Multinomial Model for ordinal data: MAR

Setting:

- For each observations, we have 3 associated variables

X1: Nomial variable with 3 possible outcomes {1,2,3} which we will generate from $Mult(1, [0.2, 0.3, 0.5])$ X2: Nomial variable with 3 possible outcomes {1,2,3} which we will generate from $Mult(1, [0.3, 0.5, 0.2])$ Y: Ordinal variable with 5 levels, {1,2,3,4,5} generated using the following process

$\epsilon_i \sim N(0,1)$ $z_i = \beta^T X_i + \epsilon$ where $X = [X1 == 1, X1 == 2, X2 == 1, X2 == 2]$ i.e. we drop category (1,1) as the baseline $g(z_i) = Y_i$.

Here $\beta = [-3, 2, 2, -4]$ and function g will be the binning function which will bin data into 5 different bins corresponding to 5 possible ordinal outcome.

- There will be data missing at random (MAR) in Y. The probability of missingness is govern by $logit(w^T x_i - 0.75)$ where $w = [0.4, -0.3, -0.8, 0.3]$. This results in 25% of missing data in the last feature.
- We will try to model Y conditioned on other variables (X1, and X2) using multinomial model with the parameter $\theta \in R^{45}$ governing the joint probability $P(X1, X2, Y)$.
- We have one unknown parameter θ which we will put Dirichlet distribution prior with parameter $\alpha = 1$ as an noninformative prior.

```
# Data generating process
set.seed(0)
n = 600
beta = c(-3, 2, 2, -4)

# noise term
epsilon = rnorm(n, mean = 0, sd = 1)

# X1
X1 = t(rmultinom(n, size = 1, prob = c(0.2, 0.3, 0.5)))

# X2
X2 = t(rmultinom(n, size = 1, prob = c(0.3, 0.5, 0.2)))

# X
X = cbind(X1[,2:3], X2[,2:3])
colnames(X) <- c('X1_cat2', 'X1_cat3', 'X2_cat2', 'X2_cat3')

# Z
Z = X%*%beta + epsilon

# Cut-off points and Y
g = quantile(Z, probs = c(0.2, 0.4, 0.6, 0.8))
Y = rep(NA, n)
Y[Z<g[1]] = 1
Y[Z>=g[1] & Z<g[2]] = 2
Y[Z>=g[2] & Z<g[3]] = 3
Y[Z>=g[3] & Z<g[4]] = 4
Y[Z>=g[4]] = 5
```

```
Z_original = Z
Y_original = Y
```

Generate missingness in data

```
# Define parameter of logistic function
w= c(0.4, -0.3, -0.8, 0.3)

# Calculate probability of missingness of features 3
prob = apply(t(w*t(X)), MARGIN = 1, FUN = sum)-0.75
prob = 1/(exp(-prob)+1)

# Indicator for X3miss
indicator = rbernoulli(n = n, p = prob)
Y[indicator] = NA
Z[indicator] = NA
```

Model specifications:

- $\theta = (\theta_1, \theta_2, ..., \theta_{45})$ a vector of Multinomial parameter
- For all samples $x_i \sim Mult(1, \theta)$ and $x_i = (x_{1i}, x_{2i}, y_i)$
- prior distribution: $\theta \sim Dir(1)$ the non-informative prior

Blocked Gibbs Sampling here consists of two major steps

1. Sample new θ from its full conditional

$\theta \mid X_1, X_2, Y \sim Dir(1 + n_{ijk})$ where n_{ijk} is the number of observations found in class i of X_1, class j of X_2 and class k of Y

2. for missing data index i, sample Y_i from its full conditional distribution:

$Y_i \mid X_{1i}, X_{2i}, \theta \sim Mult(1, \theta_{X_{1i}, X_{2i}})$ where $\theta_{X_{1i}, X_{2i}}$ is the conditional probability of Y given values of X_{1i} and X_{2i}

