Sparse Finite Discrete Mixture Model

2023-06-12

Updated Note (6/13/2023)

- The code is in the main branch on Github.
- For the empty clusters, we will not sample μ_k and σ_k^2.
- Fix the constraint on μ. I will set $\mu_{(1)} > \mu_{(2)} > \cdots > \mu_{(K_{max})}$ instead.
- Remove the trace plot for μ_k and σ_k^2. Instead, I have included the trace plot for the number of the active cluster.
- Include the another setting of analysis, the non-separated 5 cluster setting.

Note

- I will use the same datasets as in the previous reports. (FMM - R; 6/8/2023 and FMM - Rcpp; 6/10/2023)
- Based on the comment, I will run 10,000 iterations in total, but I will let the first 5,000 iterations as a burn-in.

Model

The derivation for the posterior parameters is in `derive_fmm.jpeg`.

$$
\begin{aligned}
Y_i | c_i = k, \mu, \sigma^2 &\sim \mathrm{N}\left(\mu_k, \sigma_k^2\right) \\
\mu_k &\sim \mathrm{N}\left(\mu_0, \sigma_0^2\right) \\
\sigma_k^2 &\sim \text{Inv-Gamma}\left(a_s, b_s\right) \\
c_i | \alpha &\sim \text{Multinomial}\left(1, \frac{\alpha}{\sum_{k=1}^{K+} \alpha_k}\right) \\
\alpha_k | \phi_k &\sim \phi_k \text{Gamma}\left(1, \xi_i\right) + \left(1 - \phi_k\right) \delta_0 \\
\phi_k &\sim \text{Ber}\left(\theta\right) \\
\theta &\sim \text{Beta}\left(a_\theta, b_\theta\right)
\end{aligned}
$$

Hyperparameters

- According to the model, all clusters will have the same hyperparameters $\left(\mu_0, \sigma_0^2, a_s, b_s\right)$.
- To use the noninformative prior, I will let $\mu_0 = 0$, $\sigma_0^2 = 100$, $a_s = b_s = 1$. Also, I will let $\xi_1 = \xi_2 = \cdots = \xi_K = 1$.
- I have set $a_\theta = b_\theta = 1$.
- Also, I have set the maximum possible number of clusters ($\mathrm{K_{max}}$) to 10.
- The other model's setting is the number of the launch step. I have set it to 10.

Procedure

- In this model, we will have three steps: reallocation, split-merge and parameters update.
- Reallocation step: This is similar to the finite mixture model. The difference is that we will not allow the observations to go to the new cluster. We will reallocate them to the already existing clusters only. So, there will not create any new cluster in this step. The number of the active cluster will be the same or decrease only. At the end of this step, I will update μ and σ^2. Also, I perform the relabelling in this step to prevent the label-switching issue.
- SM: For the split-merge procedure, I will expand/collapse the cluster space via the split-merge algorithm. The algorithm is followed the restricted Gibbs Sampling Proposals From a Random Launch State (Section 3.2). In this step, I have updated the parameters for the cluster (μ and σ^2) at the end of every launch iteration.
- Parameters update: I update α (based on the Dirichlet Data Augmentation Trick document), μ and σ^2. Also, I perform the relabelling in this step to prevent the label-switching issue.

