Nonpar Bayes - HW1

Rongzhao Yan

02/27/2022

1 Overview

In this project, the dirichlet process univariate gaussian mixture model (DPGMM) is implemented, to be specific, the model has the following prior structure:

$$
\begin{aligned}
Y_i|\theta_i &\overset{\text{i.i.d}}{\sim} N(\theta_i, \sigma^2) \\
\theta_i|G &\overset{\text{iid}}{\sim} G \\
G \sim DP(M, G_0), G_0 &= N(m, B) \\
m &\sim N(m_0, D) \\
B &\sim Inv - Gamma(a, b) \\
M &\sim Gamma(c, d).
\end{aligned}
$$

As the prior structure suggested, there are 7 hyper parameters:

$$\{\sigma^2, m_0, D, a, b, c, d\}$$

need to be specified. There are are 7 variables need to be sampled:

$$\{\mathbf{s}, \theta^*, m, B, M, \eta\},$$

where $\mathbf{s}$ is the component membership for each datapoint, θ^* is the center for each component, η is an auxiliary variable.

The gibbs sampler uses fast mixing to integrate out θ_j^* when sampling s the membership for each data point.

`Python` is used for the algorithm implementation. Only `numpy` is used as the framework of scientific computing.

The implementation is encapsulated into a python class: `DPGMM`. To speed up the performance, `numba.jitclass` is used to compile the class `DPGMM` into machine code. We note that some `numpy` functions are not supported to compile into machine code by `numba`, which are then reimplemented by hand.

The python code will attached in the appendix, and the corresponding source code will also be attached as a seperate file.

2 Algorithm Analysis

2.1 Starting Example

We will dive deep into a synthetic data example and see how our model performs.

The datapoints are simulated from three components: $N(5,1), N(-4,1), N(13,1)$, each component has 50 data points. Figure 1 shows the data points with the color their class membership.

The hyperparameter setting in this problem is:

$$\sigma^2 = 1, m_0 = 1, D = 1, a = 1, b = 1, c = 1, d = 1,$$

the MCMC will run 10000 iterations with burn-in period of 2000 and only picking every 5th iteration, resulting in total of 1600 posterior points.

Figure 2 examines the convergence of MCMC. The left two figure shows the ACF of the θ_1 and the population mean m. The right plot shows the 2d density plot of $\theta_1 vs \theta_{150}$ (they are points comming from different component) and that of $\theta_1 vs \theta_2$ (coming from the same component).

We can see that the autocorrelation for the sample is very low. The density plot shows that the MCMC chain has iterated over the entire sample space well, and $\theta_1 vs \theta_1 50$ are not that correlated, and $\theta_1 vs \theta_2$ are highly linear correlated, both of which are under expectation.

