CARLETON UNIVERSITY

SCHOOL OF
MATHEMATICS AND STATISTICS

HONOURS PROJECT

TITLE: State Inference in the Hidden Markov Models

AUTHOR: Yizhen Li

SUPERVISOR: Dr. Gennady Shaikhet

DATE: May 1, 2021

State Inference in the Hidden Markov Models

Yizhen Li

January 2021

Acknowledgement

For Dr.Shaikhet, couldn't have done it without your guidance and encouragement.

Contents

1 Introduction

We start with the basic definition of Discrete time Markov Chain. Then move to the hidden Markov Models(HMM).

1.1 Discrete time Markov Chain

Definition 1.1.1 (Discrete time Markov Chain) Let S be a countable set. Each element $s \in S$ is the state and S is the state-space.

1) The distribution of X_0 is so called initial distribution v

2) for $k \geqslant 0$, $P(X_{k+1} = j | X_k = i, \ldots X_0 = x_0) = P(X_{k+1} = j | X_k = i) := p_{ij}$, where $(p_{ij} : j \in S)$ is a probability distribution.

For short, we called $(X_k)_{k \geqslant 0}$ as Markov (v, P). P is a transition matrix with $(P)_{ij} = p_{ij}$.

Definition 1.1.2 (Irreducible)

If $P(X_0 = i, X_n = j \text{ for some } n \in N) > 0$ for all $i, j \in S$, we say P is irreducible. That is, we can always reach the state j, starting from state i. $(\forall i, j \in S)$

Definition 1.1.3 (Stationary distribution)

Let π be a probability distribution. π is a stationary distribution of P if and only if $\pi P = \pi$ and π is strictly positive.

For instance, let $S = (0, 1)$

$$\mathbf{P} = \begin{array}{c} \\ 0 \\ 1 \end{array} \begin{array}{c} \begin{array}{cc} 0 & 1 \end{array} \\ \left\| \begin{array}{cc} 3/4 & 1/4 \\ 4/5 & 1/5 \end{array} \right\| \end{array}$$

is a transition matrix on S and $\pi = (16/21, 5/21)$. Moreover, P is irreducible.

1.2 Definition of HMM

Definition 1.2.1 We define the hidden Markov Model $(X_k, Y_k)_{k\geqslant 0}$ as

1)$(X_k)_{k\geqslant 0}$ follows Markov(v, P)

2) The observation process $(Y_k)_{k\geqslant 0}$ is conditionally independent given the hidden chain $(X_k)_{k\geqslant 0}$. The distribution of Y_k depends on X_k only for any given k.

If Y is a discrete random variables, the distribution of Y_k given $(X_k)_{k\geqslant 0}$ can be written as $P(Y_k = y|(X_k)_{k\geqslant 0} = (x_0, x_1, x_2, \dots)) = P(Y_k = y|X_k = x_k) = P_{\theta(x_k)}(Y_k = y)$. $\theta(x_k)$ is the parameter of the distribution. Also $P(Y_0 = y_0, \dots Y_n = y_n|(X_k)_{k\geqslant 0} = (x_0, x_1, x_2, \dots)) \equiv \prod_{k=0}^{n} P(Y_k = y_k|X_k = x_k)$. If Y is a continuous random variables, we should replace the $P(Y_k = y_k|X_k = x_k)$ with the probability density function $g(y_k|X_k = x_k)$.

Such relationship between X_k and Y_k can be described in the following graph

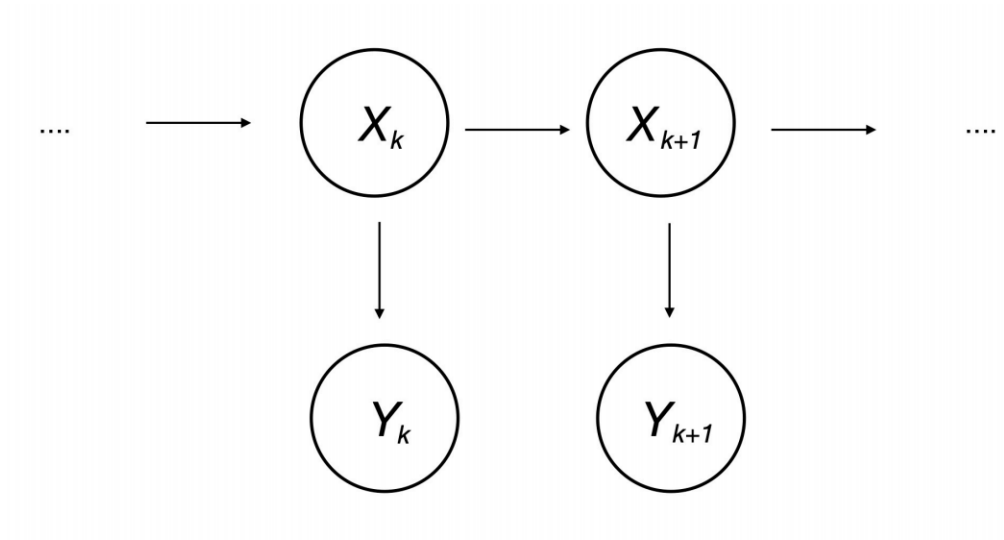

Figure 1.1: $(X_k)_k \geqslant 0$ is the unobservable Markov Chain and $(Y_k)_k \geqslant 0$ is the observable time series

Example 1. Let $(X_k)_{k\geqslant 0}$ follow a two state $(0,1)$ Markov chain with transition probability $p_{01} = P(X_k = 1|X_{k-1} = 0) = 1/4$ and $p_{10} = P(X_k = 0|X_{k-1} = 1) = 4/5$ for $k \geqslant 1$. Therefore, $p_{00} = 1 - p_{01} = 3/4$ and $p_{11} = 1 - p_{10} = 1/5$, the transition probabilities do not depend on time k.

$$\mathbf{P} = \begin{array}{c} \\ 0 \\ 1 \end{array} \begin{array}{c} \begin{array}{cc} 0 & 1 \end{array} \\ \left\| \begin{array}{cc} 3/4 & 1/4 \\ 4/5 & 1/5 \end{array} \right\| \end{array}$$

We also set the initial distribution of X_0 to $v_0 = P(X_0 = 0) = 1/2$ and $v_1 = P(X_0 = 1) = 1/2$. Let $(Y_k)_{k\geqslant 0}$ have the exponential distribution with parameter λ_k with

$$\lambda_k = \begin{cases} 1, & X_k = 0 \\ 2, & X_k = 1, \end{cases}$$

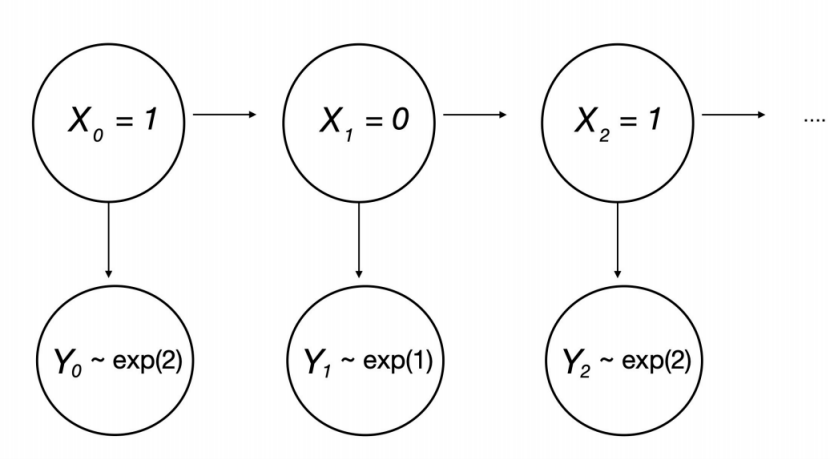

Figure 1.2: $(X_k)_k \geqslant 0$ is the unobservable Markov Chain and $(Y_k)_k \geqslant 0$ is the observable time series

As in figure 1.2, suppose we obtain the $X_0 = 1, X_1 = 0$, and $X_2 = 1$, then Y_0, Y_1, Y_2 are independent exponential variables with parameters 2,1 and 2.

$\triangle$

In this project, we are interested in 2 targets.

1) Use recursive method to find the condition distribution of X_i given the observation process $(Y_k)_{0\leqslant k\leqslant n}$ for $i = 0, 1, \ldots, n$.

2) Apply the simulation to estimate the $E(f(X_0, \ldots, X_n)|(Y_k)_{0\leqslant k\leqslant n})$ for arbitrary bounded real value function $f : S^{(n+1)} \to R$, where S is the state space of $X_k, k = 0, \ldots, n$

We will assume the hidden chain is discrete time finite state Markov Chain in our project. This implies the X_k takes value from a finite set and k is an integer.

1.3 Likelihood for HMM

Example 2. Before talking about the HMM, we start with a bivariate distribution (X, Y) and assume the joint distribution is $f(x, y)$ with (x, y) belong to some set A. So $\int_{(x,y)\in A} f(x, y)dxdy = 1$. We further assume Y is observable and X is unobservable. Thus, the likelihood of Y equal to y is the marginal probability density function $f(y) = \int f(x, y)dx$.

For any bounded function $h : A \to R$, the expected value $E(h(X, Y)|Y = y) = \int h(x, y)f(x|y)dx = \int h(x, y)f(x, y)/f(y)dx$. Therefore, in order to find the conditional expected value of $E(h(X, Y)|Y = y)$, we need to find the $f(y)$, which is the likelihood of y. △

Now, let go back to the conditional expectation of HMM. We assume the observable process Y follows the discrete distribution and $(Y_0, Y_1, \dots Y_n) = (y_0, y_1 \dots y_n)$ is given.

$$
\begin{aligned}
&E(h((X_k)_{a\leqslant k\leqslant b})|(Y_k)_{0\leqslant k\leqslant n} = y_{0:n}) \\
&= \sum_{x_a\in S\dots x_b\in S} h(x_{a:b}) \cdot P((X_k)_{a\leqslant k\leqslant b} = x_{0:n}|(Y_k)_{0\leqslant k\leqslant n} = y_{0:n})) \\
&= \sum_{x_a\in S\dots x_b\in S} h(x_{a:b}) \cdot \frac{P((X_k)_{a\leqslant k\leqslant b} = x_{0:n}, (Y_k)_{0\leqslant k\leqslant n} = y_{0:n}))}{P((Y_k)_{0\leqslant k\leqslant n} = y_{0:n})}
\end{aligned}
$$

where $x_{a:b} := (x_a, x_{a+1}, \dots x_b)$

If the observable process Y follows the continuous distribution, there is a slight difference. Let f be the joint probability density of $Y_0, Y_1, \dots Y_n$. For any bounded function h, we have

$$
\begin{aligned}
&E(h(X_0, Y_0, \dots X_n, Y_n)) = \\
&\int_{Y_0,\dots Y_n} \sum_{(X_0,\dots X_n)=(x_0,\dots x_n)} h(x_0, y_0, \dots x_n, y_n)P(X_0 = x_0, \dots X_n = x_n) \\
&\times f(y_0, \dots y_n|X_0 = x_0, \dots X_n = x_n)dy_0dy_1 \dots dy_n
\end{aligned}
$$

Meanwhile, marginalizing the part of the hidden Markov Chain $X_0, \dots X_n$, we obtain the expectation of any bounded function h with observation process is

$$
\begin{aligned}
&E(h(Y_0, \ldots Y_n)) = \\
&\int_{Y_0, \ldots Y_n} h(y_0, \ldots y_n) \sum_{(X_0, \ldots X_n) = (x_0, \ldots x_n)} P(X_0 = x_0, \ldots X_n = x_n) \\
&\times f(y_0, \ldots y_n | X_0 = x_0, \ldots X_n = x_n) dy_0 dy_1 \ldots dy_n \\
&= \int_{Y_0, \ldots Y_n} h(y_0, \ldots y_n) \times L(y_0, \ldots y_n) dy_0 dy_1 \ldots dy_n
\end{aligned}
$$

$L(y_0, \ldots y_n)$ is the probability density function of $(Y_0, Y_1, \ldots Y_n)$. Finally,

$$
\begin{aligned}
&E(h(X_0, \ldots X_n) | (Y_0, \ldots Y_n) = (y_0, \ldots y_n)) \\
&= L(y_0, \ldots y_n)^{-1} \times \sum_{(X_0, \ldots X_n) = (x_0, \ldots x_n)} h(x_0, \ldots x_n) \times P(X_0 = x_0, \ldots X_n = x_n) \\
&\times f(y_0, \ldots y_n | X_0 = x_0, \ldots X_n = x_n)
\end{aligned}
$$

Now, let us explore some details of $L(y_0, \ldots y_n)$.

Definition of Likelihood 1.3.1 a) Let the $(X_k)_{k \geqslant 0}$ be the finite state Markov(v, P) with state space S. The likelihood of the observable process $(Y_k)_{k \geqslant 0}$ for $\forall (y_0, \ldots, y_n)$ is the probability density function $L(y_0, \ldots y_n)$. We will simply set the conditional probability density of Y_k given X_k as $g(y | X_k = x_k)$. If the value y has been fixed earlier, then we write $g(y | X_k = x_k)$ as $g_k(x_k)$ for short. Since the probability density function g is now a function of x_k given y. By the conditional independency assumption, we can express $f(y_0, \ldots y_n | X_0 = x_0, \ldots X_n = x_n)$ as $\prod_{k=0}^{n} g_k(x_k)$. We will further require $\sup_{-\infty < y < \infty, X_k \in S} g_k(x_k) < \infty$ for all $k \geqslant 0$.

$$
\begin{aligned}
L(y_0, \ldots, y_n) &= \sum_{x_0 \in S} \ldots \sum_{x_n \in S} P(X_0 = x_0, \ldots X_n = x_n) \\
&\times f(y_0, \ldots y_n | X_0 = x_0, \ldots X_n = x_n)
\end{aligned} \tag{1.1}
$$

As we have described earlier, we can write this relation with a slight abuse of notation, omitting the values of y.

$$
= \sum_{x_0 \in S} \ldots \sum_{x_n \in S} P(X_0 = x_0, \ldots X_n = x_n) \times \prod_{k=0}^{n} g_k(x_k) \tag{1.2}
$$

$$= \sum_{x_0 \in S} \dots \sum_{x_n \in S} v(x_0) \times \prod_{k=1}^{n} P(X_k = x_k | X_{k-1} = x_{k-1}) \times \prod_{k=0}^{n} g_k(x_k) \tag{1.3}$$

$$= \sum_{x_0 \in S} \dots \sum_{x_n \in S} v(x_0) \times \prod_{k=1}^{n} p_{x_{k-1}x_k} \times \prod_{k=0}^{n} g_k(x_k) \tag{1.4}$$

We use the conditional independent property of $(Y_k)_{k \geqslant 0}$ at (1.2) and Markov property of $(X_k)_{k \geqslant 0}$ at (1.3). We will simply write $L(y_0, \dots, y_n) = L_n$, since once the $(y_0, \dots, y_n)$ is observed, it is just a constant under our assumption.

Definition of Likelihood 1.3.1 b) We also define $L_n(x_n)$ as the marginal probability density of $(y_0, \dots y_n, x_n)$ with $L_n(x_n) := L(y_0, \dots, y_n, x_n)$ for $n = 0, 1, 2 \dots$. It is marginal probability density of the observable process $(Y_0, Y_1, \dots Y_n) = (y_0, y_1, \dots y_n)$ and $X_n = x_n$.

$$\begin{aligned} L(y_0, \dots, y_n, x_n) &= \sum_{x_0 \in S} \dots \sum_{x_{n-1} \in S} P(X_0 = x_0, \dots X_{n-1} = x_{n-1}, X_n = x_n) \\ &\times f(y_0, \dots y_n | X_0 = x_0, \dots X_n = x_n) \\ &= \sum_{x_0 \in S} \dots \sum_{x_{n-1} \in S} v(x_0) \times \prod_{k=1}^{n-1} p_{x_{k-1}x_k} \times \prod_{k=0}^{n-1} g_k(x_k) \times p_{x_{n-1}x_n} \times g_n(x_n) \end{aligned} \tag{1.5}$$

The difference between L_n and $L_n(x_n)$ is we fixed the last state. It is not difficult to show that

$$L_n = \sum_{x_n \in S} L_n(x_n) \tag{1.6}$$

The introduction of $L_n(x_n)$ is to develop the recursive method.

Theorem 1.3.3 let L_n defined as above for all positive integer n, we have such recursive relationship

$$L_n(x_n) = \sum_{x_{n-1} \in S} L_{n-1}(x_{n-1}) \cdot p_{x_{n-1}x_n} \cdot g_n(x_n) \tag{1.7}$$

Proof. Since $\sup_{-\infty < y < \infty, X_k \in S} g_k(X_k) < \infty$ for all $k \geqslant 0$ by assumption. It is safe to interchange the order of summation

$$
\begin{aligned}
L_n(x_n) &= \sum_{x_0 \in S} \cdots \sum_{x_{n-1} \in S} v(x_0) \times \prod_{k=1}^{n-1} p_{x_{k-1}x_k} \times \prod_{k=0}^{n-1} g_k(x_k) \times p_{x_{n-1}x_n} \times g_n(x_n) \\
&= \sum_{x_{n-1} \in S} g_n(x_n) \cdot p_{x_{n-1}x_n} \sum_{x_0 \in S} \cdots \sum_{x_{n-2} \in S} v(x_0) \times \prod_{k=1}^{n-1} p_{x_{k-1}x_k} \times \prod_{k=0}^{n-1} g_k(x_k) \\
&= \sum_{x_{n-1} \in S} L_{n-1}(x_{n-1}) \cdot p_{x_{n-1}x_n} \cdot g_n(x_n)
\end{aligned}
$$

□

Therefore, we can compute L_n recursively by (1.6) and (1.7).

1.4 Marginal Distribution and Joint distribution in HMM

Definition 1.4.1 Suppose the information of $(Y_k)_{0 \leqslant k \leqslant n}$ is observed as $y_{0:n}$. For $0 \leqslant a \leqslant b \leqslant n$ and $\forall x_{a:b} \in S^{(b-a+1)}$. for any bounded function $h : S^{(b-a+1)} \to R$, We define

$$
\phi_{a:b|n}(h) = E(h((X_k)_{a \leqslant k \leqslant b}) | (Y_k)_{0 \leqslant k \leqslant n} = y_{0:n}) \tag{1.8}
$$

For example, if we let $h = I_{x_{a:b}}$, the indicator function of $X_{a:b} = x_{a:b}$.

$$
\phi_{a:b|n}(I_{x_{a:b}}) = P((X_k)_{a \leqslant k \leqslant b} = x_{a:b} | (Y_k)_{0 \leqslant k \leqslant n} = y_{0:n}) \tag{1.9}
$$

$\phi_{a:b|n}(I_{x_{a:b}})$ is the conditional probability of $(X_k)_{a \leqslant k \leqslant b} = x_{a:b}$, given $(Y_k)_{0 \leqslant k \leqslant n} = y_{0:n}$. If $a = b$, we write (1.9) as $\phi_{a|n}(I_{x_a})$ for short. $\phi_{a|n}(I_{x_a})$ is the marginal distribution of X_a, given $(Y_k)_{0 \leqslant k \leqslant n} = y_{0:n}$.

Now, we are going to develop a recursive method of $\phi_{k|n}(h)$ for any $k = 0, 1, \ldots, n$. We will decompose $\phi_{k|n}(h)$ into forward part α_k and backward part $\beta_{k|n}$. Such recursive method can find the $\phi_{k|n}(h)$ without computing the joint probability density. It is costly to compute the joint probability density when the size of observation is large. Since there have r^n states for $P(X_0 = x_0, \ldots X_n = x_n | (Y_0, \ldots Y_n) = (y_0, \ldots y_n))$. We will show the efficiency of recursive method in Example 5.

Moreover, the recursive method plays an important role in simulating the conditional joint distribution $P(X_0 = x_0, \dots X_n = x_n | (Y_0, \dots Y_n) = (y_0, \dots y_n))$. We will show it in Theorem 1.5.1(Markovian Backward Sampling).

Defination 1.4.2 Let us define some notation first. For $k = 0, 1, \dots, n$. given the observation process. Let h be any bounded function from $S \to R$.

$$\begin{aligned} \alpha_k(y_{0:k}, h) &:= \sum_{x_0 \in S} \dots \sum_{x_k \in S} P(X_0 = x_0, \dots X_k = x_k) \\ &\times f(y_0, \dots y_k | X_0 = x_0, \dots X_k = x_k) \times h(x_k) \\ &= \sum_{x_0 \in S} \dots \sum_{x_k \in S} v(x_0) \times \prod_{i=1}^{k} p_{x_{i-1} x_i} \times \prod_{i=0}^{k} g_i(x_i) \times h(x_k) \end{aligned} \tag{1.10}$$

and

$$\begin{aligned} \beta_{k|n}(y_{k+1:n}, x_k) &:= \sum_{x_{k+1} \in S} \dots \sum_{x_n \in S} P(X_{k+1} = x_{k+1}, \dots X_n = x_n | X_k = x_k) \\ &\times f(y_{k+1}, \dots y_n | X_{k+1} = x_{k+1}, \dots X_n = x_n) \\ &= \sum_{x_{k+1} \in S} \dots \sum_{x_n \in S} \prod_{i=k+1}^{n} p_{x_{i-1} x_i} \times \prod_{i=k+1}^{n} g_i(x_i) \end{aligned} \tag{1.11}$$

Then

$$\phi_{k|n}(h) = L_n^{-1} \cdot \sum_{x_k \in S} h(x_k) \cdot \alpha_k(y_{0:k}, I_{x_k}) \cdot \beta_{k|n}(y_{k+1:n}, x_k) \tag{1.12}$$

where

The $y_{0:k}$ in $\alpha_k(y_{0:k}, h)$ represent the dependence relationship. We omit the writing of $y_{0:k}$ in the rest of project. That is, we will regard $\alpha_k(h)$ as $\alpha_k(y_{0:k}, h)$ and $\beta_{k|n}(x_k)$ as $\beta_{k|n}(y_{k+1:n}, x_k)$ for short. To verify (1.12), we can use changing the order of summation and the Markov property.

Now let us show how to compute $\alpha_k(h)$ and $\beta_{k|n}(x_k)$ recursively. Once we obtain $\alpha_k(h)$ and $\beta_{k|n}(x_k)$ for $k = 0, \dots n$, we are able to compute $\phi_{k|n}(h)$ for all $k = 0, \dots n$ immediately by (1.12).

