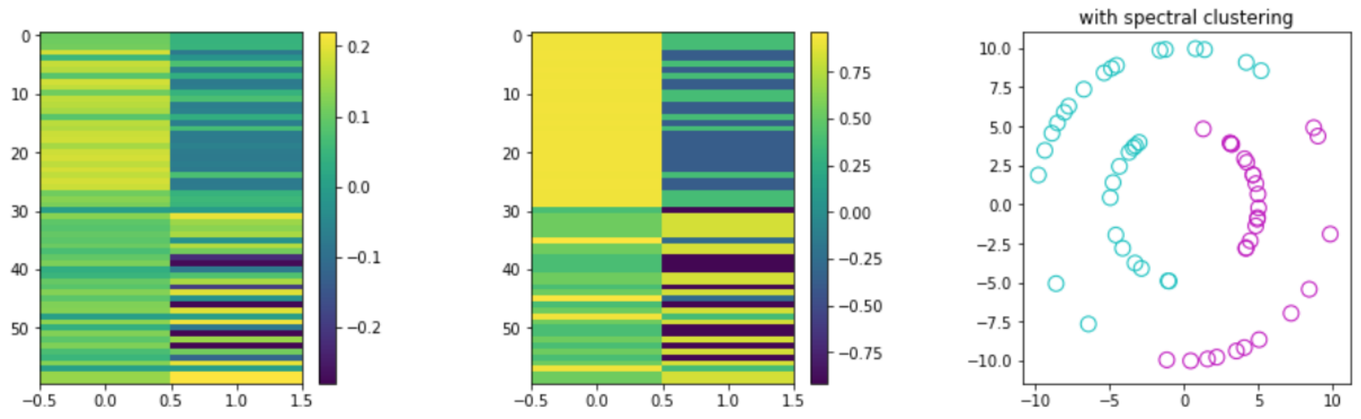
If sigsq <= .2 then part of the outer ring get mixed with a different. This clustering is not optimal.



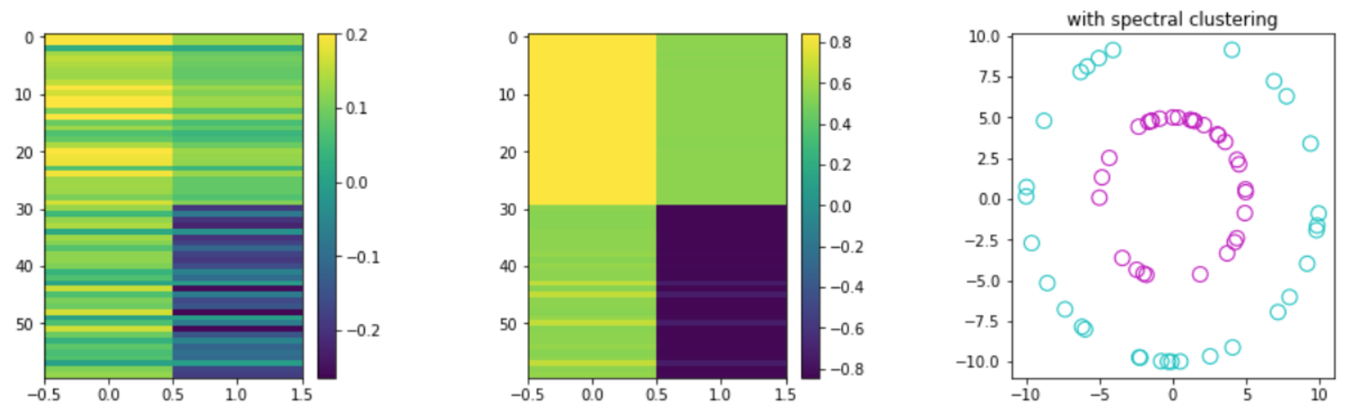
The Laplacian graphs seem to be more clear as the sigsq is smaller.

9. For this question, you will generate more two-ring datasets and run the code with your favorite value of sigsq. For this part you will have to comment and uncomment the noted lines in the code to create the data in the program (a new dataset with each run). How well does your sigsq work for different random datasets generated with the same code? Can you identify why some datasets don't work well? (be sure to run the code at least 50 times).

when randomizing the data it seems to cause noise in the data on some iterations. It seems if the data points are close together it doesnt have an issue with clustering but if there was space between points then it caused some issue. On the iteration with noisy data As the second laplacian graphs werent solidly colored.



