Incorporating Prior Information into Bayesian Mixture Models

Alex Dombowsky

2022-06-06

Introduction

In this document, we will compare the performance of two different clustering methods based on prior information using a simple example. In both cases, we will fit a finite mixture of univariate normal distributions for $K = 3$ components. That is, for all $i = 1, \ldots, n$ and $h = 1, \ldots, K$,

$$y_i \mid c_i = h, \theta_h \sim \mathcal{N}(\theta_h, 1). \tag{1}$$

In one case, we will have prior information in the form of an informed prior guess c_0. In the other, we will have linkeage information W. For both models, we will assume that the mixture components come from a hierarchical normal prior:

$$\theta_1, \ldots, \theta_K \sim \mathcal{N}(\theta_0, \sigma_0^2). \tag{2}$$

The first model we will fit is the CP process from Paganin et al. (2021). This assumes the following prior for the partition $\boldsymbol{c}$:

$$p(\boldsymbol{c}) \propto \exp\left\{-\psi d(\boldsymbol{c}, \boldsymbol{c}_0)\right\} p_0(\boldsymbol{c}) \tag{3}$$

and denoted as $\boldsymbol{c} \sim \text{CP}(\boldsymbol{c}_0, \psi, p_0(\boldsymbol{c}))$.

The next model we will fit is the constrained finite mixture model:

$$p(\boldsymbol{c}) \propto \exp\left\{\sum_{i=1}^{n} \sum_{j \neq i} W_{ij} \mathbf{1}(c_i = c_j)\right\} p_0(\boldsymbol{c}) \tag{4}$$

and denoted as $\boldsymbol{c} \sim \text{CC}(\boldsymbol{W}, p_0(\boldsymbol{c}))$.

In either case, $p_0(\boldsymbol{c})$ is a symmetric Dirichlet prior, with parameters K and $\gamma = 1$. I'll also compare it to the general Gaussian Mixture model, which results from setting $\boldsymbol{W} = 0$ in (4).

To compare these three methods, I use the clustering point estimate given by minimizing Binder's loss, using code from the `mcclust.ext` package. I then compare the clustering point estimate to the truth by using the adjusted Rand index (values close to 1 indicate recovery) and by plotting the posterior similarity matrix. For each method, I set $\theta_0 = 0$ and $\sigma_0^2 = 1$.

