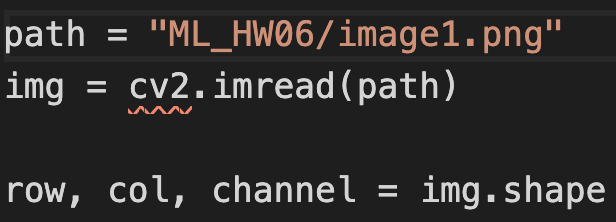
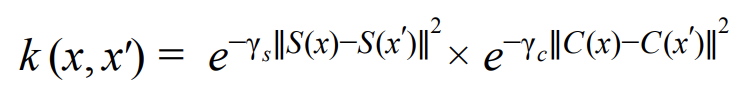
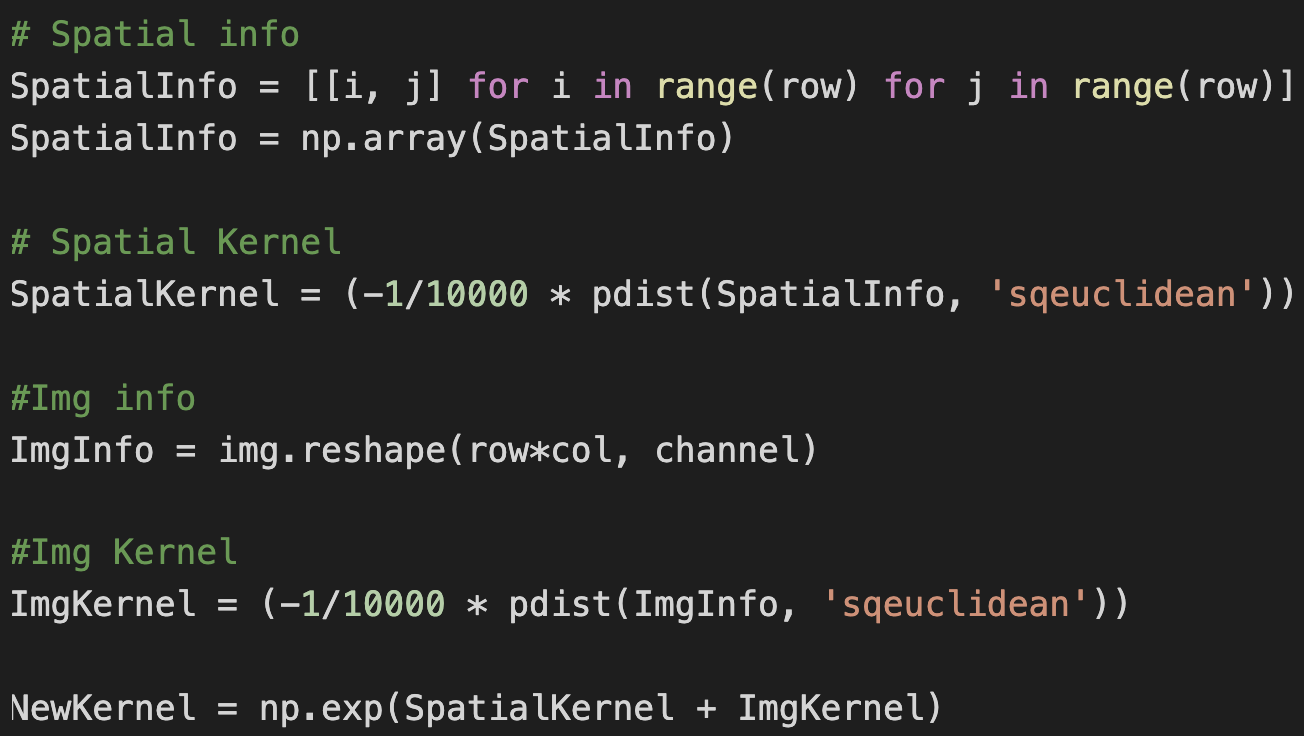
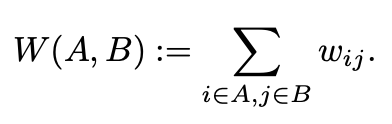
