Machine Learing

Final Project

2D Shape Analysis with Spectral Clustering

Xiang Hao
hao@cs.utah.edu

December 28, 2009

1 Introduction

In this project, the goal is to separare similar shapes by computing each pair's distance the spectral clustering. So it is basically a clustering problem. When we do this clustering problem, there are several problems, the first question is how to compute the distance between two shapes, which I will talk about in section 2. In section 3, I will generally talk about what's spectral clustering and why I use it and how to use it. In section 4, I will demonstrate the workflow of this project. In section 5, I will give a toy example, in order to give an intuitively explaination about how it works. In addition, I will also give another two examples. In section 6, I will talk give a discussion about some issues in this project.

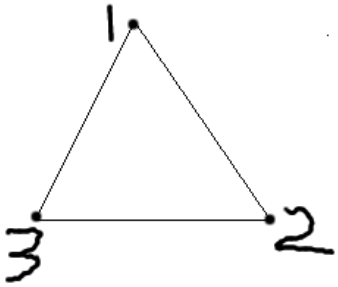
2 Distance between shapes

In the project, when computing the distance between two shapes, the assumption is that the two shapes have the same benchmmarks, which means we use the same number of poins to describe the shapes.

Befoe we compute the distance between two shapes, we need first define what is a shape. In other words, that is when two shapes are different.

Definition 1:
For a shape A,
A is rotated by certain angle θ, and we get A1, we will say A1 is the same as A;
A is translated by certain distance, and we get A2, we will say A2 is the same as A;
A is scaled by certain constant, and we get A3, we will say A3 is the same as A;

So based on the above defintion, the main problem when we compute the distance is that we need to find a metric which is invariant with rotation, translation and scaling. In the following part of this section, I will talk about this problem.



[x1,y1, x2,y2, x3,y3]

For a shape, we will select M landmarks, for these M landmarks, we will get the coordiantes for each landmark. So suppose we have M landmarks, and the shape is in a d dimention space, so for a shape will have a vector S $\in R^{Md}$. For example, in the above image, M=3, d=2

For two shapes, $S_1, S_2 \in R^{Md}$, $d(S_1, S_2)$ is the distance between S_1, S_2

1. Easiest way

   The esaiest way to compute d is $d(S_1, S_2)^2 = ||S_1 - S_2||^2$. This metric is easy but is not what we want, since it is not invariant with rotation, translation, scaling.

2. Ordinary Procrustes Analysis[1]

   Definition 2:
   $RS = (Rx_1, Ry_1, Rx_2, Ry_2, ..., Rx_M, Ry_M)$

   Definition 3:
   $RS + t = (Rx_1 + t_x, Ry_1 + t_y, Rx_2 + t_x, Ry_2 + t_y, ..., Rx_M + t_x, Ry_M + t_y)$
   Where $R \in SO(d)$, SO is special orthogonal coordinate, $RR^T = R^T R = I$, and $det(R) = 1$.

   So now we can define another new metric, $d(S_1, S_2)^2 = min||aRS_1 + t - S_2||^2, R \in SO(d), a \in R^+, t \in R^d$

3. Generalized Procrustes Analysis[1]

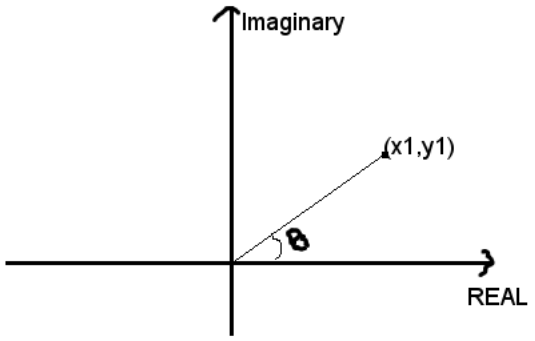
   In this case, $d(S_1, S_2)^2 = min \sum_i ||a_i R_i S_i + t_i - \mu||^2, R_i \in SO(d), a_i \in R^+, t_i \in R^d, \mu \in R^{Md}$
   In the above equation, the idea is to move each shape in order to minize the sum of distance between each shape and the μ.