Theorem 1.4.3(The forward and backward recursive method)

1) For $k = 1, 2, \dots n$ and any bounded real valued function h. The forward method is described as below

$$\alpha_k(h) = \sum_{x_k \in S} h(x_k) \sum_{x_{k-1} \in S} \alpha_{k-1}(I_{x_{k-1}}) \cdot p_{x_{k-1}x_k} \cdot g_k(x_k) \tag{1.13}$$

and initial value is

$$\alpha_0(h) = \sum_{x_0 \in S} h(x_0) \cdot v(x_0) \cdot g_0(x_0) \tag{1.14}$$

2) For $k = n-1, n-2, \dots 0$ The backward method is described as below

$$\beta_{k|n}(x_k) = \sum_{x_{k+1} \in S} p_{x_k x_{k+1}} \cdot g_{k+1}(x_{k+1}) \cdot \beta_{k+1|n}(x_{k+1}) \tag{1.15}$$

and initial value is

$$\beta_{n|n}(x_k) = 1 \tag{1.16}$$

Proof. 1) For forward method, using (1.10) and changing the order of summation, we have

$$\begin{aligned}
\alpha_k(h) &= \sum_{x_0 \in S} \dots \sum_{x_k \in S} P(X_0 = x_0, \dots X_k = x_k) \\
&\quad \times f(y_0, \dots y_k | X_0 = x_0, \dots X_k = x_k) \times h(x_k) \\
&= \sum_{x_0 \in S} \dots \sum_{x_{k-1} \in S} v(x_0) \times \prod_{i=1}^{k} p_{x_{i-1}x_i} \times \prod_{i=0}^{k} g_i(x_i) \times h(x_k) \\
&= \sum_{x_k \in S} h(x_k) \left\{ \sum_{x_0 \in S} \dots \sum_{x_{k-1} \in S} v(x_0) \times \prod_{i=1}^{k-1} p_{x_{i-1}x_i} \times \prod_{i=0}^{k-1} g_i(x_i) \right\} \times p_{x_{k-1}x_k} \times g_k(x_k) \\
&= \sum_{x_k \in S} h(x_k) \sum_{x_{k-1} \in S} \alpha_{k-1}(I_{x_{k-1}}) \cdot p_{x_{k-1}x_k} \cdot g_k(x_k)
\end{aligned}$$

The proof of the backward method follow the same logic, so we omit the detail proof. □

Proposition 1.4.4 For $k = 0, 1, \dots n$

$$\begin{aligned} L_k &= \alpha_k(1) \\ L_n &= \alpha_n(1) = \alpha_k(\beta_{k|n}(x_k)), \forall k = 0, \dots n \end{aligned} \tag{1.17}$$

Proof. We regard $\beta_{k|n}(x_k)$ as a function from S into R and use (1.10), (1.11), and (1.4).

$$\alpha_k(\beta_{k|n}(x_k)) = \sum_{x_0 \in S} \dots \sum_{x_k \in S} v(x_0) \times \prod_{i=1}^{k} p_{x_{i-1}x_i} \times \prod_{i=0}^{k} g_i(x_i) \times \beta_{k|n}(x_k) = L_n$$

□

In conclusion, we start from $\alpha_0(h)$ and compute the $\alpha_1(h), \alpha_2(h), \dots, \alpha_n(h)$ recursively. Similarly, we start from $\beta_n(x_n)$ and compute $\beta_{n-1}(x_{n-1}), \cdot\beta_0(x_0)$. For any given k, we can obtain $\phi_{k|n}(h)$ by (1.10), where $L_n = \alpha_k(\beta_{k|n}(x_k))$

Here is the algorithm for (1.13)

Algorithm 1: Forward Recursion

Input: Initial distribution v; Markov transition Matrix $P = (p_{ij} : i, j \in S)$; Conditional probability density $g_k(x_k) := g(y|x_k)$; Observation $y_{0:n}$

Output: Probability matrix $\alpha = (\alpha_k(I_i) : i \in S, k \in 0, 1, \dots n)$

```
Initialize α = n + 1 × r matrix;
(α)_{0,i} := α_0(I_i)=v(i)g_0(i) for i = 1, ... r;
for k = 1 : n do
    for i = 1 : r do
        (α)_{k,i} := α_k(I_i) = Σ_{j=1}^{r} (φ)_{k-1,j} · p_{ji} · g_k(i)
    end
end
```

The following is the algorithm for (1.15).

However, if some outlier y_k occurs, then the value of $g(y_k|X_k = x_k) := g_k(x_k)$ is close to 0. As a consequence, α_k and β_k will decrease to 0. The following theorem give a normalized method to adjust the α_k as a probability

Algorithm 2: Backward Recursion

Input: Initial distribution v; Markov transition Matrix $P = (p_{ij} : i, j \in S)$; Conditional probability density $g_k(x_k) := g(y|x_k)$; Observation $y_{0:n}$

Output: Probability matrix $\beta = (\beta_{k|n}(i) : i \in S, k \in 0, 1, \dots n)$

```
Initialize β = n + 1 × r matrix;
(β)_{n,i} := β_{n|n}(i) = 1 for i = 1, ... r;
for k = n − 1 : 0 do
    for i = 1 : r do
        (β)_{k,i} := β_{k|n}(i) = Σ_{j=1}^{r} (β)_{k+1,j} · p_{ij} · g_{k+1}(j)
    end
end
```

distribution, in order to avoid the numerical issue and reduce the storage cost for some extremely small α_k and β_k.

Proposition 1.4.5 For any bounded function h: $S \to R$.

$$\phi_{k|k}(h) = \alpha_k(h)/\alpha_k(1) = \alpha_k(h)/L_k \tag{1.18}$$

where $\phi_{k|k}(h) = E(h(X_k)|(Y_i)_{0 \leqslant i \leqslant k} = y_{0:k})$, we will write $\phi_k(h)$ as $\phi_{k|k}(h)$ for short. Applying (1.13) we have

$$\phi_k(h) = L_{k-1}/L_k \cdot \sum_{x_k} h(x_k) \cdot \sum_{x_{k-1}} \phi_{k-1}(I_{x_{k-1}}) \cdot p_{x_{k-1}x_k} \cdot g_k(x_k) \tag{1.19}$$

Now combing (1.12) and (1.18) together, we have

$$\phi_{k|n}(h) = \sum_{x_k \in S} h(x_k) \cdot \phi_k(I_{x_k}) \cdot \left\{ L_n^{-1} \cdot L_k \cdot \beta_{k|n}(x_k) \right\} \tag{1.20}$$

We could define $\hat{\beta}_{k|n}(y_{x_k}) = L_n^{-1} \cdot L_k \cdot \beta_{k|n}(y_{x_k})$ and derived the recursive method based on (1.15). The interested readers are refer to [Inference in HMM, p63]. Since we cannot regard $\hat{\beta}_{k|n}(y_{x_k})$ as a probability distribution, we are not able to normalize this part obviously. Thus, we will introduce the

new method called backward decomposition B_k to replace the $\beta_{k|n}$ in finding $\phi_{k|n}(h)$. The construction of B_k is built on ϕ_k and P entirely.

Let us give an example to show how to update the ϕ_k by hand.

Example 3. Let initial distribution $v = (1/2, 1/2), P = \begin{bmatrix} 1/4 & 3/4 \\ 4/5 & 1/5 \end{bmatrix}$

Observation process Y_k follows exponential distribution with parameter λ_k

$$\lambda_k = \begin{cases} 1, & X_k = 1 \\ 2, & X_k = 2, \end{cases}$$

The observation we obtain is $(y_0, y_1, y_2) = (1, 2, 3)$

Hence,

$\phi_0(I_{x_0=1}) := P(X_0 = 1|Y_0 = 1) = \dfrac{v(1)g_0(1)}{v(1)g_0(1) + v(2)g_0(2)}$

$= \dfrac{1/2 * exp(-1)}{1/2 * exp(-1) + 1/2 * 2 * exp(-2 * 1)} \approx 0.576$

So $\phi_0(I_{x_0=2}) := P(X_0 = 2|Y_0 = 1) = 1 - \phi_0(I_{x_0=1}) \approx 0.424$

Note that it is unnecessary to compute L_{k-1}/L_k. We only need to normalized the ϕ_k as above.

For convenience, we define the symbol $\propto$ as following: the function $f \propto g$ if and only if $f(x) = K \cdot g(x)$ for all $x \in S$ and some constant K

By (1.19), we have

$\phi_1(I_{x_1=1}) := P(X_1 = 1|Y_0 = 1, Y_1 = 2) \propto g_1(1) \cdot \{\phi_0(I_{x_0=1}) \cdot p_{11} + \phi_0(I_{x_0=2}) \cdot p_{21}\} = exp(-2) \cdot \{0.576 \cdot 1/4 + 0.424 \cdot 4/5\} \approx 0.0654$

and

$\phi_1(I_{x_1=2}) := P(X_1 = 2|Y_0 = 1, Y_1 = 2) \propto g_1(2) \cdot \{\phi_0(I_{x_0=1}) \cdot p_{12} + \phi_0(I_{x_0=2}) \cdot p_{22}\} = 2exp(-2 \cdot 2) \cdot \{0.576 \cdot 3/4 + 0.424 \cdot 1/5\} \approx 0.0189$

Hence,

$\phi_1(I_{x_1=1}) \approx 0.0654/(0.0654 + 0.0189) \approx 0.776$

and $\phi_1(I_{x_1=2}) \approx 0.224$

Finally,

$\phi_2(I_{x_2=1}) := P(X_2 = 1|Y_0 = 1, Y_1 = 2, Y_2 = 3) \propto g_2(1) \cdot \{\phi_1(I_{x_1=1}) \cdot p_{11} + \phi_1(I_{x_1=2}) \cdot p_{21}\} = exp(-3) \cdot (0.776 \cdot 1/4 + 0.224 \cdot 4/5) \approx 0.01858$

and

$\phi_2(I_{x_2=2}) := P(X_2 = 2|Y_0 = 1, Y_1 = 2, Y_2 = 3) \propto g_2(2) \cdot \{\phi_1(I_{x_1=1}) \cdot p_{12} + \phi_1(I_{x_1=2}) \cdot p_{22}\} = 2exp(-3 \cdot 2) \cdot (0.776 \cdot 3/4 + 0.224 \cdot 1/5) \approx 0.003107$

Hence,

$$\phi_2(I_{x_1=1}) \approx \frac{0.01858}{0.01858 + 0.003107} = 0.8567 \text{ and } \phi_2(I_{x_1=1}) \approx 0.14328$$

△

Proposition 1.4.6 (Backward Decomposition) for k = 1,2 ... n

$$\phi_{k-1|n}(h) = \sum_{x_k \in S} \phi_{k|n}(I_{x_k}) \cdot B_{k-1}(x_k, h) \tag{1.21}$$

where

$$B_{k-1}(x_k, h) = \sum_{x_{k-1} \in S} \phi_{k-1}(I_{x_{k-1}}) \cdot p_{x_{k-1}x_k} \cdot h(x_{k-1}) / \sum_{x_{k-1} \in S} \phi_{k-1}(I_{x_{k-1}}) \cdot p_{x_{k-1}x_k} \tag{1.22}$$

Since $\phi_{n|n}(h) = \phi_n(h)$ by definition, we can obtain $\phi_n(h)$ by (1.19). Then applying Backward Decomposition to get $\phi_{n-1|n}(h), \ldots \phi_{0|n}(h)$. The following proof can be omit, as we can apply the next theorem 1.5.1 to show (1.21).

Proof. Using (1.10), (1,13) and (1.15)

$$\begin{aligned}
\phi_{k-1|n}(h) &= L_n^{-1} \cdot \sum_{x_{k-1} \in S} h(x_{k-1}) \cdot \alpha_{k-1}(I_{x_{k-1}}) \cdot \beta_{k-1|n}(x_{k-1}) \\
&= L_n^{-1} \cdot \sum_{x_{k-1} \in S} h(x_{k-1}) \cdot \alpha_{k-1}(I_{x_{k-1}}) \times \left\{ \sum_{x_k \in S} p_{x_{k-1}x_k} \cdot g_k(x_k) \cdot \beta_{k|n}(x_k) \right\} \\
&= L_n^{-1} \cdot \sum_{x_k \in S} \beta_{k|n}(x_k) \cdot \sum_{x_{k-1} \in S} L_{k-1} \cdot \phi_{k-1}(I_{x_{k-1}}) \cdot p_{x_{k-1}x_k} \cdot g_k(x_k) \cdot h(x_{k-1}) \\
&= L_n^{-1} \cdot \sum_{x_k \in S} \beta_{k|n}(x_k) \cdot L_{k-1} \cdot \phi_k(I_{x_k}) \cdot B_{k-1}(x_k, h) \\
&= \sum_{x_k \in S} \phi_{k|n}(I_{x_k}) \cdot B_{k-1}(x_k, h)
\end{aligned}$$

□

$B_{k-1}(x_k, I_{x_{k-1}})$ can be viewed as probability distribution of X_{k-1} with parameter x_k. [*Inference in HMM*, p71] pointed out that $B_{k-1}(x_k, I_{x_{k-1}})$ in the form of (1.22) can also be viewed as the Bayesian posterior, where ϕ_{k-1} is the prior distribution and $p_{x_{k-1}x_k}$ is the condition distribution of x_k given x_{k-1}.

Example 4. Maintaining all condition in Example 3. Recall that we already obtain all the value of ϕ_k. $\phi_0(I_{x_0=1}) = 0.576, \phi_0(I_{x_0=2}) = 0.424, \phi_1(I_{x_1=1}) = 0.776, \phi_1(I_{x_1=2}) = 0.224, \phi_2(I_{x_2=1}) = 0.8567$,and $\phi_2(I_{x_2=2}) = 0.14328$.

Hence,

$$B_1(1, I_{x_1=1}) = \frac{\phi_1(I_{x_1=1}) \cdot p_{11}}{\phi_1(I_{x_1=1}) \cdot p_{11} + \phi_1(I_{x_1=2}) \cdot p_{21}} = \frac{0.776 \cdot 1/4}{0.776 \cdot 1/4 + 0.224 \cdot 4/5}$$

= 0.5198 and $B_1(1, I_{x_1=2})$ = 1-0.5198=0.48017.

Similarly,

$$B_1(2, I_{x_1=1}) = \frac{\phi_1(I_{x_1=1}) \cdot p_{12}}{\phi_1(I_{x_1=1}) \cdot p_{12} + \phi_1(I_{x_1=2}) \cdot p_{22}} = \frac{0.776 \cdot 3/4}{0.776 \cdot 3/4 + 0.224 \cdot 1/5}$$

= 0.9285 and $B_1(1, I_{x_1=2})$ = 1-0.9285=0.0715.

Finally,

$\phi_{1|2}(I_{x_1=1}) := P(X_1 = 1|Y_0 = 1, Y_1 = 2, Y_2 = 3) = \phi_2(I_{x_2=1}) \cdot B_1(1, I_{x_1=1}) + \phi_2(I_{x_2=2}) \cdot B_1(2, I_{x_1=1}) = 0.8567 \cdot 0.5198 + 0.14328 \cdot 0.9285 = 0.57834$ and $\phi_{1|2}((I_{x_1=2})) := P(X_1 = 2|Y_0 = 1, Y_1 = 2, Y_2 = 3) = 1 - 0.57834 = 0.42165$

△

Now let us give the algorithm form of (1.19) and (1.21). Assume the state space of hidden chain is $S = \{1, 2, \ldots, r\}$.

For algorithm 3, we further assume the cost of computing $g(x_k, y_k)$ for different y_k is C. The cost of multiplication, summation, and division are 1. For example, the cost of computing $4 \times 2 + 5$ is 2. The cost of computing $\sum_{i=1}^{5} 1/i$ is 9: 5 for division and 4 for summation.

In Algorithm 3, the computational cost for step 2 is rC for computing $g(x_k, y_k)$, r for multiplication, $r - 1$ for summation, and r for division. Total computational cost at step 2 is $(C + 3) \cdot r - 1$. Computational cost for the For Loop is $r \cdot C$ for computing $g(x_k, y_K)$, $2r$ for multiplication, $r - 1$ for

Algorithm 3: Normalized Forward Recursion

Input: Initial distribution v; Markov transition Matrix $P = (p_{ij} : i, j \in S)$; Conditional probability density $g_k(x_k) := g(y|x_k)$; Observation $y_{0:n}$

Output: Probability matrix $\phi = (\phi_k(I_i) : i \in S, k \in 0, 1, \dots n)$

1 Initialize $\phi = n + 1 \times r$ matrix;
2 $(\phi)_{0,i} := \phi_0(I_i) = v(i)g_0(i)/\sum_{i=1}^{r} v(i)g_0(i)$ for $i = 1, \dots r$;
3 **for** $k = 1 : n$ **do**
4 **for** $i = 1 : r$ **do**
5 $(\phi)_{k,i} := \phi_k(I_i) = \sum_{j=1}^{r}(\phi)_{k-1,j} \cdot p_{ji} \cdot g_k(i) / \sum_{i=1}^{r}\sum_{j=1}^{r}(\phi)_{k-1,j} \cdot p_{ji} \cdot g_k(i)$
6 **end**
7 **end**

the numerator summation, $r - 1$ for the numerator summation, $r - 1$ for the denominator summation, and r for the division. Thus, the total cost is $[(C + 4)r - 2] \times rn + (C + 3)r - 1$, which is proportional to $r^2 \times n$.

The computational cost for Algorithm 4 is r^2 for step 3 to step 7, the cost of step 8 is $r(r - 1) + r$ and the cost of step 9 is $(r + r - 1) \times r$. Thus, the total computational cost is $n(4r^2 - r)$, which is also proportional to $r^2 \times n$.

let us show why the forward and backward algorithm is effective. Recall the natural way to compute the marginal distribution $\phi_{k|n}(I_{x_k})$ is

$$\phi_{k|n}(I_{x_k}) = \sum_{x_0,\dots x_{k-1},x_{k+1},\dots,x_n \in S^{n+1}} \phi_{0:n|n}(I_{x_0,\dots x_{k-1},x_k,x_{k+1},\dots,x_n})$$

Let us give a example to explain why we need the forward and backward algorithm

Example 5. Let initial distribution $v = (1/2, 1/2), P = \begin{bmatrix} 1/4 & 3/4 \\ 4/5 & 1/5 \end{bmatrix}$

Observation process Y_k follows exponential distribution with parameter λ_k

$$\lambda_k = \begin{cases} 1, & X_k = 1 \\ 2, & X_k = 2, \end{cases}$$

Algorithm 4: Backward Decomposition

Input: Probability matrix ϕ; Markov transition Matrix $P = (p_{ij} : i, j \in S)$

Output: $\Phi = (\phi_{k|n}(I_i) : i \in S, k \in 0, 1, \dots n)$

```
Initialize Φ = n + 1 × r matrix and (Φ)_{n,i} = (φ)_{n,i}, for i = 1, ... r;
for k = n : 1 do
    for i = 1 : r do
        for j = 1 : r do
            (B_{k-1})_{i,j} := B_{k-1}(i, I_j) = (φ)_{k-1,i} · p_{ij}
        end
    end
    Normalized each row of B_{k-1};
    Update the (k-1)th row of Φ: (Φ)_{(k-1,·)} = (Φ)_{(k,·)} × B_{k-1}
end
```

The observation we obtain is $(y_0, y_1, y_2) = (1, 2, 3)$

Suppose we don't know the forward and backward algorithm. We need to find the joint distribution first. There have 2^3 outcomes. We assume the cost of computing $exp(x)$ for different x is C. The cost of multiplication, summation, and division are 1.

$\phi_{0:3|3}(I_{(1,1,1)}) \propto v(1) \cdot g_0(1) \cdot p_{11} \cdot g_1(1) \cdot p_{11} \cdot g_2(1) = exp(-1 + -2 + -3) \cdot (1/2) \cdot (1/3)^2 = 0.0001377$ Therefore, the cost of computing $\phi_{0:3|3}(I_{(1,1,1)}) \geqslant$ 3C+5: 3C for $exp(-1), exp(-2), exp(-3)$ and 5 for multiplication. We still need to normalized the distribution at the final step, so the cost $\geqslant 3C + 5$

We have 2^3 joint distribution need to compute. The total cost $\geqslant 8 \cdot (3C + 5) = 24C + 40$. It is clear that for general observation size n and state size r. The total cost of computing joint distribution is proportion to $r^n \times (nC + n - 1) \propto r^n \times n$. However, using the compute cost formula for Algorithm 3 and Algorithm 4, the total cost for the forward and backward method is proportion to $r^2 \times n$. Such lager difference is due to the cost of computing the joint distribution.

Finally, we are able to compute the marginal distribution as following $\phi_{0:3|3}(I_{(x_0=1)}) = \phi_{0:3|3}(I_{(1,1,1)}) + \phi_{0:3|3}(I_{(1,1,2)}) + \phi_{0:3|3}(I_{(1,2,1)}) + \phi_{0:3|3}(I_{(1,2,2)})$. △

1.5 Monte Carlo Simulation

[Chib(1996)] gives a very important interpretation of $B_{k-1}(x_k, I_{x_{k-1}})$ for the Monte Carlo Simulation.

Theorem 1.5.1(Markovian Backward Sampling), Given $y_0, \dots y_n$ and $x_0, \dots x_n$

$$\phi_{0:n|n}(I_{x_0,\dots,x_n}) = \phi_n(x_n) \times \prod_{k=1}^{n} B_{k-1}(x_k, I_{x_{k-1}}) \tag{1.23}$$

Therefore, we can simulate the Hidden Markov Chain $x_0, \dots x_n$, given $y_0, \dots y_n$. We first simulate x_n according to distribution $\phi_n(x_n)$, and then simulate the x_{n-1} according to the transition matrix $B_{n-1}(x_n, I_{x_{n-1}}), \dots$,until x_0.

Proof.

$$\begin{aligned}
\phi_{0:n|n}(x_0, \dots x_n) &= P(X_{0:n} = x_{0:n} | Y_{0:n} = y_{0:n}) \\
&= P(X_n = x_n | Y_{0:n} = y_{0:n}) \cdot P(X_{n-1} = x_{n-1} | X_n = x_n, Y_{0:n} = y_{0:n}) \\
&\times \dots \times P(X_k = x_k | X_{k+1} = x_{k+1} \dots X_n = x_n, Y_{0:n} = y_{0:n}) \\
&\times \dots \times P(X_0 = x_0 | X_1 = x_1 \dots X_n = x_n, Y_{0:n} = y_{0:n})
\end{aligned}$$

We may further assume the observation Y follows the discrete distribution. The proof for the continuous case follows the same line.

