ST740 HW3

Chenyin Gao

2023-11-17

Contents

1 Derive the full conditional distributions of (a) X_t; (b) $(p_1, \cdots, p_6)$ and (c) (P_{11}, P_{12})

(a) For the conditional distribution of X_t, we have

$$p(X_t \mid \text{rest}) = \frac{P(Y_{1:n}, X_{1:n})}{P(Y_{1:n}, X_{-t})} = \begin{cases} \frac{P_{i1}P_{1j}/6}{P_{i1}P_{1j}/6+p_y P_{i2}P_{2j}}, & X_{t-1}=i, X_t=1, X_{t+1}=j, Y_t=y, \\ \frac{p_y P_{i2}P_{2j}}{P_{i1}P_{1j}/6+p_y P_{i2}P_{2j}}, & X_{t-1}=i, X_t=2, X_{t+1}=j, Y_t=y. \end{cases}$$

which holds for $1 < t < n$. On the one hand, when $t = 1$, we have

$$p(X_t \mid \text{rest}) = \begin{cases} \frac{P_{1j}/6}{P_{1j}/6+p_y P_{2j}}, & X_t=1, X_{t+1}=j, Y_t=y, \\ \frac{p_y P_{2j}}{P_{1j}/6+p_y P_{2j}}, & X_t=2, X_{t+1}=j, Y_t=y. \end{cases}$$

On the other hand, when $t = n$, we have

$$p(X_t \mid \text{rest}) = \begin{cases} \frac{P_{i1}/6}{P_{i1}/6+p_y P_{i2}}, & X_{t-1}=i, X_t=1, Y_t=y, \\ \frac{p_y P_{i2}}{P_{i1}/6+p_y P_{i2}}, & X_{t-1}=i, X_t=2, Y_t=y. \end{cases}$$

For the conditional distribution of $(p_1, \cdots, p_6)$, we have

$$\begin{aligned} & p(p_1, \cdots, p_6 \mid \text{rest}) \\ & \propto p(p_1, \cdots, p_6) \cdot \prod_{t=1}^{n} \{p(Y_t = y \mid X_t = 2)\}^{\mathbf{1}(X_t=2)} \\ & \propto \prod_{y=1}^{6} p_y^{\sum_{t=1}^{n} \mathbf{1}(Y_t=y, X_t=2)+1/6}. \end{aligned}$$

For the conditional distribution of (P_{11}, P_{12}), we have

$$
\begin{aligned}
& p(P_{11}, P_{12} \mid \text{rest}) \\
& \propto p(P_{11}, P_{12}) \cdot \prod_{t=2}^{n} \left\{ p(X_t = 1 \mid X_{t-1} = 1)^{\mathbf{1}(X_t=1)} \cdot p(X_t = 2 \mid X_{t-1} = 1)^{\mathbf{1}(X_t=2)} \right\}^{\mathbf{1}(X_{t-1}=1)} \\
& \propto \prod_{j=1}^{2} P_{1j}^{\sum_{t=2}^{n} \mathbf{1}(X_t=j, X_{t-1}=1)+1/2}.
\end{aligned}
$$

2 MCMC algorithm to approximate the posterior distribution

```
# initialization
X.list <- list(); p.vec.list <- list(); P.mat.list <- list()
X.list[[1]] <- X <- as.factor(sample(c(1,2), size = n, replace = TRUE))
p.vec.list[[1]] <- p.vec <- rdirichlet(1, alpha = rep(1/6, 6))
P.mat.list[[1]] <- P.mat <- rbind(rdirichlet(1, alpha = rep(1/2, 2)),
               rdirichlet(1, alpha = rep(1/2, 2)))
niters <- 2000
for (iter in 1:niters) {
  # update X_t
for(t in 1:n){
  # index 1
  if(t==1){
    P.1j <- P.mat[1, X[t+1]]
    P.2j <- P.mat[2, X[t+1]]
    p.y <- p.vec[Y[t]]
    # update
    X[t] <- sample(c(1,2), size = 1, prob = c(P.1j/6,
                                              P.2j*p.y))
  }
  # index (1,n)
  if(t>1 & t<n){
    P.i1 <- P.mat[X[t-1], 1]
    P.i2 <- P.mat[X[t-1], 2]
    P.1j <- P.mat[1, X[t+1]]
    P.2j <- P.mat[2, X[t+1]]
    p.y <- p.vec[Y[t]]
    # update
    X[t] <- sample(c(1,2), size = 1, prob = c(P.i1 * P.1j/6,
                                              P.i2 * P.2j*p.y))
  }
  # index n
    if(t>1 & t<n){
    P.i1 <- P.mat[X[t-1], 1]
    P.i2 <- P.mat[X[t-1], 2]
    p.y <- p.vec[Y[t]]
    # update
    X[t] <- sample(c(1,2), size = 1, prob = c(P.i1/6, P.i1*p.y))
  }
}
X.list[[iter + 1]] <- X
# update p1, \cdots, p6
p.vec <- rdirichlet(1, alpha = table(Y[X==2]) + 1/6)
```

```
p.vec.list[[iter + 1]] <- p.vec
# update P11, P12
X.indx1 <- which(X == 1) + 1
## delete the out-of-range
X.indx1 <- X.indx1[X.indx1<=n]
P.mat[1,] <- rdirichlet(1, alpha = table(X[X.indx1]) + 1/2)
# update P21, P22
X.indx2 <- which(X == 2) + 1
## delete the out-of-range
X.indx2 <- X.indx2[X.indx2<=n]
P.mat[2,] <- rdirichlet(1, alpha = table(X[X.indx2]) + 1/2)
P.mat.list[[iter + 1]] <- P.mat
}
save(X.list, p.vec.list, P.mat.list,
     file = 'ST740_HW3_ChenyinGao_model1.RData')
```