Analysis

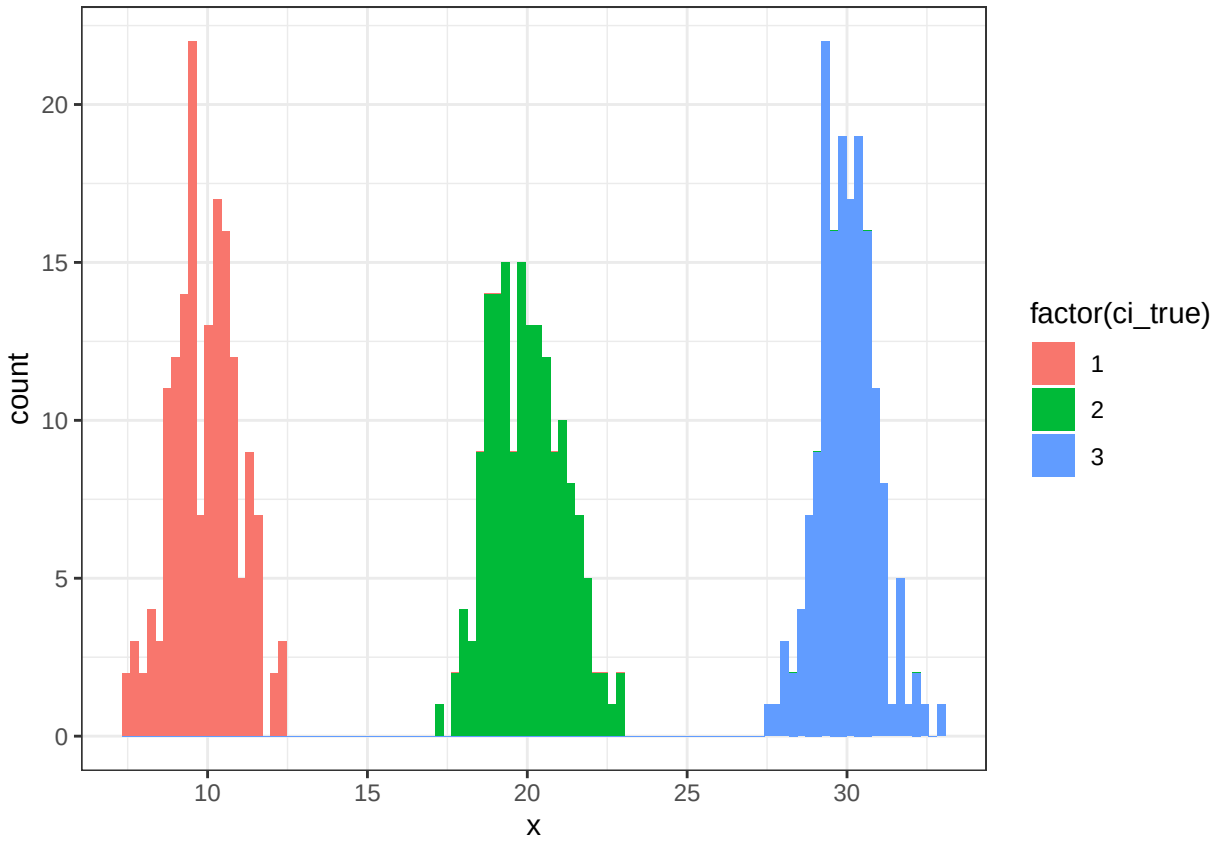
For each cases, I will run the model for the one simulated dataset only.

(1)

This is the scenario that we discuss during the one of our meeting.

```
rm(list = ls())

### Data Simulation: (1)
set.seed(1843)
N <- 500
K <- 3
ci_true <- sample(1:K, N, replace = TRUE)
dat_sim <- rnorm(N, c(10, 20, 30)[ci_true], 1)
ggplot(data.frame(x = dat_sim, ci_true), aes(x = x, fill = factor(ci_true))) +
  geom_histogram(bins = 100) +
  theme_bw()
```



First, I will run the model for 10,000 iterations.

```
start_time <- Sys.time()
test_result <- SFDMM_model(iter = 10000, K_max = 10, init_assign = rep(0, 500),
                           y = dat_sim, a0 = 1, b0 = 1, mu0 = 0, s20 = 100,
                           xi0 = 1, a_theta = 1, b_theta = 1, launch_iter = 10,
                           print_iter = 10000)
model_time <- Sys.time() - start_time
```

```
model_time
```

```
## Time difference of 51.04499 secs
```

The result looks perfect.

```
table(salso(test_result$iter_assign[-c(1:7500), ], maxNClusters = 10), ci_true)
```

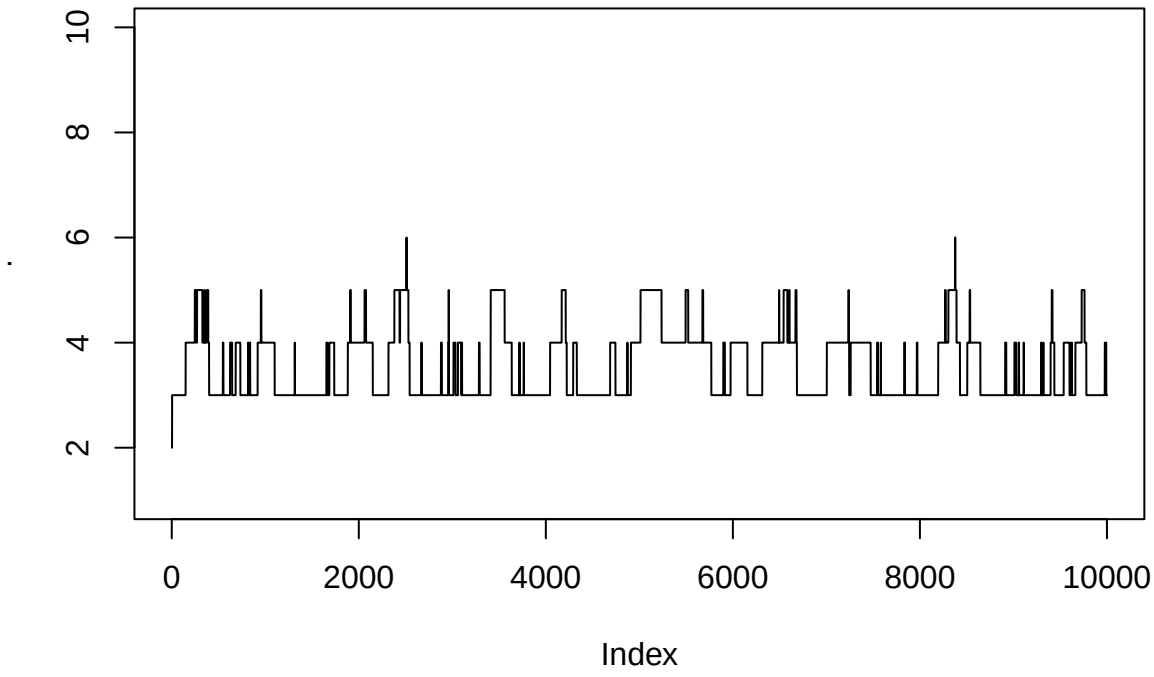
```
##    ci_true
##       1   2   3
##   1   0 170   0
##   2   0   0 166
##   3 164   0   0
```

