FMM

2023-05-29

The source code can be found in Github under the ‘debugging’ branch.

User-Defined functions

```r
### Function: Summary Quantities "Mean (SD)".
bal_quan <- function(num_vec, rounding = 4){
  mean_val <- round(mean(num_vec), 4)
  sd_val <- round(sd(num_vec), 4)
  paste0(mean_val, " (", sd_val, ")")
}

### Function: Summary from the result
summary_para <- function(result_list){
  ### Collect the data
  n_cluster_vec <- rep(NA, n_para)
  time_vec <- rep(NA, n_para)
  clus_quality <- matrix(NA, ncol = 3, nrow = n_para)

  for(i in 1:n_para){
    n_cluster_vec[i] <- result_list[[i]]$n_cluster
    time_vec[i] <- result_list[[i]]$time
    clus_quality[i, ] <- result_model[[i]]$clus_measure[c(1, 5, 22), 2]
  }

  data.frame(n_cluster = bal_quan(n_cluster_vec), time = bal_quan(time_vec)) %>%
    data.frame(t(apply(clus_quality, 2, bal_quan))) %>%
    kbl(col.names = c("# cluster", "time", "Adjusted Rand", "Jaccard", "VI"))

}

### Function: Calculate mean and variance
mean_var <- function(num_vec){
  c(mean(num_vec), var(num_vec))
}
```

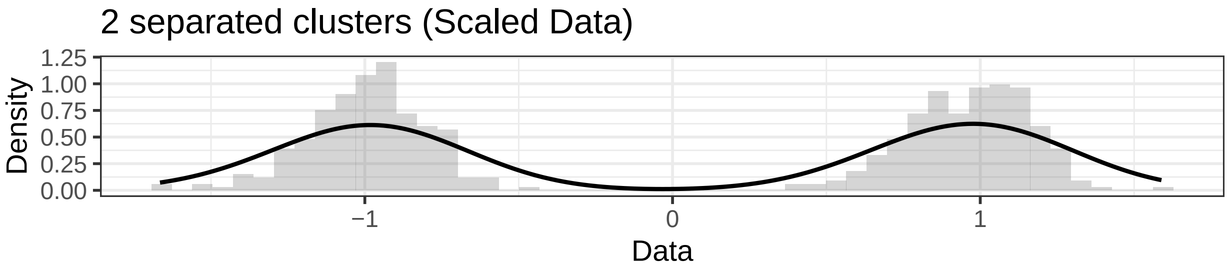
Overall Settings

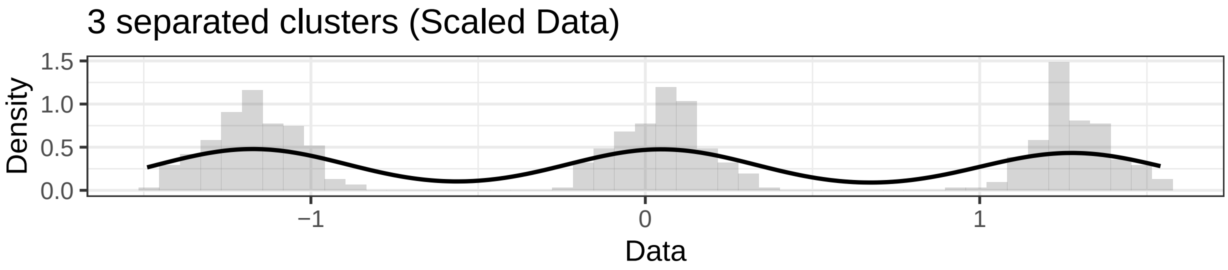
I will run the model for 5,000 iterations for all cases while using the first 2,500 iterations as a burn-in. Also, I will run the model for 10 data sets parallel for each case.