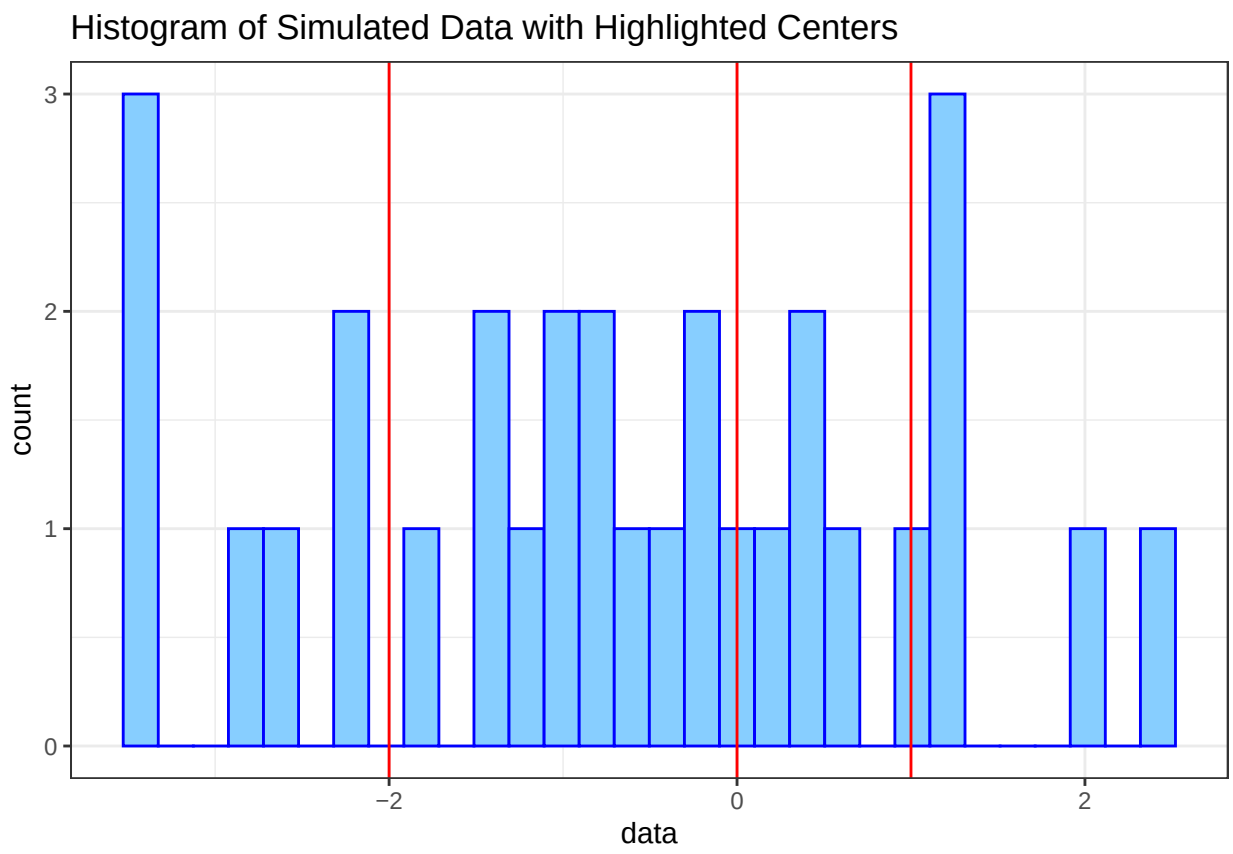
Simulating Data

We specify the true cluster centers to be $\theta_{true} = (-2, 0, 1)^T$ and fix the cluster labels. Conditional on a cluster label in cluster h, we simulate the ith data point as $y_i \sim \mathcal{N}(\theta_{true,h}, 1)$.

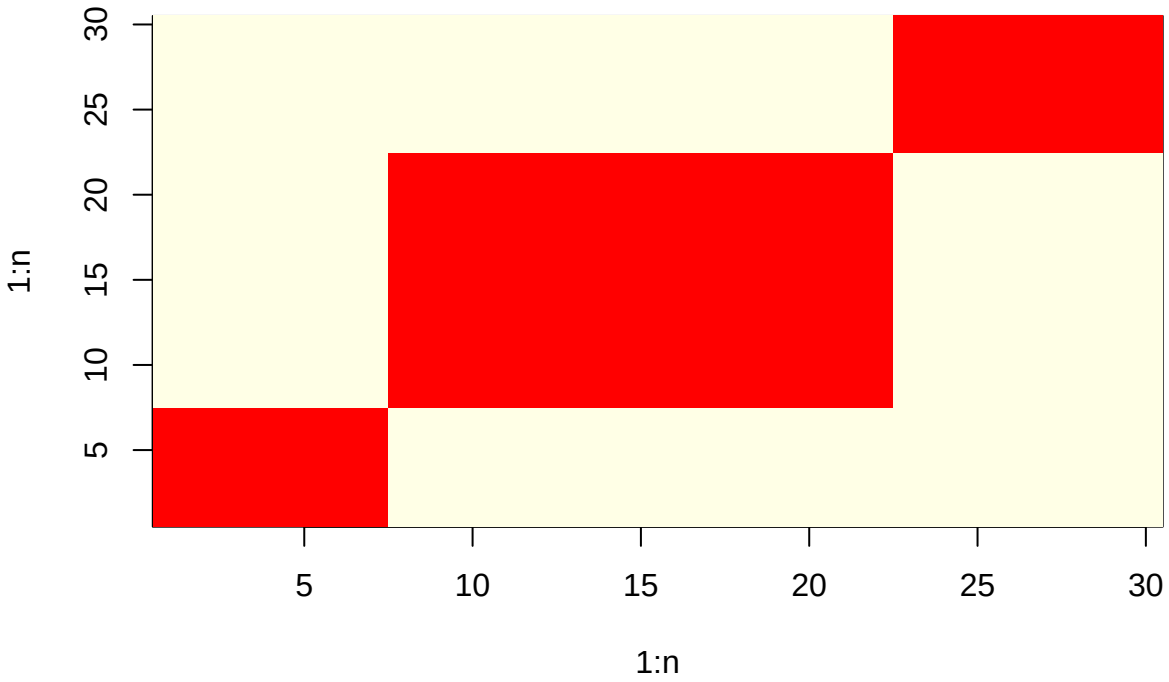
```
set.seed(1996)
K <- 3 # number of clusters
n <- 30 # sample size
theta_true <- c(-2, 0, 1) # actual centers
sigma_true <- 1 # standard deviation is the same for each cluster
```

```r
c_true <- c(rep(1, 15), rep(2, 8), rep(3,7)) # true cluster components
y <- rep(0,n)
for (i in 1:n){
  y[i] <- rnorm(n = 1, mean = theta_true[c_true[i]], sd = sigma_true)
}

# create adjacecny matrix for reference
A <- matrix(0, nrow = n, ncol = n)
for (i in 1:(n-1)) {
  for (j in (i+1):n) {
   A[i,j] <- ifelse(c_true[i] == c_true[j], 1, 0)
  }
}
A <- A + t(A) + diag(1,n)

# plotting
df.y <- data.frame(data = y)
ggplot(df.y, aes(x = data)) + geom_histogram(color = "blue", fill = "skyblue1") +
  geom_vline(xintercept = theta_true[1], color = "red") +
  geom_vline(xintercept = theta_true[2], color = "red") +
  geom_vline(xintercept = theta_true[3], color = "red") +
  labs(title = "Histogram of Simulated Data with Highlighted Centers") +
  theme(text = element_text(size = 13)) +
  theme_bw()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
# plotting adjacecny
plotpsm(A)
```