Consider the trace plot for the number of active cluster.

```
n_active <- apply(test_result$iter_assign, 1, function(x){length(unique(x))})
table(n_active)
```

```
## n_active
##    2    3    4    5    6
##    3 5738 3334  917    8
```

```
apply(test_result$iter_assign, 1, function(x){length(unique(x))}) %>%
  plot(type = "l", ylim = c(1, 10))
```



Most of the time, the model detect that the number of active clusters is 3 or 4.

```
c(mean(n_active), sd(n_active))
```

```
## [1] 3.5189000 0.6621832
```

Then, I will look at the acceptance rate of the MH-algorithm in the split-merge.

```
ac <- factor(test_result$sm_status)
levels(ac) <- c("Reject", "Accept")
sm <- factor(test_result$split_or_merge)
levels(sm) <- c("Merge", "Split")
table(ac, sm)
```

```
##         sm
## ac       Merge Split
##   Reject  6860  3060
##   Accept     1    79
```

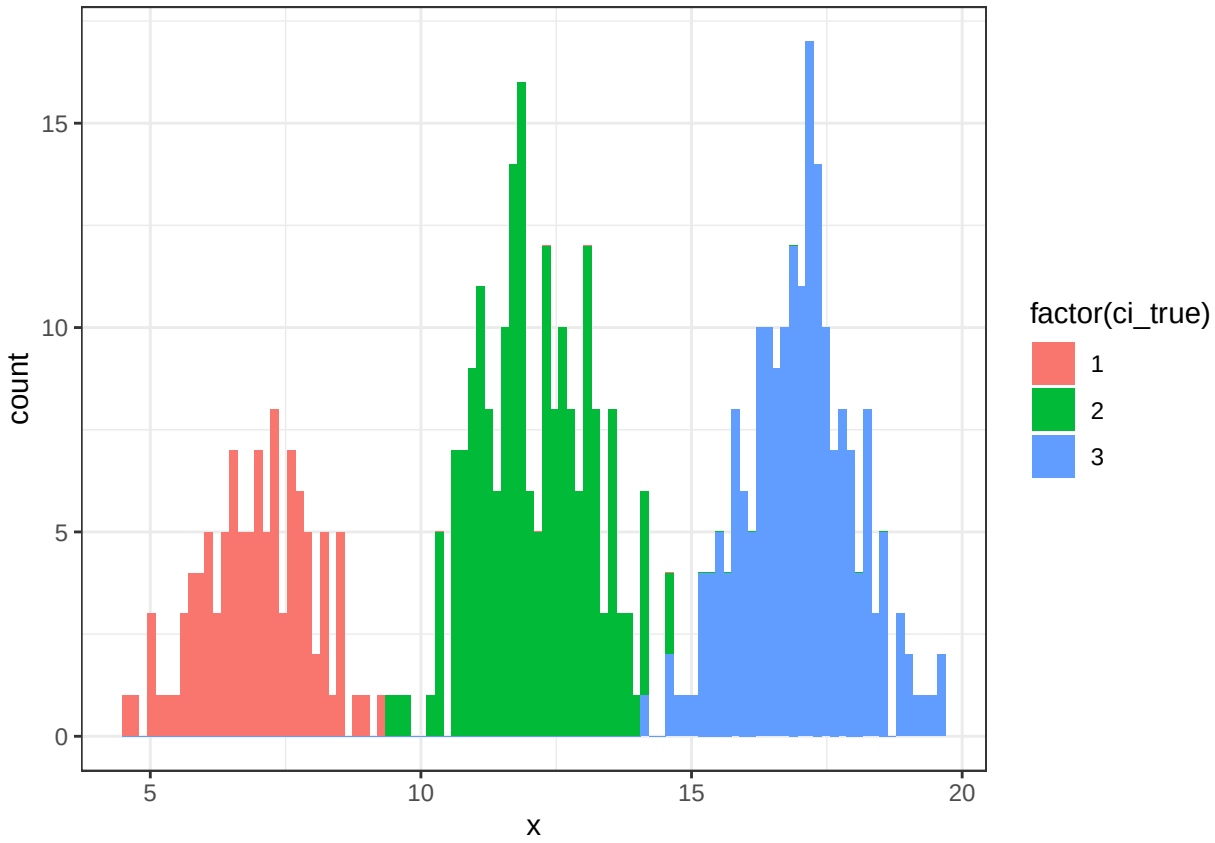
The acceptance rate is extremely low (80/10000 = 0.80%)