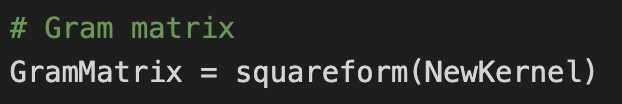
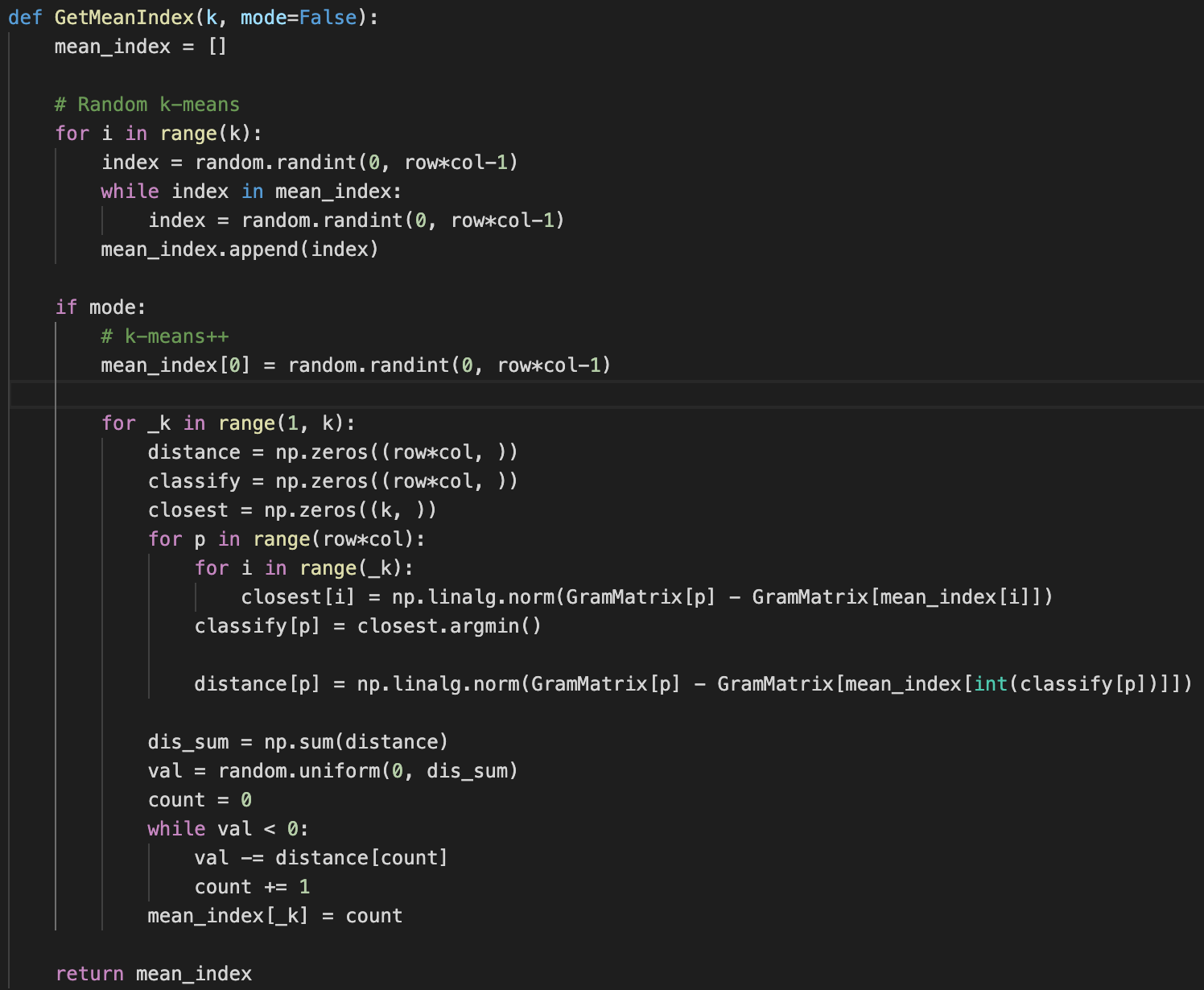
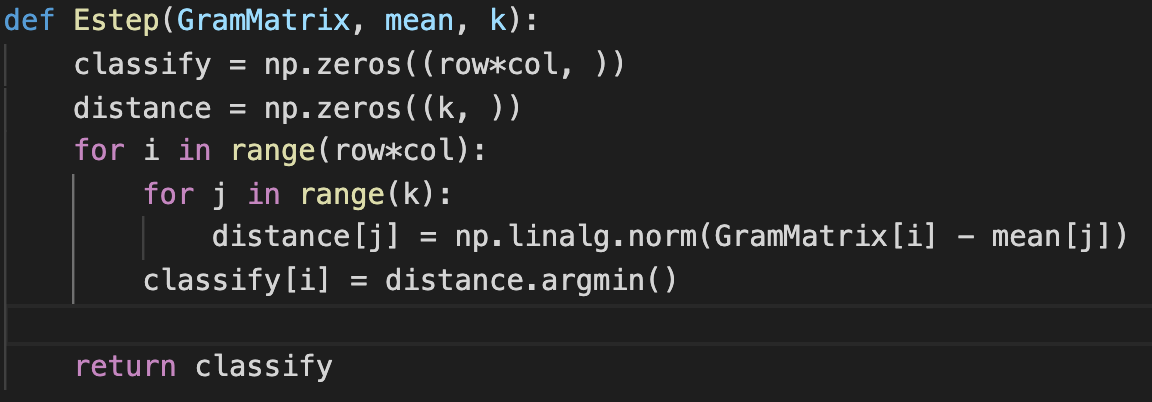