For posterior similarity plots, we hope to see three well-defined blocks, two of the same size, and one larger one.

Gibbs Samplers

CP Process

First, we must take an initial clustering c_0 as an informed prior guess. Here, I've set c_0 to be the true partition with about 1/10 of the observations contaminated by some noise.

```
set.seed(612)
accur <- rbinom(n = n, size = 1, prob = 9/10)
c_0 <- accur * c_true  + (1 - accur) * sample(1:K, size = n, replace = T, prob = rep(1/K,K))
```

The adjusted Rand index between c_0 and the truth indicates that these partitions are similar, but not exactly the same.

```
adjustedRandIndex(c_0, c_true)
```

```
## [1] 0.8958274
```

Now, we can move onto posterior computation.

```
source("ccfuncts/gibbs_CP.R")
R <- 2000
g <- 1
theta_0 = 0
sigma0_sq = 1
stops = 2000
psi = 3.5 # change this to impact penalty
```

```
fit_CP <- gibbs_CP(R = R,
                   y = y,
                   c_0 = c_0,
                   psi = psi,
                   g = g,
                   K = K,
                   theta_0 = theta_0,
                   sigma0_sq = sigma0_sq,
                   stops = stops)
```

```
## [1] "sampling"
## [1] 2000
```

```
# extract c
c_CP <- fit_CP$c
c_CP.psm <- comp.psm(cls = c_CP)
c_CP.minbinder <- minbinder(psm = c_CP.psm,
                cls.draw = c_CP,
                max.k = 10)
```