3 Posterior analysis

We first plot the posterior mean of the latent states over the roll number and label them in different colors according to the values of the observed data $Y_{1:n}$ in Figure 1. We notice that the die with successive 6 tends to the HMM flagging the die as rigged.

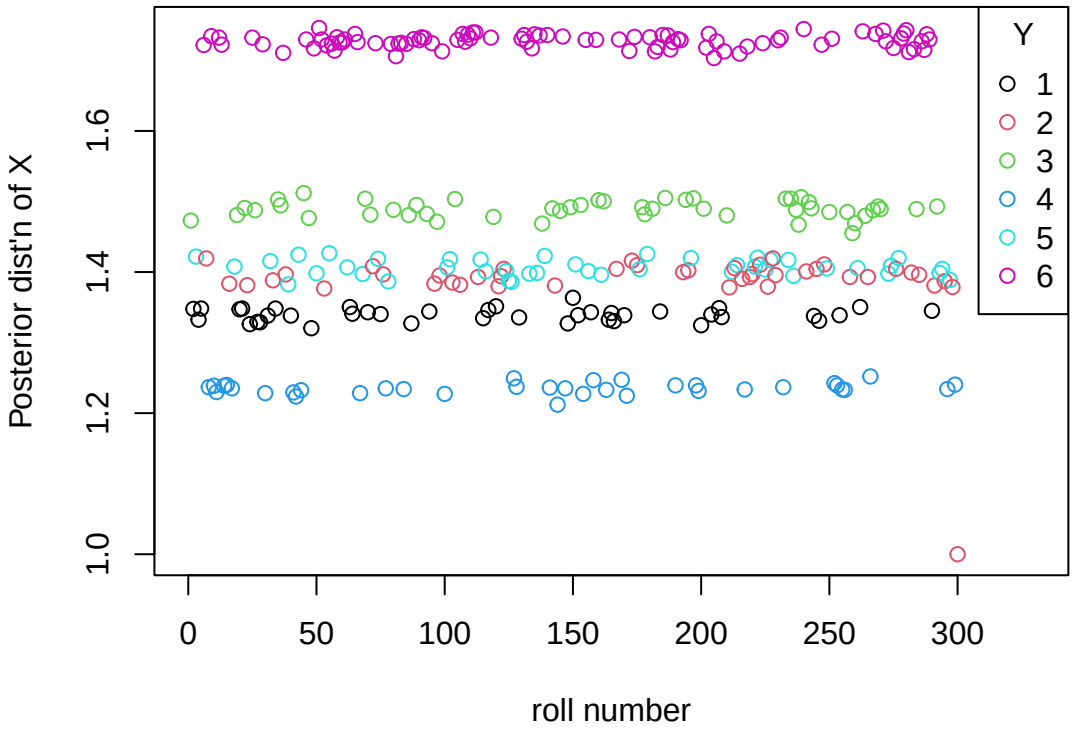

Figure 1: Posterior distribution of the latent states with the data

Besides, we compare the posterior results with the true latent states in Figure 2 and could visually conclude that our method fits the data rather well.