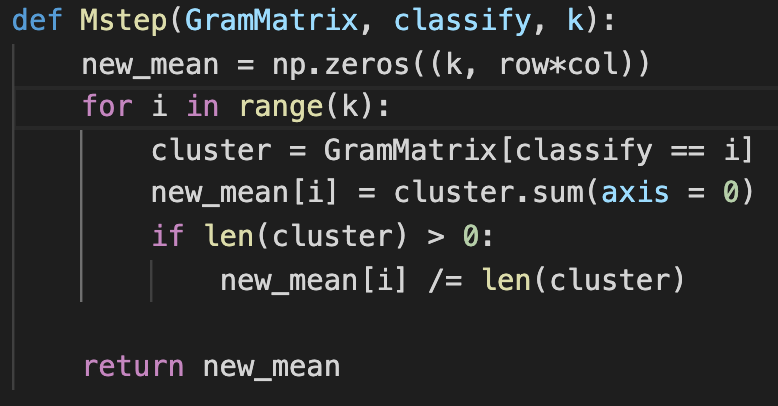
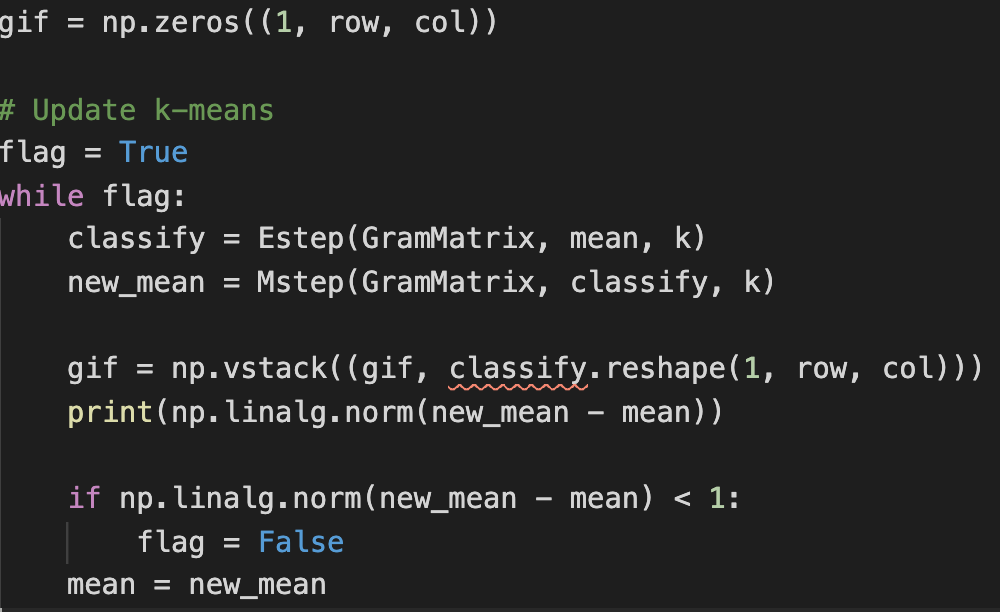
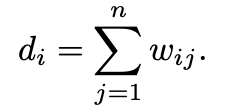
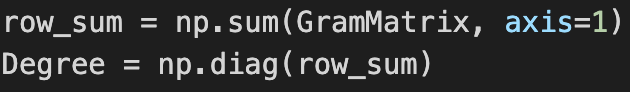
