**AI5213/EC4213/ET5402/CT5303/AI4021: Machine learning and deep learning Spring 2022**

# Report for Coding Assignment #2

(due on Fri. June 3, 2022. 11 :59PM)

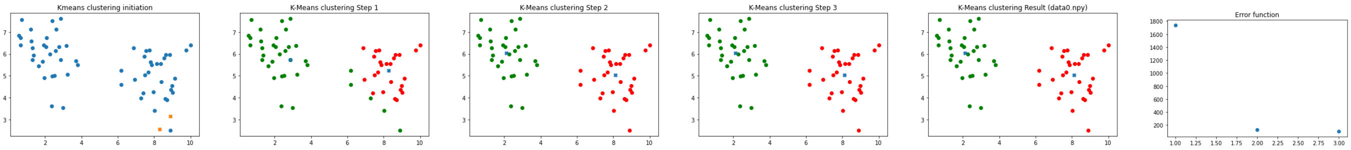
*Instructor: Jeany Son*
*Student name (GIST ID# / GitHub ID#): Sejin Park (20175068 / 59hwa)*

## REPORT1 K-means Algorithm

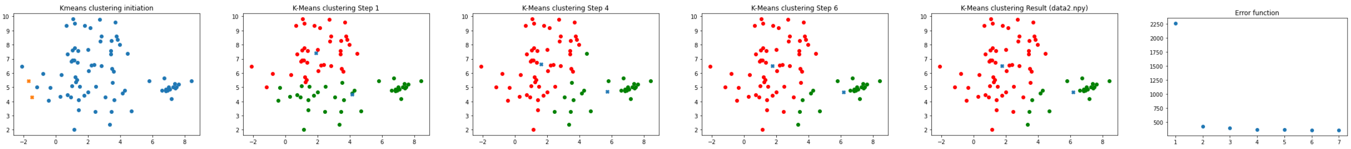
I set the random point(using random.uniform) within minimum and maximum coordinate as initiated center value. And I limited the iteration value to 10000. Although I set the limited value for iteration, every task I tried was convergence within 100 iteration.

I choose the 6 figures contain result and error function. x mark is center point each step. I couldn't upload them all because of the space, So I will upload whole of this images on gel and github.