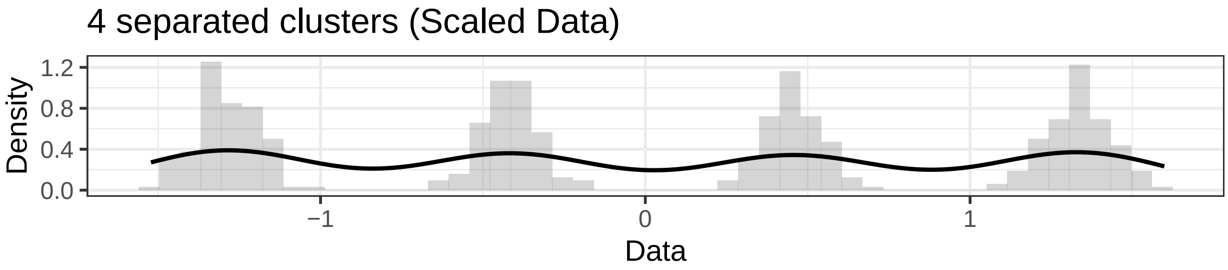
```
iter <- 5000
burn_in <- 2500
overall_seed <- 31807
n_para <- 10
```

Part I: 2 - 4 Separated Clusters

In this part, we will investigate the performance of the model when applied to the data with clearly separated clusters. I will use the model for the data with 2, 3, and 4 separated clusters. The plots below show the standardized data for each case. Also, I have initialized that all observations are in the same cluster.


2 separated clusters (Scaled Data)


3 separated clusters (Scaled Data)


4 separated clusters (Scaled Data)

Note that the hyperparameter for all cases are $\mu = 0, \lambda = a_\sigma = b_\sigma = 1, \xi = 0.1$. Besides, I have set $K_{\max} = 5$ (Default Case).

Result for the two separated clusters.

# cluster	time	Adjusted Rand	Jaccard	VI
2 (0)	15.6827 (2.2656)	1 (0)	1 (0)	0 (0)

Result for the three separated clusters.

# cluster	time	Adjusted Rand	Jaccard	VI
3 (0)	15.7675 (2.1883)	1 (0)	1 (0)	0 (0)

Result for the four separated clusters.

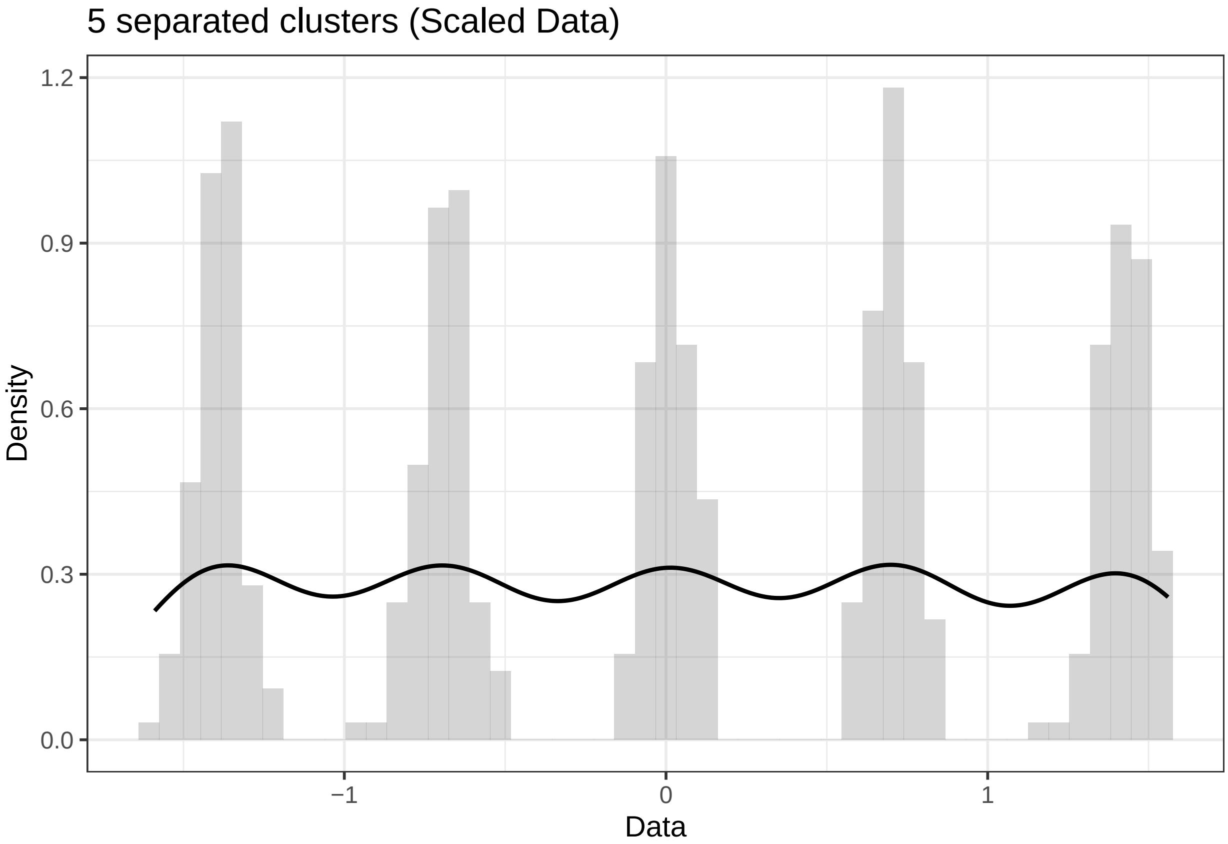
# cluster	time	Adjusted Rand	Jaccard	VI
4 (0)	16.6136 (2.4829)	1 (0)	1 (0)	0 (0)