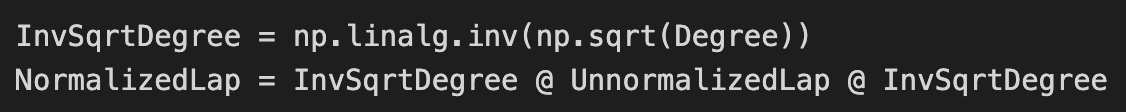
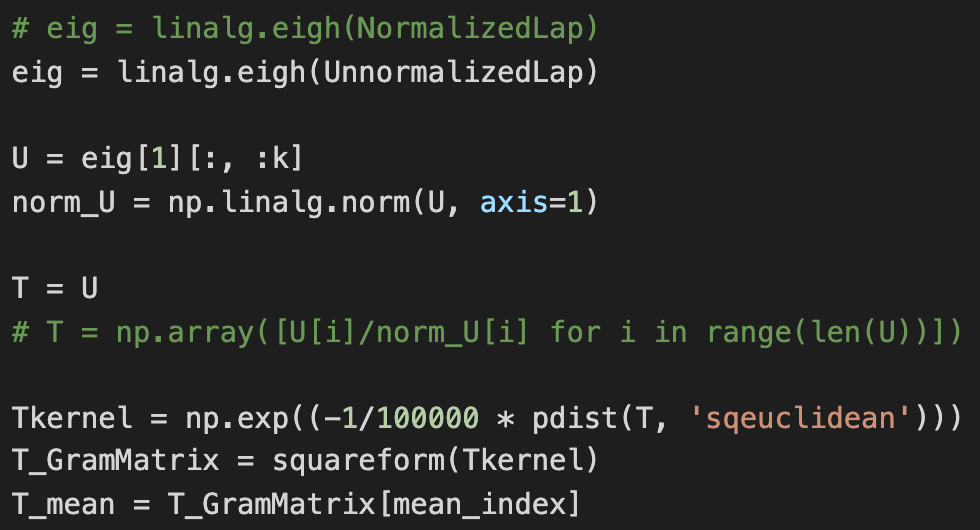
ML HW06 – Kernel K-means and Spectral Clustering