   Since we want to minimize d, the problem with the above equation is that we can make it by let $a_i \to 0$.
   So we need to deal with a_i now. The solution is we can remove a_i by nomalize S_i.
   We can also remove t_i by let $mean(S_i) = 0$

   We still have some problem in this analysis. That is we can minimize the sum, but the distance between each two shapes may not be minized.

4. Kendall Shape Space

   Kendall Shape Space is a complex space rather an a real space. So in this space, the shape vector S, where $S \in R^{Md}$, and d =2, becomes Z $\in C^M$.
   where z_i represents x_i andy_i, since z_i can be writen as $x_i + iy_i$



$z_i = x_i + iy_i = r\cos\theta + ir\sin\theta = re^{i\theta}$

$(a,\phi)z_i = are^{i(\theta+\phi)} = re^{i\theta}(ae^{i\phi}) = (ae^{i\phi})z_i$, where a is a scaling factor, and ϕ is the rotation angle

$(a,\phi)Z = (ae^{i\phi})Z$

So here we can see that a shape in Kendall Shape Space, when we rotate and scale a shape, it is equalent to just multiply the shape vector with a complex number. In addition, as I said in the Generalized Procrustes Analysis, we can remove the translation by let the mean of each shape be 0, so now we have found a metric, which is invariant with rotation, translation, scaling in the Kendall Shape Space.

In the Kendalll Shape Space, each shape is a vector V, and its rotation, translation, scaling can be represented as αV, so can pick up a α to represent a shape and all its rotation, translation and scaling. When we compute the distance between two shapes, we can simplily compute the angle between the two vectors.
PS: The program, I wrote for computing the distance between two shapes, is based on Tom Fletcher's codes.

3 Spectral Clustering

Spectral Clustering is widely used in data analysis. It can find the similar groups in the data, and what it need is only a similarity martix M,[2] where M_ij meansuers the similarity between object i and object j.

There are a lot of Spectral Clustering Algorithm, here I will state two most common spectral clustering algorithms.

Unnormalized spectral clustering[3]
Input: Similarity matrix $S \in R^{nn}$, number k of clusters to construct.

1. Construct a similarity graph by one of the ways described in Section 2. Let W be its weighted adjacency matrix.
2. Compute the unnormalized Laplacian L.
3. Compute the rst k eigenvectors $u_1, ..., u_k$ of L.
4. Let $U \in R^{nk}$be the matrix containing the vectors $u_1, ..., u_k$ as columns.
5. For i = 1, . . . , n, let $y_i \in R^k$ be the vector corresponding to the i-th row of U .
6. Cluster the points $(y_i)i = 1, ..., ninR^k$ with the k-means algorithm into clusters $C_1, ..., C_k$.

Output: Clusters $A_1, ..., A_k$ with $A_i = j|y_jC_i$.

Normalized spectral clustering according to Shi and Malik (2000)[3]
Input: Similarity matrix S Rnn , number k of clusters to construct.

1. Construct a similarity graph by one of the ways described in Section 2. Let W be its weighted adjacency matrix.
2. Compute the unnormalized Laplacian L.
3. Compute the rst k generalized eigenvectors $u_1, ..., u_k$ of the generalized eigenproblem Lu = Du.
4. Let $U \in R^{nk}$ be the matrix containing the vectors $u_1, ..., u_k$ as columns.
5. For i = 1, . . . , n, let $y_i \in R^k$ be the vector corresponding to the i-th row of U .
6. Cluster the points $(y_i)i = 1, ..., ninR^k$ with the k-means algorithm into clusters $C_1, ..., C_k$.

Output: Clusters $A_1, ..., A_k$ with $A_i = j|y_jCi$.

This is an toy example from Ulrike's papper[3]

In this example, Ulrike compare the eigenvalues and eigenvectors when using different algorithms. For more details, please refer to Ulrike's paper.