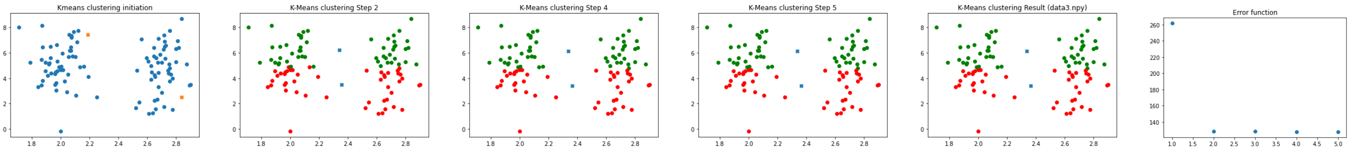
K=2 and for Data0.npy



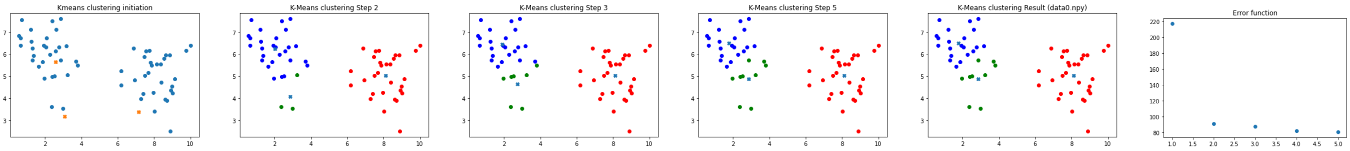
K=2 for Data2.npy



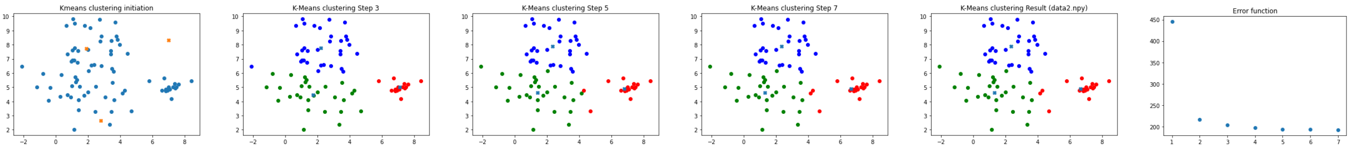
K=2 for Data3.npy



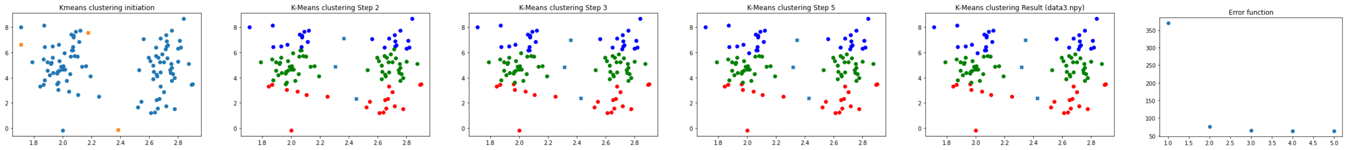
K=3 for Data0.npy



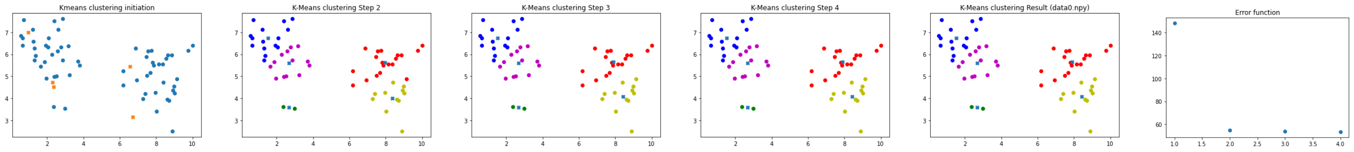
K=3 for Data2.npy



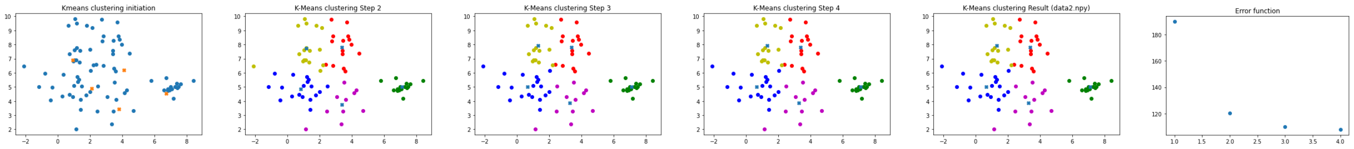
K=3 for Data3.npy



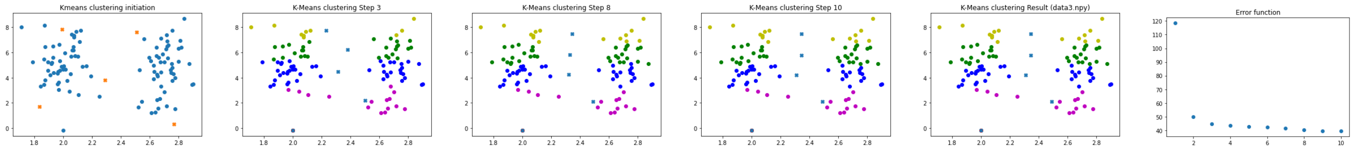
K=5 for data0.npy



K=5 for data2.npy



K=5 for data3.npy

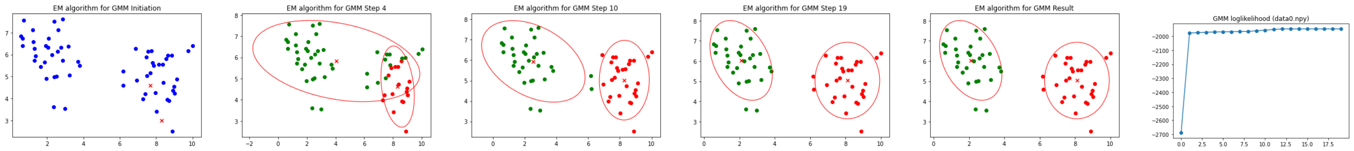


For the clustering, I draw the Each step graph and Error function. In initiation step, random k point is selected. And calculate the distance between all data and update teir center point. I can observed K means algorithm divide the point to all K =2,3,5 cluster in data0, data2 successfully, however data3 doesn't. When k is 2, it seems that the data needs to be clustered to the right and left of the cluster, but the center point is located in the center.Although Error function is converged, it seem not best clustering. How to solve this situation ? To solve this problem, there is many ways like Scaling what classmate introduce in piazza.Let's solve this problem using EM algorithms for GMM.

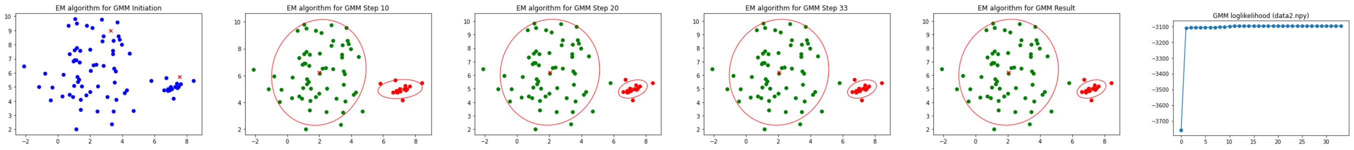
## REPORT2 EM Algorithm for GMM

I set the random point(using random.uniform) within minimum and maximum coordinate as initiated mu value. And I limited the iteration value to 10000. Although I set the limited value for iteration, every task I tried was convergence within 100 iteration. And I set the Identity matrix and 1/k for the sigma and expected for each cluster when k is numbers what we want to divide into.