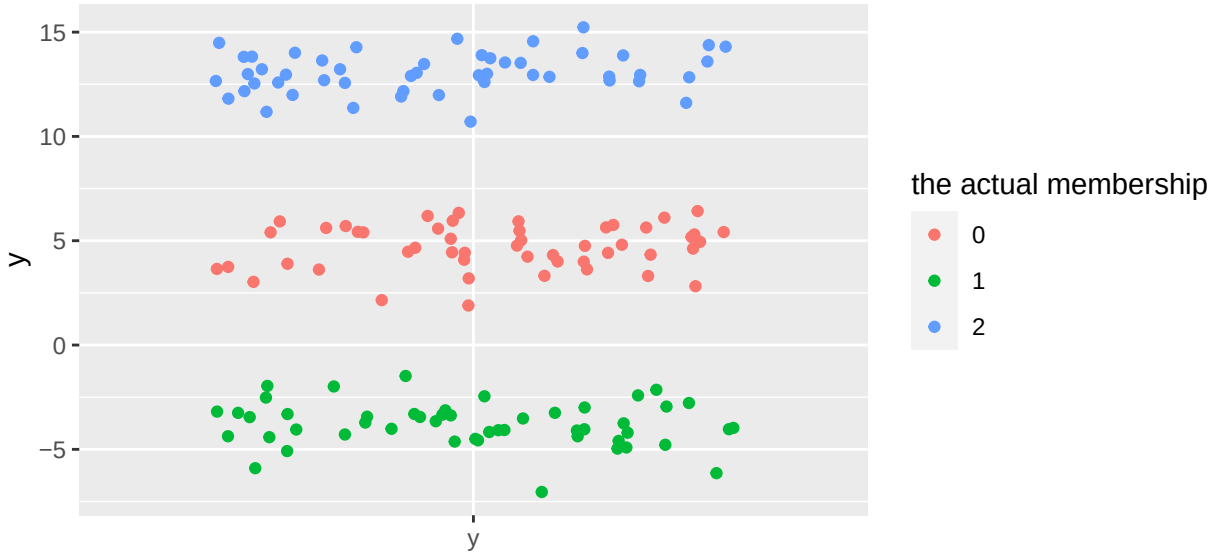

Figure 1: The jittered plot of the synthetic data with color its corresponding membership

Figure 3 shows the mean of the posterior sample of θ_j for each data point. For the left plot, both x axis and y axis is the observed value, resulting in a straight line. For the right plot, the x axis is the mean of the posterior sample of θ_j for each data point, the y axis is the observer value, the color is the actual membership for both plot.

We see that the DPGMM sucessfully finds the "center" of each datapoint in the sense that mean of θ_j for datapoints of the same true component is really close to each other.

Next, we will dive deep into one round of iteration to see how the model cluster the data in a specific round.

Figure 4 shows that in the round 8000, the sampled membership and θ_j of the model. The x axis is the index of each data point, the y axis is the sampled θ_j for each data point, the color of each point corresponding to the **sampled** membership, the shape of each point corresponding to the **actual** membership.