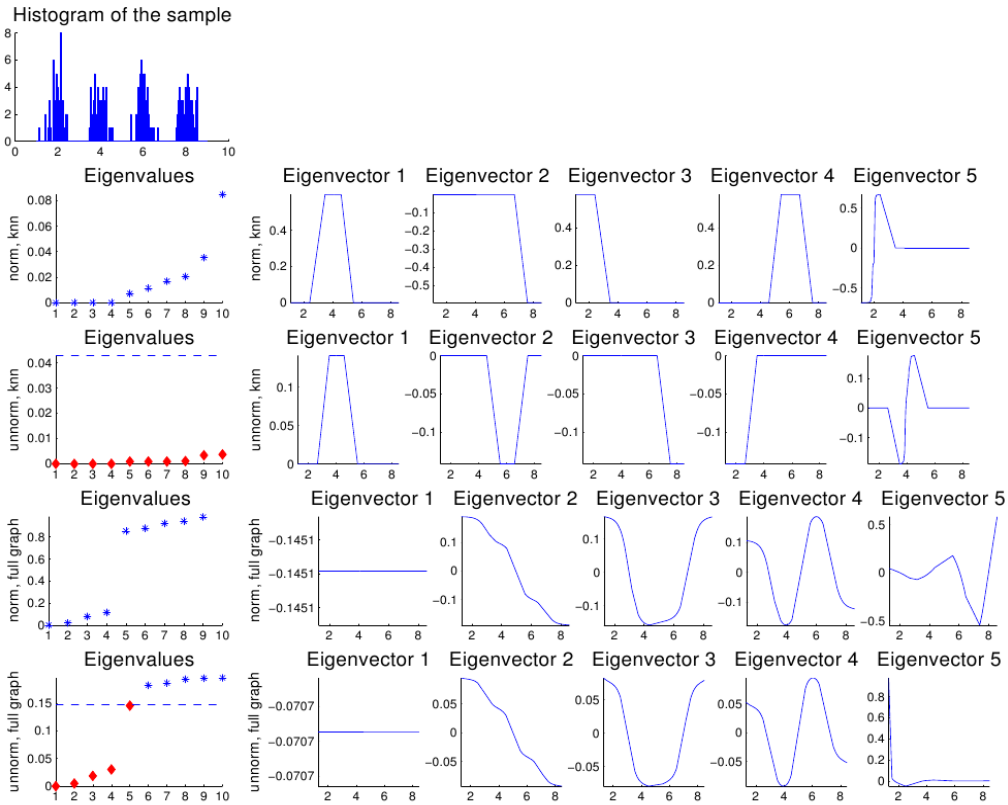
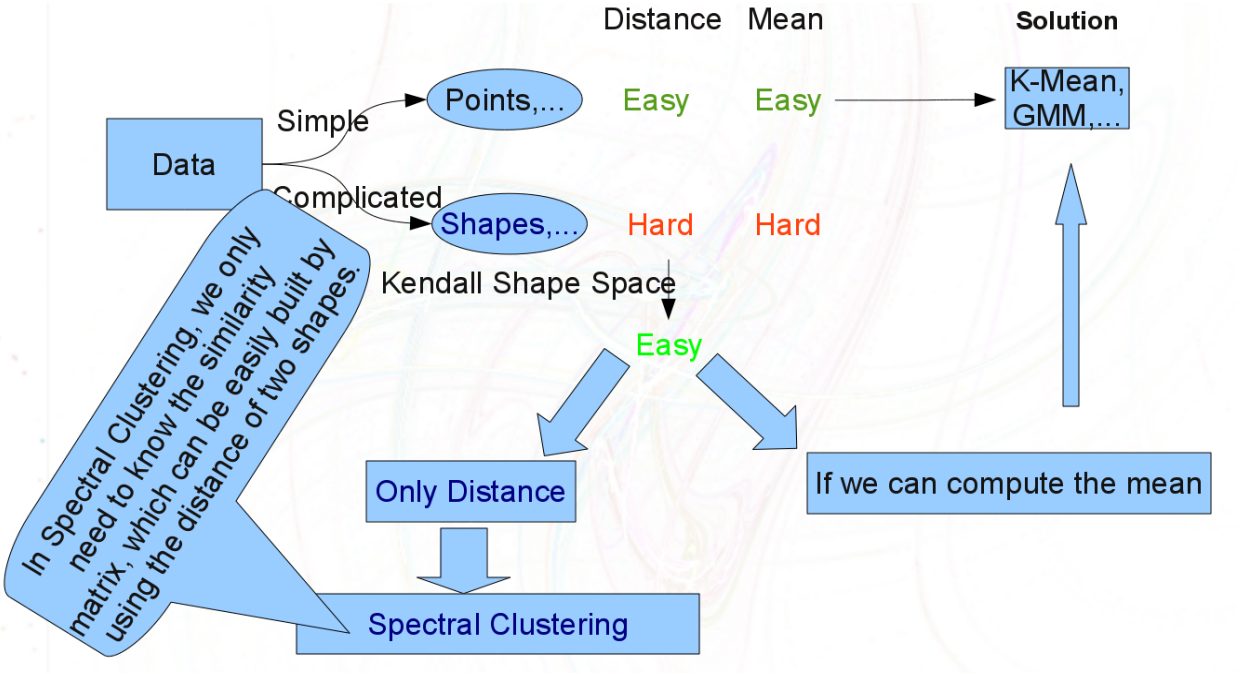