For $k = 0, 1, \dots$ n

$$\begin{aligned}
&P(X_k = x_k | X_{k+1} = x_{k+1} \dots X_n = x_n, Y_{0:n} = y_{0:n}) \\
&= \frac{P(X_k = x_k, \dots X_n = x_n, Y_{0:n} = y_{0:n})}{P(X_{k+1} = x_{k+1}, \dots X_n = x_n, Y_{0:n} = y_{0:n})} \\
&= \frac{\sum_{x_0 \in S} \dots \sum_{x_{k-1} \in S} P(Y_0 = y_0, \dots Y_n = y_n | X_0 = x_0, \dots X_n = x_n) \times P(X_0 = x_0, \dots X_n = x_n)}{\sum_{x_0 \in S} \dots \sum_{x_k \in S} P(Y_0 = y_0, \dots Y_n = y_n | X_0 = x_0, \dots X_n = x_n) \times P(X_0 = x_0, \dots X_n = x_n)}
\end{aligned}$$

Now, using the conditional independent property of Y given X and the Markov property of X, the numerator of above probability can be written as

$$= \sum_{x_0 \in S} \dots \sum_{x_{k-1} \in S} v(x_0) \{ \prod_{i=0}^{k} P(Y_i = y_i | X_i = x_i) \cdot p_{x_i x_{i+1}} \}$$
$$\times \{ \prod_{i=k+1}^{n-1} P(Y_i = y_i | X_i = x_i) \cdot p_{x_i x_{i+1}} \} \times P(Y_n = y_n | X_n = x_n)$$
$$= \{ \prod_{i=k+1}^{n-1} P(Y_i = y_i | X_i = x_i) \cdot p_{x_i x_{i+1}} \} \times P(Y_n = y_n | X_n = x_n)$$
$$\times \sum_{x_0 \in S} \dots \sum_{x_{k-1} \in S} v(x_0) \{ \prod_{i=0}^{k} P(Y_i = y_i | X_i = x_i) \cdot p_{x_i x_{i+1}} \}$$

Similarly, the denominator can be written as

$$= \{ \prod_{i=k+1}^{n-1} P(Y_i = y_i | X_i = x_i) \cdot p_{x_i x_{i+1}} \} \times P(Y_n = y_n | X_n = x_n)$$
$$\times \sum_{x_0 \in S} \dots \sum_{x_k \in S} v(x_0) \{ \prod_{i=0}^{k} P(Y_i = y_i | X_i = x_i) \cdot p_{x_i x_{i+1}} \}$$

Thus, the part of

$$\{ \prod_{i=k+1}^{n-1} P(Y_i = y_i | X_i = x_i) \cdot p_{x_i x_{i+1}} \} \times P(Y_n = y_n | X_n = x_n)$$

could be cancelled out

Finally,

$$
\begin{aligned}
&P(X_k = x_k | X_{k+1} = x_{k+1} \dots X_n = x_n, Y_{0:n} = y_{0:n}) \\
&= \frac{\sum_{x_0 \in S} \dots \sum_{x_{k-1} \in S} v(x_0) \prod_{i=0}^{k} P(Y_i = y_i | X_i = x_i) \cdot p_{x_i x_{i+1}}}{\sum_{x_0 \in S} \dots \sum_{x_k \in S} v(x_0) \prod_{i=0}^{k} P(Y_i = y_i | X_i = x_i) \cdot p_{x_i x_{i+1}}} \\
&= \frac{\sum_{x_0 \in S} \dots \sum_{x_{k-1} \in S} P(Y_0 = y_0, \dots Y_k = y_k | X_0 = x_0, \dots X_k = x_k) P(X_0 = x_0, \dots X_k = x_k)) p_{x_k x_{k+1}}}{\sum_{x_0 \in S} \dots \sum_{x_k \in S} P(Y_0 = y_0, \dots Y_k = y_k | X_0 = x_0, \dots X_k = x_k) P(X_0 = x_0, \dots X_k = x_k)) p_{x_k x_{k+1}}} \\
&= \frac{\phi_k(I_{x_k}) \cdot p_{x_k x_{k+1}}}{\sum_{x_k \in S} \phi_k(I_{x_k}) \cdot p_{x_k x_{k+1}}} \\
&= B_k(x_{k+1}, I_{x_k})
\end{aligned}
$$

For the continuous case, we need to replace the $P(Y_k = y_k | X_k = x_k)$ as the probability density of $Y_k := g(y|X_k = x_k) = g_k(x_k)$ in the above proof. □

By the strong law of large numbers, we have

$$
\hat{\pi}^{MC}(h) = 1/N \cdot \sum_{i=1}^{N} h(x_0^i, \dots x_n^i) \xrightarrow{a.s.} E(h(X_0, \dots X_n) | Y_0 = y_0, \dots Y_n = y_n) \tag{1.24}
$$

for any bounded function h, where $(x_0^i, \dots x_n^i)$ is simulated according to theorem 1.5.1

2 Markov Chain Monte Carlo(MCMC)

Given the observation, simulating the i.i.d hidden chain according to Theorem 1.5.1 is not the only way to approximate $E(f(X_0 \dots X_n)|Y_0 = y_0, \dots Y_n = y_n)$. Reader may be interested in the approach that are based on the Ergodic theorem. That is, instead of simulating from $P(X_0 = x_0, \dots X_n = x_n|Y_0 = y_0, \dots Y_n = y_n)$ directly, we want to simluate a Markov Chain whose stationary distribution will be $P(X_0 = x_0, \dots X_n = x_n|Y_0 = y_0, \dots Y_n = y_n)$.

2.1 Stationary distribution and limiting probability of Markov Chain

Definition 2.1.1 (Aperiodic) Let X be a finite state discrete time Markov Chain Markov(v, P). We call a state i aperiodic if $p_{ii}^{(n)} := (P^n)_{i,i} > 0$ for all $n > N_x$, where N_x is some constant. It can be show that this statement is equivalent to the statement that the common divisor of the set $\{n \geqslant 1 : p_{ii}^{(n)} > 0\} = 1$. If the common divisor is not equal to 1, we call the state i periodic.

Definition 2.1.2 (Limiting probability) The Markov(v, P) has a limiting distribution if $\lim_{n\to\infty} P(X_n = j|X_0 = i)$ exist for all $i, j \in S$. Also, $\sum_{j\in S} \lim_{n\to\infty} P(X_n = j|X_0 = i) = 1$ Moreover, the limit is independent of the initial state i.

Theorem 2.1.3 Suppose X is a finite state discrete time Markov Chain Markov(v, P). Then $\lim_{n\to\infty} P(X_n = j|X_0 = i)$ exists and is unique for $i, j \in S$ if X is aperiodic and irreducible.

Recll that the $\pi = (\pi_0, \dots \pi_n)$ is the stationary distribution of P if and only if $\pi = \pi \cdot P$, $\sum_{i=1}^{n} \pi_i = 1$ and π is strictly positive.

Proposition 2.1.4 let $\pi_j := \lim_{n\to\infty} P(X_n = j|X_0 = i)$. $\pi := (\pi_0, \dots \pi_n)$ is the stationary distribution of P if π is the limiting distribution of Markov(v, P).

Theorem 2.1.5 The stationary distribution is unique.

Interested reader can refer to [Markov Chain(Norris), Chapter 1, Section 7] for the proof.

Therefore, if a finite state discrete time Markov(v, P) is aperiodic and irreducible, then the stationary distribution is equal to its limiting probability

Theorem 2.1.6 Ergodic theorem Recall that in this project, we only dealing with finite state discrete time Markov Chain: $(X_k)_{0\leqslant k\leqslant n}$ follows Markov(v, P). If P is irreducible and v be arbitrary distribution. For any bounded function $f : S \to R$, we have

$$\frac{1}{N}\sum_{k=0}^{N-1} f(X_k) \xrightarrow{a.s.} \sum_{x\in S} \pi_x \cdot f(x) \tag{2.1}$$

as N goes to infinity, where $\pi = (\pi_x : x \in S)$ is the stationary distribution of X.

The proof of above theorem is based on the Strong Law of large number(SLLN). Interested reader can find the detail proof in [Markov Chain(Norris), Chapter 1, Section 10]

Therefore, if we aim to estimate the $E(f(X))$, we can construct a Markov(v, P) whose stationary distribution is π and X follow the distribution π. Thus, π will automatically be the unique stationary distribution of such artificial Markov Chain. If we simulated such Chain in the long run, the relation(2.1) will guarantee the convergence to $E(f(X))$. The reason why we apply such simulation method(MCMC), rather than simulating by π directly is that it costs less to find the p_{ij} in (2.2) than to find the exact distribution π. In some case, computing the π is impossible and we do not need to know the exact expression of π to perform MCMC.

We use detailed balance to construct such Markov(v, P)

Definition 2.1.7 Detailed balance The transition matrix P and distribution π is in detail balance if

$$\pi_i \cdot p_{ij} = \pi_j \cdot p_{ji} \tag{2.2}$$

for all $i, j \in S$.

Thus, we have $\sum_{j\in S} \pi_i \cdot p_{ij} = \sum_{j\in S} \pi_j \cdot p_{ji}$ by (2.2). This implies $\pi_i = \sum_{j\in S} \pi_j \cdot p_{ji}$ for all $i \in S$ and $\pi = \pi P$. If P is irreducible, then the distribution π is the stationary distribution of P. The part of proving π is strictly positive

is not trivial so is omit. Such stationary distribution is unique by Theorem 2.1.5.

2.2 Metropolis-Hastings Method

In this section, we will talk about how to construct the transition probability p_{ij} in (2.2) by the Metropolis-Hastings Method. We start with a one dimensional distribution X. Suppose $\pi := P(X = x), x \in S$ is the target distribution. Q is some probability transition matrix we need to be defined by ourselves. We use Q to construct the transition probability, by introducing certain "corrections" α. That is, we want to find the $\alpha(x, \hat{x})$ such that

$$\mathbf{P}(\mathbf{X} = \mathbf{x}) \cdot q(x, \hat{x}) \cdot \alpha(x, \hat{x}) = \mathbf{P}(\mathbf{X} = \hat{\mathbf{x}}) \cdot q(\hat{x}, x) \cdot \alpha(\hat{x}, x) \tag{2.3}$$

where $q(i, j)$ is the row i, column j element of matrix Q. For example, if the state space of X is finite with size r, we could define $q(x, \hat{x}) = 1/r$ for $(x, \hat{x}) \in S^2$. Hence, all the elements of Q are equal to $1/r$.

If we take $\alpha(x, \hat{x}) = min\left(\frac{P(X = \hat{x}) \cdot q(\hat{x}, x)}{P(X = x) \cdot q(x, \hat{x})}, 1\right)$. The above equation will be satisfied. The equation (2.3) constructs a new Markov Chain whose stationary distribution is $P(X = x)$. If the new Markoc Chain is irreducibe, the theorem 2.1.1(Ergodic theorem) will guarantee the equation (2.1).

There have two main advantages of applying (2.3):

1) we only require the information of the q and the ratio of $\frac{P(X = \hat{x})}{P(X = x)}$.

2) we can first simulate the $\hat{x}$ by $q(x, \hat{x})$, then deciding the update result with probability $\alpha(x, \hat{x})$. Therefore, we can choose some simple transition matrix Q and compute the $\alpha(x, \hat{x})$ later.

Now let us move to the case when the target distribution is multivariate distribution. That is, X will become a vector is the equation (2.3). We write

$$\tilde{x} = (x_0, \dots x_n)$$

and

$$\tilde{x}_{-k} = (x_0, \dots, x_{k-1}, x_{k+1}, \dots x_n)$$

We let $(x_k, \tilde{x}_{-k}) \equiv \tilde{x}$. When we express $\tilde{x}$ as $(x_k, \tilde{x}_{-k})$, it means we will focus on the kth state of the vector $\tilde{x}$.

Then we have

$$\begin{aligned} &\mathbf{P}(\mathbf{X} = (\mathbf{x_k}, \tilde{\mathbf{x}}_{-\mathbf{k}})) \cdot q((x_k, \tilde{x}_{-k}), (\hat{x}_k, \tilde{x}_{-k}))\alpha((x_k, \tilde{x}_{-k}), (\hat{x}_k, \tilde{x}_{-k})) \\ &= \mathbf{P}(\mathbf{X} = (\hat{\mathbf{x}}_\mathbf{k}, \tilde{\mathbf{x}}_{-\mathbf{k}})) \cdot q((\hat{x}_k, \tilde{x}_{-k}), (x_k, \tilde{x}_{-k}))\alpha((\hat{x}_k, \tilde{x}_{-k}), (x_k, \tilde{x}_{-k})) \end{aligned} \tag{2.4}$$

where $\alpha((x_k, \tilde{x}_{-k}), (\hat{x}_k, \tilde{x}_{-k})) = min\left(\frac{\mathbf{P}(\mathbf{X} = (\hat{\mathbf{x}}_\mathbf{k}, \tilde{\mathbf{x}}_{-\mathbf{k}})) \cdot q((\hat{x}_k, \tilde{x}_{-k}), (x_k, \tilde{x}_{-k}))}{\mathbf{P}(\mathbf{X} = (\mathbf{x_k}, \tilde{\mathbf{x}}_{-\mathbf{k}})) \cdot q((x_k, \tilde{x}_{-k}), (\hat{x}_k, \tilde{x}_{-k}))}, 1\right)$
That is, we fixed all the value of $\tilde{x}$ except x_k. We will only update the value of kth state of $\tilde{x}$ in (2.4). Then $P(X = (x_k, \tilde{x}_{-k}))$ can be regarded as a one dimension probability distribution. We could updated the state X_k according to the order $k = 0, 1, \ldots n$. This is so called systematic scan strategy. Moreover, $P(X = \tilde{x})$ is still the stationary distribution under the systematic scan strategy. Interested reader can find the proof in [Chapter 6, Inference in HMM].

$$(x_0, x_1, \ldots x_n) \longrightarrow (\hat{x}_0, x_1, \ldots x_n) \longrightarrow (\hat{x}_0, \hat{x}_1, \ldots x_n) \longrightarrow \ldots \longrightarrow (\hat{x}_0, \hat{x}_1, \ldots \hat{x}_n)$$

The reason why we transform only one state at a time is that the simulation of multivariate distribution is more complex than the simulation of one dimensional distribution.

We will illustrate two well-known choices of q in Metropolis-Hastings method.

2.3 Systematic-Scan Gibbs Sampling

One of well-known choices of q is use the conditional distribution $P(X_k = \hat{x}_k | \tilde{X}_{-k} = \tilde{x}_{-k}) := P(X_k = \hat{x}_k | X_0 = x_0, \ldots X_{k-1} = x_{k-1}, X_{k+1} = x_{k+1} \ldots X_n = x_n)$. Such choice is called Systematic-Scan Gibbs Sampling. Systematic-Scan Gibbs Sampling is popular in simulating the multivariate distribution whose conditional distribution is easy to find.

Let us show how to construct the (2.4) for multivariate distribution $X := (X_0, ..., X_n)$

$$\begin{aligned}&\mathbf{P}(\mathbf{X}=(\mathbf{x_k},\tilde{\mathbf{x}}_{-\mathbf{k}}))\cdot P(X_k=\hat{x}_k|\tilde{X}_{-k}=\tilde{x}_{-k}))\cdot\alpha((x_k,\tilde{x}_{-k}),(\hat{x}_k,\tilde{x}_{-k}))\\&=\mathbf{P}(\mathbf{X}=(\hat{\mathbf{x}}_{\mathbf{k}},\tilde{\mathbf{x}}_{-\mathbf{k}}))\cdot P(X_k=x_k|\tilde{X}_{-k}=\tilde{x}_{-k})\cdot\alpha((\hat{x}_k,\tilde{x}_{-k}),(x_k,\tilde{x}_{-k}))\end{aligned}\tag{2.5}$$

where $\alpha((x_k,\tilde{x}_{-k}),(\hat{x}_k,\tilde{x}_{-k}))=$

$$min\left(\frac{\mathbf{P}(\mathbf{X}=(\hat{\mathbf{x}}_{\mathbf{k}},\tilde{\mathbf{x}}_{-\mathbf{k}})\cdot P(X_k=\hat{x}_k|\tilde{X}_{-k}=\tilde{x}_{-k})}{\mathbf{P}(\mathbf{X}=(\mathbf{x_k},\tilde{\mathbf{x}}_{-\mathbf{k}})\cdot P(X_k=x_k|\tilde{X}_{-k}=\tilde{x}_{-k})},1\right)=1$$

Therefore, α always equal to one in Systematic-Scan Gibbs sampler. For $k=0,1\dots n$, we can update the state of X_k according to $P(X_k=\hat{x}_k|\tilde{X}_{-k}=\tilde{x}_{-k})$ then such construction is so called systematic-scan Gibbs Sampler.

For regular Gibbs Sampling, we need to choose an index i with probability $\frac{1}{n}$. Then updating the state X_i according to the conditional distribution $P(X_i=\hat{x}_i|\tilde{X}_{-i}=\tilde{x}_{-i})$. So the $q((x_i,\tilde{x}_{-i}),(\hat{x}_i,\tilde{x}_{-i})=\frac{1}{n}\cdot P(X_i=\hat{x}_i|\tilde{X}_{-i}=\tilde{x}_{-i})$. But for the Systematic-Scan Gibbs Sampling as we discussed above, we require updating the state X_k according to some fixed order such as $k=0,1,2,\dots n$.

2.4 Application in HMM(Systematic-Scan Gibbs Sampling)

In this section, we are interested in using Systematic-Scan Gibbs Sampler to simulate the hidden chain $(x_0,\dots x_n)$ according to the distribution $\phi_{0:n|n}(I_{x_0,\dots x_n})$. Such simulation allows us to estimate the value of $E(f(X_0,X_1,\dots,X_n)|(Y_k)_{0\leqslant k\leqslant n}=y_{0:n})$ for arbitrary f.

We want to apply the idea of previous section to the target distribution $\phi_{0:n|n}(I_{x_0,\dots x_n})$, the difference is that now we also have the conditioning on Y's, and we need to recalculate all our transition probability Q values taking into account conditioning on Y's. This is what's done in Theorem 2.4.1. The detail balance equation for X_k is

$$\begin{aligned}&\mathbf{P}(\mathbf{X}=(\mathbf{x_k},\tilde{\mathbf{x}}_{-\mathbf{k}})|(\mathbf{Y_k})_{\mathbf{0\leqslant k\leqslant n}}=\mathbf{y_{0:n}})\times P(X_k=\hat{x}_k|\tilde{X}_{-k}=\tilde{x}_{-k},(Y_k)_{0\leqslant k\leqslant n}=y_{0:n})\\=&\mathbf{P}(\mathbf{X}=(\hat{\mathbf{x}}_{\mathbf{k}},\tilde{\mathbf{x}}_{-\mathbf{k}})|(\mathbf{Y_k})_{\mathbf{0\leqslant k\leqslant n}}=\mathbf{y_{0:n}})\times P(X_k=x_k|\tilde{X}_{-k}=\tilde{x}_{-k},(Y_k)_{0\leqslant k\leqslant n}=y_{0:n})\end{aligned}\tag{2.6}$$

for $k = 0, 1, \ldots, n$

This is the detail balance equation for updating the state of X_k. Where x_k and $\hat{x}_k$ represent the state value of X_k. For $k = 0, 1, \ldots, n$. We should then apply the systematic scan strategy to update the state of $X_0, X_1 \ldots X_n$ one at a time according to the conditional distribution $P(X_k = \hat{x}_k | \tilde{X}_{-k} = \tilde{x}_{-k}, (Y_k)_{0 \leqslant k \leqslant n} = y_{0:n})$.

That is

$(x_0^i, x_1^i, \ldots, x_n^i) \longrightarrow (x_0^{i+1}, x_1^i, \ldots, x_n^i) \longrightarrow (x_0^{i+1}, x_1^{i+1}, \ldots, x_n^i) \longrightarrow \ldots \longrightarrow (x_0^{i+1}, x_1^{i+1}, \ldots, x_n^{i+1})$

If we further assume the irreducibility of this systematic scan method , then we have

$$\frac{1}{N} \sum_{i=1}^{N} f(x_0^i, \ldots x_n^i) \xrightarrow{a.s.} E(f(X_0, \ldots X_n) | Y_0 = y_0, \ldots Y_n = y_n) \tag{2.7}$$

as N goes to infinite. Where x_k^i represents the ith update on the X_k.

Now let us find the condition distribution of X_k given all the information of other X and Y : $P(X_k = \hat{x}_k | \tilde{X}_{-k} = \tilde{x}_{-k}, (Y_k)_{0 \leqslant k \leqslant n} = y_{0:n}) :=$
$P(X_k = \hat{x}_k | X_0 = x_0, \ldots X_{k-1} = x_{k-1}, X_{k+1} = x_{k+1} \ldots X_n = x_n, (Y_k)_{0 \leqslant k \leqslant n} = y_{0:n})$

Theorem 2.4.1

For $k = 1, \ldots, n-1$

$$\begin{aligned} &P(X_k = \hat{x}_k | X_0 = x_0, \ldots X_{k-1} = x_{k-1}, X_{k+1} = x_{k+1} \ldots X_n = x_n, (Y_k)_{0 \leqslant k \leqslant n} = y_{0:n}) \\ &\quad = P(X_k = \hat{x}_k | X_{k-1} = x_{k-1}, X_{k+1} = x_{k+1}, (Y_k)_{0 \leqslant k \leqslant n} = y_{0:n}) \\ &\quad = \frac{p_{\hat{x}_{k-1}\hat{x}_k} \cdot g_k(\hat{x}_k) \cdot p_{\hat{x}_k x_{k+1}}}{\sum_{\hat{x} \in S} p_{\hat{x}_{k-1}\hat{x}_k} \cdot g_k(\hat{x}_k) \cdot p_{\hat{x}_k x_{k+1}}} \end{aligned} \tag{2.8}$$

For $k = 0$

$$\begin{aligned} &P(X_0 = \hat{x}_0 | X_1 = x_1, \ldots X_n = x_n, (Y_k)_{0 \leqslant k \leqslant n} = y_{0:n}) \\ &\quad = P(X_0 = \hat{x_0} | X_1 = x_1, (Y_k)_{0 \leqslant k \leqslant n} = y_{0:n}) \\ &\quad = \frac{v(\hat{x}_0) \cdot g_0(\hat{x}_0) \cdot p_{\hat{x}_0 x_1}}{\sum_{\hat{x}_0 \in S} v(\hat{x}_0) \cdot g_0(\hat{x}_0) \cdot p_{\hat{x}_0 x_1}} \end{aligned} \tag{2.9}$$

For $k = n$

$$\begin{aligned} &P(X_n = \hat{x}_n | X_1 = \hat{x}_1, \dots X_{n-1} = \hat{x}_{n-1}, (Y_k)_{0 \leqslant k \leqslant n} = y_{0:n}) \\ &\quad = P(X_n = \hat{x}_n | X_{n-1} = \hat{x}_{n-1}, (Y_k)_{0 \leqslant k \leqslant n} = y_{0:n}) \\ &\quad = \frac{p_{\hat{x}_{n-1}\hat{x}_n} \cdot g_n(\hat{x}_n)}{\sum_{\hat{x}_n \in S} p_{\hat{x}_{n-1}\hat{x}_n} \cdot g_n(\hat{x}_n)} \end{aligned} \tag{2.10}$$

That is, conditionally on the previous state and the next state value. the distribution of x_k is independent of all the other state. This theorem actually reflect the Markov property of hidden Chain.

Proof. We will only prove the case for $k = 1, \dots n-1$. The proof for $k = 0$ and $k = n$ follow the same logic.

We further assume the observation process Y follows the discrete distribution, the proof of the continuous case follow the same line.

$$\begin{aligned} &P(X_k = \hat{x}_k | X_0 = x_0, \dots X_{k-1} = x_{k-1}, X_{k+1} = x_{k+1} \dots X_n = x_n, (Y_k)_{0 \leqslant k \leqslant n} = y_{0:n}) \\ &\quad = \frac{P(X_k = \hat{x}_k, X_0 = x_0, \dots X_{k-1} = x_{k-1}, X_{k+1} = x_{k+1} \dots X_n = x_n, (Y_k)_{0 \leqslant k \leqslant n} = y_{0:n})}{\sum_{\hat{x}_k \in S} P(X_k = \hat{x}_k, X_0 = x_0, \dots X_{k-1} = x_{k-1}, X_{k+1} = x_{k+1} \dots X_n = x_n, (Y_k)_{0 \leqslant k \leqslant n} = y_{0:n})} \end{aligned}$$

The numerator of the above probability is

$$\begin{aligned} &v(x_0) \times \prod_{i=1}^{k-1} p_{x_{i-1}x_i} \cdot P(Y_i = y_i | X_i = x_i) \times \left\{ p_{x_{k-1}\hat{x}_k} \times P(Y_k = y_k | X_k = \hat{x}_k) \times p_{\hat{x}_k x_{k+1}} \right\} \\ &\times \prod_{i=k+1}^{n-1} p_{x_i x_{i+1}} \cdot P(Y_i = y_i | X_i = x_i) \times P(Y_n = y_n | X_n = x_n) \end{aligned}$$

Similarly, the denominator is

$$\sum_{\hat{x}_k \in S} v(x_0) \times \prod_{i=1}^{k-1} p_{x_{i-1}x_i} \cdot P(Y_i = y_i | X_i = x_i) \times \left\{ p_{x_{k-1}\hat{x}_k} \times P(Y_k = y_k | X_k = \hat{x}_k) \times p_{\hat{x}_k x_{k+1}} \right\}$$
$$\times \prod_{i=k+1}^{n-1} p_{x_i x_{i+1}} \cdot P(Y_i = y_i | X_i = x_i) \times P(Y_n = y_n | X_n = x_n)$$
$$= v(x_0) \times \prod_{i=1}^{k-1} p_{x_{i-1}x_i} \cdot P(Y_i = y_i | X_i = x_i) \times$$
$$\times \prod_{i=k+1}^{n-1} p_{x_i x_{i+1}} \cdot P(Y_i = y_i | X_i = x_i) \times P(Y_n = y_n | X_n = x_n)$$
$$\times \sum_{\hat{x}_k \in S} \left\{ p_{x_{k-1}\hat{x}_k} \times P(Y_k = y_k | X_k = \hat{x}_k) \times p_{\hat{x}_k x_{k+1}} \right\}$$

The part of $v(x_0) \times \prod_{i=1}^{k-1} p_{x_{i-1}x_i} \cdot P(Y_i = y_i | X_i = x_i) \times \times \prod_{i=k+1}^{n-1} p_{x_i x_{i+1}} \cdot P(Y_i = y_i | X_i = x_i) \times P(Y_n = y_n | X_n = x_n)$ could be cancelled out.