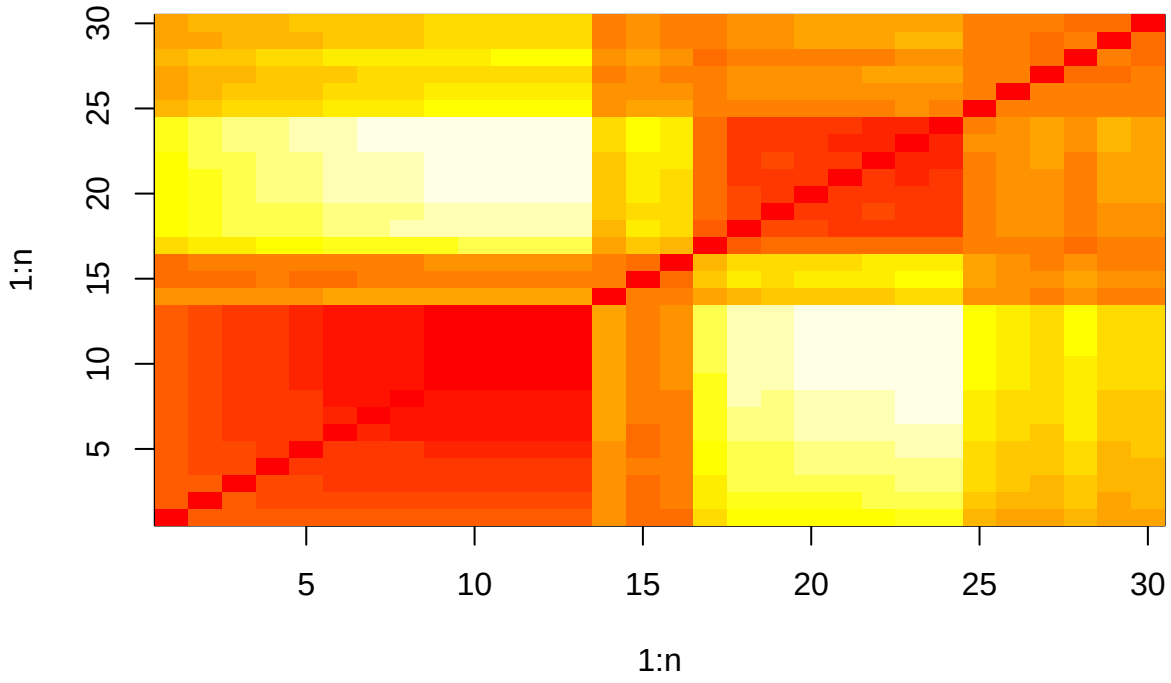
The adjusted Rand index is

```
# compare to c_true
adjustedRandIndex(c_CP.minbinder$cl, c_true) # ARI between estimated clustering and the truth
```

```
## [1] 0.4944395
```

with PSM

```
# plot the psm
plotpsm(c_CP.psm)
```



Constrained Clustering

First, we have to create the $\boldsymbol{W}$ matrix. I've found that values of $|W_{ij}| = 1$ tend to describe high certainty a priori with this data. So, for now I simulate $\boldsymbol{W}$ as $W_{ij} = Z_{ij}A_{ij}$, where $A_{ij} = 1$ if i and j are in the same (true) cluster, and $A_{ij} = -1$ if they are in different clusters. Here, $Z_{ij} \sim \text{Ber}(p)$, for some $0 < p < 1$. By convention, $W_{ii} = 0$.

```
set.seed(1996)
W <- matrix(0, nrow = n, ncol = n) # make sure to have zeros on diagonal
w <- 1 # default weight, indicates confidence, default=1
p <- 1/3 # Z_{ij} = 0 indicates that we have
# high uncertainty if i and j are in the same cluster a priori
for (i in 1:(n-1)) {
  for (j in (i+1):n) {
   W[i,j] <- w * ifelse(c_true[i] == c_true[j], 1, -1) * rbinom(n = 1, size = 1, prob = 1/10)
  }
}
W <- W + t(W)
```

Now, we can sample from the posterior distribution.

```
source("ccfuncts/gibbs_CC.R")
R <- 2000
g <- 1
theta_0 = 0
sigma0_sq = 1
stops = 2000


fit_CC <- gibbs_CC(R = R,
                   y = y,
                   W = W,
                   g = g,
                   K = K,
                   theta_0 = theta_0,
                   sigma0_sq = sigma0_sq,
                   stops = stops)
```

```
## [1] "sampling"
## [1] 2000
```

```
# extract c
c <- fit_CC$c
c.psm <- comp.psm(cls = c)
c.minbinder <- minbinder(psm = c.psm,
                cls.draw = c,
                max.k = 10)
```