Figure 1: Toy example for spectral clustering where the data points have been drawn from a mixture of four Gaussians on $\mathbb{R}$. Left upper corner: histogram of the data. First and second row: eigenvalues and eigenvectors of L_{rw} and L based on the k-nearest neighbor graph. Third and fourth row: eigenvalues and eigenvectors of L_{rw} and L based on the fully connected graph. For all plots, we used the Gaussian kernel with $\sigma = 1$ as similarity function. See text for more details.

4 Work Flow

Here is the whole work flow of this project.



1. The first step is to compute the distance between two shapes.

2. When we got the distance:
   On one hand, if we also can compute the mean of the shapes, then we can use K-Mean to deal with the problme.
   On the other hand, if we only have the distance, we an use the spectral clustering to solve the poblem.

5 Examples

1. Toy Example

   In this toy example, there are 10 shapes. The first 5 are the right angle triangle. The second 5 are the isosceles triagnles.

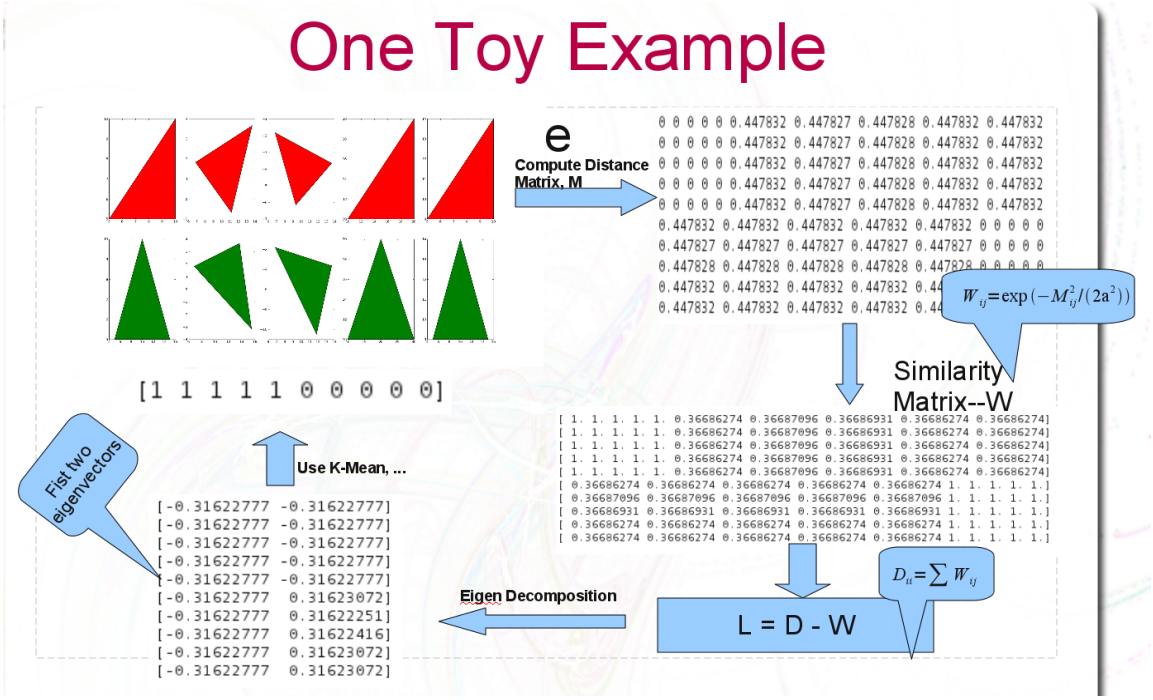
   (a) We first compute the distant marix M, where $M_{ij}$is the distance between shape i and shape j.