309553048\_多工所\_陳柏丞

* **Code Explanation：**

1. Read image：  
   
2. Spatial & Color RBF kernel：  
     
   
3. Gram matrix：  
     
   
4. Initialize k-means by random or k-means++：  
   
5. E step in k-means：  
   
6. M step in k-means：  
   
7. K-means & generate gif array：  
   
8. Colorize & generate gif：  
   
9. Degree matrix：  
     
   
10. Ratio cut：  
    
11. Normalized cut：  
    
12. Get k eigenvectors & generate T matrix & do k-means (same as above)：  
    
13. Examine whether the data points within the same cluster do have the same coordinates in the eigenspace of graph Laplacian or not & plotting：  
    

* **Experiment setting and results：**

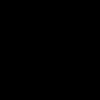
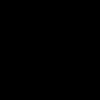
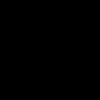
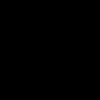
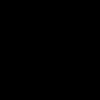
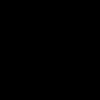
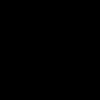
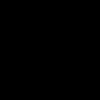
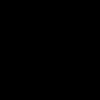
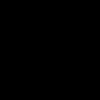
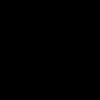
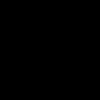
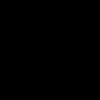
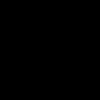
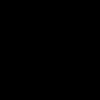
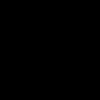
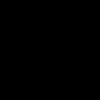
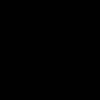
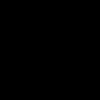
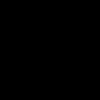
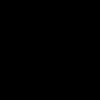
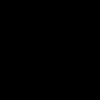
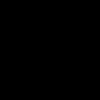
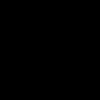
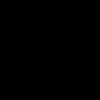
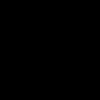
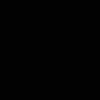
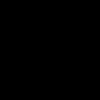
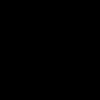
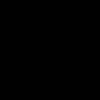
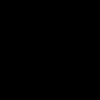
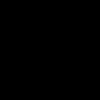
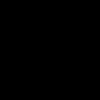
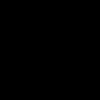
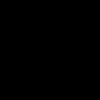
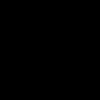
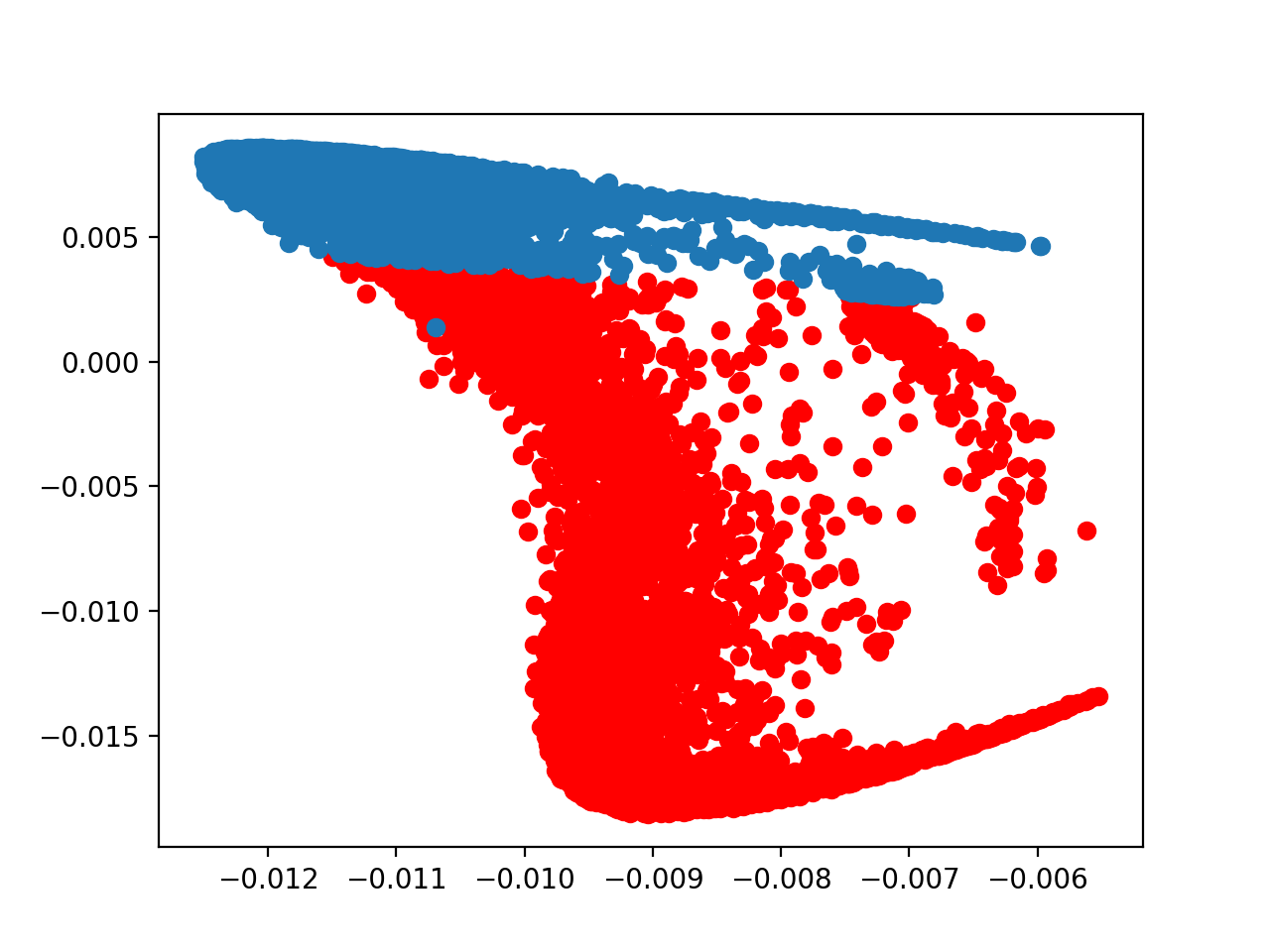
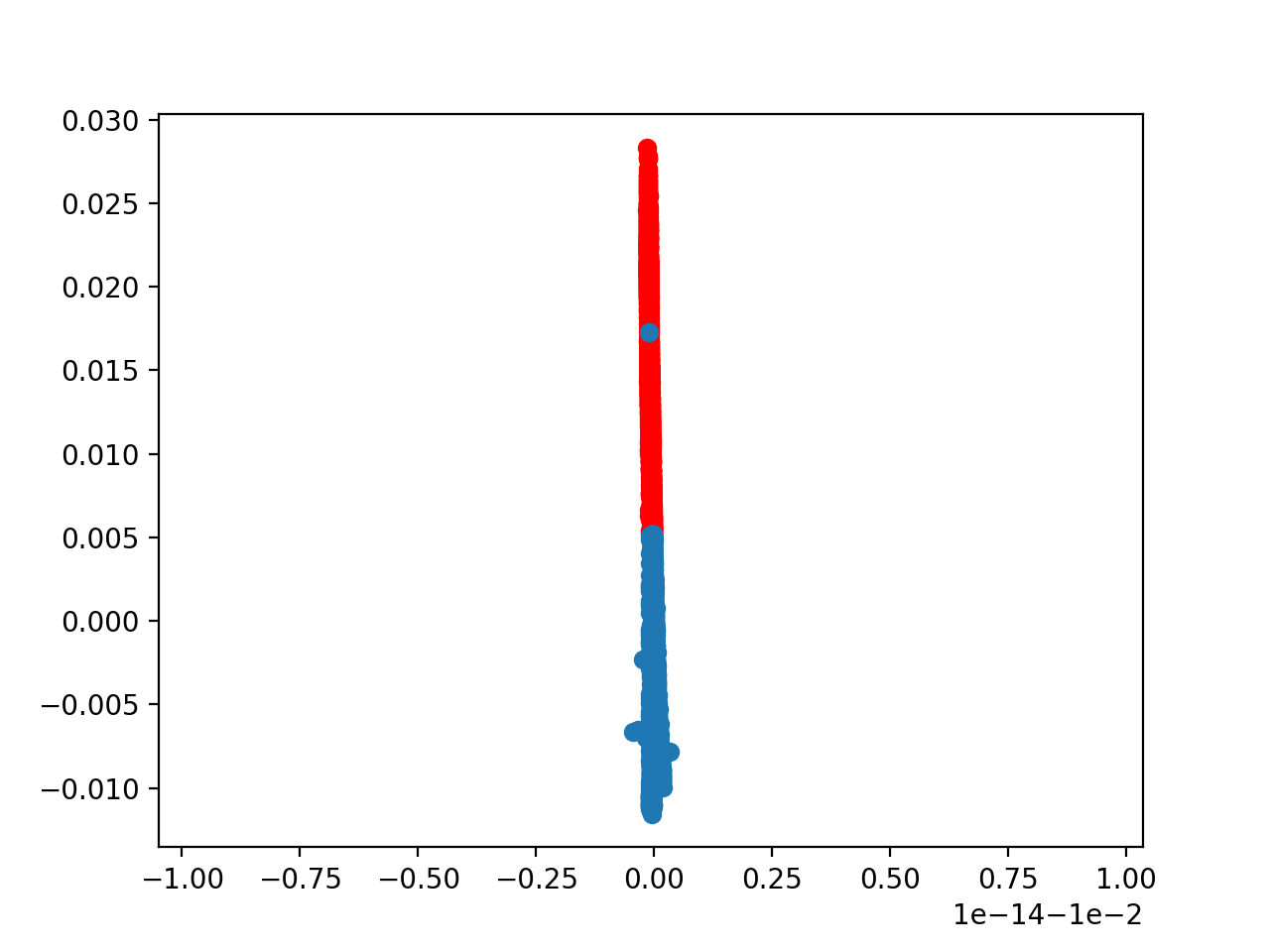
1. K-means & normalized cut & ratio cut：  
   spatial gamma=1/10000, color gamma=1/10000, k=2  
     