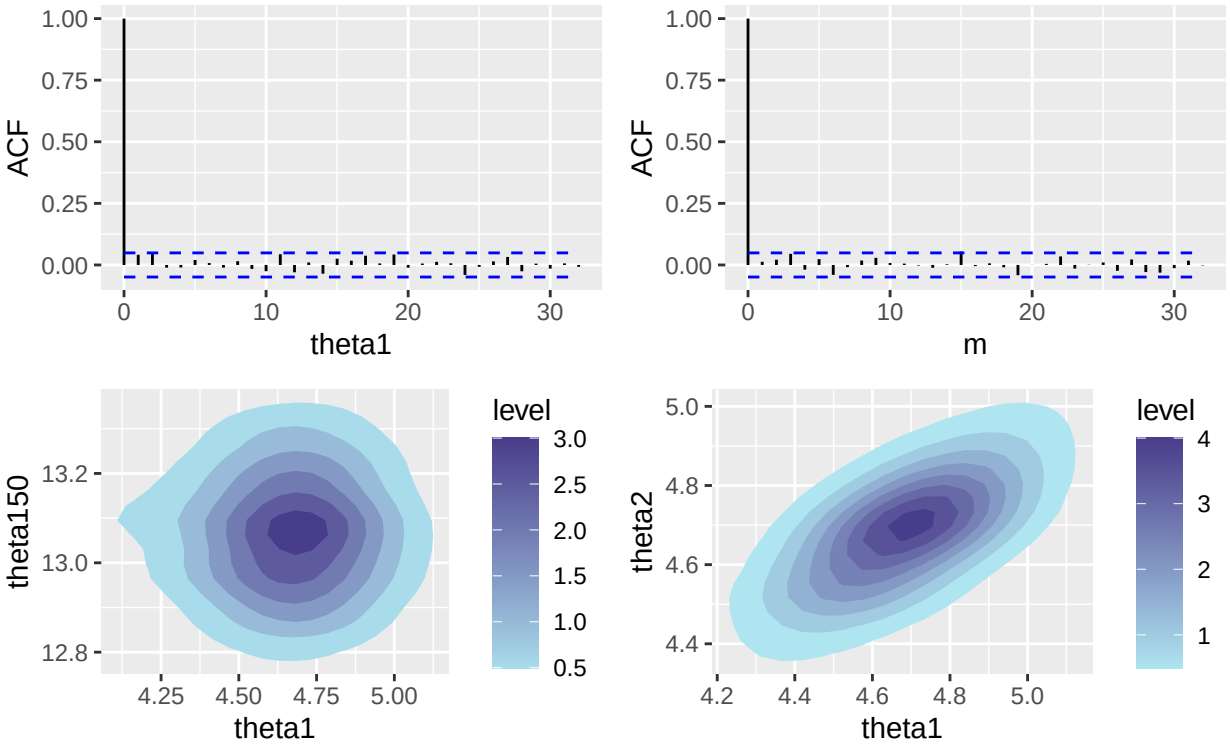

Figure 2: The ACF and 2d density plot of the MC chain

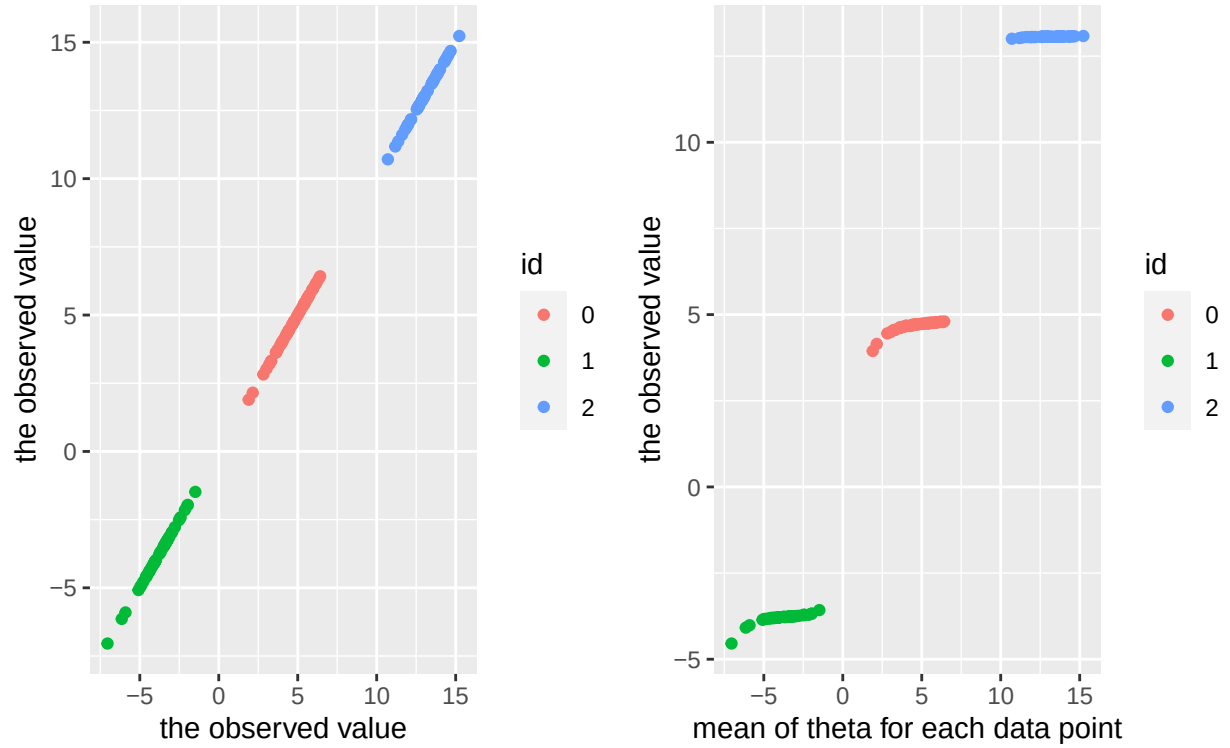

Figure 3: Left: observed value vs observed value, Right: mean of theta vs observed value

We can see that the membership of data points that are actually of component 0 and component 2 (corresponding to shape ∘ and □) are correctly sampled.