   (b) From the distance matrix, we compute the similarity matrix W, where $W_{ij} = exp(-\frac{M_{ij}^2}{2\sigma^2})$.

   (c) We compute L = D - W, where $D_{ii} = \sum_{j=1}^{N} W_ij$

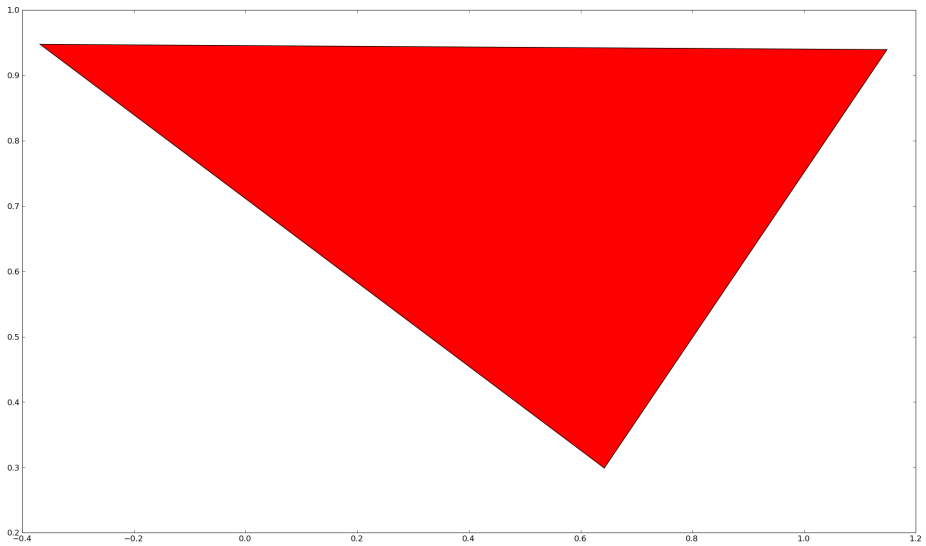
   (d) Compute the Eigen Decomposition of L, and choose the first K eigenvectors.