Finally,

$$P(X_k = \hat{x}_k | X_0 = x_0, \ldots X_{k-1} = x_{k-1}, X_{k+1} = x_{k+1} \ldots X_n = x_n, (Y_k)_{0 \leqslant k \leqslant n} = y_{0:n})$$
$$= \frac{p_{x_{k-1}\hat{x}_k} \times P(Y_k = y_k | X_k = \hat{x}_k) \times p_{\hat{x}_k x_{k+1}}}{\sum_{\hat{x}_k \in S} p_{x_{k-1}\hat{x}_k} \times P(Y_k = y_k | X_k = \hat{x}_k) \times p_{\hat{x}_k x_{k+1}}}$$

If the observation follows the continuous distribution, we could replace the $P(Y_k = y_k | X_k = x_k)$ by the probability density function $g_k(x_k) := g(y_k | X_k = x_k)$.

□

For algorithm 5, we assume the cost to simulate a r state discrete probability distribution is $h(r)$ and the cost to compute $g_k(x_k)$ is C. The computational cost for step 3 is $Cr + 2r + h(r)$. The computation cost for step 4 to 6 is $(n-1) \times (Cr + 2r + r + h(r))$. The cost for step 7 is the same as step 3. Thus, the total cost is $(n-1)(3+C)r + 2(2+C)r + (n+1)h(r)$.

Here is one small example of the Systematic-Scan Gibbs Sampler in HMM.

Algorithm 5: Systematic-Scan Gibbs Sampler in HMM

Input: $P = (p_{ij} : i, j \in S)$; Conditional probability density $g_k(x_k) = g(y|x_k)$; Observation $y_{0:n}$; The sample size N; The initial hidden chain $(x_0^0, x_1^0, \dots x_n^0)$.

Output: N sample of $(x_0, x_1, \dots x_n)$

Initialize $M = N \times n$ matrix , to store the simulation chain.

for $i = 1 : N$ **do**

  Simulate the x_0^i according to discrete probability distribution $p(x_0) = v(x_0) \cdot g_0(x_0) \cdot p_{x_0 x_1^{i-1}} / \sum_{x_0 \in S} v(x_0) \cdot g_0(x_0) \cdot p_{x_0 x_1^{i-1}}$(equation (2.8));

  for $k = 1 : n-1$ **do**

    Simulate the x_k^i according to discrete probability distribution $p(x_k) = p_{x_{k-1}^i x_k} \cdot g_k(x_k) \cdot p_{x_k x_{k+1}^{i-1}} / \sum_{x_k \in S} p_{x_{k-1}^i x_k} \cdot g_k(x_k) \cdot p_{x_k x_{k+1}^{i-1}}$ (equation (2.9))

  end

  Simulate the x_n^i according to discrete probability distribution $p(x_n) = p_{x_{n-1}^i x_n} \cdot g_n(x_n) / \sum_{x_n \in S} p_{x_{n-1}^i x^n} \cdot g_n(x_n)$ (equation (2.10));

  Store the simulated chain $(x_0^i, x_1^i, \dots x_n^i)$ into the ith row of matrix M

end

Example 6. Let the state size $r = 3$, v be the initial distribution and p_{ij} is the transition probability from state i to state j. Suppose we start at the initial chain (2,2,2,2,2)

Choosing (2,2,2,2,2) as the initial hidden chain. Applying the Algorithm 5. Here is one possible path to update the chain in Algorithm 5.

$$(\mathbf{2}, 2, 2, 2, 2) \to (\mathbf{1}, 2, 2, 2, 2), p = \frac{v(1) \cdot g_0(1) \cdot p_{12}}{v(1) \cdot g_0(1) \cdot p_{12} + v(2) \cdot g_0(2) \cdot p_{22} + v(3) \cdot g_0(3) \cdot p_{32}}$$

$$(1, \mathbf{2}, 2, 2, 2) \to (1, \mathbf{2}, 2, 2, 2), p = \frac{p_{12} \cdot g_1(2) \cdot p_{22}}{p_{11} \cdot g_1(1) \cdot p_{12} + p_{12} \cdot g_1(2) \cdot p_{22} + p_{13} \cdot g_1(3) \cdot p_{32}}$$

$\ldots$

$$(1, 2, 3, 3, \mathbf{2}) \to (1, 2, 3, 3, \mathbf{3}), p = \frac{p_{33} \cdot g_4(3)}{p_{33} \cdot g_4(3) + p_{31} \cdot g_4(1) + p_{32} \cdot g_4(2)}$$

Then above transition is the one step transition in the sense of Systematic-Scan Gibbs Sample.

$\triangle$

Let us start with a simple example.

Example 7. Let $(X_k)_{0 \leqslant k \leqslant n}$ be a hidden Markov(v, P) with state space $\{1, 2\}$, where probability transition matrix P is

$$\mathbf{P} = \begin{array}{c} \\ 1 \\ 2 \end{array} \begin{array}{c} \begin{array}{cc} 1 & 2 \end{array} \\ \left\| \begin{array}{cc} 1/4 & 3/4 \\ 3/4 & 1/4 \end{array} \right\| \end{array}$$

The initial distribution v is (1/5,4/5).

and

$$Y_k = X_k + \begin{cases} 1/2, & p = 1/2 \\ -1/2, & p = 1/2, \end{cases}$$

For instance, $P(Y_k = 5/2 | X_k = 1) = 0$ and $P(Y_k = 3/2 | X_k = 1) = 1/2$

The observed value $(Y_0, Y_1, Y_2) = (3/2, 3/2, 3/2)$ and the original values of the hidden chain X were (2,1,2). Let us try to recover them.

We will also use the true marginal distribution of $\phi_{k|n}(I_{x_k})$ to verify the result of MCMC simulation.

Recall in Chapter 1. To compute the exact $\phi_{k|n}(I_{x_k})$. We need to apply the Algorithm 3 and Algorithm 4 together. We first apply the Algorithm 3 to compute the $\phi_k(I_{x_k})$ with k from 0 to n. After we obtained the $\phi_n(I_{x_n})$ for $x_n \in \{1, 2\}$, we implemented the Algorithm 4 to compute the $\phi_{k|n}(I_{x_k})$ with k from n to 0. Don't forget that $\phi_k(I_{x_k}) := \phi_{k|k}(I_{x_k}) = P(X_k = x_k|(Y_i)_{0 \leqslant i \leqslant k})$.

Now let us apply the Systematic-Scan Gibbs Sampler(Algorithm 3). Choosing (2,1,2) as the initial hidden chain.

After we simulated 1000 hidden chains. Choosing $f = I_{x_k=j}$ for $j = 1, 2$ and $k = 0, 1, 2$. Computing

$$\hat{\phi}_{k|2}(I_{x_k=j}) = 1/1000 \cdot \sum_{i=1}^{1000} I_{x_k=j}(x_0^i, x_1^i, x_2^i)$$

We have

```
[1] "Marginal distribution"
      [,1]  [,2]
[1,] 0.200 0.800
[2,] 0.650 0.350
[3,] 0.425 0.575
[1] "Marginal distribution estimator under Gibbs Sampling"
      [,1]  [,2]
[1,] 0.187 0.813
[2,] 0.663 0.337
[3,] 0.428 0.572
```

Figure 2.1: The row i, column j element of the first matrix represents the conditional distribution of $\phi_{i-1|2}(I_j) := P(X_{i-1} = j|Y_0 = 3/2, Y_1 = 3/2, Y_2 = 3/2)$. The second matrix represent the Systematic-Scan Gibbs Sampler estimator of $(\hat{\phi})_{i,j} = \hat{\phi}_{i-1|2}(I_j) := \hat{P}(X_{i-1} = j|Y_0 = 3/2, Y_1 = 3/2, Y_2 = 3/2)$ based on 1000 sample.

Thus, the Algorithm 5 did a very good estimator for the conditional marginal distribution.

△

Example 8. Let $(X_k)_{0\leqslant k\leqslant n}$ be a hidden Markov(v, P) with state space $\{1, 2, 3\}$, where probability transition matrix P is

$$\mathbf{P} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \left\| \begin{array}{ccc} 1/8 & 5/8 & 2/8 \\ 1/9 & 7/9 & 1/9 \\ 2/5 & 1/5 & 2/5 \end{array} \right\| \end{array}$$

The initial distribution v is (1/3,1/3,1/3).

Let Y be defined as

$$Y_k = \mu(X_k) + \sqrt{s(X_k)} \cdot V_k$$

where $\mu(X)$ and $s(X)$ represent the parameters of the distribution of Y_k. Assume $\mu(\{1, 2, 3\}) = \{-5, 0, 5\}$ and $s(\{1, 2, 3\}) = \{2, 4, 2\}$. $(V_k)_{0\leqslant k\leqslant n}$ is the independent N(0,1) sequence. For example, if $X_k = 1$, then Y_k follows N(-5,2).

Again, let us using the true marginal distribution of $\phi_{k|n}(I_{x_k})$ to verify the result of MCMC simulation.

To do this, we first simulate the $(X_k, Y_k)_{0\leqslant k\leqslant 4}$ by above model. Let the $(Y_k)_{0\leqslant k\leqslant 4}$ we simulated be the observed process. $(Y_0, Y_1, Y_2, Y_3, Y_4) \approx (-4.445, 2.024, -6.553, 3.838, -2.044)$ and the hidden Markov Chain is (1,2,1,2,2)

```
> hmm
     X         Y
[1,] 1 -4.445142
[2,] 2  2.024224
[3,] 1 -6.552508
[4,] 2  3.837976
[5,] 2 -2.044038
```

Figure 2.2: The above matrix represent the $(X_k, Y_k)_{0\leqslant k\leqslant 4}$ we simulated, the first column is the state of hidden chain and the second column is the value of observation. The random seed is 1234 in this R example, interested reader can verify this result by themselves.

Now let us apply the Systematic-Scan Gibbs Sampler(Algorithm 5). Choosing (2,2,2,2,2) as the initial hidden chain.

After we simulated 1000 hidden chains. Choosing $f = I_{x_k=j}$ for $j = 1, 2, 3$ and $k = 0, 1, 2, 3, 4$. Computing

$$\hat{\phi}_{k|4}(I_{x_k=j}) = 1/1000 \cdot \sum_{i=1}^{1000} I_{x_k=j}(x_0^i, x_1^i, x_2^i, x_3^i, x_4^i)$$

We have

```
> round(phi_x,4)
       [,1]   [,2]   [,3]
[1,] 0.9098 0.0902 0.0000
[2,] 0.0000 0.8108 0.1892
[3,] 0.9563 0.0437 0.0000
[4,] 0.0000 0.6256 0.3744
[5,] 0.1195 0.8805 0.0000
> round(phi_mcmc,4)
       [,1]   [,2]   [,3]
[1,] 0.9108 0.0892 0.0000
[2,] 0.0000 0.8093 0.1907
[3,] 0.9571 0.0429 0.0000
[4,] 0.0000 0.6278 0.3722
[5,] 0.1167 0.8833 0.0000
```

Figure 2.3: The row i, column j element of the first matrix represents the conditional distribution of $\phi_{i-1|4}(I_j) := P(X_{i-1} = j|Y_0 = y_0, \dots Y_4 = y_4)$. The second matrix represent the Systematic-Scan Gibbs Sampler estimator of $(\hat{\phi})_{i,j} = \hat{\phi}_{i-1|4}(I_j)$ based on 1000 sample.

We can see that the MCMC estimator did a very good approximation of the real marginal distribution $\phi_{k|n}$. However, the using of MCMC to estimate $\phi_{k|n}$ will cost more than finding the real distribution by Algorithm 3 and 4. Thus, the chain we simulated should be used to estimate $E(f(X_0, X_1, \dots, X_n)|(Y_k)_{0\leqslant k\leqslant n} = y_{0:n})$ for some complex function f. Also, we simulated the X_k condition on the information X_{k-1} and X_{k+1}. This approach will make the simulated samples dependent on each other in general case.

We may use the auto correlation function(ACF) to discover the such dependence of MCMC simulation. The ACF is a function to estimate $\gamma(h) =$

$Cor(x_k, x_{k+h})$. In realistic, we always use the sample auto corrletaion$\hat{\gamma}(h)$ to approximate ACF.

$$\hat{\gamma}(h) = \frac{\frac{1}{N-h}\sum_{i=1}^{N-h}(X_i - \bar{X}_{1\leqslant i\leqslant N-h})(X_{i+h} - \bar{X}_{1+h\leqslant i\leqslant N})}{\frac{1}{N-h}\sqrt{\sum_{i=1}^{N-h}(X_i - \bar{X}_{1\leqslant i\leqslant N-h})^2 \cdot \sum_{i=1}^{N-h}(X_{i+h} - \bar{X}_{1+h\leqslant i\leqslant N})^2}} \tag{2.11}$$

where $\bar{X}_{1\leqslant i\leqslant N-h}$ is the sample mean of $(X_i)_{1\leqslant i\leqslant N-h}$ and $\hat{\gamma}(0) = 1$.

If the sample ACF decrease with h very slowly, then this result suggested such MCMC simulation is not efficient. We may either increase the sample size or use different initial hidden chain to simulate.

For this example, the size of each sample is 5, so we can determine the sample ACF for the state with time index $k = 0, 1, 2, 3, 4$.

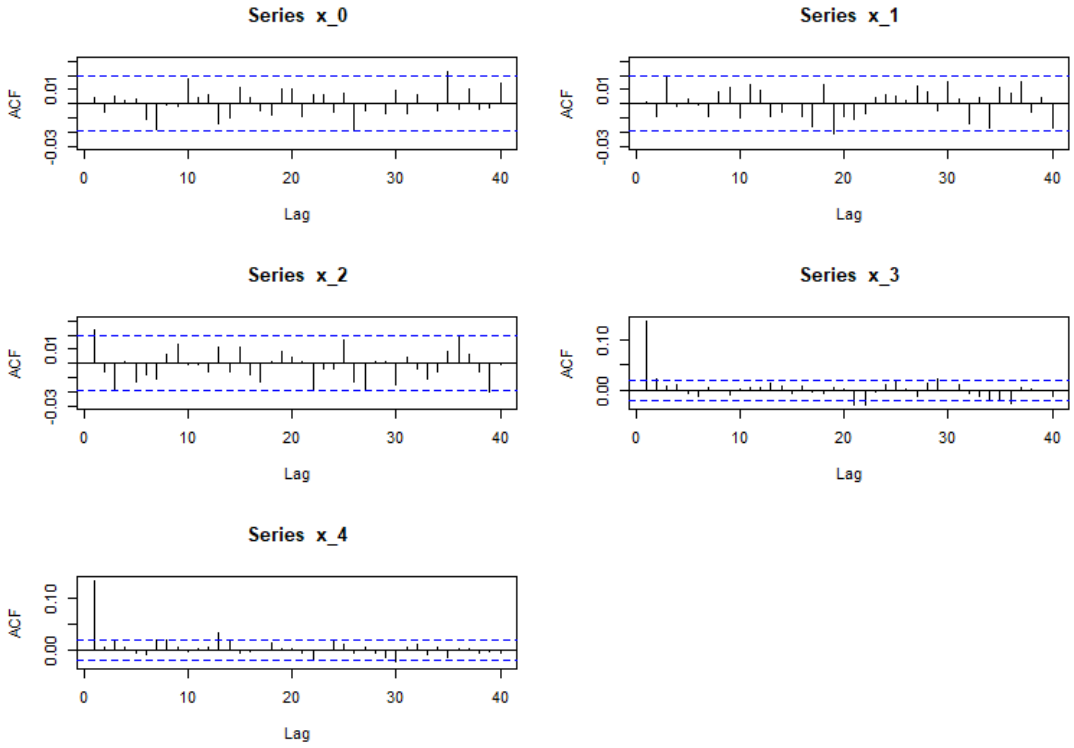

Figure 2.4: The sample ACF plot for $(X_0, \dots X_4)$. There have small peak at X_3 and X_4. Overall, the sample ACF of this MCMC simulation suggests the lower dependency between each sample.

Therefore, we do not need to increase the sample size or choose different initial hidden chain in this example. △

2.5 Random Walk Metropolis-Hastings(RWMH)

In this section, we will talk about another choice of q. Recall in the previous section, we let the q become the conditional distribution. Now we set the q become the discrete uniform distribution.

Let r be the state size of X and $S = \{1, 2, \dots r\}$. Set $q(x, \hat{x})$=1/r. Then $\alpha(x, \hat{x}) = min\left(\frac{P(X = \hat{x})}{P(X = x)}, 1\right)$ in (2.3).

That is

$$\mathbf{P}(\mathbf{X}=\mathbf{x})\cdot 1/r\cdot min\left(\frac{P(X=\hat{x})}{P(X=x)},1\right)=\mathbf{P}(\mathbf{X}=\hat{\mathbf{x}})\cdot 1/r\cdot min\left(\frac{P(X=x)}{P(X=\hat{x})},1\right)$$

Let us show how to construct the (2.4) for multivariate distribution $X:=(X_0,...,X_n)$. We first assume the size of the state X is r.

$$\begin{aligned}&\mathbf{P}(\mathbf{X}=(\mathbf{x_k},\tilde{\mathbf{x}}_{-\mathbf{k}}))\cdot q((x_k,\tilde{x}_{-k}),(\hat{x}_k,\tilde{x}_{-k}))\alpha((x_k,\tilde{x}_{-k}),(\hat{x}_k,\tilde{x}_{-k}))\\&=\mathbf{P}(\mathbf{X}=(\hat{\mathbf{x}}_{\mathbf{k}},\tilde{\mathbf{x}}_{-\mathbf{k}}))\cdot q((\hat{x}_k,\tilde{x}_{-k}),(x_k,\tilde{x}_{-k}))\alpha((\hat{x}_k,\tilde{x}_{-k}),(x_k,\tilde{x}_{-k}))\end{aligned}\tag{2.12}$$

where $q((x_k,\tilde{x}_{-k}),(\hat{x}_k,\tilde{x}_{-k}))=1/r$

and $\alpha((x_k,\tilde{x}_{-k}),(\hat{x}_k,\tilde{x}_{-k}))=min\left(\frac{P(X=(\hat{x}_k,\tilde{x}_{-k}))\cdot 1/r}{P(X=(x_k,\tilde{x}_{-k}))\cdot 1/r},1\right)$

$$=min\left(\frac{P(X_k=\hat{x}_k|\tilde{X}_{-k}=\tilde{x}_{-k}))}{P(X_k=x_k|\tilde{X}_{-k}=\tilde{x}_{-k}))},1\right)$$

Hence

$$\begin{aligned}&\mathbf{P}(\mathbf{X}=(\mathbf{x_k},\tilde{\mathbf{x}}_{-\mathbf{k}}))\cdot\frac{\mathbf{1}}{\mathbf{r}}\cdot\mathbf{min}\left(\frac{\mathbf{P}(\mathbf{X_k}=\hat{\mathbf{x}}_{\mathbf{k}}|\tilde{\mathbf{X}}_{-\mathbf{k}}=\tilde{\mathbf{x}}_{-\mathbf{k}}))}{\mathbf{P}(\mathbf{X_k}=\mathbf{x_k}|\tilde{\mathbf{X}}_{-\mathbf{k}}=\tilde{\mathbf{x}}_{-\mathbf{k}}))},\mathbf{1}\right)\\&=\mathbf{P}(\mathbf{X}=(\hat{\mathbf{x}}_{\mathbf{k}},\tilde{\mathbf{x}}_{-\mathbf{k}}))\cdot\frac{\mathbf{1}}{\mathbf{r}}\cdot\mathbf{min}\left(\frac{\mathbf{P}(\mathbf{X_k}=\mathbf{x_k}|\tilde{\mathbf{X}}_{-\mathbf{k}}=\tilde{\mathbf{x}}_{-\mathbf{k}}))}{\mathbf{P}(\mathbf{X_k}=\hat{\mathbf{x}}_{\mathbf{k}}|\tilde{\mathbf{X}}_{-\mathbf{k}}=\tilde{\mathbf{x}}_{-\mathbf{k}}))},\mathbf{1}\right)\end{aligned}$$

(2.13)

Example 9. Now, let us consider the bivariate distribution example (X,Y) with X takes value in $\{1,2\dots r\}$ and Y takes value in $\{1,2\dots t\}$. r and t are some positive integer value. Define $q_X(x,\hat{x})=1/r$ and $q_Y(y,\hat{y})=1/t$.

$$\begin{aligned}&P(X=x,Y=y)\cdot\frac{1}{r}\cdot min\left(\frac{P(X=\hat{x}|Y=y)}{P(X=x|Y=y)},1\right)\\&=P(X=\hat{x},Y=y)\cdot\frac{1}{r}\cdot min\left(\frac{P(X=x|Y=y)}{P(X=\hat{x}|Y=y)},1\right)\end{aligned}\tag{2.14}$$

Similarly, we have

$$\begin{aligned} & P(X=\hat{x}, Y=y) \cdot \frac{1}{t} \cdot min\left(\frac{P(Y=\hat{y}|X=\hat{x})}{P(Y=y|X=\hat{x})}, 1\right) \\ & = P(X=\hat{x}, Y=\hat{y}) \cdot \frac{1}{t} \cdot min\left(\frac{P(Y=y|X=\hat{x})}{P(Y=\hat{y}|X=\hat{x})}, 1, 1\right) \end{aligned} \tag{2.15}$$

Again, we should apply the Systematic-Scan strategy. We first choose a state of X according to probability $1/r$ and accept this new value with probability $min\left(\frac{P(X=\hat{x}|Y=y)}{P(X=x|Y=y)}, 1\right)$. Then we do the same thing with Y.

$$(x, y) \longrightarrow (\hat{x}, y) \longrightarrow (\hat{x}, \hat{y})$$

We express $\frac{P(X=\hat{x}, Y=y)}{P(X=x, Y=y)}$ as $\frac{P(X=\hat{x}|Y=y)}{P(X=x|Y=y)}$ is because $P(X=\hat{x}|Y=y)$ is easier to obtain in the case we will deal with.

△

Let us discuss a little bit of the difference between the single state transition probability $Q((x,y) \to (\hat{x},y)) := q((x,y) \to (\hat{x},y)) \cdot \alpha((x,y) \to (\hat{x},y))$ under Gibbs sampler and Random walk Metropolis-Hastings method. r is a finite number and $P(X=x|Y=y)$ is the discrete distribution for $x \in S$. We are gonna show the computation cost of applying the RWMH is less than Gibbs sampler when state size is large, However, the state with high condition probability is less likely to jump when we implement the RWMH.