But the membership of data points that are actually of component 1 (corresponding to shape △) are not perfectly sampled. We see that there are two very small estimated components consisting of only very few datapoints, whose center (i.e θ_j^*) are really close to the center of (actual) component 1.

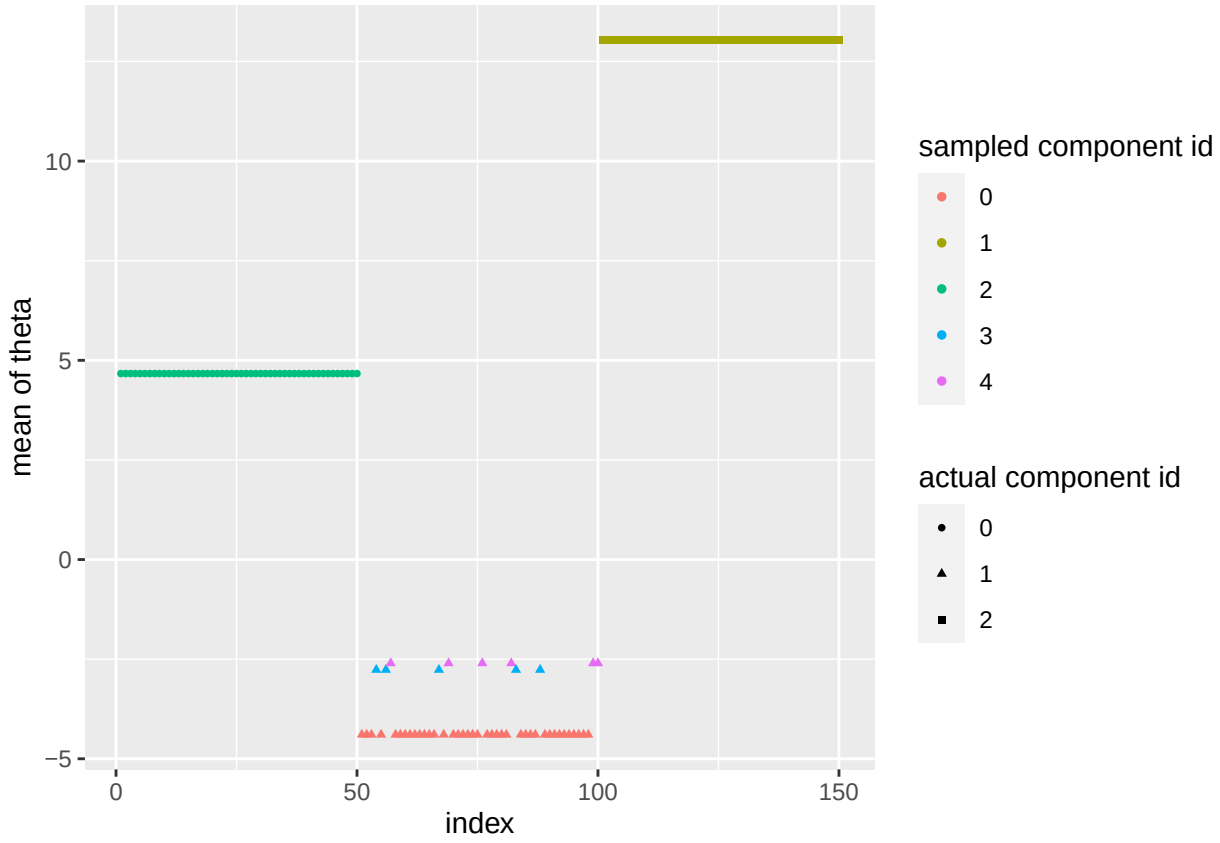

Figure 4: The sampled membership and actuall membership in round 8000

This example shows us that DP has the potential to generate some very small components. Figure 5 shows us the barplot of the number of components for each round of the MCMC. We can see that the mode is 4, and most of the time the sampled number of components will be in $[3, 6]$.