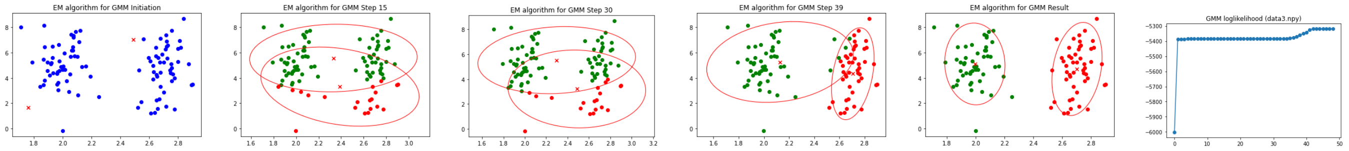
K=2 and for Data0.npy



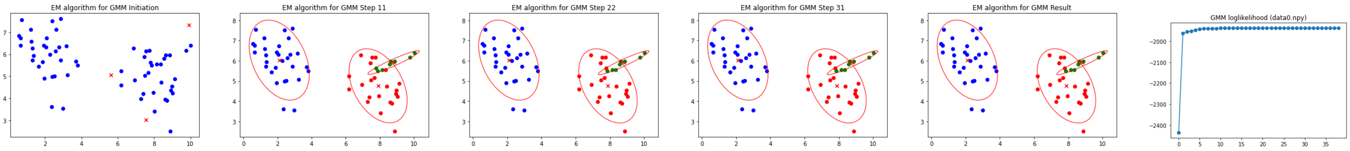
K=2 for Data2.npy



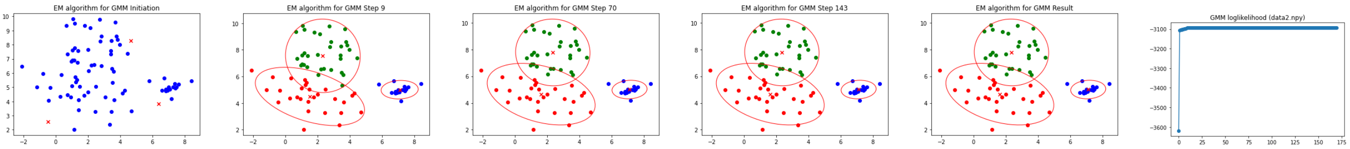
K=2 for Data3.npy



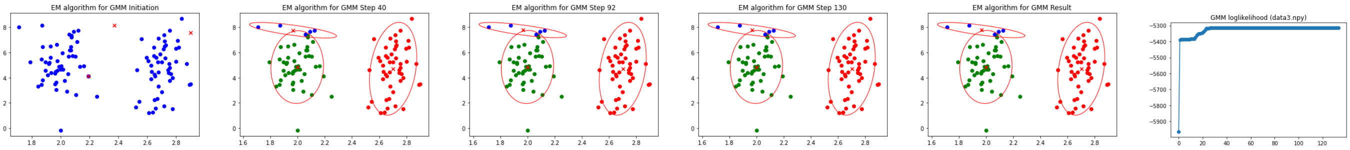
K=3 for Data0.npy



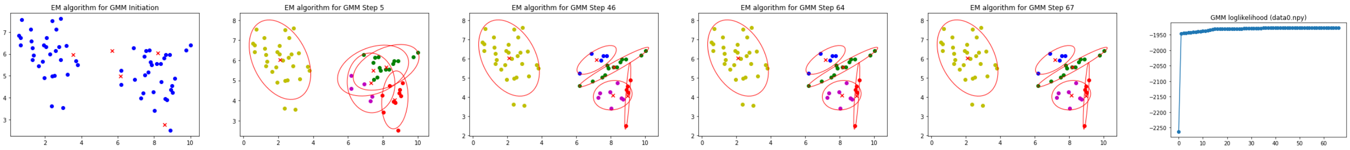
K=3 for Data2.npy



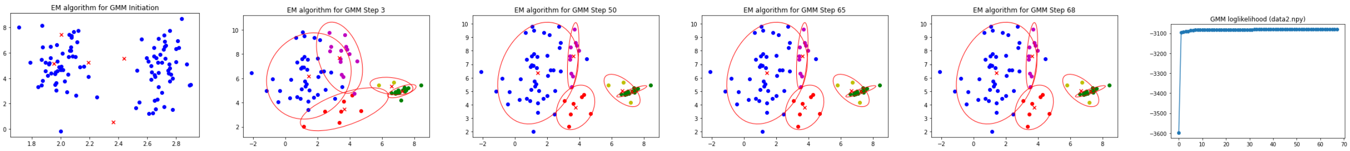
K=3 for Data3.npy



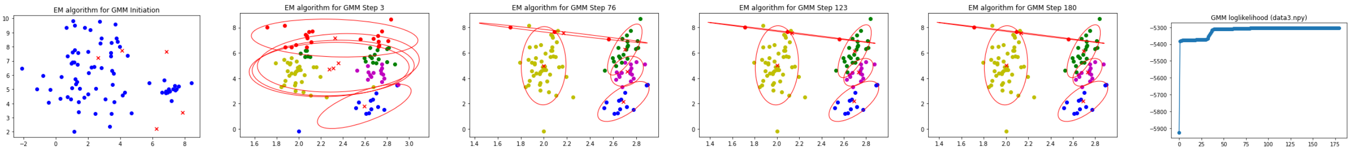
K=5 for data0.npy



K=5 for data2.npy



K=5 for data3.npy

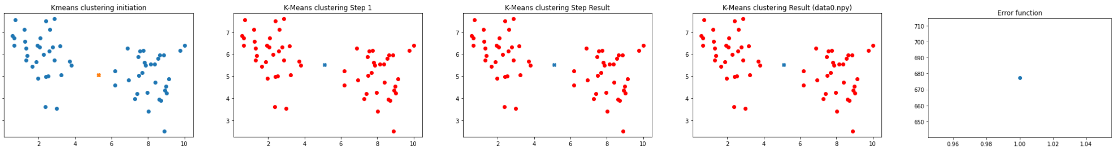


In initiation step, k random point is selected as mu. this mu is update based on probability for Gaussian mixture model. Trough the step, mu is move to appropriated side. Through the Maximize step, We can observed that log-likelihood increase until these likelihood converged. I try to draw ellipse on myself use eigenvalue and vector of sigma. However it was not appropriated. So I reference the https ://github.com/mr-easy/GMM-EM-Python/blob/master/GMM.py for draw the ellipse. EM algorithms need to large iteration compare to the K-means. And, unlike K-means clustering, EM algorithms can divide the data3 efficiently. However EM algorithms also has a problem. when k =5, I observed the very strange ellipse for cluster. And data point seem not to divide appropriately. I try the this case more time, I conclude that it maybe cause by the initiation point of mu and probability.

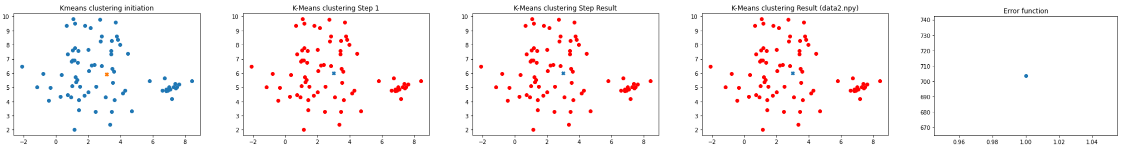
## REPORT3 Discussion

Most of difference between the K-means algorithm, and EM algorithm is how to decide the what cluster is assigned to data. K-means algorithm decide this using distance, however EM algorithms decide this using probability. Because of this difference, data3 cannot divide efficiently using K-means algorithms, however, EM algorithm can divide. Let the number of cluster K=2. And let change the starting point at two algorithms. Before the set the starting point, let think about the how to set. I set the starting center/ mu point using random function before. Let set the case 1 is two point is same and case 2 is one point have minimum point and another starting point is maximum point of data range. It is easy to understand setting point see the first image of each test. I test K-means, EM algorithm for 2 case using 3 data file.