(2)

For this case, we will have three (almost) separated clusters. The proportion for each group is 0.25, 0.35, and 0.4

```
rm(list = ls())

### Data Simulation: (2)
set.seed(12441)
N <- 500
K <- 3
ci_true <- sample(1:K, N, replace = TRUE, prob = c(0.25, 0.35, 0.4))
dat_sim <- rnorm(N, c(7, 12, 17)[ci_true], 1)
ggplot(data.frame(x = dat_sim, ci_true), aes(x = x, fill = factor(ci_true))) +
  geom_histogram(bins = 100) +
  theme_bw()
```



Then, I will run the model for 10,000 iterations.

```
start_time <- Sys.time()
test_result <- SFDMM_model(iter = 10000, K_max = 10, init_assign = rep(0, 500),
                           y = dat_sim, a0 = 1, b0 = 1, mu0 = 0, s20 = 100,
                           xi0 = 1, a_theta = 1, b_theta = 1, launch_iter = 10,
                           print_iter = 10000)
model_time <- Sys.time() - start_time
```

```
model_time
```

```
## Time difference of 51.18652 secs
```

```
table(salso(test_result$iter_assign[-c(1:7500), ], maxNClusters = 10), ci_true)
```

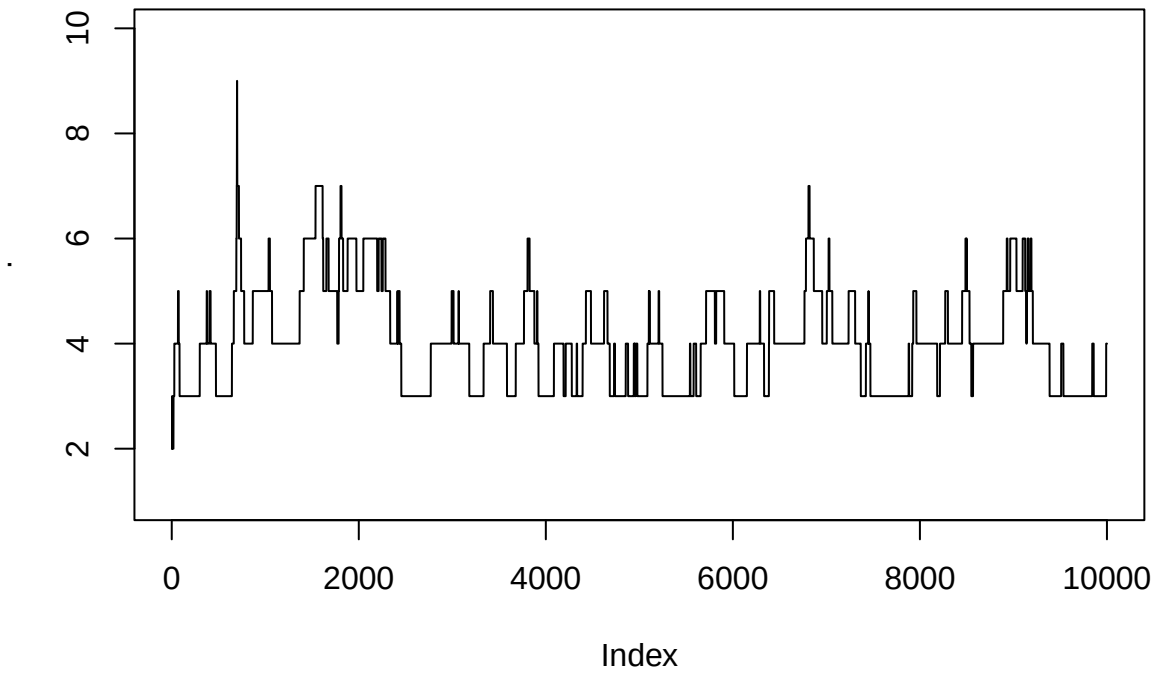
```
##    ci_true
##       1   2   3
##   1   0 195   1
##   2   0   1 196
##   3 106   1   0
```