Next We will examine the averaged within clusters variance (WCV) for each k among all iterations, in comparison with the true within cluster variance (the square difference of y and its true center).

$$WCV_k = |\{K^{(j)} = k\}|^{-1} \sum_{\{j:K^{(j)}=k\}} \sum_{i=1}^{n} (y_i - \theta_i^{(j)})^2,$$

where $K^{(j)}$ is the number of componentes in jth iteration.

We see that when the posterior sample yields $k = 3$ the true number of components, the avg-WCV is mostly closed to the true WCV. When k is increasing, the avg-WCV are much smaller than true WCV which is not abnormal as when you increasing the number of clusters, the WCV must decreased.

We see our model has a very good approximation to the underlying truth if the sampled number of clusters is equal to the tue number of clusters.

Finally we examine the averaged co-clustered adjacency matrix heatmap among all iterations.

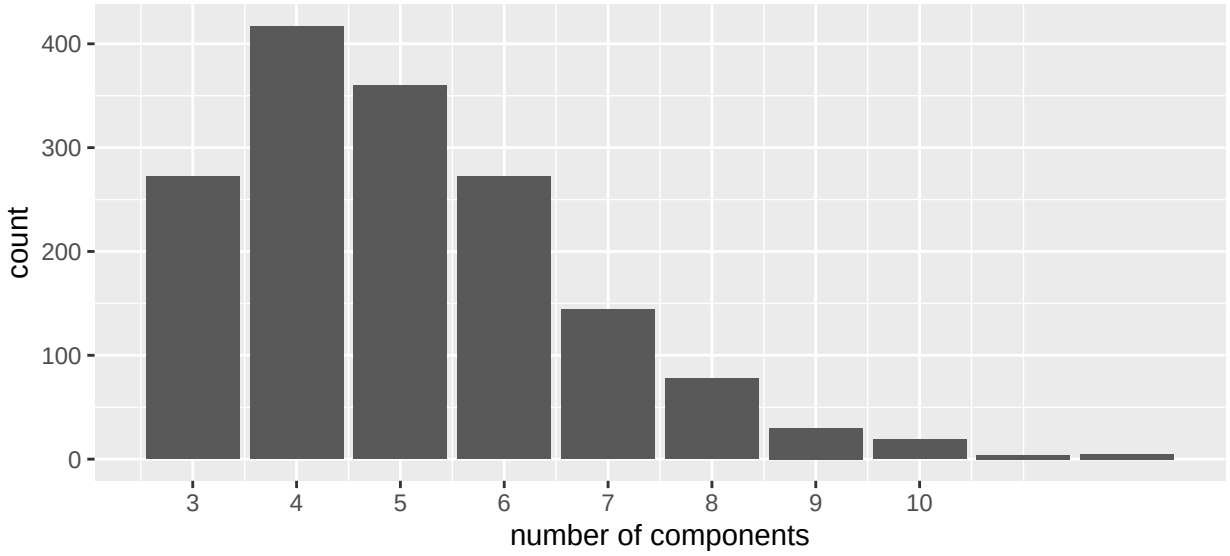

Figure 5: The barplot of the number of sampled membership

Table 1: The average WCV for different number of clusters among all iterations, and the true WCV (using the true center for each data point), and the the proportion of averaged WCV for each k to the true WCV

k	freq	avg_wcv	true_wcv	reduced_proportion
3	272	160.9319	165.0916	0.0251965
4	417	154.2746	165.0916	0.0655213
5	360	148.7135	165.0916	0.0992062
6	272	143.8175	165.0916	0.1288628
7	144	143.4513	165.0916	0.1310808
8	78	139.2238	165.0916	0.1566878
9	30	137.1597	165.0916	0.1691906
10	19	138.8145	165.0916	0.1591667
11	4	124.6804	165.0916	0.2447804
12	5	132.1285	165.0916	0.1996656

