DPMPM for missing data imputation (using R package): MAR

Setting:

- K = 3 real clusters (classes) (π denotes the mixing proportion)
- For each k, we sample values for three categorical (qualitative) features ($p = 3$).
- For each feature, there are 3 possible outcome ($J = 3$) and the probability of them can be characterized by the multinomial probability parameter θ_{kp} for k = 1, 2, 3 and p = 1, 2, 3
- There will be data missing at random (MAR) in the last features. The probability of missingness is govern by $logit(w_1^T x_{i1} + w_2^T x_{i2})$ where $w_1 = [-0.75, -1, 0.2]$ and $w_2 = [0.6, -2, 0.5]$. This results in 42% of missing data in the last feature.
- For modelling process, we will use NPBayesImputeCat package for multiple imputation.

Summary on modelling process

$X_{ip} \mid z_i = k, \theta \sim Mult(\theta_{kp})$ for p = 1, 2, 3

$z \mid \pi \sim Mult(\pi_1, \pi_2, ...)$

$\pi_h = V_h \Pi_{g<h}(1 - V_g)$

$V_h \sim Beta(1, \alpha)$

$\alpha \sim Gamma(a_\alpha, b_\alpha)$

$\theta_{kp} \sim Dirichlet(1)$

```
# Define mixing proportion
set.seed(0)
pi_true = c(0.3, 0.1, 0.6)

# Theta 1 for mixture cluster 1
theta_11_true = c(0.7,0.2,0.1)
theta_12_true = c(0.1,0.8,0.1)
theta_13_true =c(0.2,0.1,0.7)

# Theta 2 for mixture cluster 2
theta_21_true = c(0.05,0.75,0.2)
theta_22_true = c(0.2,0.15,0.65)
theta_23_true =c(0.7,0.2,0.1)

# Theta 3 for mixture cluster 3
theta_31_true = c(0.1,0.1,0.8)
theta_32_true = c(0.7,0.15,0.15)
theta_33_true =c(0.1,0.7,0.2)

# Theta row i is cluster i, column j is category j
theta_p1_true = rbind(theta_11_true, theta_21_true, theta_31_true)
theta_p2_true = rbind(theta_12_true, theta_22_true, theta_32_true)
theta_p3_true = rbind(theta_13_true, theta_23_true, theta_33_true)

# Create simulated data
```

```r
n = 300
class_i = rmultinom(n, size = 1, prob = pi_true)
x1 = c()
x2 = c()
x3_original = c()
for (i in 1:n) {
  x1 = cbind(x1, rmultinom(1, size = 1, prob = theta_p1_true[class_i[, i]==1,]))
  x2 = cbind(x2, rmultinom(1, size = 1, prob = theta_p2_true[class_i[, i]==1,]))
  x3_original = cbind(x3_original, rmultinom(1, size = 1, prob = theta_p3_true[class_i[, i]==1,]))
}

# Format data input
col1 = as.factor(apply(c(1,2,3)*x1, MARGIN = 2, FUN = sum))
col2 = as.factor(apply(c(1,2,3)*x2, MARGIN = 2, FUN = sum))
col3 = as.factor(apply(c(1,2,3)*x3_original, MARGIN = 2, FUN = sum))
df = data.frame(col1, col2, col3)
colnames(df)<- c('feature1', 'feature2', 'feature3')
```

Generate missingness in data

```r
# Define parameter of logistic function
w1 =  c(-0.75, -1, 0.2)
w2 = c(0.6, -2, 0.5)

# Calculate probability of missingness of features 3
prob = apply(w1*x1, MARGIN = 2, FUN = sum) + apply(w2*x2, MARGIN = 2, FUN = sum)
prob = 1/(exp(-prob)+1)

# Indicator for X3miss
indicator = rbernoulli(n = 300, p = prob)
df[indicator, 3] = c(NA)
```

Multiple imputation using NPBayesImputeCat package

Ref: https://cran.r-project.org/web/packages/NPBayesImputeCat/NPBayesImputeCat.pdf

1. Create and initialize the Rcpp_Lcm model object using CreateModel with the following arguments:

- X: dataframe to be imptuted = df
- MCZ: dataframe with the definition of structural zero = NULL
- K: the maximum number of mixture components = 30
- Nmax: An upper truncation limit for the augmented sample size = 0
- aalpha: the hyper parameter alpha in stick-breaking prior = 0.25
- balpha: the hyper parameter beta in stick-breaking prior = 0.25
- seed = 0

2. Set the tracer for the sampling process

- k_star: the effective cluster number
- psi: conditional multinomial probabilties
- ImputedX: imputation result