Consider the trace plot for the number of active cluster.

```
n_active <- apply(test_result$iter_assign, 1, function(x){length(unique(x))})
table(n_active)
```

```
## n_active
##    2    3    4    5    6    7    8    9
##   16 3276 4118 1705  762  119    2    2
```

```
apply(test_result$iter_assign, 1, function(x){length(unique(x))}) %>%
  plot(type = "l", ylim = c(1, 10))
```



Most of the time, the model detect that the number of active clusters is 3, 4 or 5.

```
c(mean(n_active), sd(n_active))
```

```
## [1] 4.0296000 0.9611536
```

Then, I will look at the acceptance rate of the MH-algorithm in the split-merge.

```
ac <- factor(test_result$sm_status)
levels(ac) <- c("Reject", "Accept")
sm <- factor(test_result$split_or_merge)
levels(sm) <- c("Merge", "Split")
table(ac, sm)
```

```
##         sm
## ac       Merge Split
##   Reject  6883  3026
##   Accept     1    90
```

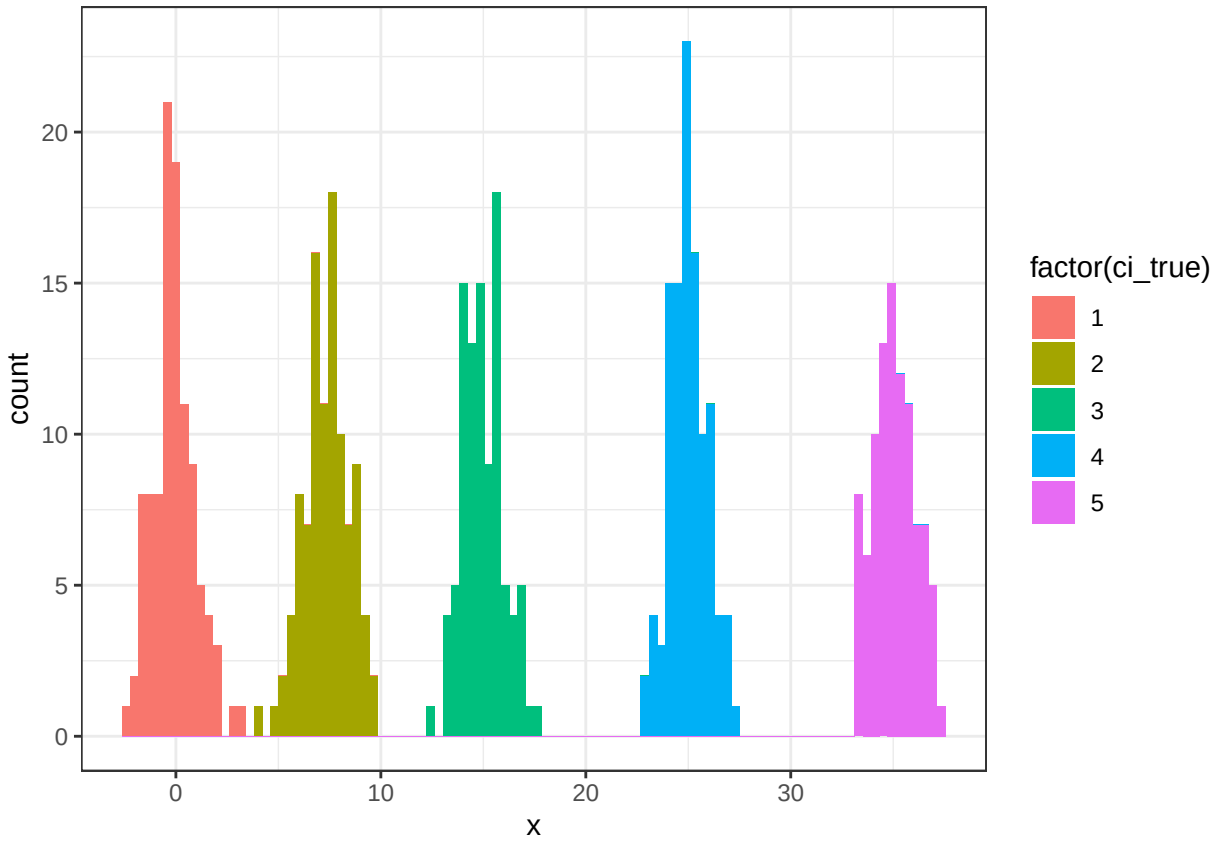
The acceptance rate is extremely low (91/10000 = 0.91%)