	Gibbs sampler	Random walk Metropolis-Hastings method
$Q((x,y) \to (\hat{x},y)$	$P(X=\hat{x}\|Y=y)$	$\frac{1}{r} \cdot min\left(\frac{P(X=\hat{x}\|Y=y)}{P(X=x\|Y=y)}, 1\right)$

Case 1: Suppose $P(X=x|Y=y) > \frac{\sum_{x \in S} P(X=x|Y=y)}{r} = 1/r$ and we start with (x,y). That is, we start with a state with conditional probability higher than average.

If $P(X = \hat{x}|Y = y) \geqslant P(X = x|Y = y)$, then the transition probability from (x, y) to $(\hat{x}$,y) is $1/r < P(X = \hat{x}|Y = y)$. Hence, it is less likely to transform (x, y) to $(\hat{x}, y)$ under RWMH.

If $P(X = \hat{x}|Y = y) \leqslant P(X = x|Y = y)$, then the transition probability from (x, y) to $(\hat{x}$,y) is $\dfrac{P(X = \hat{x}|Y = y)}{r \cdot P(X = x|Y = y)} < P(X = \hat{x}|Y = y)$ by our assumption. That is, if $P(X = x|Y = y) > 1/r$, it is more likely to stay at the (x, y) in RWMH, comparing to Gibbs Sampling.

Case 2: Suppose $P(X = x|Y = y) < 1/r$ and we start with (x, y). That is, we start with a state with conditional probability lower than average.

If $P(X = \hat{x}|Y = y) \geqslant P(X = x|Y = y)$, then the transition probability from (x, y) to $(\hat{x}, y)$ is 1/r. If $P(X = \hat{x}|Y = y) < P(X = x|Y = y)$ then the transition probability for RWMH equal to $\dfrac{1}{r} \cdot \dfrac{P(X = \hat{x}|Y = y)}{P(X = x|Y = y)} > P(X = \hat{x}|Y = y)$.

The situation in case 2 is not as clear as it in case 1. When the chain reach the state with high probability, it will stay there for a long time in RWMH, comparing to the Gibbs Sampling. Such feature may affect the converge speed significantly in practice.

The above table also implies the compute cost of RWMH method less than the Gibbs Sampler. For the Gibbs Sampler, we need to compute the distribution $P(X = \hat{x}|Y = y)$ first. When the state size of X is large, the advantage of applying the discrete uniform distribution is significant. The RWMH choose the $\hat{x}$ according to the discrete uniform discrete distribution. Then accept it with probability $min\left(\dfrac{P(X = \hat{x}|Y = y)}{P(X = x|Y = y)}, 1\right)$. Hence, we only require the computation of $\dfrac{P(X = \hat{x}|Y = y)}{P(X = x|Y = y)}$ for fixd x and $\hat{x}$, rather than the entire condition distribution $P(X = x|Y = y), x \in S$.

Notice that all the argument in this section is no difficult to extent to the multivariate distribution.

2.6 Application in HMM(RWMH)

Let us talk about the construction of the single site detail balance equation when we let the $q = 1/r$. We should build the detail balance equation as

$$\begin{aligned}&\mathbf{P}(\mathbf{X} = (\mathbf{x_k}, \tilde{\mathbf{x}}_{-\mathbf{k}})|(\mathbf{Y_k})_{\mathbf{0\leqslant k\leqslant n}} = \mathbf{y_{0:n}}) \times \frac{1}{r} \times min\left(\frac{P(X_k = \hat{x}_k|\tilde{X}_{-k} = \tilde{x}_{-k}, (Y_k)_{0\leqslant k\leqslant n} = y_{0:n})}{P(X_k = x_k|\tilde{X}_{-k} = \tilde{x}_{-k}, (Y_k)_{0\leqslant k\leqslant n} = y_{0:n})}, 1\right)\\ =&\mathbf{P}(\mathbf{X} = (\hat{\mathbf{x}}_\mathbf{k}, \tilde{\mathbf{x}}_{-\mathbf{k}})|(\mathbf{Y_k})_{\mathbf{0\leqslant k\leqslant n}} = \mathbf{y_{0:n}}) \times \frac{1}{r} \times min\left(\frac{P(X_k = x_k|\tilde{X}_{-k} = \tilde{x}_{-k}, (Y_k)_{0\leqslant k\leqslant n} = y_{0:n})}{P(X_k = \hat{x}_k|\tilde{X}_{-k} = \tilde{x}_{-k}, (Y_k)_{0\leqslant k\leqslant n} = y_{0:n})}, 1\right)\end{aligned} \tag{2.16}$$

for $k = 0, 1, \ldots, n$

Now applying the Theorem 2.4.1 to simplify the (2.15). We have

$$\begin{aligned}&P(X_0 = \hat{x}_0, \ldots, X_{k-1} = \hat{x}_{k-1}, X_k = x_k, \ldots, X_n = x_n|Y_{0\leqslant k\leqslant n} = y_{0:n})\\ &\cdot \frac{1}{r} \cdot min\left(\frac{p_{\hat{x}_{k-1}\hat{x}_k} \cdot g_k(\hat{x}_k) \cdot p_{\hat{x}_k x_{k-1}}}{p_{\hat{x}_{k-1}x_k} \cdot g_k(x_k) \cdot p_{x_k x_{k-1}}}, 1\right)\\ &= P(X_0 = \hat{x}_0, \ldots, X_{k-1} = \hat{x}_{k-1}, X_k = \hat{x}_k, \ldots, X_n = x_n|Y_{0\leqslant k\leqslant n} = y_{0:n})\\ &\cdot \frac{1}{r} \cdot min\left(\frac{p_{\hat{x}_{k-1}x_k} \cdot g_k(x_k) \cdot p_{x_k x_{k-1}}}{p_{\hat{x}_{k-1}\hat{x}_k} \cdot g_k(\hat{x}_k) \cdot p_{\hat{x}_k x_{k-1}}}, 1\right)\end{aligned} \tag{2.17}$$

for $k = 0, 1, \ldots n$. Recall that we need to adjust the above equation for $k = 0$ and $k = n$.

Here is a simple example about how to simulate the hidden chain under RWMH method.

Example 10. Let the state size $r = 3$, v be the initial distribution and p_{ij} is the transition probability from state i to state j. Suppose we start at the initial chain (2,2,2,2,2)

Here is one possible path to update the chain in Algorithm 6.

$$(\mathbf{2},2,2,2,2) \to (\mathbf{1},2,2,2,2), p = \frac{1}{3} \cdot min\left(\frac{v(1) \cdot g_0(1) \cdot p_{12}}{v(2) \cdot g_0(2) \cdot p_{22}}, 1\right)$$
$$(1,\mathbf{2},2,2,2) \to (1,\mathbf{2},2,2,2), p = \frac{1}{3} \cdot min\left(\frac{p_{12} \cdot g_1(2) \cdot p_{22}}{p_{12} \cdot g_1(2) \cdot p_{22}}, 1\right) = \frac{1}{3}$$
$$\dots$$
$$(1,2,3,3,\mathbf{2}) \to (1,2,3,3,\mathbf{3}), p = \frac{1}{3} \cdot min\left(\frac{p_{33} \cdot g_2(3)}{p_{32} \cdot g_2(2)}, 1\right)$$

$\triangle$

Example 11. We maintain all the condition in Example 8.

Let $(X_k)_{0 \leqslant k \leqslant n}$ is Markov(v, P) with

$$\mathbf{P} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \left\| \begin{array}{ccc} 1/8 & 5/8 & 2/8 \\ 1/9 & 7/9 & 1/9 \\ 2/5 & 1/5 & 2/5 \end{array} \right\| \end{array}$$

The initial distribution v is (1/3,1/3,1/3).

Let Y be defined as

$$Y_k = \mu(X_k) + \sqrt{s(X_k)} \cdot V_k$$

$\mu(\{1,2,3\}) = \{-5,0,5\}$ and $s(\{1,2,3\}) = \{2,4,2\}$. $(V_k)_{0 \leqslant k \leqslant n}$ is the independent N(0,1) sequence.

Again, let us using the true marginal distribution of $\phi_{k|n}(I_{x_k}) := P(X_k = x_k | Y_0 = y_0, \dots Y_k = y_k)$ to verify the result of MCMC simulation.

To do this, we first simulate the $(X_k, Y_k)_{0 \leqslant k \leqslant 4}$ by above model. Let the $(Y_k)_{0 \leqslant k \leqslant 4}$ we simulated be the observed process. $(Y_0, Y_1, Y_2, Y_3, Y_4) \approx (-4.445, 2.024, -6.553, 3.838, -2.044)$ and the hidden Markov Chain is (1,2,1,2,2)

We will compare the algorithm 5 with algorithm 6 in the sense of section 2.5. Let the simulation size $N = 1000$. We have the following output.

Algorithm 6: Random Walk Metropolis-Hastings in HMM

Input: P=$(p_{ij} : i, j \in S)$; Conditional probability density $g_k(x_k) = g(y|x_k)$; Observation $y_{0:n}$; The sample size N; The initial hidden chain $(x_0^0, x_1^0, \dots x_n^0)$.

Output: N sample of $(x_0, x_1, \dots x_n)$

Initialize M = N × n matrix , to store the information of simulation chain.

for $i = 1 : N$ **do**

  Simulate the x_0 with probability 1/r. Compute $p(x_0) = min(\frac{v(x_0) \cdot g_0(x_0) \cdot p_{x_0 x_1^{i-1}}}{v(x_0^{i-1}) \cdot g_0(x_0^{i-1}) \cdot p_{x_0^{i-1} x_1^{i-1}}}, 1)$ (equation (2.17)) ;

  Simulating a uniform(0,1) random variable u;

  if $u < p(x_k)$ **then**

    $x_0^i = x_0$

  else

    $x_0^i = x_0^{i-1}$

  end

  for $k = 1 : n - 1$ **do**

    Simulate the x_k with probability 1/r. Compute $p(x_k) = min\left(\frac{p_{x_{k-1}^i x_k} \cdot g_k(x_k) \cdot p_{x_k x_{k+1}^{i-1}}}{p_{x_{k-1}^i x_k^{i-1}} \cdot g_k(x_k^{i-1}) \cdot p_{x_k^{i-1} x_{k+1}^{i-1}}}, 1\right)$ (equation (2.17)) ;

    Simulating a uniform(0,1) random variable u;

    if $u < p(x_0)$ **then**

      $x_k^i = x_k$

    else

      $x_k^i = x_k^{i-1}$

    end

  end

  Simulate the x_n with probability 1/r. Compute $p(x_n) = min(\frac{p_{x_{n-1}^i x_n} \cdot g_n(x_n)}{p_{x_{n-1}^i x_n^{i-1}} \cdot g_n(x_n^{i-1})}, 1)$ (equation (2.17)) ;

  Simulating a uniform(0,1) random variable u;

  if $u < p(x_n)$ **then**

    $x_n^i = x_n$

  else

    $x_n^i = x_n^{i-1}$

  end

end

```
[1] "Running time of Gibbs Sampling"
Time difference of 1.011295 secs
[1] "Running time of Metropolis-Hastings"
Time difference of 0.5335729 secs
[1] "Marginal distribution"
       [,1]   [,2]   [,3]
[1,] 0.9098 0.0902 0.0000
[2,] 0.0000 0.8108 0.1892
[3,] 0.9563 0.0437 0.0000
[4,] 0.0000 0.6256 0.3744
[5,] 0.1195 0.8805 0.0000
[1] "Marginal distribution estimator under Gibbs Sampling"
       [,1]   [,2]   [,3]
[1,] 0.9108 0.0892 0.0000
[2,] 0.0000 0.8093 0.1907
[3,] 0.9571 0.0429 0.0000
[4,] 0.0000 0.6278 0.3722
[5,] 0.1167 0.8833 0.0000
[1] "Marginal distribution estimator under  Metropolis-Hastings"
       [,1]   [,2]   [,3]
[1,] 0.8945 0.1055 0.0000
[2,] 0.0000 0.8085 0.1915
[3,] 0.9601 0.0399 0.0000
[4,] 0.0001 0.6236 0.3763
[5,] 0.1181 0.8818 0.0001
```

Figure 2.5: The running time of Metropolis-Hastings is significantly less than the running time of Gibbs Sampling. As the size of state increases, such gap will become bigger. Both algorithm achieve very accurate result.

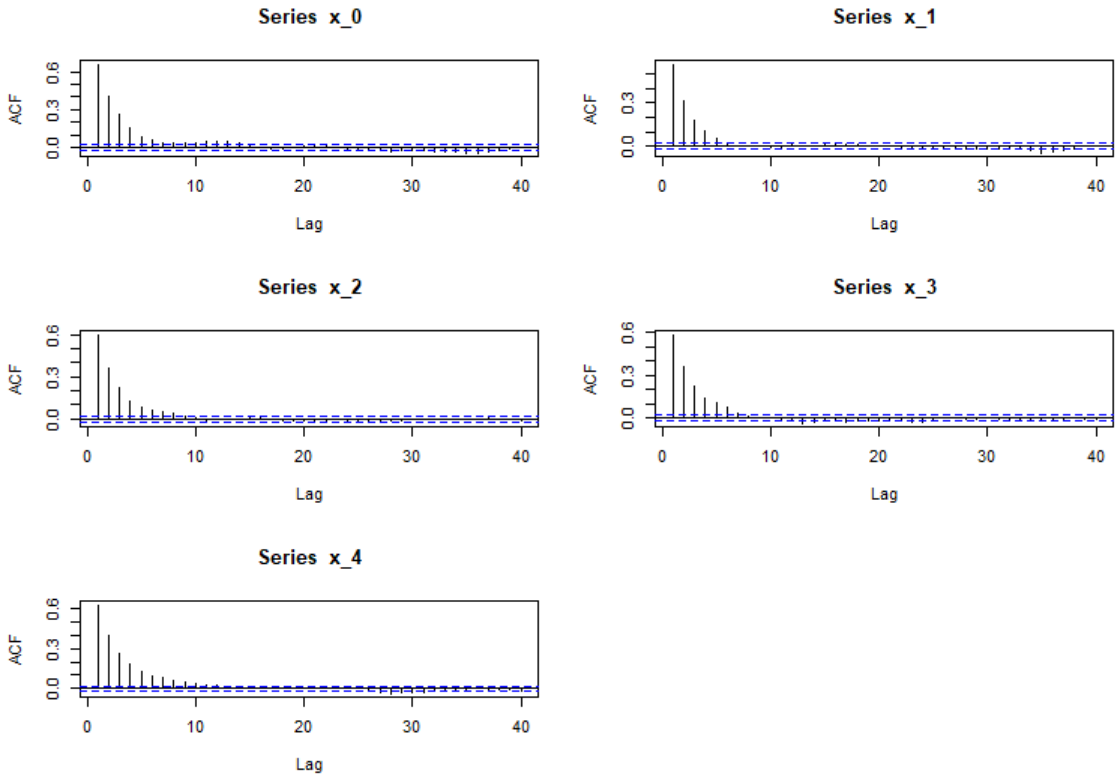

Figure 2.6: The sample ACF suggests the high dependency.

Since the running time of Metropolis-Hastings is small, comparing to Gibbs Sampling. We can increase the simulation size. Also, to reduce the dependency, we should start with several different hidden chain in Metropolis-Hastings Method. △

2.7 Hierarchical Hidden Markov Models

Now let us move to a more complex method which is so called Hierarchical Hidden Markov Model. The Metropolis-Hastings Method plays a very important role in sampling the hidden chain of Hierarchical Hidden Markov Model.

In general, the Hierarchical Hidden Markov Model are defined as follow-

ing:

Defination 2.7.1 (Hierarchical Hidden Markov Model)

1)$(X_k)_{k\geqslant 0}$ follows Markov(v, P)

2)Conditionally on $(X_k)_{k\geqslant 0}$, $(Z_k)_{k\geqslant 0}$ is a non-homogeneous Markov Chain. That is, the transition probability distribution of Z_k depends on the value of Z_{k-1} and X_k

$$\begin{aligned} &P(Z_k = z_k|(Z_0, \dots Z_{k-1}) = (z_0, \dots z_{k-1}); (X_0, \dots X_k) = (x_0, \dots x_k)) \\ &= P(Z_k = z_k|Z_{k-1} = z_{k-1}; X_k = x_k) =: p^{x_k}_{z_{k-1}z_k} \end{aligned} \tag{2.18}$$

The initial distribution of Z_0 is $v^{x_0}(z_0) := P(Z_0 = z_0|X_0 = x_0)$

3)$(Y_k)_{k\geqslant 0}$ are conditionally independent given $(X_k)_{k\geqslant 0}$ and $(Z_k)_{k\geqslant 0}$. The conditional distribution of Y_k depends on (X_k, Z_k) only.

If Y follows a discrete probability distribution, we have

$$\begin{aligned} &P(Y_0 = y_0, \dots Y_n = y_n|X_0 = x_0, \dots X_n = x_n; Z_0 = z_0, \dots Z_n = z_n) \\ &= \prod_{i=0}^{n} P(Y_i = y_i|X_i = x_i; Z_i = z_i) \end{aligned}$$

If Y follows a continuous probability distribution, we can replace the $P(Y_i = y_i|X_i = x_i; Z_i = z_i)$ as $g_i(x_i, z_i) := g(y_i|X_i = x_i; Z_i = z_i)$ as before, where g is the probability density of Y.

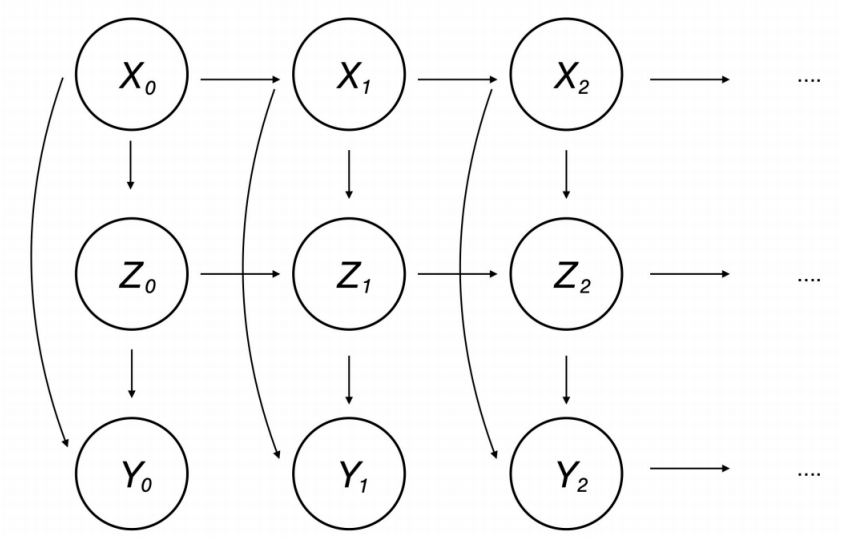

Figure 2.7: Y is the observation, (X, Z) is the hidden chain

We assume the state spaces of X and Z are finite in our project. If Y follows a discrete distribution, we have

$$\begin{aligned}
&P(X_0 = x_0, \dots X_n = x_n; Z_0 = z_0, \dots Z_n = z_n; Y_0 = y_0, \dots Y_n = y_n) \\
&= P(X_0 = x_0, \dots X_n = x_n) \times P(Z_0 = z_0, \dots Z_n = z_n | X_0 = x_0, \dots X_n = x_n) \\
&\times P(Y_0 = y_0, \dots Y_n = y_n | X_0 = x_0, \dots X_n = x_n; Z_0 = z_0, \dots Z_n = z_n) \\
&= v(x_0) \prod_{i=1}^{n} p_{x_{i-1}x_i} \times v(z_0)^{x_0} \prod_{i=1}^{n} p_{z_{i-1}z_i}^{x_i} \times \prod_{i=0}^{n} P(Y_i = y_i | X_i = x_i; Z_i = z_i)
\end{aligned} \tag{2.19}$$

We are interesting in estimating the $P(X_0 = x_0, \dots X_n = x_n; Z_0 = z_0, \dots Z_n = z_n | Y_0 = y_0, \dots Y_n = y_n)$ as before. Then we can get $\phi_{0:n|n}(h) := E(h(X_0, \dots X_n, Z_0, \dots Z_n) | Y_0 = y_0, \dots Y_n = y_n)$.

One way to solve the above problem is make a new Markov Chain(X, Z).

Since

$$
\begin{aligned}
&P((X_k, Z_k) = (x_k, z_k)|(X_0, Z_0) = (x_0, z_0), \dots (X_{k-1}, Z_{k-1}) = (x_{k-1}, z_{k-1})) \\
&= P(X_k = x_k|(X_0, Z_0) = (x_0, z_0), \dots (X_{k-1}, Z_{k-1}) = (x_{k-1}, z_{k-1})) \\
&\times P(Z_k = z_k|X_k = x_k; (X_0, Z_0) = (x_0, z_0), \dots (X_{k-1}, Z_{k-1}) = (x_{k-1}, z_{k-1})) \\
&= P(X_k = x_k|X_{k-1} = x_{k-1}) \times P(Z_k = z_k|Z_{k-1} = z_{k-1}; X_k = x_k) \\
&= p_{x_{k-1}x_k} \times p^{x_k}_{z_{k-1}z_k}
\end{aligned}
$$

which only depends on the values of x_k, x_{k-1} and z_k, z_{k-1}. Then this model again becomes a HMM(Y is the observation process and (X, Z) is the hidden chain). we can use all the previous method to estimate the $\phi_{0:n|n}(h)$

But with the help of the Metropolis-Hastings Method , we don't need to construct any new Markov chain. This will be very useful when the state space of X and Z are particularly large.

We could use the Systematic-Scan strategy to update the state(X, Z) as following:

$$
\begin{aligned}
&\big((x_0^j, z_0^j) \dots (x_n^j, z_n^j)\big) \to \big((x_0^{j+1}, z_0^j) \dots (x_n^j, z_n^j)\big) \to \big((x_0^{j+1}, z_0^{j+1}) \dots (x_n^j, z_n^j)\big) \\
&\to \dots \to \big((x_0^{j+1}, z_0^{j+1}) \dots (x_n^{j+1}, z_n^{j+1})\big)
\end{aligned}
$$

where the transition probability is based on (2.20) and (2.22)

First, let us define some notation .