3. Run the model using the method Run of Rcpp_Lcm class with the following arguments:

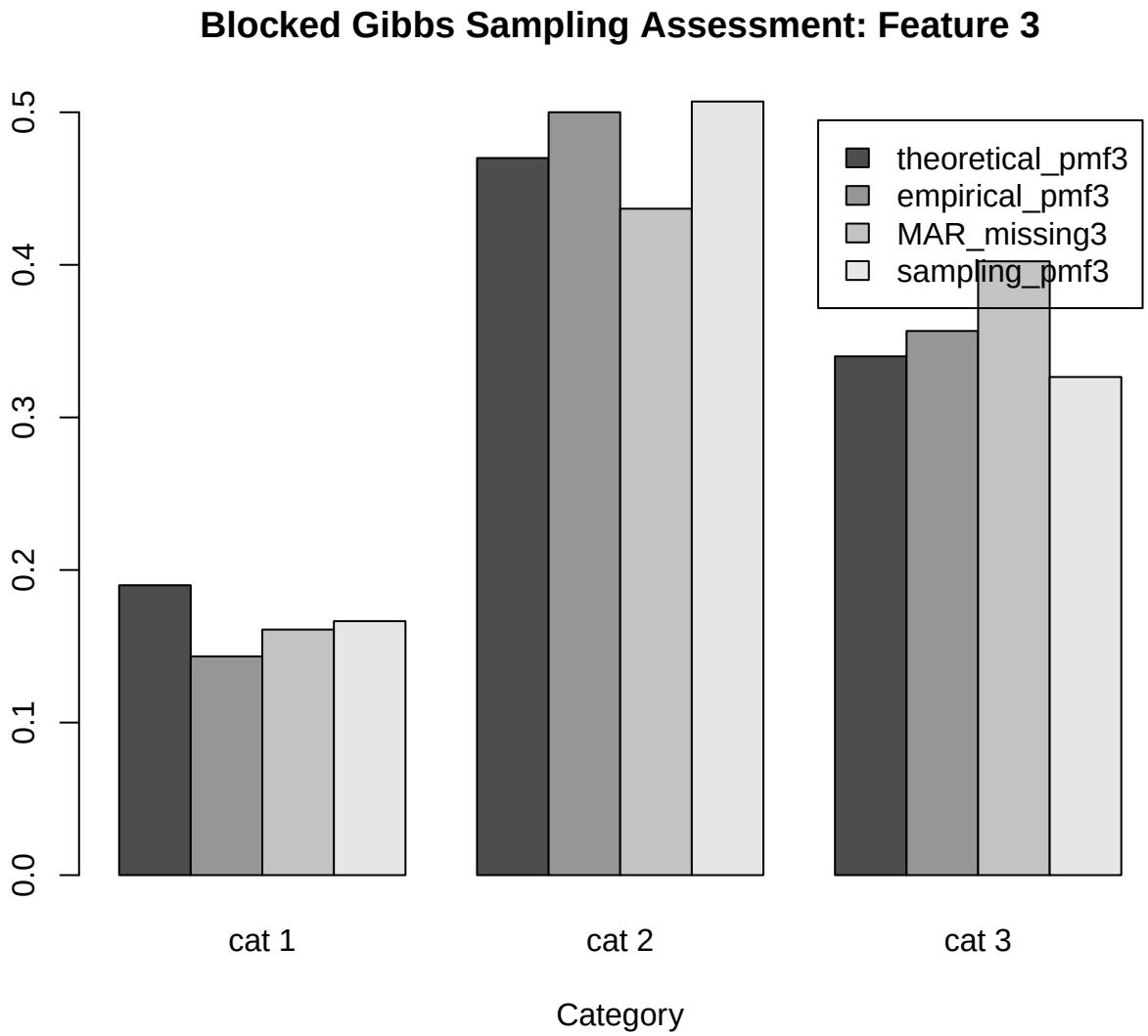
- burnin = 5000
- iter = 10000
- thinning = 1