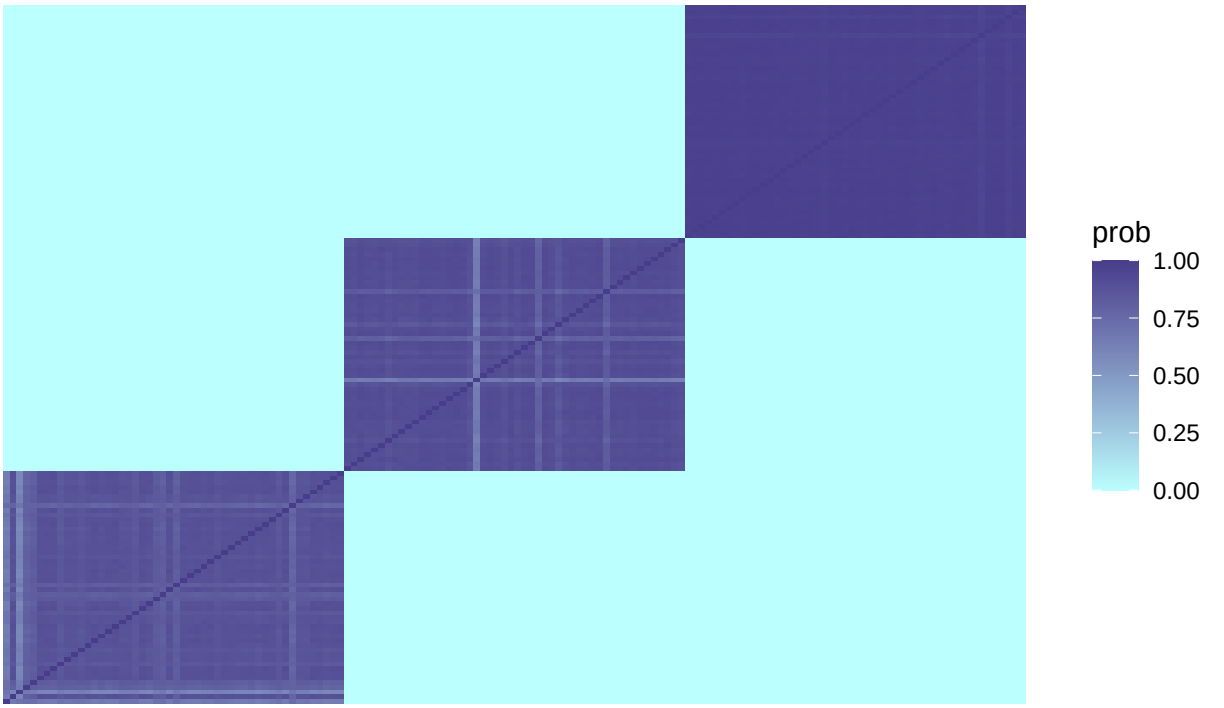

Figure 6: The co-clustered probabilites heatmap among all iterations

2.2 Some Other Scenerios

Above example shows that the model has good performance when number of clusters is small and size of data is medium.

We next examine three different scenerios:

1. there exists one components with small size of data.
2. there are many components in medium size of data.
3. there are many components in large size of data.

In scenerio 1, we assume data coming from 4 centers: $0, -5, 10, 5$, with the first three having number of observations 50 and the last one having number of observations 10.

In scenerio 2, we assume data coming from 10 centers: [-25, 25] with equal distance. The number of observations for each class will be 20.

In scenerio 3, we assume data coming from 10 centers: [-25, 25] with equal distance. The number of observations for each class will be 100.

All the other settings are remained the same as the previous section, including hyperparameters setting and MCMC setting.

Figure 7 shows the same contents of plot as the that with figure 3. We can see that each mean θ_j is largely seperated compared to the observed value. But as the distance between each true center is relatively small in this 3 scenerios compared to previous section (from 9 to 5), we see that there are also some data points have their mean of θ a bit far away from the true center.

We also note that in scenerio 1, the minor class ($\theta_j^* = 5$) actually has relatively good clustering result, only one data point is far away from its true center, the mean θ of others is really close to each other.

Next we examine the co-clustered probabilities heatmap among all iterations in each of the scenerio, see in figure 8.