As before

$$\tilde{x} = (x_0, \dots x_n)$$

and

$$\tilde{x}_{-k} = (x_0, \dots, x_{k-1}, x_{k+1}, \dots x_n)$$

We let $(x_k, \tilde{x}_{-k}) \equiv \tilde{x}$

Similarly,

$$\tilde{z} = (z_0, \dots z_n)$$

and

$$\tilde{z}_{-k} = (z_0, \ldots, z_{k-1}, z_{k+1}, \ldots z_n)$$

Again, let $(z_k, \tilde{z}_{-k}) \equiv \tilde{z}$

We want to construct a new Markov Chain whose stationary distribution is $P(X = \tilde{x}, Z = \tilde{z}|Y_0 = y_0, \ldots Y_n = y_n)$. Applying the same idea as in (2.4)

We have

$$\begin{aligned} &P(X = (x_k, \tilde{x}_{-k}); Z = \tilde{z}|(Y_0, \ldots Y_n) = (y_0, \ldots y_n))) \\ &\cdot q_X((x_k, \tilde{x}_{-k}), (\hat{x}_k, \tilde{x}_{-k})) \cdot \alpha_X((x_k, \tilde{x}_{-k}), (\hat{x}_k, \tilde{x}_{-k})) \\ &= P(X = (\hat{x}_k, \tilde{x}_{-k}); Z = \tilde{z}|(Y_0, \ldots Y_n) = (y_0, \ldots y_n))) \\ &\cdot q_X((\hat{x}_k, \tilde{x}_{-k}), (x_k, \tilde{x}_{-k}) \cdot \alpha_X((\hat{x}_k, \tilde{x}_{-k}), (x_k, \tilde{x}_{-k})) \end{aligned} \tag{2.20}$$

where

$$\begin{aligned} &\alpha_X((x_k, \tilde{x}_{-k}), (\hat{x}_k, \tilde{x}_{-k})) \\ &= min\left(\frac{P(X = (\hat{x}_k, \tilde{x}_{-k}); Z = \tilde{z}|(Y_0, \ldots Y_n) = (y_0, \ldots y_n))) \cdot q_X((\hat{x}_k, \tilde{x}_{-k}), (x_k, \tilde{x}_{-k}))}{P(X = (x_k, \tilde{x}_{-k}); Z = \tilde{z}|(Y_0, \ldots Y_n) = (y_0, \ldots y_n))) \cdot q_X((x_k, \tilde{x}_{-k}), (\hat{x}_k, \tilde{x}_{-k}))}, 1\right) \end{aligned} \tag{2.21}$$

Now we replace the X with Z in (2.20), we have

$$\begin{aligned} &P(Z = (z_k, \tilde{z}_{-k}); X = \tilde{x}|(Y_0, \ldots Y_n) = (y_0, \ldots y_n))) \\ &\cdot q_Z((z_k, \tilde{z}_{-k}), (\hat{z}_k, \tilde{z}_{-k})) \cdot \alpha_Z((z_k, \tilde{z}_{-k}), (\hat{z}_k, \tilde{z}_{-k})) \\ &= P(Z = (\hat{z}_k, \tilde{z}_{-k}); X = \tilde{x}|(Y_0, \ldots Y_n) = (y_0, \ldots y_n))) \\ &\cdot q_Z((\hat{z}_k, \tilde{z}_{-k}), (z_k, \tilde{z}_{-k}) \cdot \alpha_Z((\hat{z}_k, \tilde{z}_{-k}), (z_k, \tilde{z}_{-k})) \end{aligned} \tag{2.22}$$

where

$$\begin{aligned} &\alpha_Z((z_k, \tilde{z}_{-k}), (\hat{z}_k, \tilde{z}_{-k})) \\ &= min\left(\frac{P(X = (\hat{z}_k, \tilde{z}_{-k}); X = \tilde{x}|(Y_0, \ldots Y_n) = (y_0, \ldots y_n))) \cdot q_Z((\hat{z}_k, \tilde{z}_{-k}), (z_k, \tilde{z}_{-k}))}{P(Z = (z_k, \tilde{z}_{-k}); X = \tilde{x}|(Y_0, \ldots Y_n) = (y_0, \ldots y_n))) \cdot q_Z((z_k, \tilde{z}_{-k}), (\hat{z}_k, \tilde{z}_{-k}))}, 1\right) \end{aligned} \tag{2.23}$$

One choice of $q_Z((x_k, \tilde{x}_{-k}), (\hat{x}_k, \tilde{x}_{-k}))$ is $P(X_k = \hat{x}_k|\tilde{X}_{-k} = \tilde{x}_{-k}; Z = \tilde{z}; (Y_0, \ldots Y_n) = (y_0, \ldots y_n))$. This is exactly the Systematic-Scan Gibbs sampler, so $\alpha_X((x_k, \tilde{x}_{-k}), (\hat{x}_k, \tilde{x}_{-k})) = 1$

Similarly, we can let $q_Z((z_k, \tilde{z}_{-k}), (\hat{z}_k, \tilde{z}_{-k}))$ as $P(Z_k = \hat{z}_k | \tilde{Z}_{-k} = \tilde{z}_{-k}; X = \tilde{x}; (Y_0, \ldots Y_n) = (y_0, \ldots y_n))$ and $\alpha_Z((z_k, \tilde{z}_{-k}), (\hat{z}_k, \tilde{z}_{-k})) = 1$

Theorem 2.7.2 Let S^X be the state space of X.

for $k = 1, \ldots, n-1$

$$\begin{aligned} &P(X_k = \hat{x}_k | \tilde{X}_{-k} = \tilde{x}_{-k}; Z = \tilde{z}; (Y_0, \ldots Y_n) = (y_0, \ldots y_n)) \\ &= \frac{p_{x_{k-1}\hat{x}_k} \cdot g_k(\hat{x}_k, z_k) \cdot p^{\hat{x}_k}_{z_{k-1}z_k} \cdot p_{\hat{x}_k x_{k+1}}}{\sum_{\hat{x}_k \in S^X} p_{x_{k-1}\hat{x}_k} \cdot g_k(\hat{x}_k, z_k) \cdot p^{\hat{x}_k}_{z_{k-1}z_k} \cdot p_{\hat{x}_k x_{k+1}}} \end{aligned} \tag{2.24}$$

for $k = 0$

$$\begin{aligned} &P(X_0 = \hat{x}_0 | \tilde{X}_{-0} = \tilde{x}_{-0}; Z = \tilde{z}; (Y_0, \ldots Y_n) = (y_0, \ldots y_n)) \\ &= \frac{v(\hat{x}_0) \cdot g_0(\hat{x}_0, z_0) \cdot v(z_0)^{\hat{x}_0} \cdot p_{\hat{x}_0 x_1}}{\sum_{\hat{x}_0 \in S^X} v(\hat{x}_k) \cdot g_0(\hat{x}_0, z_0) \cdot v(z_0)^{\hat{x}_0} \cdot p_{\hat{x}_0 x_1}} \end{aligned} \tag{2.25}$$

for $k = n$

$$\begin{aligned} &P(X_n = \hat{x}_n | \tilde{X}_{-n} = \tilde{x}_{-n}; Z = \tilde{z}; (Y_0, \ldots Y_n) = (y_0, \ldots y_n)) \\ &= \frac{p_{x_{n-1}\hat{x}_n} \cdot g_n(\hat{x}_n, z_n) \cdot p^{\hat{x}_n}_{z_{n-1}z_n}}{\sum_{\hat{x}_n \in S^X} p_{x_{n-1}\hat{x}_n} \cdot g_n(\hat{x}_n, z_n) \cdot p^{\hat{x}_n}_{z_{n-1}z_n}} \end{aligned} \tag{2.26}$$

Theorem 2.7.3 Let S^Z be the state space of Z.

for k= 1,. . . , n-1

$$\begin{aligned} &P(Z_k = \hat{z}_k | \tilde{Z}_{-k} = \tilde{z}_{-k}; X = \tilde{x}; (Y_0, \ldots Y_n) = (y_0, \ldots y_n)) \\ &= \frac{p^{x_k}_{z_{k-1}\hat{z}_k} \cdot g_k(x_k, \hat{z}_k) \cdot p^{x_{k+1}}_{\hat{z}_k z_{k+1}}}{\sum_{\hat{z}_k \in S^Z} p^{x_k}_{z_{k-1}\hat{z}_k} \cdot g_k(x_k, \hat{z}_k) \cdot p^{x_{k+1}}_{\hat{z}_k z_{k+1}}} \end{aligned} \tag{2.27}$$

for k = 0

$$\begin{aligned} &P(Z_0 = \hat{z}_0 | \tilde{Z}_{-0} = \tilde{z}_{-0}; X = \tilde{x}; (Y_0, \ldots Y_n) = (y_0, \ldots y_n)) \\ &= \frac{v(z_0)^{x_0} \cdot g_0(x_0, \hat{z}_0) \cdot p^{x_1}_{\hat{z}_0 z_1}}{\sum_{\hat{z}_0 \in S^Z} v(z_0)^{x_0} \cdot g_0(x_0, \hat{z}_0) \cdot p^{x_1}_{\hat{z}_0 z_1}} \end{aligned} \tag{2.28}$$

for k = n

$$P(Z_n = \hat{z}_n | \tilde{Z}_{-n} = \tilde{z}_{-n}; X = \tilde{x}; (Y_0, \dots Y_n) = (y_0, \dots y_n))$$
$$= \frac{p^{x_n}_{z_{n-1}\hat{z}_n} \cdot g_n(x_n, \hat{z}_n)}{\sum_{\hat{z}_n \in S^Z} p^{x_n}_{z_{n-1}\hat{z}_n} \cdot g_n(x_n, \hat{z}_n)} \tag{2.29}$$

The proof of the theorem 2.7.2 and 2.7.3 are similar to the proof of theorem 2.4.1. Let us give an intuitive explanation. In the theorem 2.7.2, we replace the $g_k(x_k)$ as $g_k(x_k, z_k) \cdot p^{x_k}_{z_{k-1}z_k}$. In the theorem 2.7.3, since conditionally on the X, Z is a non-homogeneous Markov chain, we replace the $p_{z_{k-1}z_k}$ as $p^{x_k}_{z_{k-1}z_k}$. Actually, in the proof of Theorem 2.4.1, we do not use the property of homogeneous Markov Chain, so there is no problem when we focus on the non-homogeneous Markov chain.

Once the theorem 2.7.2 and theorem 2.7.3 are done, there is no difficult to apply the idea of RWMH.

3 Importance Sampling

In the previous Chapter, we are dealing with the sampling of hidden chain $(x_0, \dots x_n)$ with fixed length $n+1$ given the observation. When the new observation occurs, we need to give up all the previous sampling result and implement the whole algorithm again. But we can avoid such tedious action in the sampling method based on Importance sampling. We will also talking about how to use resampling to solve the weight degeneracy phenomenon in Importance sampling method.

3.1 Introduction of Importance Sampling

Theorem 3.1.1 Strong Law of Large Number let $(X_0, X_1, \dots X_N)$ be independent identical random variables. f be the probability density of X. Then

$$\frac{\sum_{i=1}^N X_i}{N} \xrightarrow{a.s.} E_f(X_1) := \int x \cdot f(x) dx$$

as N goes to infinity. $E_f(X_1)$ means the expect value of X respect to probability density f.

Let $f(x)$ be some probability density function. Notice that $E_f(X) = \int x \cdot f(x)\ dx = \int x \cdot \dfrac{f(x)}{g(x)} \cdot g(x)\ dx =: E_g\left(x \cdot \dfrac{f(x)}{g(x)}\right)$ for other probability density $g(x)$ where $g(x) > 0$ if $f(x) > 0$.

Thus, if we simulate the X according to the probability density $g(x)$, rather than $f(x)$ itself. We have

$$\frac{\sum_{i=1}^{N} h(x_i) \cdot \dfrac{f(x_i)}{g(x_i)}}{N} \xrightarrow{a.s.} E_g\left(h(X) \cdot \frac{f(X)}{g(X)}\right)$$

as N goes to infinity. But

$$E_g\left(h(X) \cdot \frac{f(X)}{g(X)}\right) = \int h(x) \cdot \frac{f(x)}{g(x)} \cdot g(x)dx = E_f[h(X)]$$

Hence,

$$\frac{\sum_{i=1}^{N} h(x_i) \cdot \dfrac{f(x_i)}{g(x_i)}}{N} \xrightarrow{a.s.} E_f[h(X)]$$

The idea of the importance sampling is that the probability density $g(x)$ is more appropriate to simulate in some situation.

Theorem 3.1.2 Let $f(x)$ be a probability density, and $g(x)$ be another probability density which satisfied $g(x) > 0$ if $f(x) > 0$. We want to estimate $E_f[h(X)]$ by simulating X according to probability density g.

$$\mu_N^{IS}(h) := \frac{\sum_{i=1}^{N} h(x_i) \cdot \dfrac{f(x_i)}{g(x_i)}}{\sum_{i=1}^{N} \cdot \dfrac{f(x_i)}{g(x_i)}} \xrightarrow{a.s.} E_f[h(X)] \tag{3.1}$$

Proof. We have

$$\frac{\sum_{i=1}^{N} h(x_i) \cdot \dfrac{f(x_i)}{g(x_i)}}{N} \xrightarrow{a.s.} E_f[h(X)] \tag{3.2}$$

and replacing the function $h(x)$ with constant 1, we obtain

$$\frac{\sum_{i=1}^{N} 1 \cdot \dfrac{f(x_i)}{g(x_i)}}{N} \xrightarrow{a.s.} E_f[1] = 1 \tag{3.3}$$

Finally, let the (3.2) divided by (3.3). We have

$$\mu_N^{IS}(h) = \frac{\sum_{i=1}^{N} h(x_i) \cdot \dfrac{f(x_i)}{g(x_i)}}{\sum_{i=1}^{N} \cdot \dfrac{f(x_i)}{g(x_i)}} \xrightarrow{a.s.} E_f[h(X)] \tag{3.4}$$

□

We may regard

$$w_N^i = \frac{\dfrac{f(x_i)}{g(x_i)}}{\sum_{i=1}^{N} \dfrac{f(x_i)}{g(x_i)}} \tag{3.5}$$

as the normalized importance weight. Also,

$$\sum_{i=1}^{N} w_N^i = \sum_{i=1}^{N} \frac{\dfrac{f(x_i)}{g(x_i)}}{\sum_{i=1}^{N} \dfrac{f(x_i)}{g(x_i)}} = 1$$

.

We can simplify $\mu_N^{IS}(h) = \sum_{i=1}^{N} h(x_i) \cdot w_N^i$. Hence, $\mu_N^{IS}(h)$ can be regarded as the expectation $E_w(h(X))$ of the following discrete probability distribution

$$X = \begin{cases} x_1, & p = w_N^1 \\ x_2, & p = w_N^2 \\ \cdots & \\ x_N, & p = w_N^N \end{cases}$$

This method is so called Sampling Importance Resamping.

Let us give an example to explain why we need $\mu_N^{IS}(h)$ rather than the general Importance sampling estimator.

Example 12. Suppose the probability density function of random variable X is $K \cdot f(x)$ and $K = \dfrac{1}{\int f(x)dx}$ is some unknown constant that is hard to compute. We want to estimate $E_f(h(X))$ by sampling X according to another probability density g.

The constant K makes it hard to construct the general Importance estimator $\dfrac{\sum_{i=1}^N h(x_i) \cdot \dfrac{K \cdot f(x_i)}{g(x_i)}}{N}$. But the

$$\mu_N^{IS}(h) = \frac{\sum_{i=1}^N h(x_i) \cdot \dfrac{K \cdot f(x_i)}{g(x_i)}}{\sum_{i=1}^N \cdot \dfrac{K \cdot f(x_i)}{g(x_i)}} = \frac{\sum_{i=1}^N h(x_i) \cdot \dfrac{f(x_i)}{g(x_i)}}{\sum_{i=1}^N \cdot \dfrac{f(x_i)}{g(x_i)}}$$

and $\mu_N^{IS}(h) \xrightarrow{a.s.} E_f(h(X))$. Hence, we don't need to evaluate the K in $\mu_N^{IS}(h)$.

$\triangle$

3.2 Application in HMM

We want to estimate

$$\begin{aligned}
&\phi_{0:n|n}(h(X_0,\dots,X_n)) := E_f(h(X_0,\dots X_n)|Y_0=y_0,\dots Y_n=y_n)\\
&=\sum_{x_0\in S}\dots\sum_{x_n\in S} h(x_0,\dots x_n)\cdot f(y_0,\dots y_n|X_0=x_0,\dots X_n=x_n)\\
&\times P(X_0=x_0,\dots X_n=x_n)\times f(y_0,\dots y_n)^{-1}\\
&=\sum_{x_0\in S}\dots\sum_{x_n\in S}\frac{h(x_0,\dots x_n)}{L_n}\cdot v(x_0)\prod_{i=1}^{n}p_{x_{i-1}x_i}\cdot\prod_{i=0}^{n}g_i(x_i)
\end{aligned}\tag{3.6}$$

for some bounded function h. $L_n := f(y_0,\dots y_n)$ is the likelihood of $Y_0,\dots Y_n$

$L_n^{-1}\cdot v(x_0)\prod_{i=1}^{n}p_{x_{i-1}x_i}\cdot\prod_{i=0}^{n}g_i(x_i)$ is the conditional probability of chain $(x_0,\dots x_n)$. We can construct another distribution $Q(X_0=x_0,\dots X_n=x_n)$ such that $Q(X_0=x_0,\dots X_n=x_n)>0$ if $L_n^{-1}\cdot v(x_0)\prod_{i=1}^{n}p_{x_{i-1}x_i}\cdot\prod_{i=0}^{n}g_i(x_i)>0$ for any n. We will talk about the choices for Q, later, in section 3.3, and right now we can concentrate on the mathematics of the estimation, We have

$$\begin{aligned}
\phi_{0:n|n}(h(X_0,\dots,X_n)) &= \sum_{x_0\in S}\dots\sum_{x_n\in S}\frac{h(x_0,\dots x_n)}{L_n}\cdot v(x_0)\prod_{i=1}^{n}p_{x_{i-1}x_i}\cdot\prod_{i=0}^{n}g_i(x_i)\\
&=\sum_{x_0\in S}\dots\sum_{x_n\in S}h(x_0,\dots x_n)\cdot\frac{L_n^{-1}\cdot v(x_0)\prod_{i=1}^{n}p_{x_{i-1}x_i}\cdot\prod_{i=0}^{n}g_i(x_i)}{Q(X_0=x_0,\dots X_n=x_n)}\\
&\cdot Q(X_0=x_0,\dots X_n=x_n)
\end{aligned}\tag{3.7}$$

Thus, we can first simulate the $(x_0,\dots x_n)$ according to the joint distribution $Q(X_0=x_0,\dots X_n=x_n)$ and then evaluate the function $h(x_0,\dots x_n)\cdot\frac{L_n^{-1}\cdot v(x_0)\prod_{i=1}^{n}p_{x_{i-1}x_i}\cdot\prod_{i=0}^{n}g_i(x_i)}{Q(X_0=x_0,\dots X_n=x_n)}$. Recall that we needed a recursive algorithm(Theorem 1.3.3) to evaluate L_n, we can avoid this as in example 12, using as constant K.

We regard $L_n^{-1}\cdot v(x_0)\prod_{i=1}^{n}p_{x_{i-1}x_i}\cdot\prod_{i=0}^{n}g_i(x_i)$ as f and $Q(X_0=x_0,\dots X_n=x_n)$ as g in (3.4).

Therefore

$$\mu_N^{IS}(h) = \frac{\sum_{i=1}^N h(x_i)\cdot\dfrac{f(x_i)}{g(x_i)}}{\sum_{i=1}^N \cdot\dfrac{f(x_i)}{g(x_i)}} = \frac{\sum_{j=1}^N h(x_0^j,\dots x_n^j)\cdot\dfrac{v(x_0^j)\prod_{i=1}^n p_{x_{i-1}^j x_i^j}\cdot\prod_{i=0}^n g_i(x_i^j)}{Q(X_0=x_0^j,\dots X_n=x_n^j)}}{\sum_{j=1}^N \dfrac{v(x_0^j)\prod_{i=1}^n p_{x_{i-1}^j x_i^j}\cdot\prod_{i=0}^n g_i(x_i^j)}{Q(X_0=x_0^j,\dots X_n=x_n^j)}}$$
$$\xrightarrow{a.s.} E_f(h(X_0,\dots X_n)|Y_0=y_0,\dots Y_n=y_n) \tag{3.8}$$

as N goes to infinity.

Such expression do not require the estimation of L_n. Since L_n is the constant K in example 12 which will be canceled out. If we set

$$w_n^j = \frac{\dfrac{f(x_i)}{g(x_i)}}{\sum_{i=1}^N \dfrac{f(x_i)}{g(x_i)}} = \frac{\dfrac{v(x_0^j)\prod_{i=1}^n p_{x_{i-1}^j x_i^j}\cdot\prod_{i=0}^n g_i(x_i^j)}{Q(X_0=x_0^j,\dots X_n=x_n^j)}}{\sum_{j=1}^N \dfrac{v(x_0^j)\prod_{i=1}^n p_{x_{i-1}^j x_i^j}\cdot\prod_{i=0}^n g_i(x_i^j)}{Q(X_0=x_0^j,\dots X_n=x_n^j)}} \tag{3.9}$$

Notice that $\sum_{j=1}^N w_n^j = 1$. Then we could regard (3.8) as $\sum_{j=1}^N h(x_0^j,\dots x_n^j)\cdot w_n^j$,where w_n^j is the weight of each sample. We related the symbol w_n^j with n is because, when we obtain the new observation $Y_{n+1}=y_{n+1}$. We could update the w_n^j recursively.

Theorem 3.2.1 If $f(x) = K\cdot g(x)$ for any x and given constant K, we write $f\propto g$. For $j=1,\dots N$, we have

$$w_{n+1}^j \propto w_n^j \cdot \frac{p_{x_n^j x_{n+1}^j}\cdot g_{n+1}(x_{n+1}^j)}{Q(X_{n+1}=x_{n+1}^j|X_0=x_0^j,\dots X_n=x_n^j)} \tag{3.10}$$

and $\sum_j^N w_{n+1}^j = 1$

Proof.

$$w_{n+1}^j = \frac{\frac{v(x_0^j)\prod_{i=1}^{n+1} p_{x_{i-1}^j x_i^j} \cdot \prod_{i=0}^{n+1} g_i(x_i^j)}{Q(X_0 = x_0^j, \dots X_{n+1} = x_{n+1}^j)}}{\sum_{j=1}^N \frac{v(x_0^j)\prod_{i=1}^{n+1} p_{x_{i-1}^j x_i^j} \cdot \prod_{i=0}^{n+1} g_i(x_i^j)}{Q(X_0 = x_0^j, \dots X_{n+1} = x_{n+1}^j)}}$$

$$= \frac{v(x_0^j)\prod_{i=1}^{n} p_{x_{i-1}^j x_i^j} \cdot \prod_{i=0}^{n} g_i(x_i^j)}{Q(X_0 = x_0^j, \dots X_n = x_n^j)} \cdot K_{n+1}^{-1} \cdot \frac{p_{x_n^j x_{n+1}^j} \cdot g_{n+1}(x_{n+1}^j)}{Q(X_{n+1} = x_{n+1}^j | X_0 = x_0^j, \dots X_n = x_n^j)}$$

where $K_{n+1} = \sum_{j=1}^N \frac{v(x_0^j)\prod_{i=1}^{n+1} p_{x_{i-1}^j x_i^j} \cdot \prod_{i=0}^{n+1} g_i(x_i^j)}{Q(X_0 = x_0^j, \dots X_{n+1} = x_{n+1}^j)}$.

Hence,

$$w_{n+1}^j = w_n^j \cdot \frac{p_{x_n^j x_{n+1}^j} \cdot g_{n+1}(x_{n+1}^j)}{Q(X_{n+1} = x_{n+1}^j | X_0 = x_0^j, \dots X_n = x_n^j)} \cdot \frac{K_n}{K_{n+1}}$$

$$\propto w_n^j \cdot \frac{p_{x_n^j x_{n+1}^j} \cdot g_{n+1}(x_{n+1}^j)}{Q(X_{n+1} = x_{n+1}^j | X_0 = x_0^j, \dots X_n = x_n^j)}$$

for $j = 1, \dots N$

□

Thus, we simulate the new state of the jth chain according to $Q(X_{n+1} = x_{n+1}^j | X_0 = x_0^j, \dots X_n = x_n^j)$. Updating the w_{n+1}^j by (3.10) and estimating the $\phi_{0:n+1|n+1}(h(X_0, \dots, X_n, X_{n+1}))$ by $\sum_{j=1}^N h(x_0^j, \dots x_n^j, x_{n+1}^j) \cdot w_{n+1}^j$.