   (e) Use K-Mean to do the clustering on the first K eigenvectors



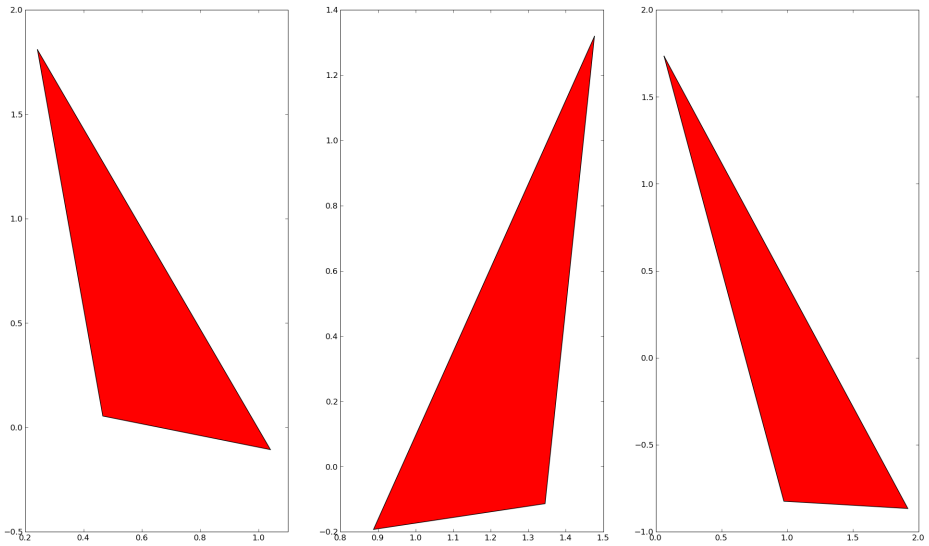
2. In this example, I randomly generate 100 tritangles.

   When I choose k=5, $\sigma = 1$, the cluster 1 is the right triangles.



   There is only one element in theis cluster, but I think this is reasonable since I randomly generate the data.

   I also find this cluster.

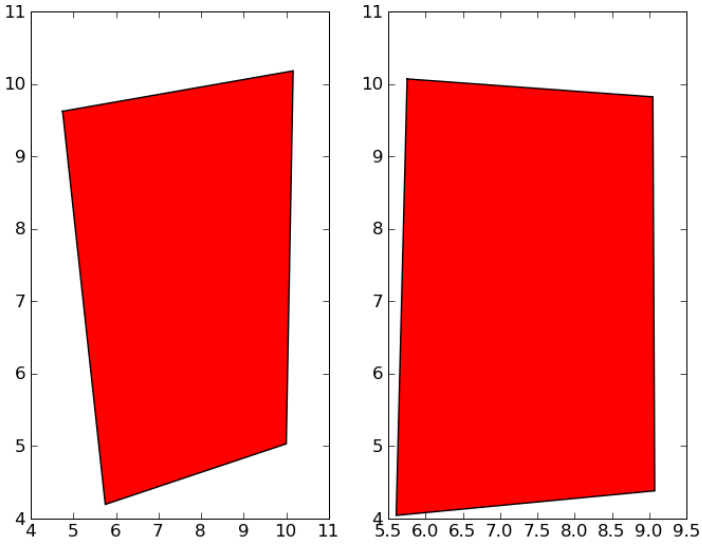


Obviously, they aer pretty simillar.