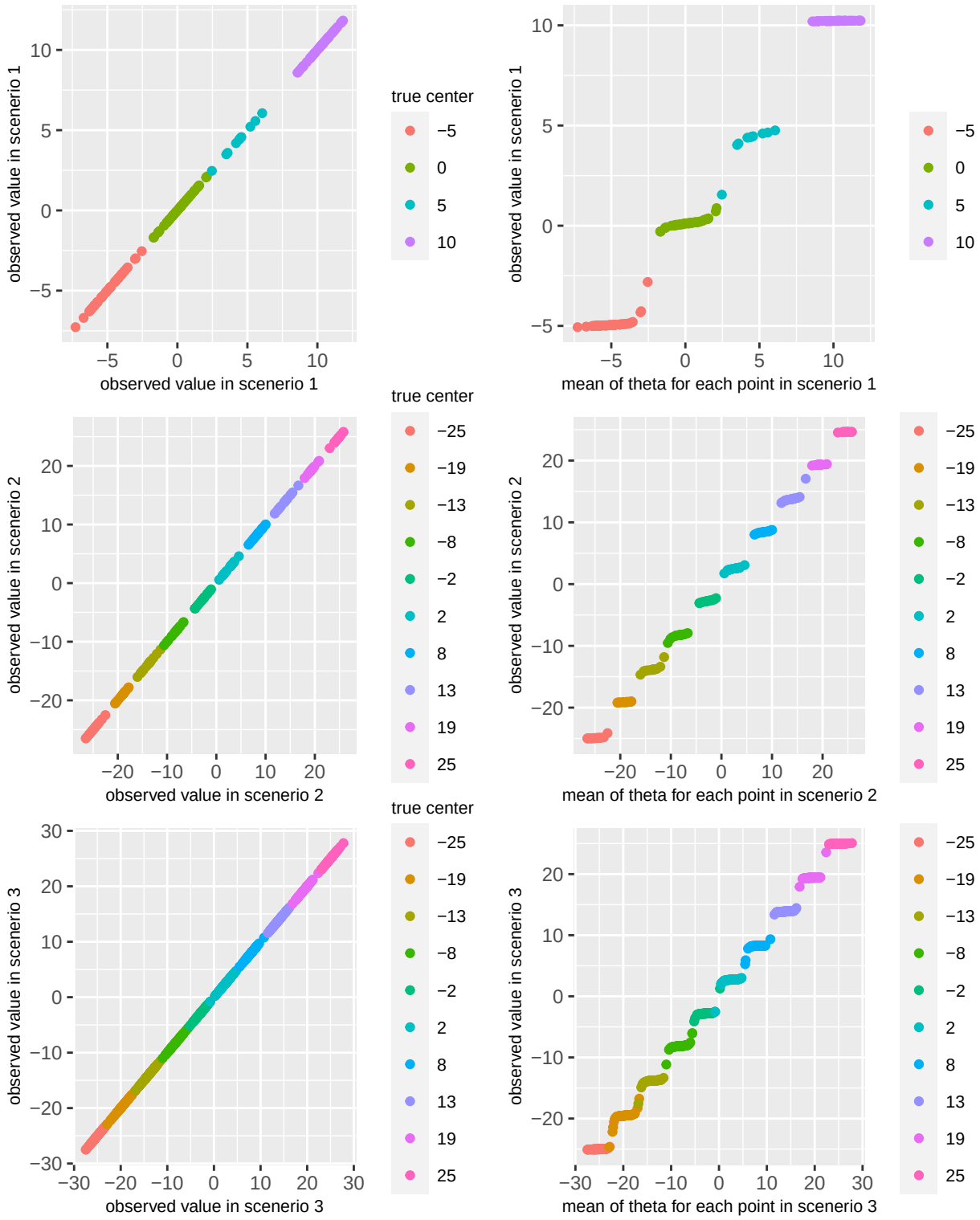

Figure 7: Left: observed value vs observed value, Right: mean of theta vs observed value

We see that in the 3 scenerios the output is relatively good, despite that there are some of the noises. Especially in scenerio 1, there are two points from two componentes are being identified within the same components (and seperated from other points) significantly, we note that one of which coming from the minor class.

In other two scenerios, we see the same situation for only very few points in some components.