We define

$$\hat{\phi}_{0:n+1|n+1}^{IS,N}(h) := \sum_{j=1}^N h(x_0^j, \dots x_n^j, x_{n+1}^j) \cdot w_{n+1}^j \tag{3.11}$$

This is a very important advantage, comparing to MCMC method. Since the new observation will change the stationary distribution: $\phi_{0:n|n} \to \phi_{0:n+1|n+1}$ We need to reset the MCMC simulation entirely. However, under the Importance sampling, we only need to update the new weight w_{n+1}^j. Moreover,

if the conditional distribution of the hidden chain is not strictly positive, so it cannot be the stationary distribution. We are still able to estimate the $\phi_{0:n|n}(f(X_0,\dots,X_n))$ by importance sampling. This is because the estimator of Importance sampling is constructed on SLLN, but the estimator of MCMC is constructed on Theorem 2.1.1 Ergodic theorem.

Algorithm 7: Importance Sampling in HMM

Input: P=$(p_{ij} : i,j \in S)$; Conditional probability density $g_k(x_k) = g(y|x_k)$; Observation $y_{0:n}$; The sample size N; The joint distribution Q.

Output: N sample of $(x_0, x_1, \dots x_n)$;W=$((W)_{i,j} = w_i^j, 1 \leqslant j \leqslant N, 0 \leqslant i \leqslant n)$

Initialize $M = N \times n + 1$ matrix , to store the simulation chain;
$W = N \times n + 1$ matrix , to store the w_i^j
for $j = 1 : N$ **do**
  Simulate the x_0^j according to distribution $Q(X_0 = x_0)$;
  $w_0^j = v(x_0^j) \cdot g_0(x_0^j)/Q(X_0 = x_0^j)$
end
Normalized w_0^j
for $i = 1 : n$ **do**
  for $j = 1 : N$ **do**
    Simulate the x_i^j according to the probability distribution $Q(X_i = x_i|X_{i-1} = x_{i-1}^j, \dots X_0 = x_0^j)$
    $$w_i^j = w_{i-1}^j \cdot \frac{p_{x_{i-1}^j x_i^j} \cdot g_i(x_i^j)}{Q(X_i = x_i^j|X_0 = x_0^j, \dots X_{i-1} = x_{i-1}^j)}$$
  end
  Normalized w_i^j and saved in the matrix W
end

Finally, we can use $\sum_{j=1}^N h(x_0^j, \dots x_k^j) \cdot w_k^j$ to estimate the $\phi_{0:k|k}(h(X_0,\dots,X_k))$ for any $k \in \{0, 1, \dots n\}$ and bounded function h.

3.3 Prior Kernel

The natural choice of Q is $P :=$ transition matrix of X, without any observation. That is $Q(X_0 = x_0, \dots X_n = x_n) = v(x_0) \cdot p_{x_0x_1} \dots p_{x_{n-1}x_n}$. Such Q is called prior kernel. Also, $v(x_0) \cdot p_{x_0x_1} \dots p_{x_{n-1}x_n} > 0$ if $L_n^{-1} \cdot v(x_0) \prod_{i=1}^n p_{x_{i-1}x_i} \cdot \prod_{i=0}^n g_i(x_i) > 0$.

We have

$$\begin{aligned} w_{n+1}^j &\propto w_n^j \cdot \frac{p_{x_n^j x_{n+1}^j} \cdot g_{n+1}(x_{n+1}^j)}{p_{x_n^j x_{n+1}^j}} = w_n^j \cdot g_{n+1}(x_{n+1}^j) \\ w_0^j &\propto \frac{v(x_0^j) \cdot g_0(x_0^j)}{v(x_0^j)} = g_0(x_0^j) \end{aligned} \tag{3.12}$$

Example 13. We maintain all the condition in Example 9 Applying the Algorithm 7 with prior kernel. We have the following result.

```
[1] "Marginal distribution"
       [,1]   [,2]   [,3]
[1,] 0.9098 0.0902 0.0000
[2,] 0.0000 0.8108 0.1892
[3,] 0.9563 0.0437 0.0000
[4,] 0.0000 0.6256 0.3744
[5,] 0.1195 0.8805 0.0000
[1] "Marginal distribution under Importance Sampling"
       [,1]   [,2]   [,3]
[1,] 0.9042 0.0958 0.0000
[2,] 0.0000 0.8494 0.1506
[3,] 0.9607 0.0393 0.0000
[4,] 0.0000 0.6922 0.3078
[5,] 0.0762 0.9238 0.0000
```

Figure 3.1: Let h=$I_{\{X_k=i\}}$. Estimating $\phi_{k|4}(I_i)$ by $\sum_{j=1}^N I_{\{x_k^j=i\}} \cdot w_i^j$. The sample size N=1000

The result is not good, comparing to the MCMC method. After increasing the sample size N to 10000. The importance sampling method converge to the true marginal distribution. The intuitive reason to explain such slow converge is that the simulation step didn't consider the information of observation. But the MCMC method consider the observation in each simulation.

In order to address this problem, we introduce another distribution Q, called Optimal Instrumental Kernel. $\triangle$

3.4 Optimal Instrumental Kernel

Here, let's consider another choice of Q. We can start by setting

$$\begin{aligned} &Q(X_n = x_n | X_{n-1} = x_{n-1}, \dots X_0 = x_0) = Q(X_n = x_n | X_{n-1} = x_{n-1}) \\ &= p_{x_{n-1}x_n} \cdot g_n(x_n) / \sum_{x_n \in S} p_{x_{n-1}x_n} \cdot g_n(x_n) \\ &Q(X_0 = x_0) = v(x_0) \cdot g_0(x_0) / \sum_{x_0 \in S} v(x_0) \cdot g_0(x_0) \end{aligned} \tag{3.13}$$

Thus, $Q(X_n = x_n, X_{n-1} = x_{n-1}, \dots X_0 = x_0) \propto v(x_0) \prod_{i=1}^{n} p_{x_{i-1}x_i} \cdot \prod_{i=0}^{n} g_i(x_i)$. Such Q is called Optimal Instrumental Kernel. If $Q(X_n = x_n, X_{n-1} = x_{n-1}, \dots X_0 = x_0) > 0$ if $L_n^{-1} \cdot v(x_0) \prod_{i=1}^{n} p_{x_{i-1}x_i} \cdot \prod_{i=0}^{n} g_i(x_i) > 0$. Therefore, the Optimal Instrumental Kernel kernel satisfies the minimal requirement of Importance Sampling.

Since the state space of X is finite, $Q(X_n = x_n | X_{n-1} = x_{n-1})$ is a discrete distribution. The transition probability Q consider the value of observation. So the converge speed should faster then the prior kernel.

We rewrite the (3.10) as following

$$\begin{aligned} w_{n+1}^j &\propto \frac{p_{x_n^j x_{n+1}^j} \cdot g_{n+1}(x_{n+1}^j)}{p_{x_{n-1}x_n} \cdot g_n(x_n) / \sum_{x_n \in S} p_{x_{n-1}x_n} \cdot g_n(x_n)} = w_n^j \cdot \sum_{x_{n+1} \in S} p_{x_n^j x_{n+1}} \cdot g_{n+1}(x_{n+1}) \\ w_0^j &\propto \frac{v(x_0) \cdot g_0(x_0)}{v(x_0) \cdot g_0(x_0) / \sum_{x_0 \in S} v(x_0) \cdot g_0(x_0)} = \sum_{x_0 \in S} v(x_0) \cdot g_0(x_0) \propto 1 \end{aligned} \tag{3.14}$$

Example 14. Applying the Algorithm 7 with Optimal Instrumental Kernel. We have the following result.

```
[1] "Marginal distribution"
       [,1]   [,2]   [,3]
[1,] 0.9098 0.0902 0.0000
[2,] 0.0000 0.8108 0.1892
[3,] 0.9563 0.0437 0.0000
[4,] 0.0000 0.6256 0.3744
[5,] 0.1195 0.8805 0.0000
[1] "Marginal distribution under Importance Sampling_prior kernel"
       [,1]   [,2]   [,3]
[1,] 0.9097 0.0903 0.0000
[2,] 0.0000 0.7311 0.2689
[3,] 0.9421 0.0579 0.0000
[4,] 0.0000 0.8039 0.1961
[5,] 0.1670 0.8330 0.0000
[1] "Marginal distribution under Importance Sampling_Optimal Instrumental Kernel"
       [,1]   [,2]   [,3]
[1,] 0.9068 0.0932 0.0000
[2,] 0.0000 0.7739 0.2261
[3,] 0.9579 0.0421 0.0000
[4,] 0.0000 0.6171 0.3829
[5,] 0.1217 0.8783 0.0000
```

Figure 3.2: The sample size $N = 1000$. It is obvious that the Importance Sampling with optimal instrumental kernel is more accurate than the Importance Sampling with prior kernel

As we can see, the Importance sampling estimator of $\phi_4(I_1)$ didn't converge to the true probability in both method when simulation size $N = 1000$. Figure 3.3 shows the result of 100 independent runs. Each runs contain N particles.

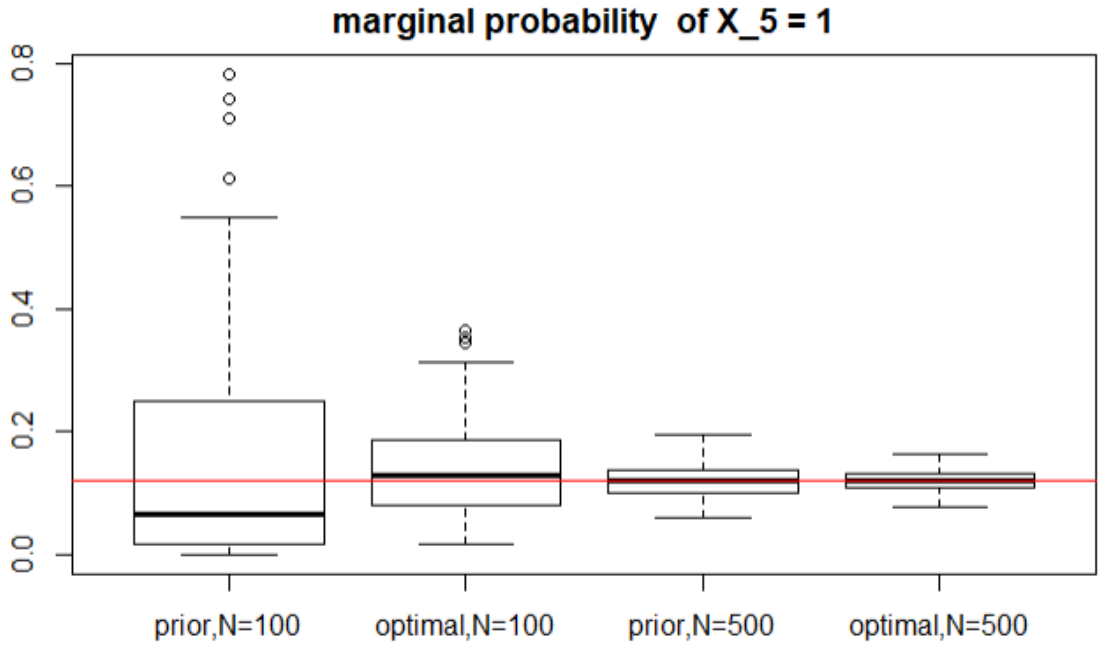

Figure 3.3: The red line represent the true probability of $\phi_4(I_1)$. The Importance sampling estimator with Optimal instrumental kernel provide a more accurate result.

△

3.5 Weight Degeneracy

In the previous section, we have $\hat{\phi}^{IS,N}_{0:n|n}(f) = \sum_{j=1}^{N} f(x_0^j, \dots x_n^j) \cdot w_n^j$,where w_n^j is defined in (3.6) and f is the bounded function. We may write $w_n^j = \nu_n^j / \sum_{j=1}^{N} \nu_n^j$, where

$$\nu_n^j = \frac{v(x_0^j) \prod_{i=1}^{n} p_{x_{i-1}^j x_i^j} \cdot \prod_{i=0}^{n} g_i(x_i^j)}{Q(X_0 = x_0^j, \dots X_n = x_n^j)}$$

We call such ν_n^j as the importance weight and w_n^j as the normalized importance weight. Notice that $\nu_n^j \propto \dfrac{\phi_{0:n|n}(I_{x_0,\dots x_n})}{Q(X_0 = x_0^j, \dots X_n = x_n^j)}$. Thus, the

small importance weight implies that the simulated chain is far from the joint distribution $\phi_{0:n|n}(I_{x_0,\dots x_n})$ and will contribute a little to the $\hat{\phi}^{IS,N}_{0:n|n}$. If most of the simulated chain have small importance weight, the $\hat{\phi}^{IS,N}_{0:n|n}$ is ineffective. This is because we put too much time on computing the importance weight that are close to 0 and updating the chain that is irreverent to the main body. As a consequence, only a few simulated chain will have $w^j_n = \nu^j_n / \sum_{j=1}^N \nu^j_n$ not close to 0.

Such situation is call **weight degeneracy** and is very likely to happen as the time index N increases. Let us introduce such phenomenon in a simple example.

Example 15. Again, we maintain all the model condition in Example 9 We choose the prior kernel as our Q. The simulated simple size= 1000 and the size of observation $n = 10$.

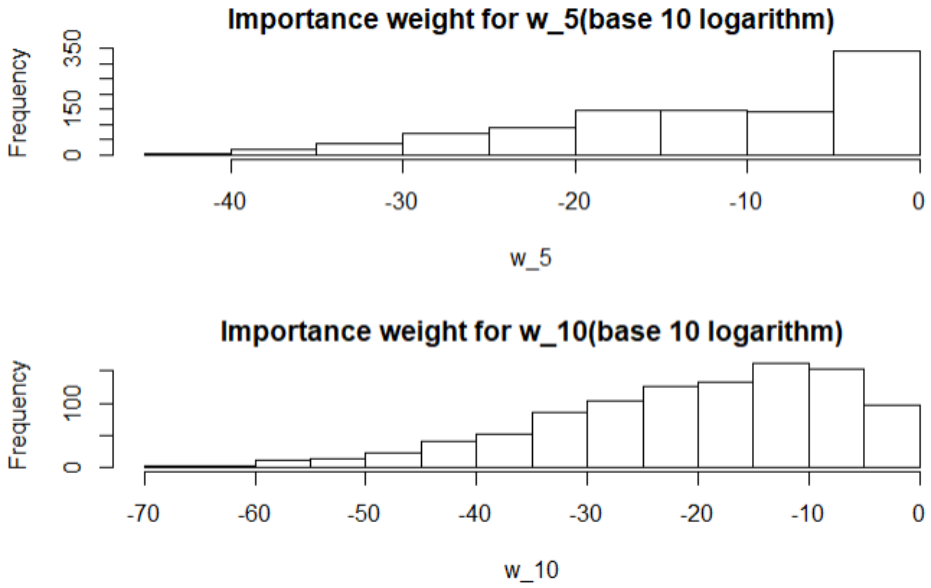

Figure 3.4: The first plot represent the histogram of the base 10 logarithm of normalized importance weight w^j_5. The second plot represent the histogram of the base 10 logarithm of normalized importance weight w^j_{10} .

Notice that $log_{10}(1/1000) = -3$ and $log_{10}(1/1000 * 1/1000) = -6$. The normalized importance weight are still reasonable when time index $n = 5$.

However, most of the normalized importance weight less than 1/1000*1/1000 when time index $n = 10$. The exact marginal probability $\phi_{10|10}(I_2) \approx 0.9745$. There have 837 of 1000 simulated chain whose normalized importance weight less than 1/1000*1/1000 and they only contribute 4.4613×10^{-6} to the $\hat{\phi}^{IS,1000}_{0:10|10}(I_{x_{10}=2})$ =0.9658087. That is, we spend over 80% time to update the irrelevant chain. $\triangle$

There have several useful tool to check the weight degeneracy. A simple test is used by Kong(1994), which is defined by

$$CV_N = \sqrt{N \sum_{j=1}^{N} (w^j - \frac{1}{N})^2} \tag{3.15}$$

CV_N achieve its minimum 0 when $w^j = \frac{1}{N}$. CV_N achieve its maximum $\sqrt{N-1}$ when $w^j = 1$ for some j.

Example 16. We maintain all the condition in example 9. The plot shows that the CV_N increase as the time index increase. Therefore, most of normalized importance weight will decrease to 0 as time index increase.

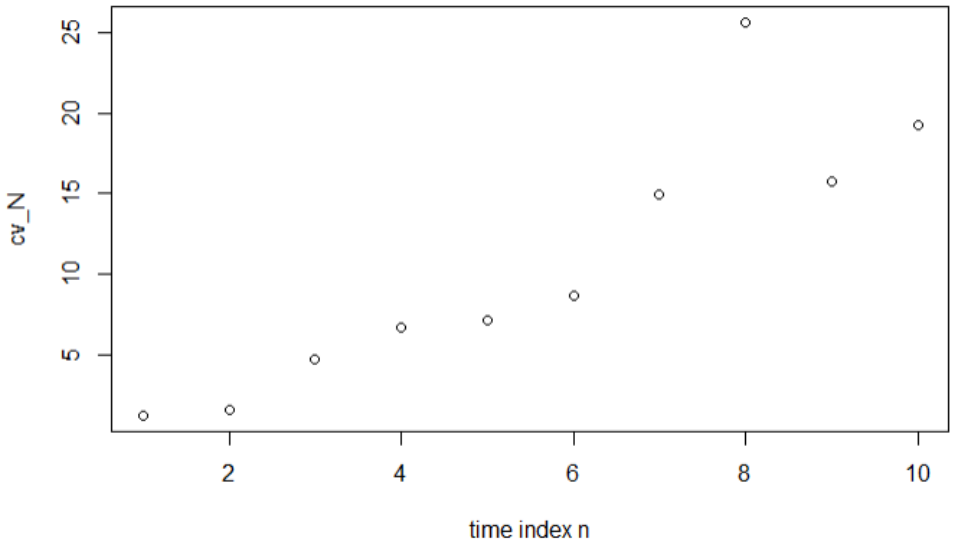

Figure 3.5: The value of CV_N. The model and data is the same in example 3.5.1

△

3.6 Resampling

Gordon(1993) provide the solution to reduce the weight degeneracy. The basic idea is based on the resampling approach. It suggests that we could resample the simulated chain $(x_0^j, \dots x_n^j)$ according to the multinomial distribution with probability eaqual to the normalized importance weight w_n^j. Resampling allows the simulated chain with small importance weight disappear. The chain with large importance weight are more likely to survive in the resampling round.

Thus we can adjust the Algorithm 7 as the following algorithm 8.

Notice that once we apply the resampling approach, the simulated chain is no longer independent. This is because the resampling distribution $P\left((X_0, \dots X_i) = (x_0^j, \dots x_i^j)\right)$ $= w_i^j, j = 1, 2, \dots N$ depends on all the importance weight $v_i^j, j = 1, 2, \dots N$. Therefore, to proof that $\phi_{0:n|n}^{ISR,N} = \sum_{j=1}^{N} f(x_0^j, \dots x_n^j) \cdot w_n^j$ is a consistent es-

Algorithm 8: Importance Sampling with Resampling

Input: P=$(p_{ij} : i, j \in S)$; Conditional probability density $g_k(x_k) = g(y|x_k)$; Observation $y_{0:n}$; The sample size N; The joint distribution Q; C the resampling constant.

Output: N sample of $(x_0, x_1, \dots x_n)$;$W = ((W)_{i,j} = w_i^j, 1 \leqslant j \leqslant N, 0 \leqslant i \leqslant n)$

Initialize $M = N \times n + 1$ matrix , to store the simulation chain;
$W = N \times n + 1$ matrix , to store the w_i^j

for $j = 1 : N$ **do**
  Simulating the x_0^j according to distribution $Q(X_0 = x_0)$;
  $w_0^j = v(x_0^j) \cdot g_0(x_0^j)/Q(X_0 = x_0^j)$
end
Normalized w_0^j;
if $\sqrt{N \sum_{j=1}^{N} (w_0^j - \frac{1}{N})^2} > C$ **then**
  Simulating N independent chain according to the distribution$P(X_0 = x_0^j) = w_0^j, j = 1, 2, \dots N$;
  Replacing the old chain with the new one;
  Set $w_0^j = 1/N$
else
end
for $i = 1 : n$ **do**
  for $j = 1 : N$ **do**
    Simulating the x_i^j according to the probability distribution $Q(X_i = x_i | X_{i-1} = x_{i-1}^j, \dots X_0 = x_0^j)$
    $$w_i^j = w_{i-1}^j \cdot \frac{p_{x_{i-1}^j x_i^j} \cdot g_i(x_i^j)}{Q(X_i = x_i^j | X_0 = x_0^j, \dots X_{i-1} = x_{i-1}^j)}$$
  end
  Normalized w_i^j;
  if $\sqrt{N \sum_{j=1}^{N} (w_i^j - \frac{1}{N})^2} > C$ **then**
    Simulating N independent chain according to the distribution $P((X_0, \dots, X_i) = (x_0^j, \dots x_i^j)) = w_i^j, j = 1, 2, \dots N$;
    Replacing the old chain with the new one;
    Set $w_i^j = 1/N$
  else
end

timator of $\phi_{0:n|n}(f(X_0,\ldots,X_n))$ is no longer a trivial task. Moreover, the resampling increase the variance of the original Importance sampling estimator.

In this project, we will only prove a special case: Resampling at the final round. Let us define one new notation first

$$\xi_n^j := (x_0^j, \ldots x_n^j)$$

That is (ξ_n^j,w_n^j) represent the jth simulated chain and corresponding normalized improtance weight, up to state x_n.

Theorem 3.6.1 Dominated Convergence Theorem(DOM) Suppose $(X_n)_{n\geqslant 0}$ is a sequence of random variables, and X is also a random variable with X_n converge to X with probability 1. If $|X_n|\leqslant Y$ for all n and Y is a random variable with finite expectation. Then

$$\lim_{n\to\infty} E(X_n) = E(X) \tag{3.16}$$

The proof can be found in the Chapter 5 of *Probability with Martingale(Williams)*.

Theorem 3.6.2 If we only take the resampling approach in the final round of Algorithm 6. f is any bounded function with $\sup f \leqslant B$. The importance sampling with resampling estimator

$$\hat{\phi}_{0:n|n}^{SIR,N}(f) = \frac{1}{N}\cdot\sum_{j=1} f(\hat{\xi}^j) \tag{3.17}$$

where $\hat{\xi}^j$ is the jth simulated chain after resampling.