Comment

The model works well with the set of default hyperparameters in these case.

Part II: 5 Separated Clusters

Below is the plot for the standardized data for five separated clusters. I will change the value for $K_{\max}$ and a_σ for each setting. The other parameters still be the same as the previous part. ($\mu = 0, \lambda = b_\sigma = 1, \xi = 0.1$)



For this case, I have set $K_{\max} = 5$, which is the same as the truth while letting a_σ equals 1. I have initialized that all observations are in the same cluster. The result looks perfect here.

# cluster	time	Adjusted Rand	Jaccard	VI
5 (0)	17.8387 (2.764)	1 (0)	1 (0)	0 (0)

Then, I have changed $K_{\max}$ to 10 instead while keeping a_σ equals 1 (same as before). I have initialized that all observations are in the same cluster. (also, same as before) It turns out that the model can detect that we have only 2 to 3 clusters instead. The run time also increases because the model have to go through every possible cluster (which is 10 in this case, but it was 5 previously.)

# cluster	time	Adjusted Rand	Jaccard	VI
2.7 (0.6749)	25.4889 (3.8541)	0.535 (0.1636)	0.5055 (0.1277)	0.9763 (0.3579)

The result below is $K_{\max}$ to 10 and a_σ equals 1 (same as before). However, I let the model cheated by initilizing the observation by using the correct cluster assignment. The result works better, but it is still bad.

# cluster	time	Adjusted Rand	Jaccard	VI
3.8 (0.6325)	24.1619 (3.4428)	0.7688 (0.1395)	0.7116 (0.116)	0.4654 (0.3109)

However, when I changed a_σ to 10, while still let $K_{\max}$ to 10 (same as before). I have initialized that all observations are in the same cluster. The result looks perfect here.

# cluster	time	Adjusted Rand	Jaccard	VI
5 (0)	24.2006 (3.5203)	1 (0)	1 (0)	0 (0)

Comment

By increasing the value of a_σ, we are using a stronger prior belief that the variances of the components are small, which leads to more concentrated and compact clusters, making it less likely for the model to assign observations to separate components and preferring a smaller number of active clusters.