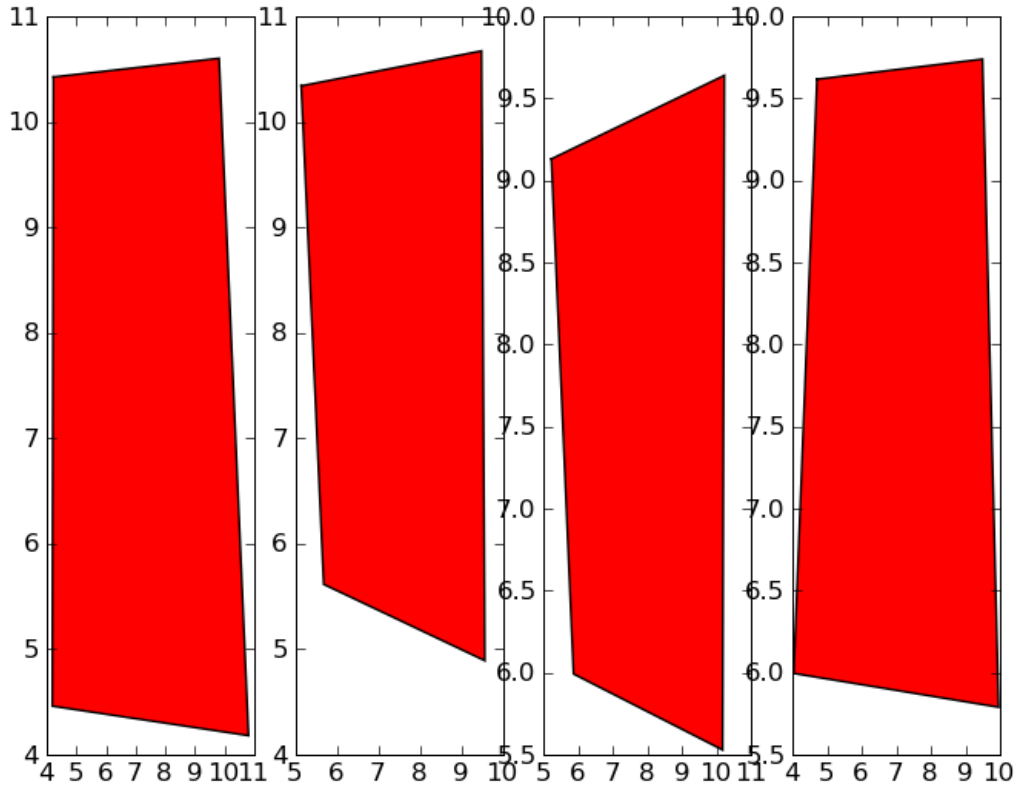
3. In this example, I randomly generate 100 quadrilateral, and I am tring to find some special quadrilateral.

   Here are some results:

   When I choose k=2, $\sigma = 0.1$, approximately, I find the square and non-square quadrilateral.



When I change k to 10, the algorithms separate the non-square quadrilateral to several clusters, the largest cluster is the trapezoid cluster.



6 Discussion

In practical, there are several issues we need to be careful.

1. Choose the σ

   We use the fucntion $W_{ij} = exp(-\frac{M_{ij}^2}{2\sigma^2})$. So the first problem is how to choose σ, if you make the σ too large or too small, basically you are elimating the difference between two shapes.

   For example,
   If $\sigma \to \infty$, $W_{ij} \to 1$
   If $\sigma \to 0$, $W_{ij} \to 0$

   So we need to carefully change the σ. Since the absolute value of the derivative of the functin $W_{ij} = exp(-\frac{M_{ij}^2}{2\sigma^2})$ is decreasing when M_{ij} is increasing, so if we want more difference between two shapes, we can make σ a little larger. On the other hand, if we want less difference between two shapes, we can make σ a liitle smaller.

2. Choose the ϵ

   Another issue is about ϵ. The ϵ means the pairwise distance s should be smaller than ϵ. If two shapes' distance is smaller than ϵ, then the two shapes are connected, otherwise, they are not connected.

   The ϵ determines if two shapes are connected or not. Usually, if you want the similarity matrix is sparse, make the *epsilon* bigger.

   However, if the ϵ is two smaller, we may lose a lot of information. If the ϵ is too larger, the similarity matrix is not sparse, when the data size is large, the program will run a long time. In addition, when the data set is large, if the ϵ is very large, the algorithm may not work well.

3. Choose k

   Choosing k is always a problem when using clustering.Usually, we can use elbow, AIC, BIC to find the k.

   In this project, since the data are shapes, when the data size is not large, we can choose different k and see whether the results make sense or not.

   Another choice is that we can determine how may clusters. For examples, if we are dealing with quadrilateral. We could ask ourselves how many different kinds of quadrilateral are there? We know there are squres, parallelogram, trapezoid, rectangles,... But, in practice, this may not work well. Usually, the bigger the k is, the more special shapes we can find in the data.

7 Reference

1. http://en.wikipedia.org/wiki/Procrustes_analysis
2. "Spectral clustering rocks!" http://www.kimbly.com/blog/000489.html
3. Ulrike von Luxburg, "A tutorial on spectral clustering". Statistic and Computing, 17(4), 2007