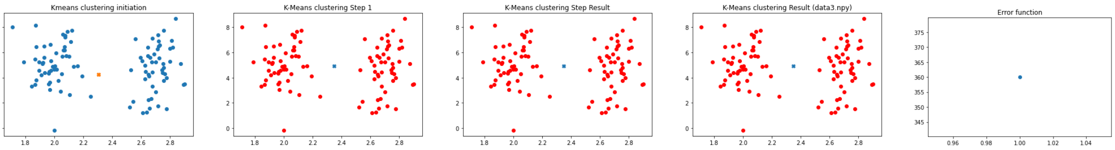
K means case1 (same initiate point) on data0.npy



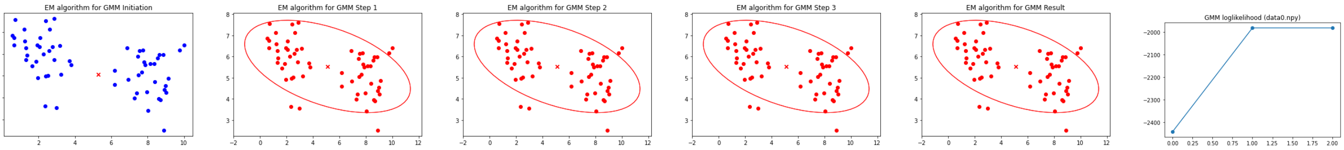
K means case1 (same initiate point) on data2.npy



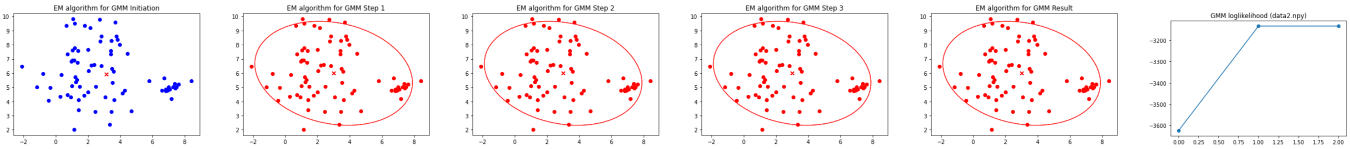
K means case1 (same initiate point) on data3.npy



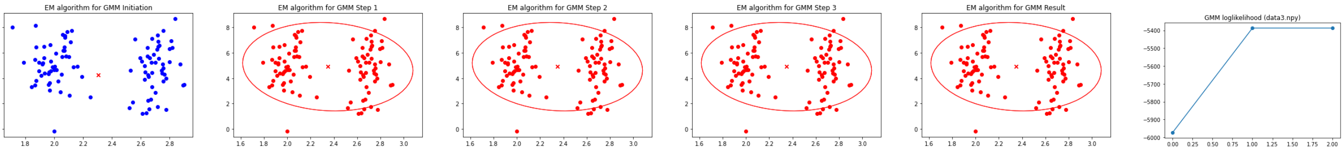
EM algorithm case1 (same initiate point) on data0.npy



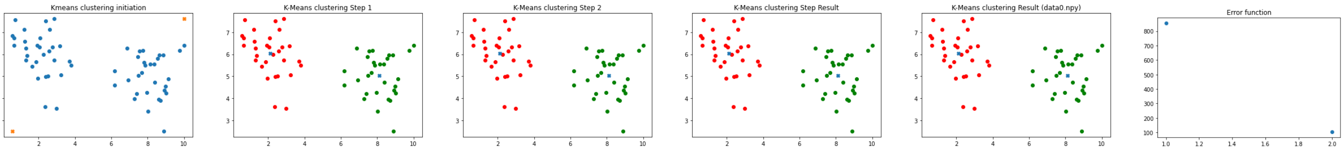
EM algorithm case1 (same initiate point) on data2.npy



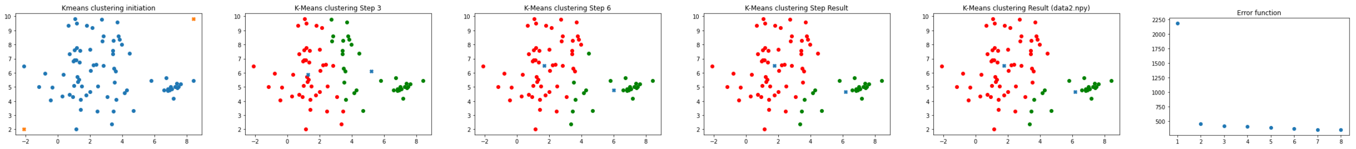
EM algorithm case1 (same initiate point) on data3.npy



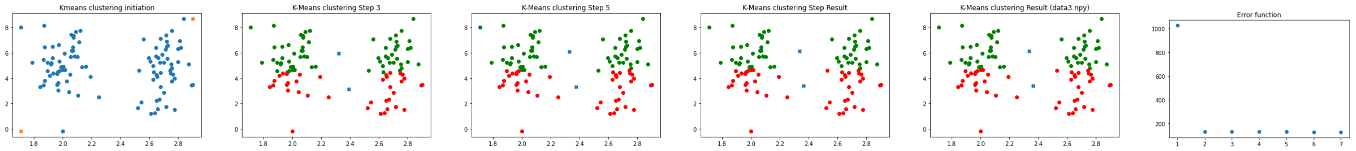
K means case2 (one point is minimum and another is maximum) on data0.npy