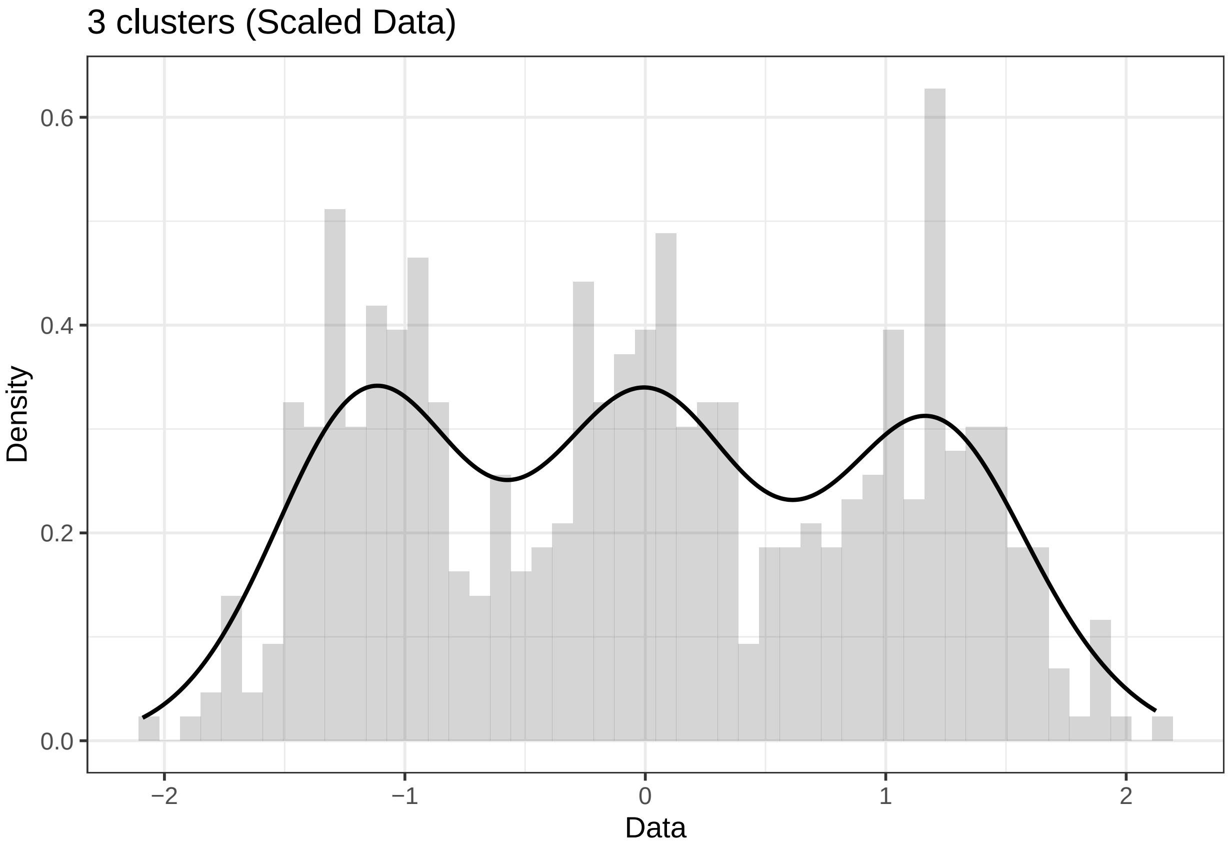
$$E\left[\sigma_k^2\right] = \frac{b_\sigma}{(a_\sigma - 1)} Var\left[\sigma_k^2\right] = \frac{b_\sigma^2}{(a_\sigma - 2)(a_\sigma - 1)^2}$$

In other words, I think this can be acted as one type of regularization, preventing the model from creating a new cluster unless there is strong evidence to support it.

Also, I think that increasing a_σ will have an effect on the posterior distribution. By setting a larger value for a_σ, the prior belief becomes more influential in determining the posterior distribution, leading to a bias towards smaller variances and more concentrated clusters.

Part III: 3 Mixing clusters

Below is the plot for the standardized data for three mixing clusters. I will change the value for λ while I keep the other parameters still be fixed. ($\mu = 0, a_\sigma = b_\sigma = 1, \xi = 0.1$)



This is the result when $\lambda = 1$ (Default Case)

# cluster	time	Adjusted Rand	Jaccard	VI
2.5 (0.527)	17.5091 (2.6257)	0.6869 (0.1735)	0.6894 (0.1375)	0.8147 (0.2271)

This is the result when $\lambda = 0.1$ (lowering λ)

# cluster	time	Adjusted Rand	Jaccard	VI
2.7 (0.483)	17.6616 (2.6677)	0.747 (0.1464)	0.7346 (0.1143)	0.7414 (0.1533)

This is the result when $\lambda = 10$ (increasing λ)

# cluster	time	Adjusted Rand	Jaccard	VI
1.6 (0.5164)	17.048 (2.6091)	0.2876 (0.2482)	0.4479 (0.0997)	1.344 (0.2083)

Comment

We might notice that when the $\lambda = 0.1$, the result looks better that the default case, while $\lambda = 10$ provides the worst result among these three cases. Based on our model, λ controls the precision of the prior belief about the mean of the components. When λ is small, it implies a weaker prior belief in tightly concentrated means, allowing for more variability in the positioning of the cluster centers. Therefore, the model is more likely to assign data points to overlapping or mixed clusters.