(3)

For this case, we will have five separated clusters.

```
rm(list = ls())

### Data Simulation: (3)
set.seed(12441)
N <- 500
K <- 5
ci_true <- sample(1:K, N, replace = TRUE)
dat_sim <- rnorm(N, c(0, 7.5, 15, 25, 35)[ci_true], 1)
ggplot(data.frame(x = dat_sim, ci_true), aes(x = x, fill = factor(ci_true))) +
  geom_histogram(bins = 100) +
  theme_bw()
```



I will run the model for 10,000 iterations.

```
start_time <- Sys.time()
test_result <- SFDMM_model(iter = 10000, K_max = 10, init_assign = rep(0, 500),
                           y = dat_sim, a0 = 1, b0 = 1, mu0 = 0, s20 = 100,
                           xi0 = 1, a_theta = 1, b_theta = 1, launch_iter = 10,
                           print_iter = 10000)
model_time <- Sys.time() - start_time
```

```
model_time
```

```
## Time difference of 40.79258 secs
```

The result looks perfect.

```
table(salso(test_result$iter_assign[-c(1:7500), ], maxNClusters = 10), ci_true)
```

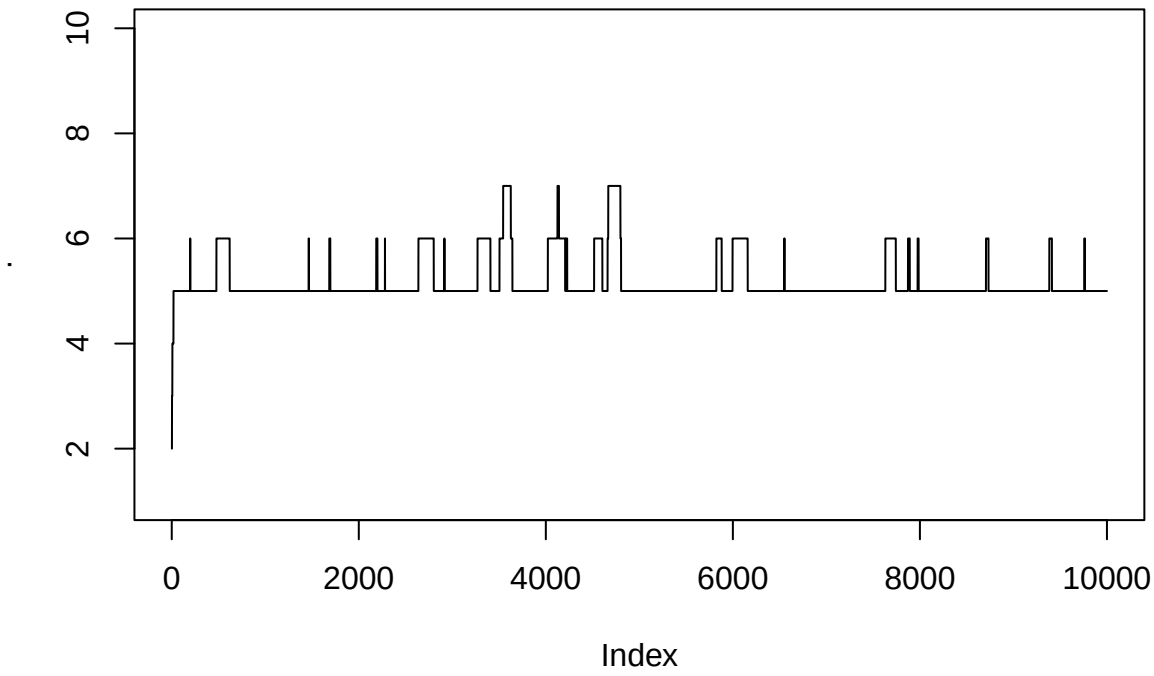
```
##    ci_true
##       1   2   3   4   5
##   1   0   0   0 108   0
##   2   0 100   0   0   0
##   3   0   0   0   0  95
##   4 101   0   0   0   0
##   5   0   0  96   0   0
```

Consider the trace plot for the number of active cluster.

```
n_active <- apply(test_result$iter_assign, 1, function(x){length(unique(x))})
table(n_active)
```

```
## n_active
##    2    3    4    5    6    7
##    2    4   12 8507 1247  228
```

```
apply(test_result$iter_assign, 1, function(x){length(unique(x))}) %>%
  plot(type = "l", ylim = c(1, 10))
```