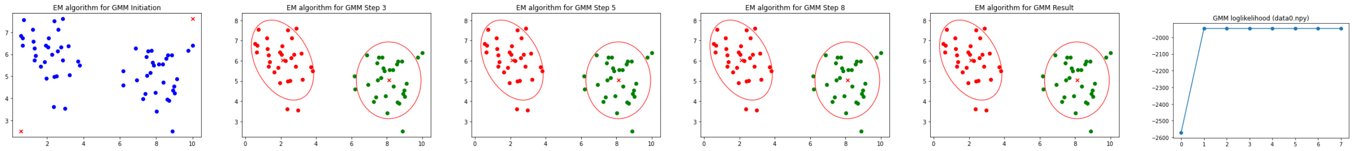
K means case2 (one point is minimum and another is maximum) on data2.npy



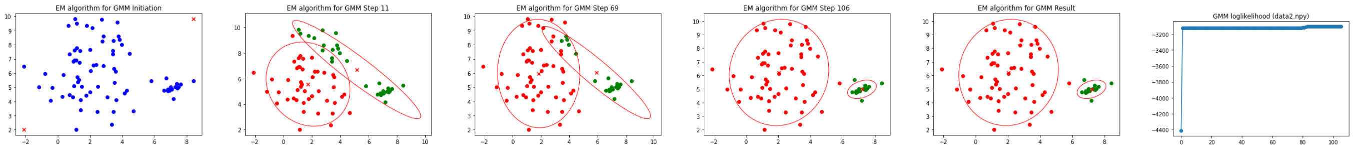
K means case2 (one point is minimum and another is maximum) on data3.npy



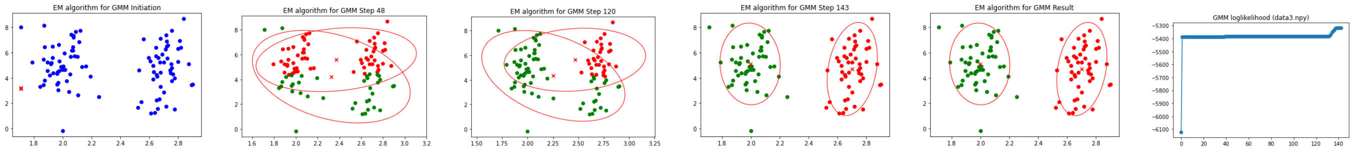
EM algorithm case2 (one point is minimum and another is maximum) on data0.npy



EM algorithm case2 (one point is minimum and another is maximum) on data2.npy

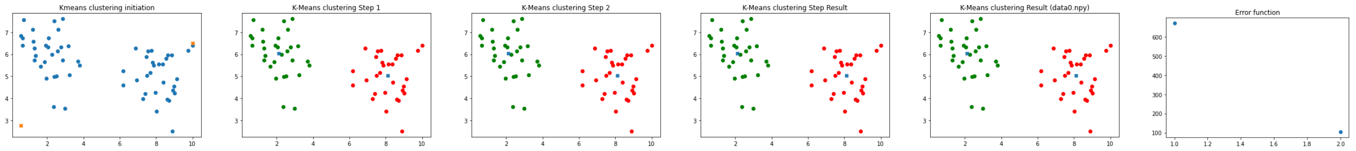


EM algorithm case2 (one point is minimum and another is maximum) on data3.npy

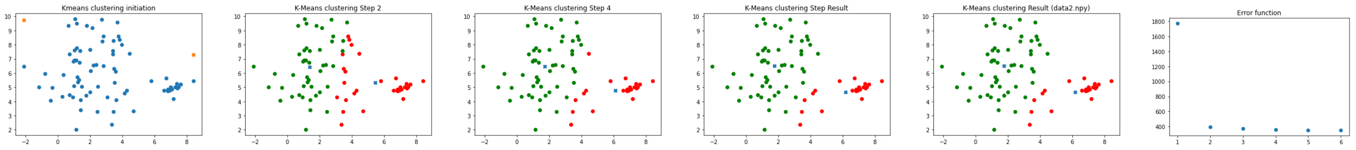


In case 1, if center point same, EM and K-means algorithm didn't divide the cluster because two center/mu point update, M-step calculating process are same(I test case1 in three different case,But I just upload one because every result are same). In case 2, K-Means and EM algorithm work efficiently. But, can we evolution this ? Let think GMM process time. EM algorithm consume the much time to convergence. How can we reduce the time on data ? Let case 3 is both side random starting point what one point is on left side and another point on right side. it is because why that we already know 2 cluster exist on both side. And let case 4 is same side starting point and compared case 3 and 4.

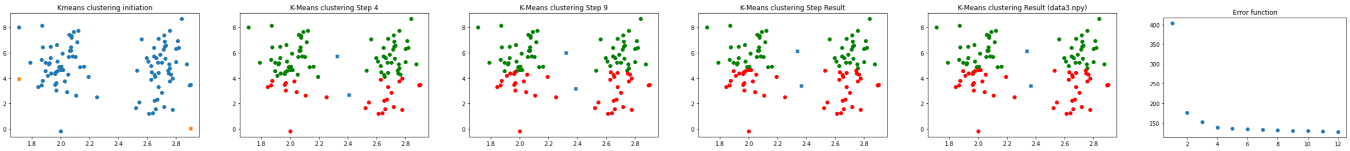
K-means algorithm case3 (both side initiate point) on data0.npy



