Markov Chain Monte Carlo Method

Johnny Zhang

Caution!

Four levels of lies

- Lie
- Damned lie

Caution!

Four levels of lies

- Lie
- Damned lie
- Statistics

Caution!

Four levels of lies

- Lie
- Damned lie
- Statistics
- MCMC

Markov chain

A *Markov chain* is a sequence of random variables (*a stochastic process*) $X_1, X_2, ..., X_t \ldots$ with the *Markov property*, namely that, given the present state, the future and past states are independent. In other words, the present state only depends on the immediately past state. Formally,

$$\begin{aligned} \Pr(X_{t+1} &= x_{t+1} | X_t = x_t, X_{t-1} = x_{t-1}, \ldots, X_1 = x_1) \\ &= \Pr(X_{t+1} = x_{t+1} | X_t = x_t). \end{aligned}$$

The possible values of X_t form a set $\mathcal{S}$ called state space that can be discrete or continuous.

Discrete random walk

Consider a random walk process

$$X_{t+1} = X_t + e_t$$

where

$$e_t = \begin{cases} -1 & \textit{with probability } .5 \\ 1 & .5 \end{cases}$$

Transition probabilities

At each step the system may change its state from the current state to another state, or remain in the same state, according to a certain probability distribution. The changes of states are called *transitions,* and the probabilities associated with various state-changes are called *transition probabilities* or *transition kernel* K. The transition kernel can be viewed as a conditional density such that $X_{t+1}|X_t \sim K(X_t, X_{t+1})$.

An example - Text analysis

- The writing of text can be viewed as a Markov process.
  - Grammar
  - Personal habits
- Generate text with a certain pattern
  - Original: the boy and the dog went to the park.
    - the 3/9, the other 1/9
    - if x_0=the; x_1=boy/dog/park
  - Generated: the boy and the boy and the dog went to the boy.

Random walk

An example of the Markov chain is the *random walk* process that satisfies

$$X_{t+1} = X_t + \epsilon_t$$

where ϵ_t is a random process generated independently of X_t. For example, ϵ_t iid $N(0, 1)$. In this example, the state space is continuous and the transition kernel (probability) is a normal distribution,

$$K(X_t, X_{t+1}) \propto \exp \frac{(X_{t+1} - X_t)^2}{2}.$$

Discrete Markov chain

Assume one wants to look at the emotion dynamics of a participant. The participant can feel either happy or unhappy on a day.Thus, the state space is $\mathcal{S} = \{\text{happy, unhappy}\}$. Furthermore, we assume if he/she feels happy today, he/she will feel happy next day with a probability of .8 and if he/she feels unhappy today, he/she will feel unhappy next day with a probability of .4. Thus, the transition probability matrix is

$$K = \begin{array}{cc} & \begin{array}{cc} \text{tomorrow} & \\ \text{happy} & \text{unhappy} \end{array} \\ \begin{array}{c} \text{today} \\ \text{happy} \\ \text{unhappy} \end{array} & \begin{array}{cc} .8 & .2 \\ .6 & .4 \end{array} \end{array}$$

State probability

Let $p_1 = \Pr(happy)$ and $p_2 = \Pr(unhappy) = 1 - p_1$ as probabilities of the two states. We can calculate the probability $P = (p_1, p_2)$ at any time using the transition probability matrix. As an example, we assume at the initial state, for example, the state when the participant enters the experiment, the probability is

$$P_0 = [.5, .5]$$

that indicates a state that the participant feels either happy or unhappy. After one day, the probability will change to

$$P_1 = P_0 K = [.5, .5] \begin{bmatrix} .8 & .2 \\ .6 & .4 \end{bmatrix} = [.7, .3].$$

Then at time t, the probability will be

$$P_t = P_0 K^t.$$

It can be shown easily that after a while, this will enter a stationary probability where

$$P_n = [.75, .25].$$

Stationary distribution

- The final stable probability does not depend on the initial probability. There is an initial phase before the stationary probability.
- The stable probability can be solved for the simple example from $PK = P$.