Most of the time, the model detect that the number of active clusters is 5.

```r
c(mean(n_active), sd(n_active))
```

```
## [1] 5.1677000 0.4386296
```

Then, I will look at the acceptance rate of the MH-algorithm in the split-merge.

```r
ac <- factor(test_result$sm_status)
levels(ac) <- c("Reject", "Accept")
sm <- factor(test_result$split_or_merge)
levels(sm) <- c("Merge", "Split")
table(ac, sm)
```

```
##         sm
## ac       Merge Split
##   Reject  8013  1934
##   Accept     3    50
```

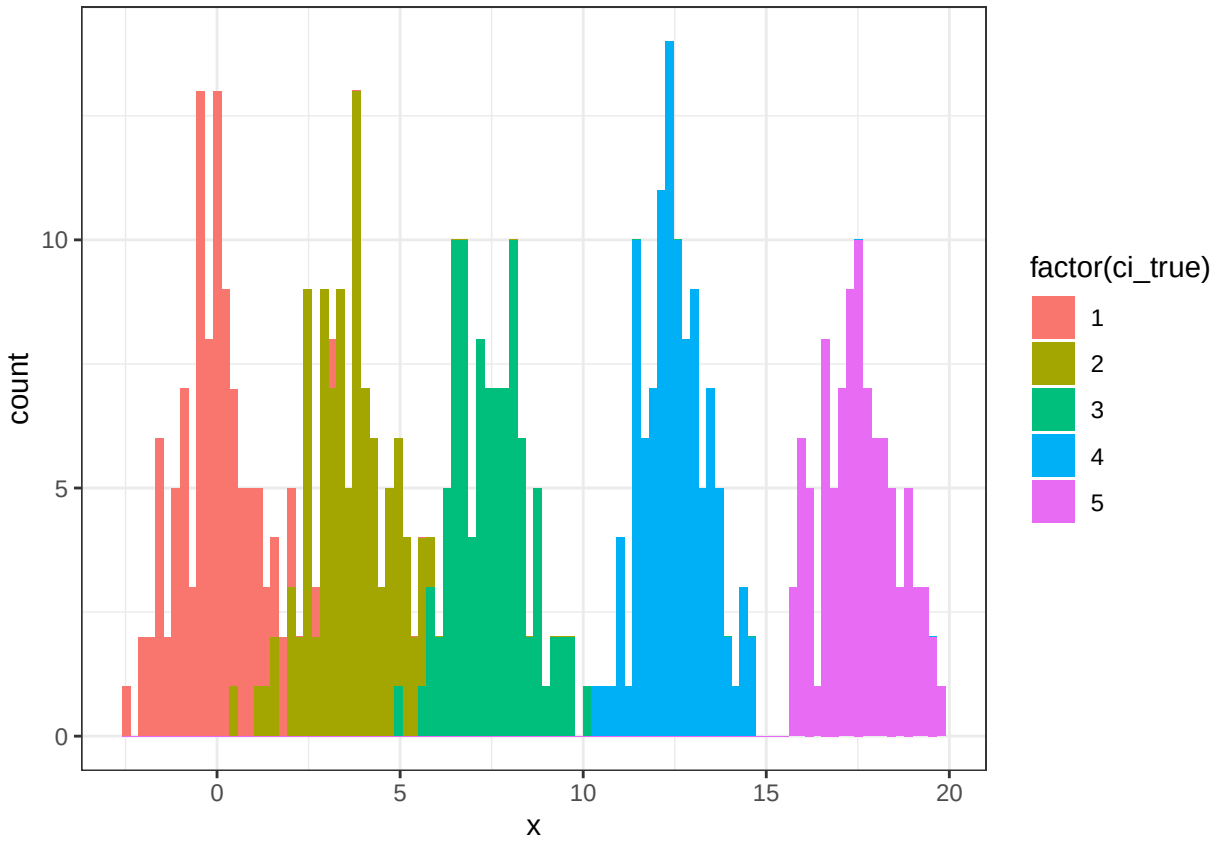
The acceptance rate is extremely low ($53/10000 = 0.53\%$)

(4)

For this case, we will have five non-separated clusters.

```r
rm(list = ls())

### Data Simulation: (4)
set.seed(12441)
N <- 500
K <- 5
ci_true <- sample(1:K, N, replace = TRUE)
dat_sim <- rnorm(N, (c(0, 7.5, 15, 25, 35)[ci_true])/2, 1)
ggplot(data.frame(x = dat_sim, ci_true), aes(x = x, fill = factor(ci_true))) +
  geom_histogram(bins = 100) +
  theme_bw()
```