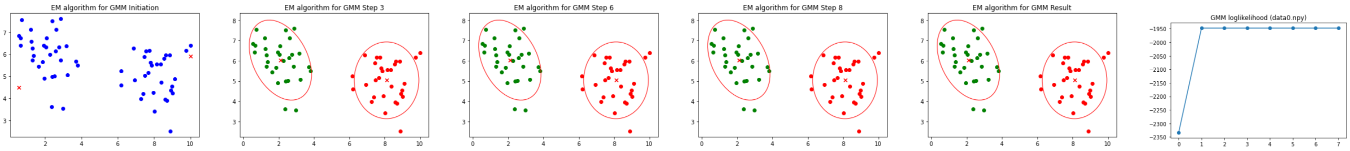
K-means algorithm case3 (both side initiate point) on data2.npy



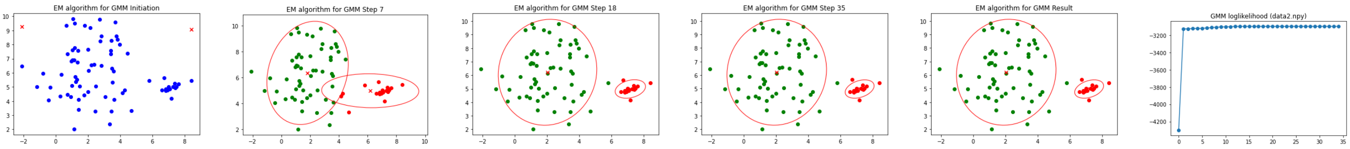
## K-means algorithm case3 (both side initiate point) on data3.npy



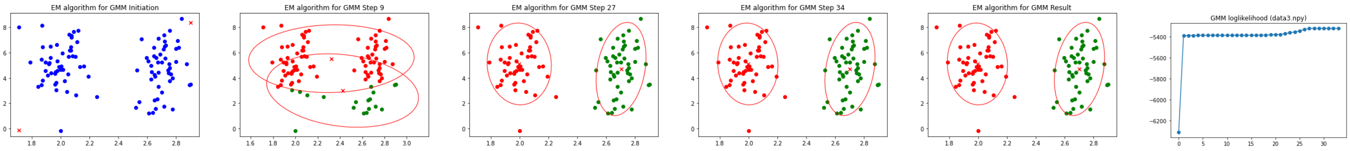
## EM algorithm case3 (both side initiate point) on data0.npy



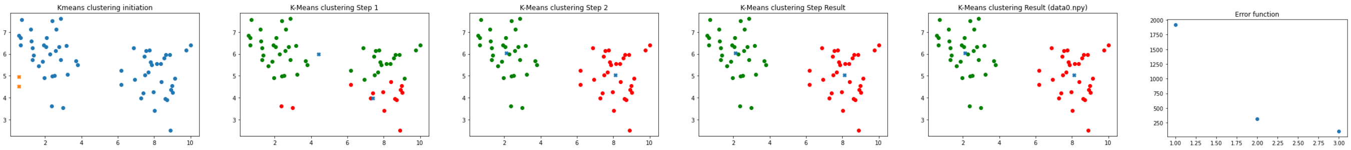
## EM algorithm case3 (both side initiate point) on data2.npy



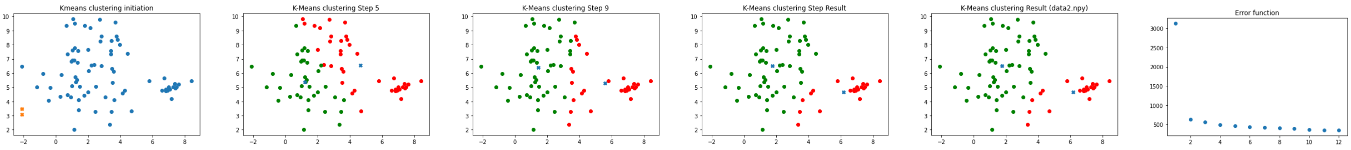
## EM algorithm case3 (both side initiate point) on data3.npy



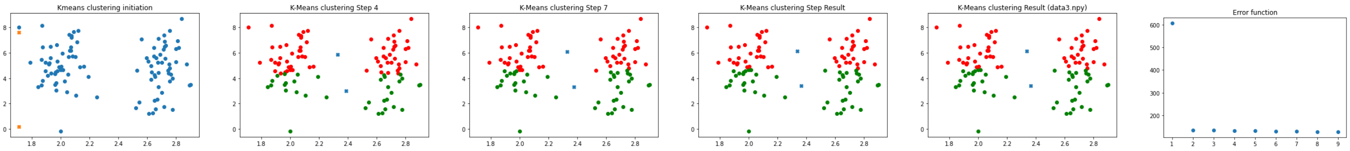
## K-means algorithm case4-1 (left side) on data0.npy



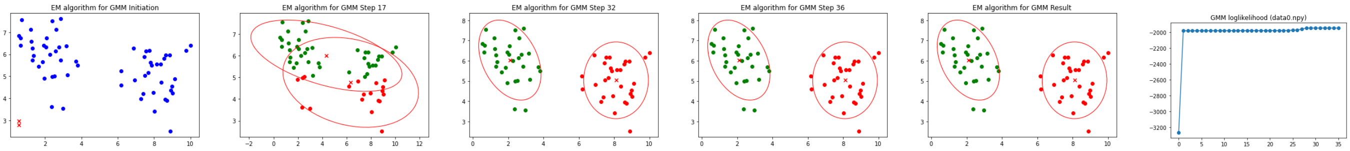
## K-means algorithm case4-1 (left side) on data2.npy



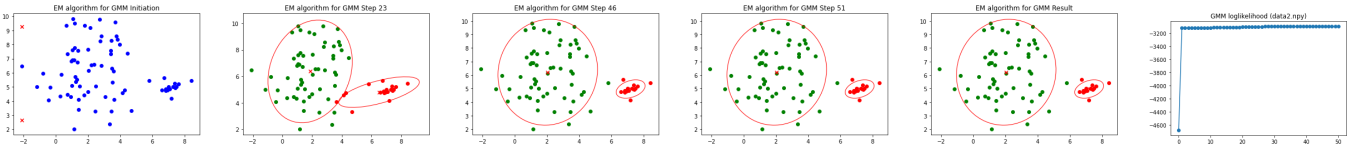
## K-means algorithm case4-1 (left side) on data3.npy