   image1：  
   k-means. Normalize. ratio  
       
   image2：  
   k-means. Normalize. ratio  
     
2. K-means & normalized cut & ratio cut：  
   spatial gamma=1/10000, color gamma=1/10000, k=3

image1：  
k-means. Normalize. Ratio  
    
image2：

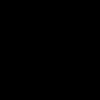
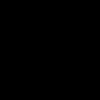
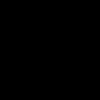
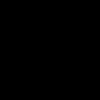
k-means. Normalize. Ratio  
  

spatial gamma=1/10000, color gamma=1/10000, k=4  
  
image1：  
k-means. Normalize. Ratio  
    
image2：  
k-means. Normalize. Ratio  
  

1. K-means++：  
   spatial gamma=1/10000, color gamma=1/10000  
     
   image1：  
   k=2. k=3. k=4  
       
   image2：  
   k=2. k=3. k=4  
       
     
   image1(normalized cut)：  
   k=2. k=3. k=4  
       
   image2(normalized cut)：  
   k=2. k=3. k=4  
       
   image1(ratio cut)：  
   k=2. k=3. k=4  
       
   image2(ratio cut)：  
   k=2. k=3. k=4  
     
2. Plotting：  
   normalized cut：  
     
   ratio cut：  
   

* **Observations and discussion：**

1. In spatial, color gamma=1/10000 and k=2, the results of three situations are the same. However, when k is larger than 2, the result of ratio cut would be worst. Among the three situations, ratio cut and normalized cut are slower than origin k-means because of inverse of matrix which is 10000x10000.
2. After k-means++, the count of iterations is smaller than performance without k-means. Then, I found that no matter which k is, the results of ratio cut and normalized cut are always clustering in 2 categories. Perhaps, I will adjust the values of spatial gamma and color gamma. When k=3, result of ratio cut is worst among all.
3. Above all, I tried to set different spatial gamma and color gamma.  
     
   image1：

spatial gamma=1/10000, color gamma=1/100, k=3  
  
spatial gamma=1/10000, color gamma=1/1000, k=3  
  
spatial gamma=1/100, color gamma=1/10000, k=3  
  
spatial gamma=1/1000, color gamma=1/10000, k=3  


Summary, spatial gamma and color gamma would lead to different clustering, and I thought spatial RBF kernel had great influence in k-means clustering.