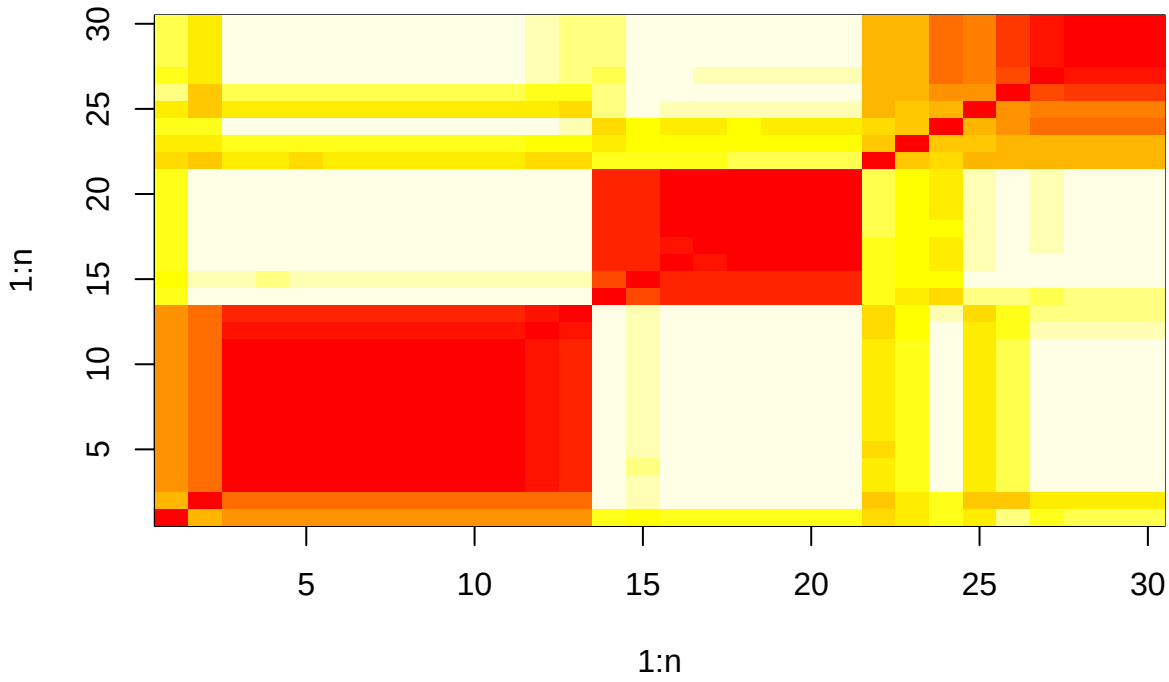
The adjusted Rand index is

```
# compare to c_true
adjustedRandIndex(c.minbinder$cl, c_true) # ARI between estimated clustering and the truth
```

```
## [1] 0.7854281
```

with PSM

```
# plot psm
plotpsm(c.psm)
```



Gaussian Mixture Model

```
source("ccfuncts/gibbs_CC.R")
R <- 2000
g <- 1
theta_0 = 0
sigma0_sq = 1
stops = 2000


fit_GMM <- gibbs_CC(R = R,
                    y = y,
                    W = matrix(0, nrow = n, ncol = n),
                    g = g,
                    K = K,
                    theta_0 = theta_0,
                    sigma0_sq = sigma0_sq,
                    stops = stops)
```

```
## [1] "sampling"
## [1] 2000
```

```
# extract c
c_gmm <- fit_GMM$c
c_gmm.psm <- comp.psm(cls = c_gmm)
c_gmm.minbinder <- minbinder(psm = c_gmm.psm,
                 cls.draw = c_gmm,
                 max.k = 10)
```

The adjusted Rand index is

```
# compare to c_true
adjustedRandIndex(c_gmm.minbinder$cl, c_true) # ARI between estimated clustering and the truth
```

```
## [1] 0.4944395
```

with PSM

```
plotpsm(c_gmm.psm)
```