I will run the model for 10,000 iterations.

```
start_time <- Sys.time()
test_result <- SFDMM_model(iter = 10000, K_max = 10, init_assign = rep(0, 500),
                           y = dat_sim, a0 = 1, b0 = 1, mu0 = 0, s20 = 100,
                           xi0 = 1, a_theta = 1, b_theta = 1, launch_iter = 10,
                           print_iter = 10000)
model_time <- Sys.time() - start_time
```

```
model_time
```

```
## Time difference of 42.15335 secs
```

The result looks good.

```
table(salso(test_result$iter_assign[-c(1:7500), ], maxNClusters = 10), ci_true)
```

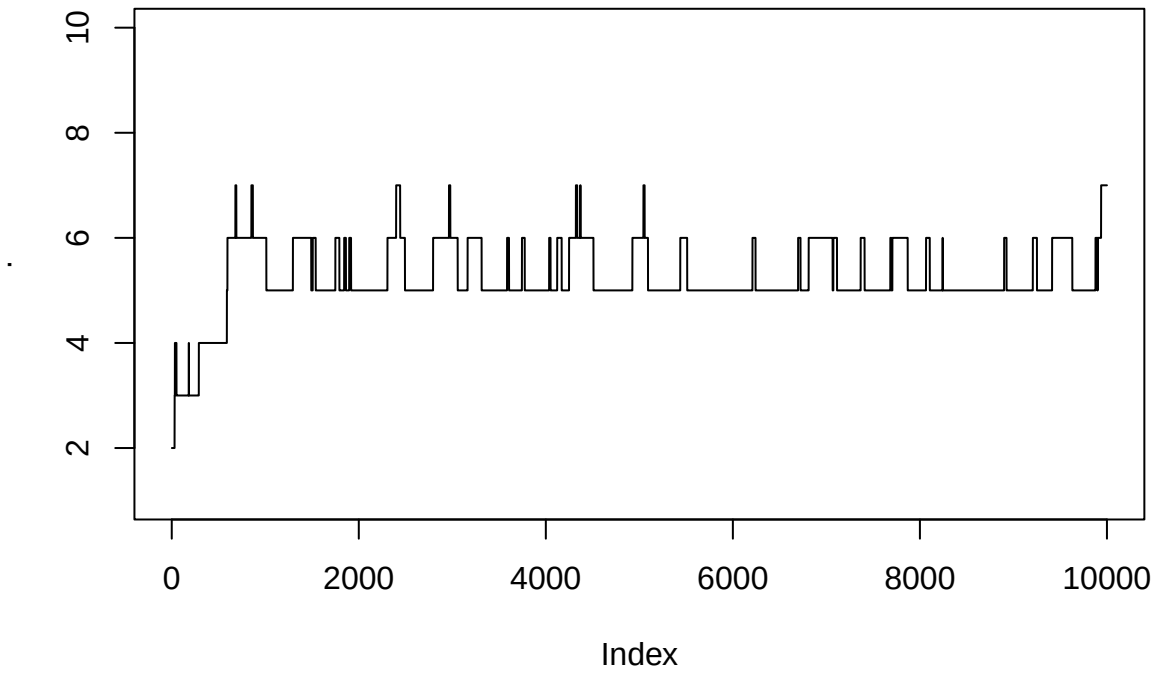
```
##    ci_true
##       1   2   3   4   5
##   1   0   0   0 108   0
##   2   7  92   1   0   0
##   3   0   0   0   0  95
##   4  94   3   0   0   0
##   5   0   5  95   0   0
```

Consider the trace plot for the number of active cluster.

```
n_active <- apply(test_result$iter_assign, 1, function(x){length(unique(x))})
table(n_active)
```

```
## n_active
##    2    3    4    5    6    7
##   30  241  318 6451 2783  177
```

```
apply(test_result$iter_assign, 1, function(x){length(unique(x))}) %>%
  plot(type = "l", ylim = c(1, 10))
```



Most of the time, the model detect that the number of active clusters is 5 or 6.

```
c(mean(n_active), sd(n_active))
```

```
## [1] 5.2247000 0.6736878
```

Then, I will look at the acceptance rate of the MH-algorithm in the split-merge.

```
ac <- factor(test_result$sm_status)
levels(ac) <- c("Reject", "Accept")
sm <- factor(test_result$split_or_merge)
levels(sm) <- c("Merge", "Split")
table(ac, sm)
```

```
##         sm
## ac       Merge Split
##   Reject  7910  2040
##   Accept     0    50
```

The acceptance rate is extremely low (50/10000 = 0.50%)