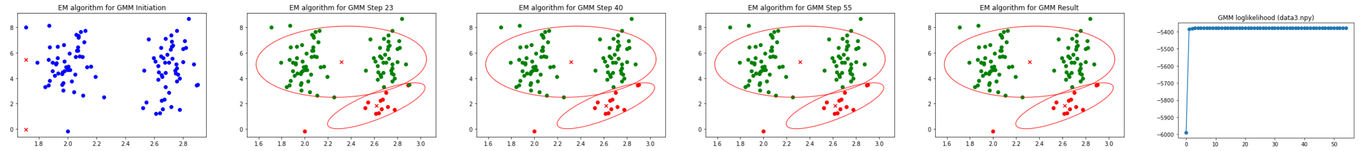
## EM algorithm case4-1 (left side) on data0.npy



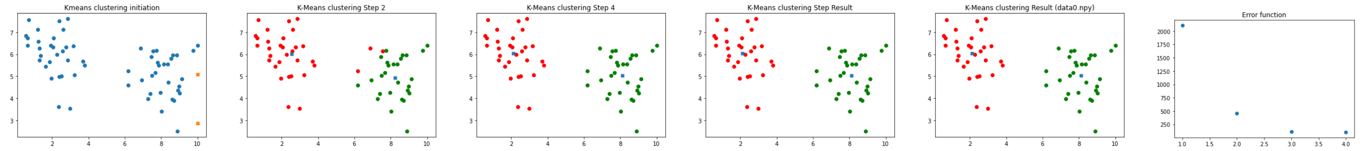
## EM algorithm case4-1 (left side) on data2.npy



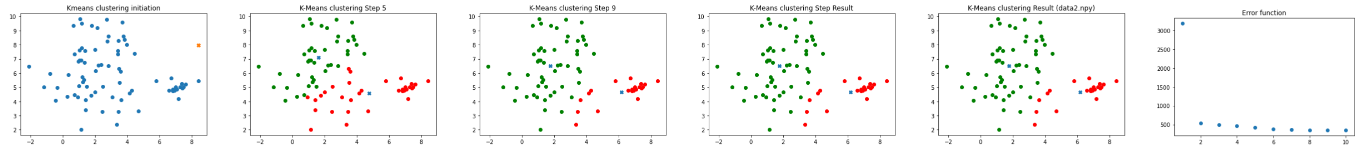
EM algorithm case4-1 (left side) on data3.npy



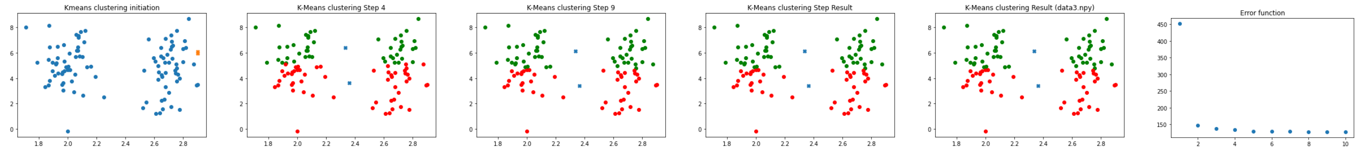
K-means algorithm case4-2 (right side) on data0.npy



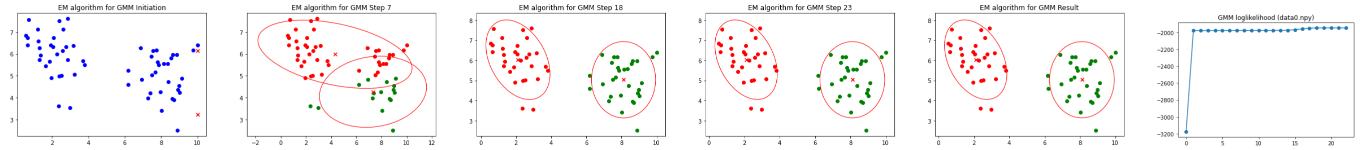
K-means algorithm case4-2 (right side) on data2.npy



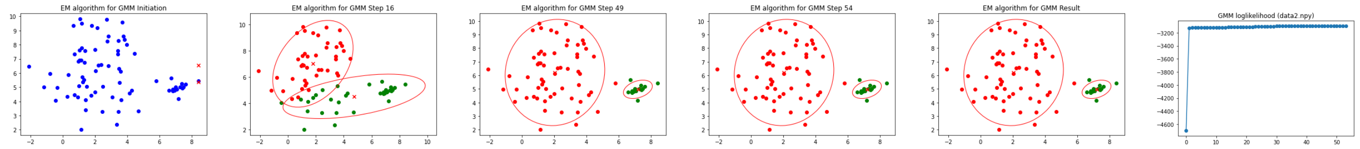
K-means algorithm case4-2 (right side) on data3.npy



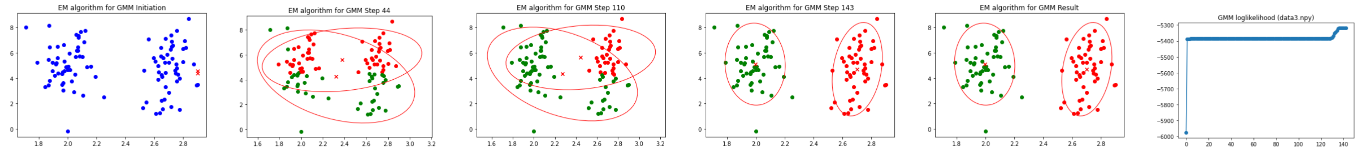
EM algorithm case4-2 (right side) on data0.npy