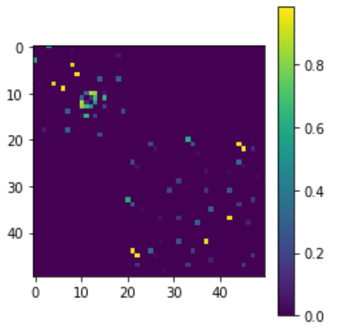
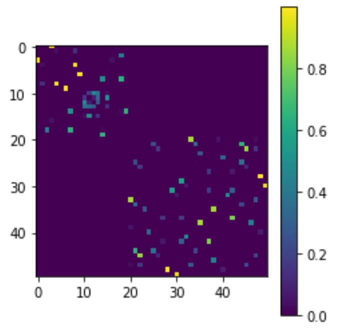
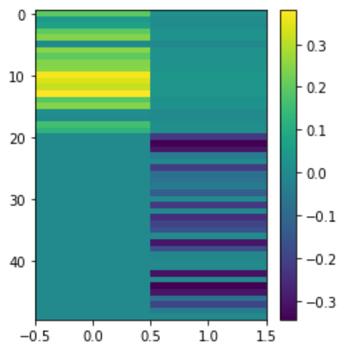
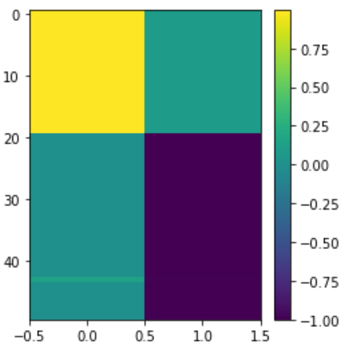
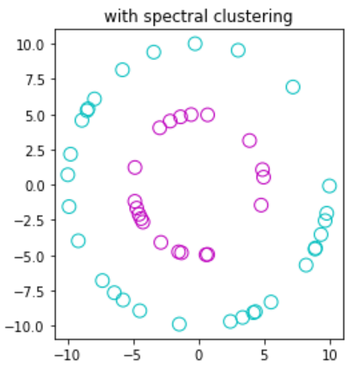
Ive realized that when I get a perfect cluster that the second laplacian graphs were more uniformly colored and dont look to have that musch static



lower sigsq gave me better odds at getting perfect clustering where the inner ring is one color and the outer ring is a different color.

In [3]:
```
#8 code
plt.subplots_adjust(left=0, right=2, bottom=0, top=2, wspace=0.5, hspace
=0.5)

## compute A (step 1)
####  experiment with sigsq in question 8
sigsq = .3
Aisq = allpts[:,[0]]**2+allpts[:,[1]]**2
Dotprod = np.dot(allpts, allpts.T)
distmat = -np.tile(Aisq.T, (goto, 1)) - np.tile(Aisq, (1, goto)) + 2*Dot
prod
Afast = np.exp(distmat/(2*sigsq))
A = Afast - np.diag(np.diag(Afast))
plt.subplot(2, 3, 2)
plt.imshow(A)
plt.colorbar()

## step 2
D = np.diag(np.sum(A.T, axis=0))
L = (np.diag(np.diag(np.power(D, -.5, where=D!=0))).dot(
    A)).dot( # NumPy can't handle D**-.5 very well :(
    np.diag(np.diag(np.power(D, -.5, where=D!=0))))
plt.subplot(2, 3, 3)
plt.imshow(L)
plt.colorbar()

## step 3
di, X = np.linalg.eig(L)
Xsort, Dsort = eigsort(X, np.diag(di))
Xuse = Xsort[:,0:2]
plt.subplot(2, 3, 4)
plt.imshow(Xuse, aspect='auto')
plt.colorbar()

## normalize X to get Y (step 4)
Xsq = Xuse*Xuse
divmat = np.tile(np.sqrt(np.sum(Xsq.T, axis=0).reshape((1, -1)).T), (1,
2))
Y = Xuse/divmat
plt.subplot(2, 3, 5)
plt.imshow(Y, aspect='auto')
plt.colorbar()

## step 5/6
centroids, labels = kmeans2(Y[:,[1]], 2, minit='points')
plt.subplot(2, 3, 6)
plt.scatter(allpts[labels==0, 0], allpts[labels==0, 1], s=100, marker=
'o', facecolors='none', edgecolors='c')
plt.scatter(allpts[labels==1, 0], allpts[labels==1, 1], s=100, marker=
'o', facecolors='none', edgecolors='m')
plt.title('with spectral clustering')

## for comparison run kmeans on original data
centroids2, labels2 = kmeans2(allpts, 2, minit='points')
plt1.scatter(allpts[labels2==0, 0], allpts[labels2==0, 1], s=100, marker
='o', facecolors='none', edgecolors='c')
```

```
plt1.scatter(allpts[labels2==1, 0], allpts[labels2==1, 1], s=100, marker
='o', facecolors='none', edgecolors='m')
plt1.set_title('with k-means')

plt.show()
```