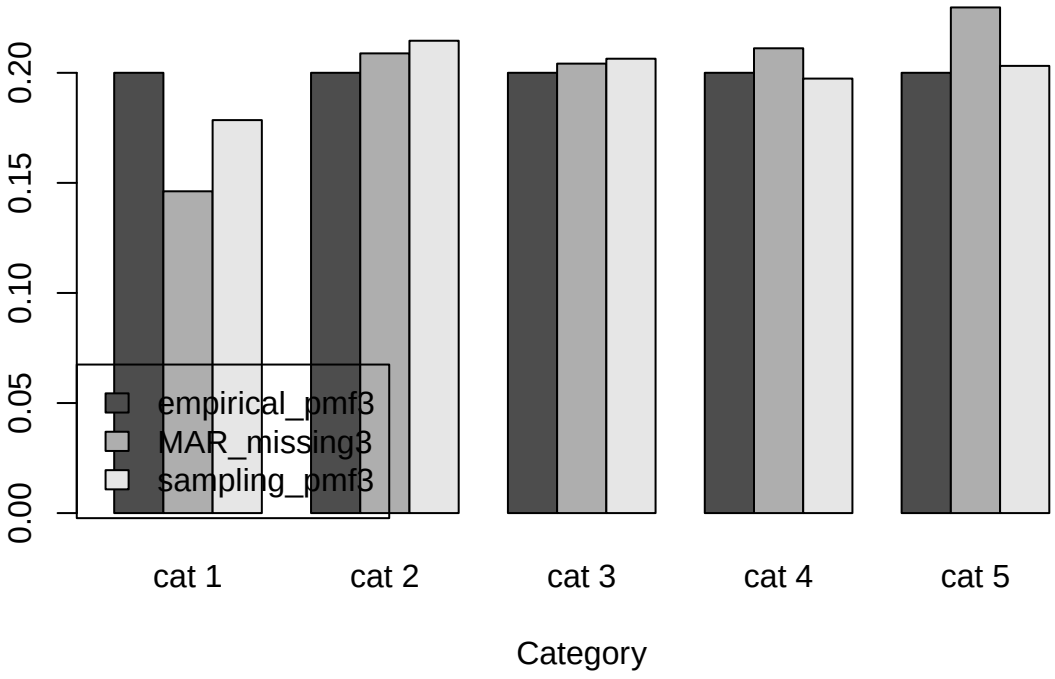
```
# Blocked Gibbs Sampling
set.seed(0)

# Imputation for first trial
df <- data.frame(cbind(apply(t(t(X1)*c(1,2,3)),1,sum),
                       apply(t(t(X2)*c(1,2,3)),1,sum), Y))
colnames(df) <- c('X1', 'X2', 'Y')
df$Y <- as.factor(df$Y)
df$X1 <- as.factor(df$X1)
df$X2 <- as.factor(df$X2)
mod <- multinom(Y~., data=df[!is.na(df$Y),])
```

```
## # weights:  30 (20 variable)
## initial  value 693.667740
## iter  10 value 395.144711
## iter  20 value 323.336665
## iter  30 value 320.213739
## iter  40 value 320.035440
## iter  50 value 320.031325
## final  value 320.031319
## converged
```

```r
Y_i = Y
Y_i[indicator] = predict(mod, newdata = df[is.na(df$Y),], 'class')

# Initialize the sampling matrix
S = 30000
SAMPLED_Y = matrix(nrow = S, ncol = n)
THETA = matrix(nrow=S,ncol=45)

for (round in 1:S) {
  # Step 1: Sample Theta
  # Data summary: Contingency table
  df <- data.frame(cbind(apply(t(t(X1)*c(1,2,3)),1,sum),
                         apply(t(t(X2)*c(1,2,3)),1,sum), Y_i))
  colnames(df) = c('X1', 'X2', 'Y')
  contingency_table = table(df$X1, df$X2, df$Y)

  # Update posterior parameter and asmple theta
  alpha = rep(1, 45) + matrix(contingency_table, nrow =1)
  theta = rdirichlet(1, alpha)
  THETA[round,] <- theta

  # Step 2: Sample missing Y
  for (i in 1:n) {
    if (indicator[i]==TRUE) {
      # The data on Y is missing
      conditional_prob = theta[seq(df$X1[i] + df$X2[i]*3 - 3, 45, 9)]
      Y_i[i] = sum(rmultinom(1, size = 1, prob = conditional_prob)*c(1,2,3,4,5))
    }
  }
  SAMPLED_Y[round,] <-  Y_i
}
```

```r
burnin = 10000
thining = 100
# Imputation accuracy
empirical_pmf3 = table(Y_original)/n
MAR_missing3 = table(Y[!indicator])/(n-sum(indicator))
sampling_pmf3 = table(SAMPLED_Y[seq(burnin, dim(THETA)[1], thining),])/
  sum(table(SAMPLED_Y[seq(burnin, dim(THETA)[1], thining),]))

df3 = rbind(empirical_pmf3, MAR_missing3, sampling_pmf3)
colnames(df3)<- c('cat 1', 'cat 2', 'cat 3', 'cat 4', 'cat 5')
barplot(df3, xlab = 'Category', beside = TRUE,
        legend = TRUE, args.legend=list(x='bottomleft'),
        main = 'Blocked Gibbs Sampling Assessment: Marginal Y pmf')
```

```r
# Imputation accuracy
MAR_missing3 = table(Y_original[indicator])/sum(indicator)
sampling_pmf3 = table(SAMPLED_Y[seq(burnin, dim(THETA)[1], thining),indicator])/
  sum(table(SAMPLED_Y[seq(burnin, dim(THETA)[1], thining), indicator]))

df3 = rbind(MAR_missing3, sampling_pmf3)
colnames(df3)<- c('cat 1', 'cat 2', 'cat 3', 'cat 4', 'cat 5')
barplot(df3, xlab = 'Category', beside = TRUE,
```