EM algorithm case4-2 (right side) on data2.npy



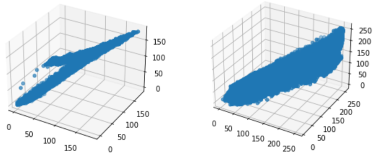
EM algorithm case4-2 (right side) on data3.npy



However I cannot observe the critical effect of starting point. I reviewed the two paper(Choosing starting values for the EM algorithm for getting the highest likelihood in multivariate Gaussian mixture models, initial seed selection algorithm for k-means clustering of georeferenced data to improve replicability of cluster assignments for mapping application), So I guess initiating value of EM algorithm and K-means are important. But I didn't observed this influenced. However I observed effect of initiate point in EM algorithm on data3.npy. I set the initiate point what case 4, set point at right side together. In that case, clustering are not efficiently unlike before setting. But as I hoped, In case 3, can reduce the iteration time compared with case 4 in EM algorithm.

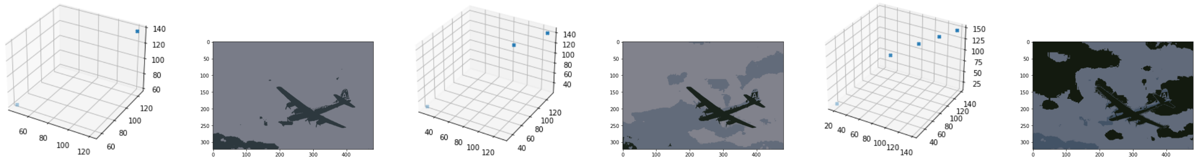
### REPORT4 Image segmentation

For segmentation, I set the limited of iteration to 10 because this task consume the a lot of times. For the task,I get the image R,G,B value of each pixel of image and I assume the these value as coordinate for 3 dimension like this.

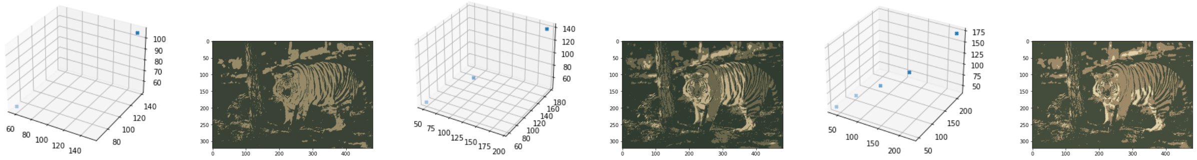


Consequently, K-means and EM algorithms divide the point to K clusters, and points in each cluster is assigned to color for center/mu of their cluster have. This is the result of segmentation of k-means and EM algorithms

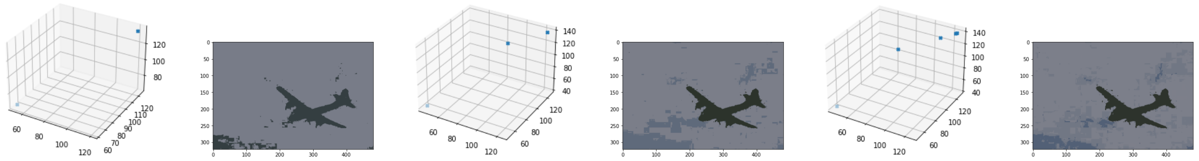
K-Means clusteing Segmentation for plane image



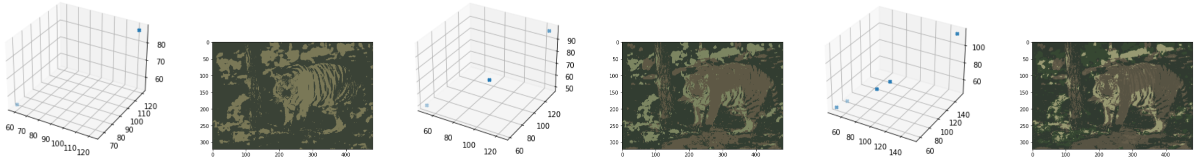
K-Means clusteing Segmentation for tiger image



EM Algorithm Segmentation for plane image

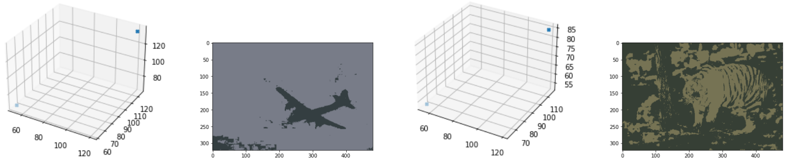


EM Algorithm Segmentation for tiger image



3d point is center/mu point for each cluster and next image is segmentation result image. left side is when K=2, center is when k=3 and right side is when k=5. Unlike the point clustering, I observed k-means is better than EM algorithm for segmentation. And I doubted this result was caused by limited iteration value, because, for point clustering, EM algorithms need more number of iteration for convergence. So I retried the EM algorithm for Segmentation when they reached the convergence.

EM Algorithm Segmentation what iterate until converged



I get the converged segmentation image through the 89, 176 interation for the plane, tiger respectively. However, It seem not much difference with when limit the iteration to 10. And also it seem not efficient compared to K-means algorithms Although EM algorithms consume the more iteration and more time compared to K means algorithm, K-means algorithm is better than EM algorithm for image segmentation.

**Reference**

Christophe Biernacki, Gilles Celeux, Gérard Govaert,Choosing starting values for the EM algorithm for getting the highest likelihood in multivariate Gaussian mixture models,Computational Statistics  Data Analysis,Volume 41, Issues 3–4,2003,Pages 561-575

Fouad Khan,An initial seed selection algorithm for k-means clustering of georeferenced data to improve replicability of cluster assignments for mapping application,Applied Soft Computing,Volume 12, Issue 11,2012,Pages 3698-3700