Then

1) $E(\hat{\phi}_{0:n|n}^{SIR}(f)) \to \phi_{0:n|n}(f(X_0,\ldots X_n))$

2) $E(\hat{\phi}_{0:n|n}^{SIR}(f)-\phi_{0:n|n}(f(X_0,\ldots X_n)))^2 \geqslant E(\phi_{0:n|n}^{IS,N}(f)-\phi_{0:n|n}(f(X_0,\ldots X_n)))^2$

Proof. 1) First notice that

$$
\begin{aligned}
&E(\hat{\phi}^{SIR}_{0:n|n}(f)|\xi^1_n,\dots\xi^N_n)\\
&= E(\frac{1}{N}\cdot\sum_{j=1} f(\hat{\xi}^j)|\xi^1_n,\dots\xi^N_n) = E(f(\hat{\xi}^1)|\xi^1_n,\dots\xi^N_n) = \sum_{j=1}^{N} w^j_n\cdot f(\xi^j_n)\\
&= \phi^{IS,N}_{0:n|n}(f)
\end{aligned}
$$

Since $\phi^{IS,N}_{0:n|n}(f) \to \phi_{0:n|n}(f(X_0,\dots X_n))$ with probability 1 as N goes to infinity. Also, $\phi^{IS,N}_{0:n|n}(f) \leqslant \sup f = B$

Applying the Theorem 3.6.1 Dominated Convergence Theorem, we have

$$
\lim_{N\to\infty} E(\hat{\phi}^{SIR}_{0:n|n}(f)) = \lim_{N\to\infty} E(E(\hat{\phi}^{SIR}_{0:n|n}(f)|\xi^1_n,\dots\xi^N_n)) = \phi_{0:n|n}(f(X_0,\dots X_n))
$$

2)

$$
\begin{aligned}
&E(\hat{\phi}^{SIR}_{0:n|n}(f) - \phi_{0:n|n}(f(X_0,\dots X_n)))^2\\
&= E(\hat{\phi}^{SIR}_{0:n|n}(f) - \phi^{IS,N}_{0:n|n}(f) + \phi^{IS,N}_{0:n|n}(f) - \phi_{0:n|n}(f(X_0,\dots X_n)))^2\\
&= E(\hat{\phi}^{SIR}_{0:n|n}(f) - \phi^{IS,N}_{0:n|n}(f))^2 + E(\phi^{IS,N}_{0:n|n}(f) - \phi_{0:n|n}(f(X_0,\dots X_n)))^2\\
&+ 2\cdot E\{\hat{\phi}^{SIR}_{0:n|n}(f) - \phi^{IS,N}_{0:n|n}(f))(\phi^{IS,N}_{0:n|n}(f) - \phi_{0:n|n}(f(X_0,\dots X_n))\}
\end{aligned}
\tag{3.18}
$$

But

$$
\begin{aligned}
&E\{\hat{\phi}^{SIR}_{0:n|n}(f) - \phi^{IS,N}_{0:n|n}(f))(\phi^{IS,N}_{0:n|n}(f) - \phi_{0:n|n}(f(X_0,\dots X_n))\}\\
&= E\{E(\hat{\phi}^{SIR}_{0:n|n}(f) - \phi^{IS,N}_{0:n|n}(f))(\phi^{IS,N}_{0:n|n}(f) - \phi_{0:n|n}(f(X_0,\dots X_n))|\xi^1_n,\dots\xi^N_n)\}\\
&= E\{(E(\hat{\phi}^{SIR}_{0:n|n}(f)|\xi^1_n,\dots\xi^N_n) - \phi^{IS,N}_{0:n|n}(f))(\phi^{IS,N}_{0:n|n}(f) - \phi_{0:n|n}(f(X_0,\dots X_n))\}\\
&= 0
\end{aligned}
$$

Therefore, $E(\hat{\phi}^{SIR}_{0:n|n}(f) - \phi_{0:n|n}(f(X_0,\dots X_n)))^2 \geqslant E(\phi^{IS,N}_{0:n|n}(f) - \phi_{0:n|n}(f(X_0,\dots X_n)))^2$ □

It can be shown that the Importance Sampling with resampling estimator under Algorithm 6 converge to $\phi_{0:n|n}(f(X_0,\dots X_n))$ in probability. Interested reader can refer to Chapter 9 of [Inference in HMM(Cappe)].

Example 17. To illustrate the power of resampling, we maintain all the condition in previous example. Applying the Algorithm 6 with prior kernel and the resampling constant $C = 5$.

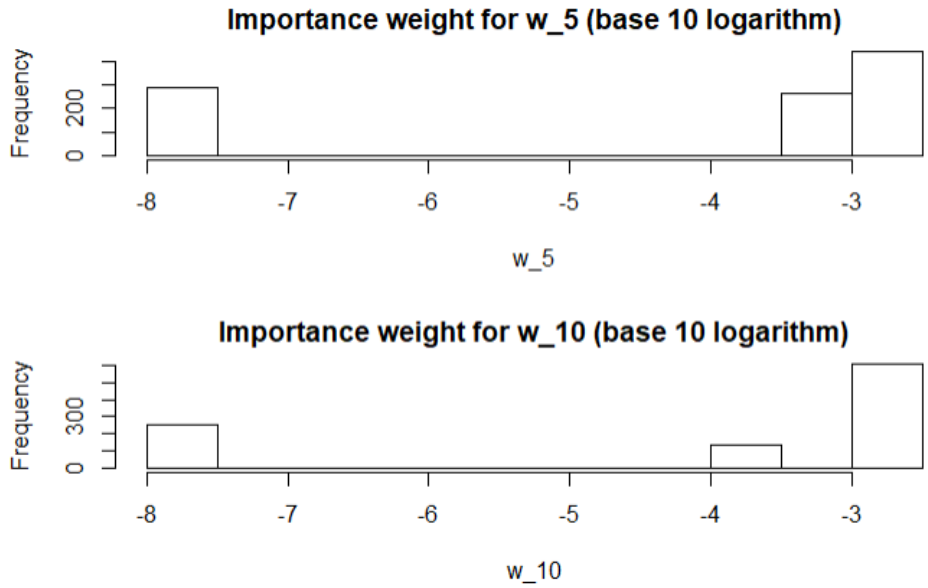

Figure 3.6: The first plot represent the histogram of the base 10 logarithm of normalized importance weight w_5^j. The second plot represent the histogram of the base 10 logarithm of normalized importance weight w_{10}^j .

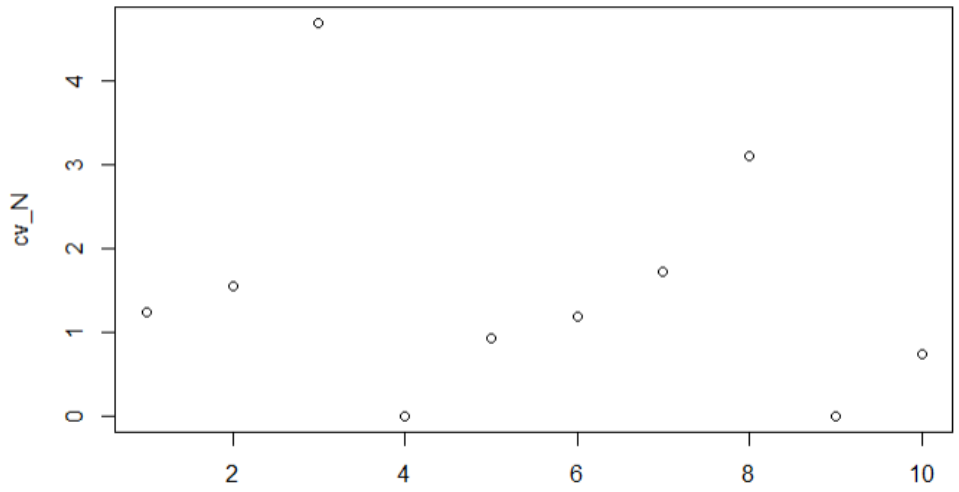

Figure 3.7: The value of CV_N.

We apply the resampling method in algorithm 6 whenever the $CV_N \geqslant 5$. In figure 11, the normalized importance weight increased significantly, comparing to figure 9. Showing that the weight degeneracy phenomenon has been controlled. There are 612 of 1000 simulated chain whose normalized importance weight larger than 1/1000 and contribute 0.9721142 to the $\hat{\phi}^{SIR,1000}(I_{x_{10}=2}) := \hat{P}(X_10 = 2|Y_0 = y_0, \ldots Y_{10} = y_{10}) = 0.97211423$. Therefore, there have over 60% simulated chains that are significantly contributing to the estimator. Such estimator is more effective than the method without resampling.

△

4 Appendices

4.1 R code

HMM

yizhenli

2021/3/1

#Forward and backward Decompostion

#discrete distribution simulation(accept-reject)

```
ds<-function(x){
  k=0
  n=length(x)
  C=max(x)
  while(k==0){
    u=runif(1,0,1)
    y=floor(n*u)+1
    if(runif(1,0,1)<=x[y]/C){k=1;final=y}
  }
  return(y)
}
```

#finding phi_{k|n} by recursive method

```
phi<-function(obs,v,Q,g2){
  N=length(obs)
  r=ncol(Q)
  forw=matrix(0,nrow=N,ncol=r)
  phi=v

  for(i in 1:N){
    c=sum(phi*g2(c(1:r),obs[i]))
    forw[i,]=phi*g2(c(1:r),obs[i])/c
    phi=forw[i,]%*%Q
  }
  backw=matrix(0,nrow=N,ncol=r)
  backw[N,]=forw[N,]
  for ( i in (N-1):1){
    B=matrix(0,nrow=r,ncol=r)
    for( j in 1:r){
      B[j,]=forw[i,]*Q[,j]/sum(forw[i,]*Q[,j])
    }
    backw[i,]=backw[i+1,]%*%B
  }
  return(backw)
}
```

#MCMC_Systematic-Scan_Gibbs_Sampling

```
singlesite<-function(N,obs,initial,Q,v,g2){
  n=length(initial)
  r=ncol(Q)
  chain=initial
```

```
  store=matrix(0,nrow=N,ncol=n)
  store[1,]=chain
  new_chain=rep(0,n)

  for( j in 1:N){
  x=chain[2]
  p=v*g2(c(1:r),obs[1])*Q[,x]
  p=p/sum(p)
  new_chain[1]=ds(p)
  for (i in 2:(n-1)){
    x=new_chain[i-1]
    y=chain[i+1]
    p=Q[x,]*g2(c(1:r),obs[i])*Q[,y]
    p=p/sum(p)
    new_chain[i]=ds(p)
  }
  p=Q[new_chain[n-1],]*g2(c(1:r),obs[n])
  new_chain[n]=ds(p/sum(p))
  store[j,]=new_chain
  chain=new_chain
  }
  return(store)
}
```

#phi_MCMC,to estimate the marginal distribution

```
phi_MCMC<-function(MCMC,r){
  nrow=nrow(MCMC)
  ncol=ncol(MCMC)
  phi=matrix(0,nrow = ncol,ncol=r)
  for (j in 1:r){
    for(i in 1:ncol){
      phi[i,j]=sum((MCMC[,i]==j))/nrow
    }
  }
  return(phi)
}
```

#MCMC_RWMH

```
MH<-function(N,obs,initial,Q,v,g){
  n=length(initial)
  r=ncol(Q)
  chain=initial
  store=matrix(0,nrow=N,ncol=n)
  store[1,]=chain
  new_chain=rep(0,n)

  for( j in 1:N){
  x=chain[2]
  a=floor(r*runif(1,0,1))+1 #simulate new state
  a_old=chain[1]#old state
  p=min(v[a]*g(a,obs[1])*Q[a,x]/(v[a_old]*g(a_old,obs[1])*Q[a_old,x]),1)
```

```
  new_chain[1]=ifelse(runif(1,0,1)<p,a,a_old)
  for (i in 2:(n-1)){
    x=new_chain[i-1]
    y=chain[i+1]
    a=floor(r*runif(1,0,1))+1
    a_old=chain[i]
    p=min(Q[x,a]*g(a,obs[i])*Q[a,y]/(Q[x,a_old]*g(a_old,obs[i])*Q[a_old,y]),1)
    new_chain[i]=ifelse(runif(1,0,1)<p,a,a_old)
  }
  a=floor(r*runif(1,0,1))+1
  a_old=chain[n]
  p=min(Q[new_chain[n-1],a]*g(a,obs[n])/(Q[new_chain[n-1],a_old]*g(a_old,obs[n])),1)
  new_chain[n]=ifelse(runif(1,0,1)<p,a,a_old)
  store[j,]=new_chain
  chain=new_chain
  }
  return(store)
}
```

#Importance Sampling_prior kernel

```
ISP<-function(N,obs,Q,v,g2,cv,C){
  n=length(obs)
  r=ncol(Q)
  store=matrix(nrow=N,ncol=n)
  W=store
  for( j in 1:N){
    x=ds(v)
    store[j,1]=x
    W[j,1]=g2(x,obs[1])
  }
  W[,1]=W[,1]/sum(W[,1])
  if(cv(W[,1])>=C){
    k=store
    for(t in 1:N){ store[t,]=k[ds(W[,1]),]}
    W[,1]=1/N
  }
  for(i in 2:n){
    for(j in 1:N){
      x=ds(Q[store[j,i-1],])
      store[j,i]=x
      W[j,i]=W[j,i-1]*g2(x,obs[i])
    }
      W[,i]=W[,i]/sum(W[,i])
      if(cv(W[,i])>=C){
      k=store
      for(t in 1:N){ store[t,]=k[ds(W[,i]),]}
      W[,i]=1/N
  }
  }
  A=list(store=store,w=W)
  return(A)
}
```

#Importance Sampling_Optimal Instrumental Kernel

```
ISO<-function(N,obs,Q,v,g2){
  n=length(obs)
  r=ncol(Q)
  store=matrix(nrow=N,ncol=n)
  W=store
  for( j in 1:N){
    x=ds(v*g2(c(1:r),obs[1]))
    store[j,1]=x
    W[j,1]=1/N
  }
  for(i in 2:n){
    for(j in 1:N){
      x=ds(Q[store[j,i-1],]*g2(c(1:r),obs[i]))
      store[j,i]=x
      W[j,i]=W[j,i-1]*sum(Q[store[j,i-1],]*g2(c(1:r),obs[i]))
    }
      W[,i]=W[,i]/sum(W[,i])
  }
  A=list(store=store,w=W)
  return(A)
}
```

#phi_IS,to estimate the marginal distribution

```
phi_IS<-function(IS,r){
  store=IS$store
  w=IS$w
  nrow=nrow(store)
  ncol=ncol(store)
  phi=matrix(0,nrow = ncol,ncol=r)
  for (j in 1:r){
    for(i in 1:ncol){
      phi[i,j]=sum((store[,i]==j)*w[,ncol]/sum(w[,ncol]))
    }
  }
  return(phi)
}
```

#CV_N

```
cv<-function(w){
  N=length(w)
  cv_N=(N*(sum((w-1/N)^2)))^0.5
  return(cv_N)
}
```

#Weight Degeneracy

```
cv_wd<-function(w){
  n=ncol(w)
  N=nrow(w)
```

```
  cv_N=vector()
  for(i in 1:n){
    cv_N[i]=(N*sum((w[,i]-1/N)^2))^(1/2)
  }
  return(cv_N)
}
```

#Test Section #parameter

```
Q=matrix(c(1/8,5/8,2/8,1/9,7/9,1/9,2/5,1/5,2/5),byrow=TRUE, nrow=3)
v=c(1/3,1/3,1/3)
u=c(-5,0,5)
s=c(2,4,2)
g<- function(x,y){
  z=exp(-(y-u[x])^2/(2*s[x]))
  return(z)
}
g2<-function(x,y){
  n=length(x)
  z=vector(length=n)
  for( i in 1:n){
    z[i]=exp(-(y-u[x[i]])^2/(2*s[x[i]]))
  }
  return(z)
}
```

#Example 7

```
simb<-function(N,v,Q){
  stat=matrix(0,nrow = N, ncol=2)
  stat[1,1]=ds(v)
  stat[1,2]=stat[1,1]+ifelse(runif(1,0,1)<1/2,1/2,-1/2)
  for( i in 2:N){
    stat[i,1]=ds(Q[stat[i-1,1],])
    stat[i,2]=stat[i,1]+ifelse(runif(1,0,1)<1/2,1/2,-1/2)
  }
  colnames(stat)<-c("X","Y")
  return(stat)
}
#sim(100,v,Q,u,s)
```

```
Q=matrix(c(1/4,3/4,3/4,1/4),byrow=TRUE, nrow=2)
v=c(1/5,4/5)
g2<-function(x,y){
  n=length(x)
  z=vector(length=n)
  for( i in 1:n){
    if(y==5/2){z[i]=ifelse(x[i]==2,1/2,0)
    }else if(y==3/2){z[i]=1/2
    }else{z[i]=ifelse(x[i]==1,1/2,0)}
  }
  return(z)
}
```

```r
set.seed(1234)
N=3
#hmm=simb(N,v,Q)
obs=c(3/2,3/2,3/2)
phi_x=phi(obs,v,Q,g2)
N=1000
r=ncol(Q)


MCMC=singlesite(N,obs,c(1,1,1),Q,v,g2)

#MCMC_2=rbind(singlesite(N/3,obs,c(1,1,1,1,1),Q,v,g2),singlesite(N/3,obs,c(2,2,2,2,2),Q,v,g2),singlesit

phi_mcmc=phi_MCMC(MCMC,r)


print("Marginal distribution")
round(phi_x,4)
print("Marginal distribution estimator under Gibbs Sampling")
round(phi_mcmc,4)
```

#Example 8

#simulate(X_k,Y_k)_for normal Y

```r
sim<-function(N,v,Q,u,s){
  stat=matrix(0,nrow = N, ncol=2)
  stat[1,1]=ds(v)
  stat[1,2]=rnorm(1,u[stat[1,1]],s[stat[1,1]])
  for( i in 2:N){
    stat[i,1]=ds(Q[stat[i-1,1],])
    stat[i,2]=rnorm(1,u[stat[i,1]],s[stat[i,1]])
  }
  colnames(stat)<-c("X","Y")
  return(stat)
}
#sim(100,v,Q,u,s)
```

```r
set.seed(1234)
N=5
hmm=sim(N,v,Q,u,s)
obs=hmm[,2]
phi_x=phi(obs,v,Q,g2)
N=1000
r=ncol(Q)

timestart1<-Sys.time();
MCMC=singlesite(N,obs,c(2,1,2,1,2),Q,v,g2)
timesend1<-Sys.time();
#MCMC_2=rbind(singlesite(N/3,obs,c(1,1,1,1,1),Q,v,g2),singlesite(N/3,obs,c(2,2,2,2,2),Q,v,g2),singlesit
timestart2<-Sys.time();
MCMC2=MH(N,obs,c(2,1,2,1,2),Q,v,g)
timesend2<-Sys.time();
```

```
print("Running time of Gibbs Sampling")
timesend1-timestart1
print("Running time of Metropolis-Hastings")
timesend2-timestart2

phi_mcmc=phi_MCMC(MCMC,r)
phi_mcmc2=phi_MCMC(MCMC2,r)

print("Marginal distribution")
round(phi_x,4)
print("Marginal distribution estimator under Gibbs Sampling")
round(phi_mcmc,4)
print("Marginal distribution estimator under  Metropolis-Hastings")
round(phi_mcmc2,4)
```

#Example 11

```
set.seed(1111)
N=5
hmm=sim(N,v,Q,u,s)
obs=hmm[,2]
S0=100

S=vector()
for(i in 1:N){
  S[i]=exp(sum(obs[1:i]))*S0
}


phi_x=phi(obs,v,Q,g2)
N=1000
r=ncol(Q)
```

```
timestart1<-Sys.time();
MCMC=singlesite(N,obs,c(2,1,2,1,2),Q,v,g2)
timesend1<-Sys.time();
#MCMC_2=rbind(singlesite(N/3,obs,c(1,1,1,1,1),Q,v,g2),singlesite(N/3,obs,c(2,2,2,2,2),Q,v,g2),singlesit
timestart2<-Sys.time();
MCMC2=MH(N,obs,c(2,1,2,1,2),Q,v,g2)
timesend2<-Sys.time();

print("Running time of Gibbs Sampling")
timesend1-timestart1
print("Running time of Metropolis-Hastings")
timesend2-timestart2

phi_mcmc=phi_MCMC(MCMC,r)
phi_mcmc2=phi_MCMC(MCMC2,r)

print("Marginal distribution")
round(phi_x,4)
print("Marginal distribution estimator under Gibbs Sampling")
```

```
round(phi_mcmc,4)
print("Marginal distribution estimator under  Metropolis-Hastings")
round(phi_mcmc2,4)
```

```
library(forecast)
x_0<-MCMC[,1]
x_1<-MCMC[,2]
x_2<-MCMC[,3]
x_3<-MCMC[,4]
x_4<-MCMC[,5]
par(mfrow=c(3,2))
Acf(x_0)
Acf(x_1)
Acf(x_2)
Acf(x_3)
Acf(x_4)
```

#test for IS #Example 14

```
set.seed(1234)
N=5
hmm=sim(N,v,Q,u,s)
obs=hmm[,2]
phi_x=phi(obs,v,Q,g2)
r=ncol(Q)
```

```
N=1000
IS1=ISP(N,obs,Q,v,g2)
phi_IS1=phi_IS(IS1,r)

IS2=ISO(N,obs,Q,v,g2)
phi_IS2=phi_IS(IS2,r)
print("Marginal distribution")
round(phi_x,4)
print("Marginal distribution under Importance Sampling_prior kernel")
round(phi_IS1,4)
print("Marginal distribution under Importance Sampling_Optimal Instrumental Kernel")
round(phi_IS2,4)
```

```
N0=100
N1=100
N2=500
x1=vector()
x2=vector()
x3=vector()
x4=vector()


for(i in 1:N0){
  IS1=ISP(N1,obs,Q,v,g2)
  x1[i]=phi_IS(IS1,r)[5,1]  #prior,N=100
  IS1=ISP(N2,obs,Q,v,g2)
```

```
  x3[i]=phi_IS(IS1,r)[5,1] #prior,N=500
  IS2=ISO(N1,obs,Q,v,g2)
  x2[i]=phi_IS(IS2,r)[5,1] #optimal,N=100
  IS2=ISO(N2,obs,Q,v,g2)
  x4[i]=phi_IS(IS2,r)[5,1] #optimal,N=500
}
```

```
x=cbind(x1,x3,x2,x4)
colnames(x)<-c("prior,N=100","optimal,N=100","prior,N=500","optimal,N=500")
boxplot.matrix(x,main="marginal probability  of X_5 = 1")
abline(a=phi_x[5,1],b=0,col="red")
```

#Example 15 & Example 17.

```
set.seed(1234)
N=10
hmm=sim(N,v,Q,u,s)
obs=hmm[,2]
r=ncol(Q)

N=1000
IS1=ISP(N,obs,Q,v,g,cv,5) #C=5
w_5=log(IS1$w[,5],base=10)
w_10=log(IS1$w[,10],base=10)
par(mfrow=c(2,1))
hist(w_5,main="Importance weight for w_5 (base 10 logarithm)")
hist(w_10,main="Importance weight for w_10 (base 10 logarithm)")
```

```
sum((IS1$w[,10]<=1/1000000))
sum((IS1$w[,10]<=1/1000000)*IS1$w[,10]*(IS1$store[,10]==2))#contribute
```

#Example 16.

```
cv_N=cv_wd(IS1$w)
plot(cv_N,xlab="time index n")
```

```
sum((IS1$w[,10]>=1/1000))
sum((IS1$w[,10]>=1/1000)*IS1$w[,10]*(IS1$store[,10]==2))#contribute
```

4.2 Reference

[1] Cappé O, Moulines O, Rydén T.(2005) Ch.2 Main Definitions and Notations. In *Inference in Hidden Markov Models* (pp.35-50). Springer: New York.

[2] Cappé O, Moulines O, Rydén T.(2005) Ch.3 Filtering and Smoothing Recursions. In *Inference in Hidden Markov Models* (pp.51-76). Springer: New York.

[3] Cappé O, Moulines O, Rydén T.(2005) Ch.6 Monte Carlo Methods. In *Inference in Hidden Markov Models* (pp.161-185). Springer: New York.

[4] Cappé O, Moulines O, Rydén T.(2005) Ch.7 Sequential Monte Carlo Methods. In *Inference in Hidden Markov Models* (pp.209-241). Springer: New York.

[5] Siddhartha Chib (1996), *Calculating posterior distributions and modal estimates in Markov mixture models*, Journal of Econometrics, Volume 75, Issue 1. (pp. 79-97)

[6] Norris, J.(1997) Ch.1 Discrete-time Markov chains. In *Markov Chains*. (pp. 1-10 & 52-56) Cambridge University Press, Cambridge.