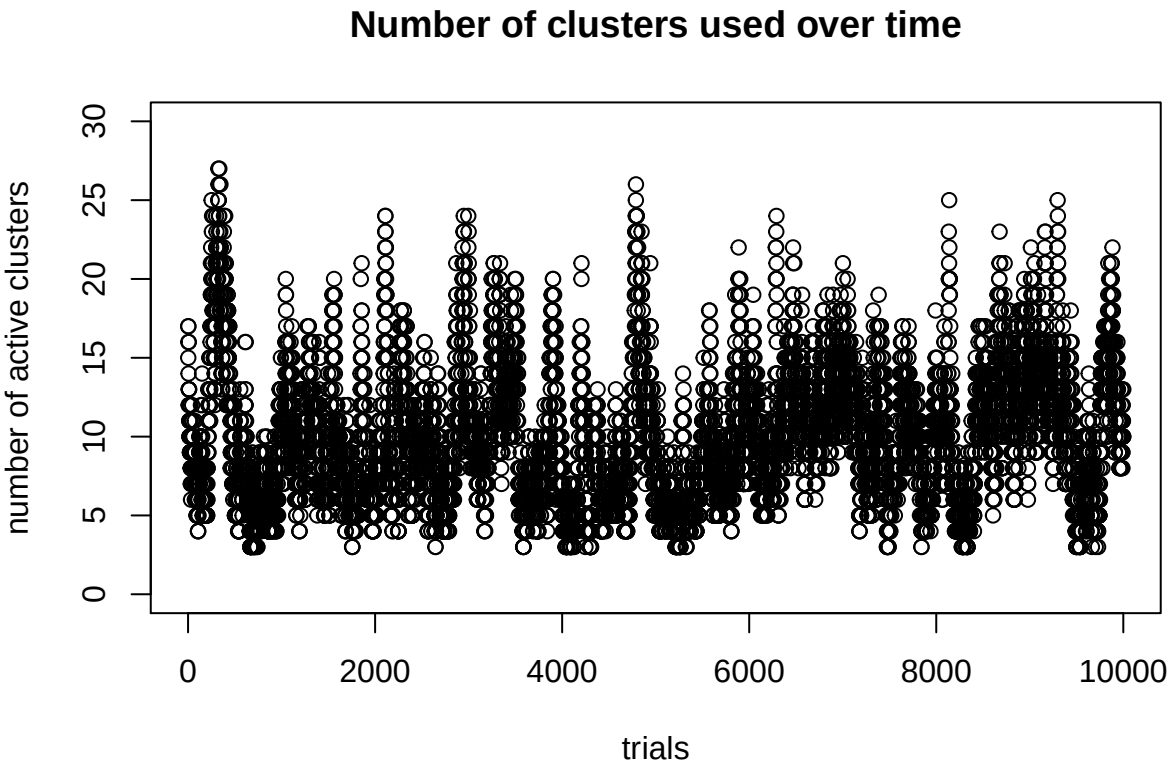
4. Obtain result

```
iter = 10000
# 1. Create and initialize the Rcpp_Lcm model object
model = CreateModel(X = df, MCZ = NULL, K = 30, Nmax = 0,
                    aalpha = 0.25, balpha = 0.25, seed = 1)
# 2. Set tracer
model$SetTrace(c('k_star', 'psi', 'ImputedX'),iter)

# 3. Run model
model$Run(5000,iter,1)
```

```
# Extract results
output <- model$GetTrace()
k_star <- output$k_star
psi <- output$psi
imputed_df <- output$ImputedX

#retrieve parameters from the final iteration
result <- model$snapshot

#convert ImputedX matrix to dataframe, using proper factors/names etc.
ImputedX <- GetDataFrame(result$ImputedX,df)
```

```
# Model checking with empirical data

# Feature 3
theoretical_pmf3 = apply(as.vector(pi_true)*theta_p3_true,MARGIN = 2, sum)
empirical_pmf3 = apply(x3_original, MARGIN = 1, mean)

MAR_missing3 = c(mean(df[!indicator,3]==1), mean(df[!indicator,3]==2),
                 mean(df[!indicator,3]==3))

# Extract only features 3 from imputed data
sampling_pmf3 = table(imputed_df[,seq(3, dim(imputed_df)[2], 3)])
sampling_pmf3 = sampling_pmf3/sum(sampling_pmf3)

df3 = rbind(theoretical_pmf3,empirical_pmf3, MAR_missing3, sampling_pmf3)
colnames(df3)<- c('cat 1', 'cat 2', 'cat 3')
barplot(df3, xlab = 'Category', beside = TRUE,
        legend = TRUE, main = 'Blocked Gibbs Sampling Assessment: Feature 3')
```


Blocked Gibbs Sampling Assessment: Feature 3
0.0
0.1
0.2
0.3
0.4
0.5
theoretical_pmf3
empirical_pmf3
MAR_missing3
sampling_pmf3
cat 1
cat 2
cat 3
Category


Number of clusters used over time
number of active clusters
0
5
10
15
20
25
30
0
2000
4000
6000
8000
10000
trials