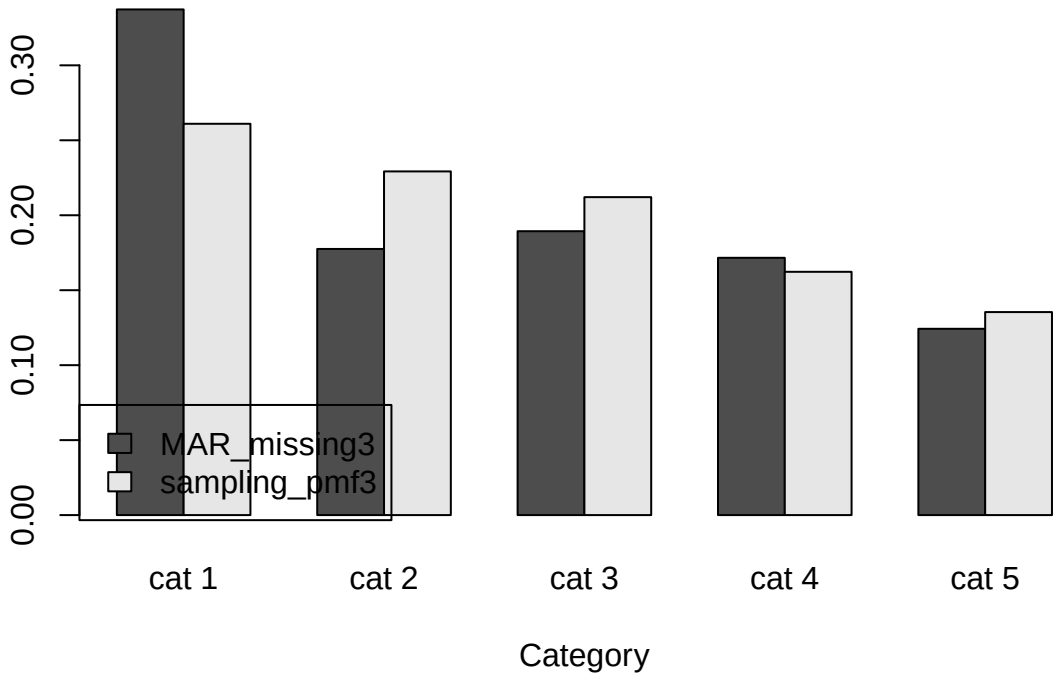

```
        legend = TRUE, args.legend=list(x='bottomleft'),
        main = 'Blocked Gibbs Sampling Assessment: Missing Y pmf')
```

```
# Imputation accuracy
true_label = Y_original[indicator]
sampled_label = SAMPLED_Y[seq(burnin, dim(THETA)[1], thining),indicator]
mean(t(sampled_label) == true_label)
```