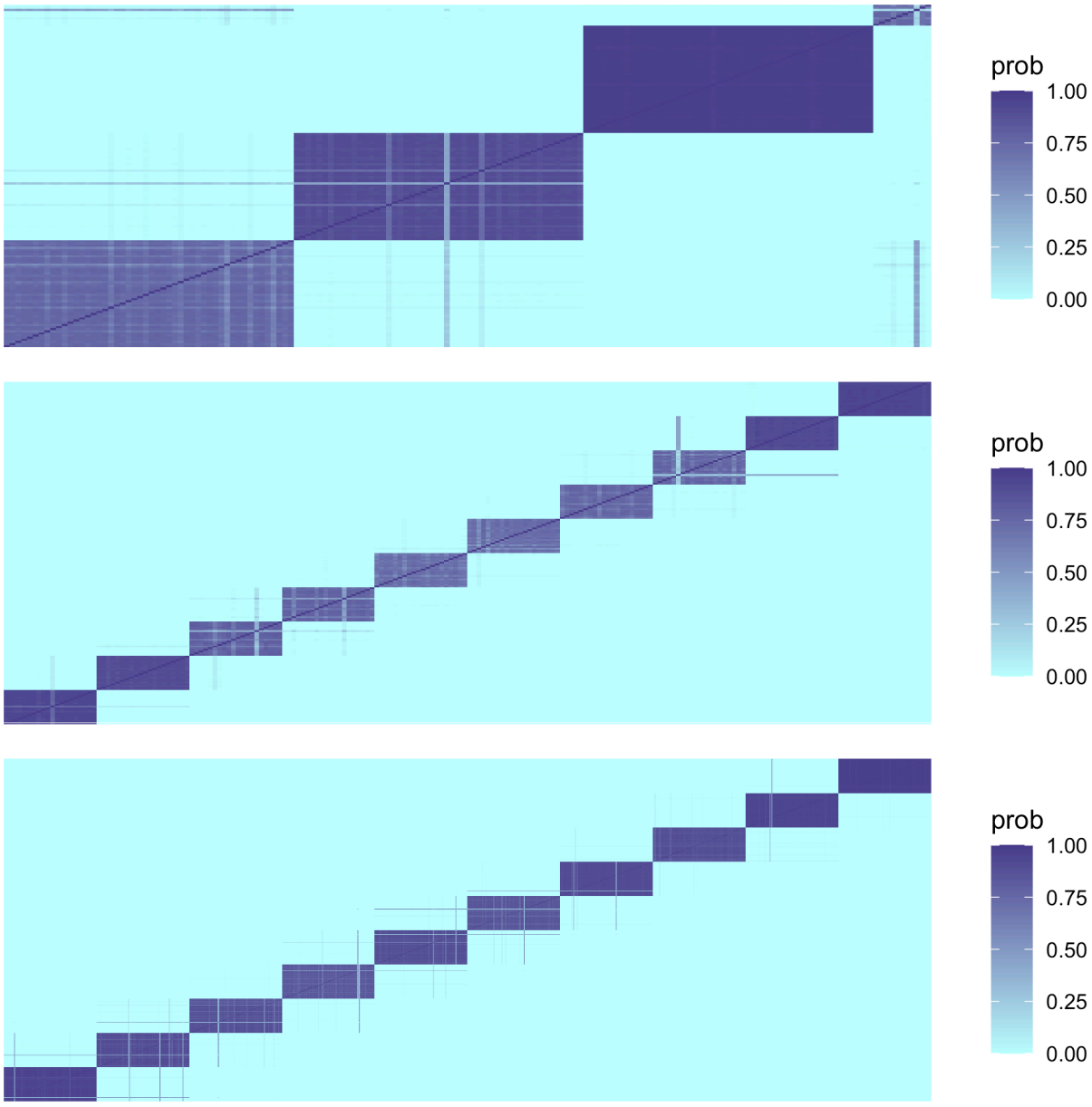

Figure 8: The co-clustered probabilities heatmap among iterations in 3 scenerios