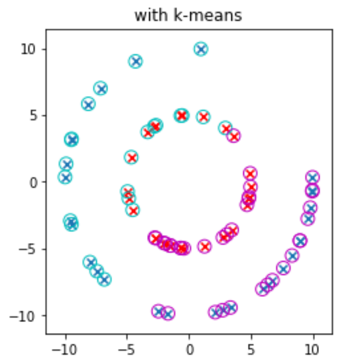






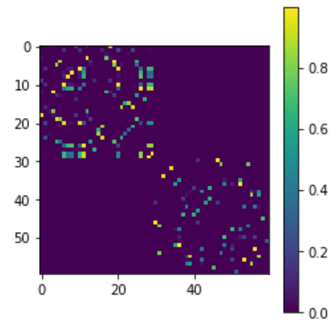
In [1]:
```
allpts = loadmat('HW3.mat')['allpts']
d1 = allpts[0:20,:].T
d2 = allpts[20:50,:].T
####################

plt.subplots_adjust(left=0, right=2, bottom=0, top=2, wspace=0.5, hspace
=0.5)

plt1 = plt.subplot(2, 3, 1)
plt1.scatter(d1[0,:], d1[1,:], c='r', marker='x')
plt1.scatter(d2[0,:], d2[1,:], marker='x')
```

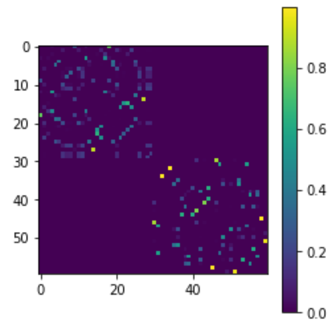
```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-1-e910d4d327a7> in <module>()
----> 1 allpts = loadmat('HW3.mat')['allpts']
      2 d1 = allpts[0:20,:].T
      3 d2 = allpts[20:50,:].T
      4 ####################
      5 

NameError: name 'loadmat' is not defined
```

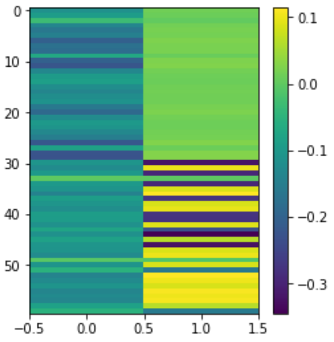
In [5]:
```
#9 code
## refs
##[1]   Shi, J., and J. Malik (1997) "Normalized Cuts and Image Segmenta
tion",
##       in Proc. of IEEE Conf. on Comp. Vision and Pattern Recognition,
##       Puerto Rico.

##[2]   Kannan, R., S. Vempala, and A. Vetta (2000) "On Clusterings - Go
od, Bad ##       and Spectral", Tech. Report, CS Dept., Yale University.

## This code is from ref [3]
##[3]   Ng, A. Y., M. I. Jordan, and Y. Weiss (2001) "On Spectral Cluste
ring:
##       Analysis and an algorithm", in Advances in Neural Information Pr
ocessing##       Systems 14.


##[4]   Weiss, Y. (1999) "Segmentation using eigenvectors: a unifying vi
ew",
##       Tech. Rep., CS. Dept., UC Berkeley.
import numpy as np
import matplotlib.pyplot as plt
from scipy.io import loadmat
from scipy.cluster.vq import kmeans2

######## For question 9 uncomment this code to make your own random dat
a of the same form
r1 = 5
r2 = 10
set1 = np.random.random((1, 30))*2*np.pi
set2 = np.random.random((1, 30))*2*np.pi
d1 = np.concatenate((r1*np.cos(set1), r1*np.sin(set1)), axis=0)
d2 = np.concatenate((r2*np.cos(set2), r2*np.sin(set2)), axis=0)
######################

############# For question 9 comment out the next 3 lines
#allpts = loadmat('HW3.mat')['allpts']
#d1 = allpts[0:20,:].T
#d2 = allpts[20:50,:].T
####################

plt.subplots_adjust(left=0, right=2, bottom=0, top=2, wspace=0.5, hspace
=0.5)

plt1 = plt.subplot(2, 3, 1)
plt1.scatter(d1[0,:], d1[1,:], c='r', marker='x')
plt1.scatter(d2[0,:], d2[1,:], marker='x')

cluster1 = d1.T
cluster2 = d2.T

allpts = np.concatenate((cluster1, cluster2), axis=0)
goto = len(allpts)

## compute A (step 1)
####  experiment with sigsq in question 8
```