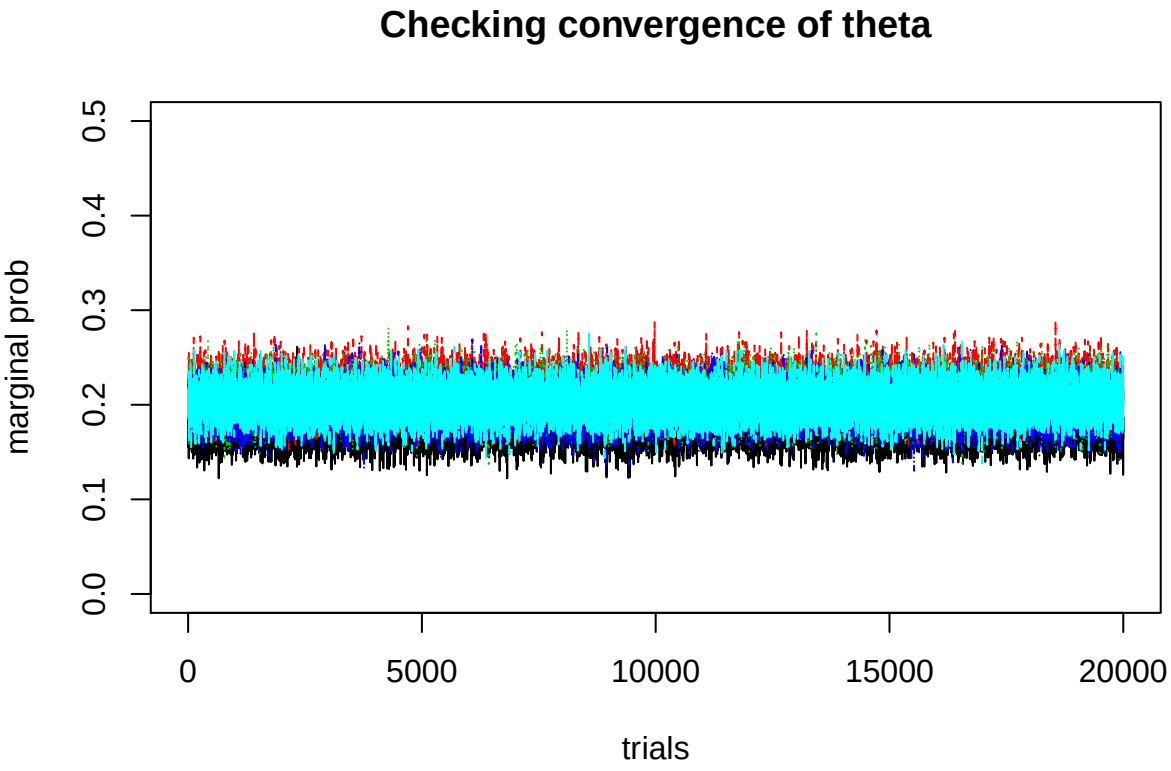
```
## [1] 0.4921546
```

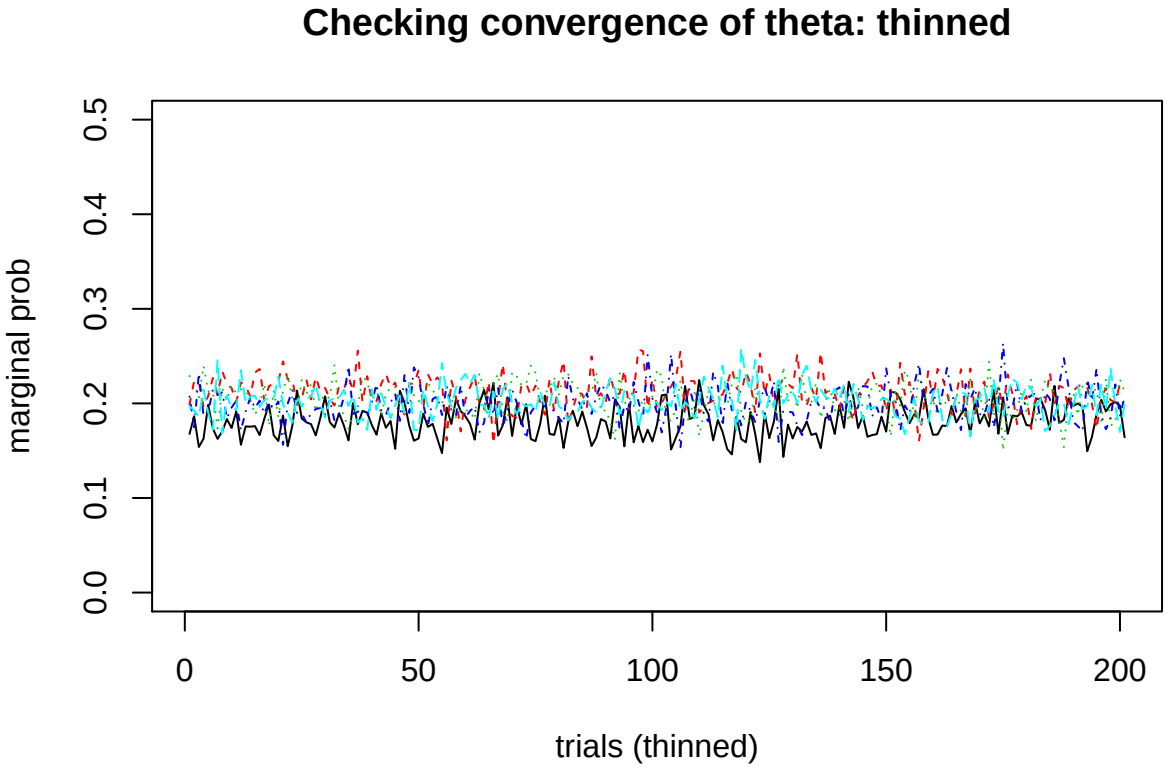
```
# Check convergence on theta
posterior_pmf = cbind(apply(THETA[burnin:dim(THETA)[1], 1:9],
                            MARGIN = 1, sum),
                      apply(THETA[burnin:dim(THETA)[1], 10:18],
                            MARGIN = 1, sum),
                      apply(THETA[burnin:dim(THETA)[1], 19:27],
                            MARGIN = 1, sum),
                      apply(THETA[burnin:dim(THETA)[1], 28:36],
                            MARGIN = 1, sum),
                      apply(THETA[burnin:dim(THETA)[1], 37:45],
                            MARGIN = 1, sum))
colnames(posterior_pmf) <- c('order 1', 'order 2', 'order 3', 'order 4', 'order 5')
matplot(1:dim(posterior_pmf)[1], posterior_pmf,
        type = 'l', ylab = 'marginal prob', xlab = 'trials',
        main = 'Checking convergence of theta', ylim = c(0, 0.5))
```

```
# Check convergence on theta
posterior_pmf = cbind(apply(THETA[seq(burnin, dim(THETA)[1], thining), 1:9],
                            MARGIN = 1, sum),
```




```
                    apply(THETA[seq(burnin, dim(THETA)[1], thining), 10:18],
                                MARGIN = 1, sum),
                    apply(THETA[seq(burnin, dim(THETA)[1], thining), 19:27],
                                MARGIN = 1, sum),
                    apply(THETA[seq(burnin, dim(THETA)[1], thining), 28:36],
                                MARGIN = 1, sum),
                    apply(THETA[seq(burnin, dim(THETA)[1], thining), 37:45],
                                MARGIN = 1, sum))
colnames(posterior_pmf) <- c('order 1', 'order 2', 'order 3', 'order 4', 'order 5')
matplot(1:dim(posterior_pmf)[1], posterior_pmf,
        type = 'l', ylab = 'marginal prob', xlab = 'trials (thinned)',
        main = 'Checking convergence of theta: thinned', ylim = c(0, 0.5))
```


Checking convergence of theta
marginal prob
0.0
0.1
0.2
0.3
0.4
0.5
0
5000
10000
15000
20000
trials


Checking convergence of theta: thinned
marginal prob
0.0
0.1
0.2
0.3
0.4
0.5
0
50
100
150
200
trials (thinned)