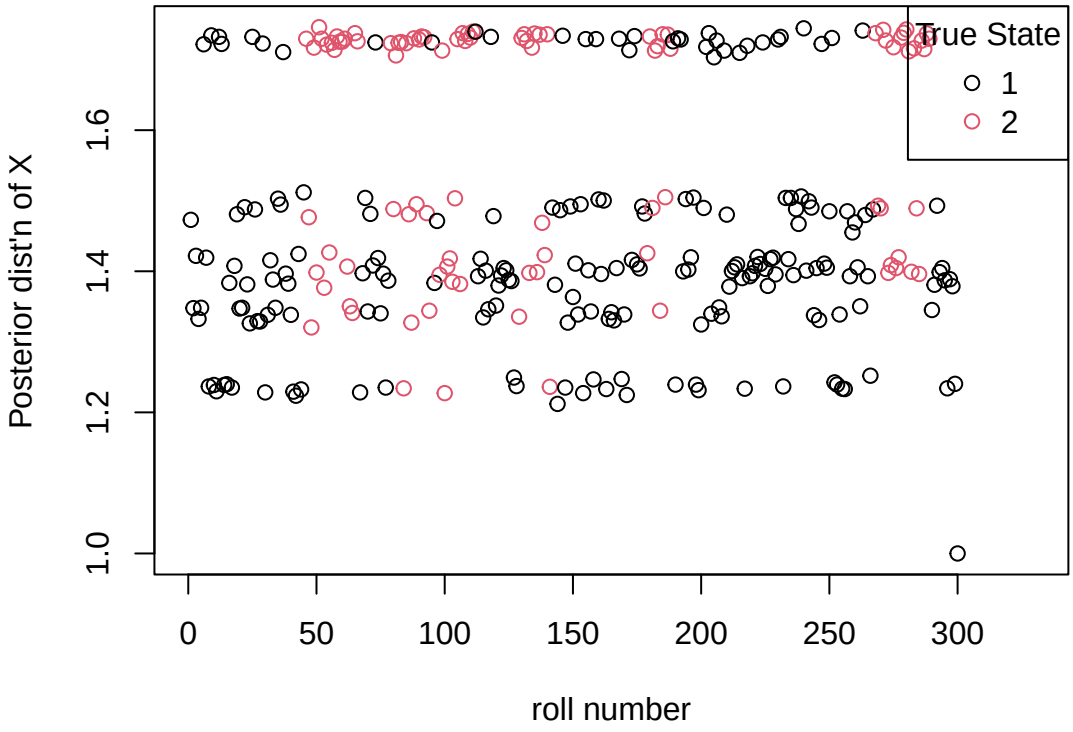

Figure 2: Posterior distribution of the latent states with the truth latent states

4 Compare the performances of model (1), (2) and (3)

We choose to calculate the DIC which is defined by the likelihood:

$$\prod_{t=1}^{n}(1/6)^{\mathbf{1}(X_t=1)}\prod_{y=1}^{6}p_y^{\mathbf{1}(Y_t=y.X_t=2)} = (1/6)^{\sum_{t=1}^{n}\mathbf{1}(X_t=1)}\prod_{y=1}^{6}p_y^{\sum_{t=1}^{n}\mathbf{1}(Y_t=y.X_t=2)}$$

Then the deviance is

$$-2\log\{f(Y_{1:n},X_{1:n})\} = -2\sum_{t=1}^{n}\mathbf{1}(X_t=1)\log(1/6) - 2\sum_{y=1}^{6}\sum_{t=1}^{n}\mathbf{1}(Y_t=y,X_t=2)\log(p_y).$$

```
dic <- function(Y, X, p.vec = NULL){
  sum(-2 * (X == 1) * log(1/6) ) -
    sum(2 * table(Y[X==2]) * log(p.vec))
}
```

```r
# full model
load('ST740_HW3_ChenyinGao_model1.RData')
dic(Y = Y, X = X.list[[length(X.list)]],
    p.vec = apply(do.call(rbind, p.vec.list)[-(1:burn), ], 2, mean))
```

```
## [1] 999.7123
```

```r
# reduced model
load('ST740_HW3_ChenyinGao_model2.RData')
dic(Y = Y, X = X.list[[length(X.list)]],
    p.vec = apply(do.call(rbind, p.vec.list)[-(1:burn), ], 2, mean))
```

```
## [1] 1054.371
```

```r
# no model
dic(Y = Y, X = rep(1, n), p.vec = 1/6)
```

```
## [1] 1075.056
```

In terms of DIC, the full model has the lowest value, which indicates that the full model is the best among the considered three models.

5 Verify that the full model fits the data well

To verify the full model fit, we employ the posterior predictive checks, where multiple (e.g., S) data sets $Y_{1:n}$ are sampled from the posterior predictive distribution and compare the induced statistics with the observed data. In particular, we consider the Bayesian p-value, which is $pVal = \sum_{s=1}^{S} \mathbf{1}(d_s > d_0)/S$. We choose the average of $Y_{1:n}$ to be the statistic of interest. As the computed Bayesian p-value is 0.50, we claim our full model is a relative good fit of the observed data.

```r
set.seed(27695)
S <- 100
# full model
load('ST740_HW3_ChenyinGao.RData')
X.post <-  X.list[[length(X.list)]]
p.vec.post <- apply(do.call(rbind, p.vec.list)[-(1:burn), ], 2, mean)

d.S <- replicate(S, {Y.gen <- sapply(1:n, function(t){
  if(X.post[t] == 1){sample(1:6, 1, prob = rep(1,6))}
  else{
  if(X.post[t] == 2){sample(1:6, 1, prob = p.vec.post)}
  }
})
mean(Y.gen)})
# compute the Bayesian p-value
mean(d.S > mean(as.numeric(Y)))
```

```
## [1] 0.5
```