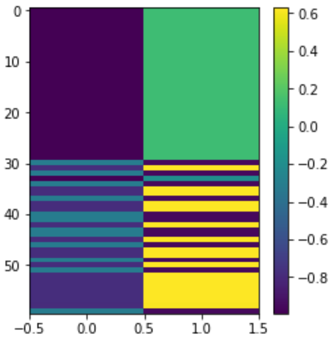
```
sigsq = .9
Aisq = allpts[:,[0]]**2+allpts[:,[1]]**2
Dotprod = np.dot(allpts, allpts.T)
distmat = -np.tile(Aisq.T, (goto, 1)) - np.tile(Aisq, (1, goto)) + 2*Dot
prod
Afast = np.exp(distmat/(2*sigsq))
A = Afast - np.diag(np.diag(Afast))
plt.subplot(2, 3, 2)
plt.imshow(A)
plt.colorbar()

## step 2
D = np.diag(np.sum(A.T, axis=0))
L = (np.diag(np.diag(np.power(D, -.5, where=D!=0))).dot(
    A)).dot( # NumPy can't handle D**-.5 very well :(
    np.diag(np.diag(np.power(D, -.5, where=D!=0))))
plt.subplot(2, 3, 3)
plt.imshow(L)
plt.colorbar()

## step 3
di, X = np.linalg.eig(L)
Xsort, Dsort = eigsort(X, np.diag(di))
Xuse = Xsort[:,0:2]
plt.subplot(2, 3, 4)
plt.imshow(Xuse, aspect='auto')
plt.colorbar()

## normalize X to get Y (step 4)
Xsq = Xuse*Xuse
divmat = np.tile(np.sqrt(np.sum(Xsq.T, axis=0).reshape((1, -1)).T), (1,
2))
Y = Xuse/divmat
plt.subplot(2, 3, 5)
plt.imshow(Y, aspect='auto')
plt.colorbar()

## step 5/6
centroids, labels = kmeans2(Y[:,[1]], 2, minit='points')
plt.subplot(2, 3, 6)
plt.scatter(allpts[labels==0, 0], allpts[labels==0, 1], s=100, marker=
'o', facecolors='none', edgecolors='c')
plt.scatter(allpts[labels==1, 0], allpts[labels==1, 1], s=100, marker=
'o', facecolors='none', edgecolors='m')
plt.title('with spectral clustering')

## for comparison run kmeans on original data
centroids2, labels2 = kmeans2(allpts, 2, minit='points')
plt1.scatter(allpts[labels2==0, 0], allpts[labels2==0, 1], s=100, marker
='o', facecolors='none', edgecolors='c')
plt1.scatter(allpts[labels2==1, 0], allpts[labels2==1, 1], s=100, marker
='o', facecolors='none', edgecolors='m')
plt1.set_title('with k-means')

plt.show()
```

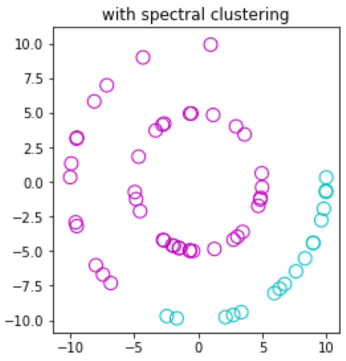












```
In [ ]:
```

```
In [ ]:
```