Conditions for stationary probability

Let's consider a general Markov chain with two discrete states. The transition probability matrix can be

$$K = \begin{bmatrix} a & 1-a \\ 1-b & b \end{bmatrix}.$$

Assume the stationary probability (distribution) is $P = (c \quad d)$. Then we should have

$$[c, d] \begin{bmatrix} a & 1-a \\ 1-b & b \end{bmatrix} = [c, d].$$

Solve the above equation, we will have

$$(1-a)c = (1-b)d.$$

Because $c + d = 1$, we have

$$c = \frac{1-b}{2-a-b} \ \& \ d = \frac{1-a}{2-a-b}.$$

Thus, unless $a = b = 1$, we will always have a unique c and d for this simple Markov process. In other words, the Markov chain will always converge to a stationary probability.

Properties of Markov chain

- Homogeneous
- Irreducibility
- Recurrence
- Stationary
- Periodicity
- Ergodicity

Homogeneous

A Markov chain is said to be time-homogeneous or simply homogeneous if for all t,

$$\Pr(X_{t+1} = x | X_t = y) = \Pr(X_t = x | X_{t-1} = y).$$

In other words, the transition probability does not depend on t. Both the random walk and the discrete Markov examples are homogeneous Markov chains.

Irreducibility

A state j is *accessible* from a different state i if there is a non-zero probability that after a number of steps the system can move from state i to state j. Formally, state j is accessible from state i if there exists $t \geq 0$ such that

$$\Pr(X_t = j | X_0 = i) > 0.$$

Furthermore, a state i *communicates* with state j if both state i is accessible from state j and state j is accessible from state i.
A Markov chain is said to be irreducible if any state can communicate with all the other states. In other words, it is possible to get to any state from another state.

Recurrence

Irreducibility ensures that every state in the state space can be visited by the Markov chain. However, it may not guarantee that the state will be visited often enough. A state $s \in \mathcal{S}$ is *transient* if the average number of visits to s, $E_s(n_s)$, is finite and *recurrent* if $E_s(n_s) = \infty$. Analogy to the random number generation from the normal distribution, if we generate $N = \infty$ times of random numbers, then each number can occur ∞ times. Not strictly speaking, a Markov chain is recurrent if every state is recurrent. If the time to return to a state s is bounded or finite, the state is *positive recurrent*.

Stationary

Let $\pi(s), s \in \mathcal{S}$ denote the probability of the state s. $\pi(s)$ is said to be stationary if

$$\begin{aligned} \sum_k \pi^t(k)p(k,s) &= \pi^{t+1}(s) \text{ for discrete Markov chain} \\ \int \pi^t(k)p(k,s)dk &= \pi^{t+1}(s) \text{ for continuous Markov chain.} \end{aligned}$$

This implies that the probability of the state does not change anymore. Note that $\pi(s)$ is also the marginal distribution. Thus the marginal distribution does not depend on t. With stationarity, all the states jumped to can be viewed as from the same stationary distribution.

Periodicity

A state i has period m if any return to state i must occur in multiples of m steps. Note that even though a state has period m, it may not be possible to reach the state in m steps. If $m = 1$, then the state is said to be *aperiodic*; otherwise ($m > 1$), the state is said to be *periodic*. It can be shown that every state in a irreducible state space has the same period.

Ergodicity

A Markov chain is *ergodic* if it is irreducible, positive recurrent, and aperiodic. For ergodic Markov chain, it eventually will converge to a stationary distribution of interest.

Gibbs Sampler

Gibbs sampler, also called *Gibbs sampling*, is an algorithm to generate a sequence of samples from the joint probability distribution of two or more random variables. The purpose of such a sequence is to approximate the joint distribution, or to compute an integral (such as an expectation). Gibbs sampler is especially useful when the joint probability distribution is too complex or unknown at all but the conditional distribution of each random variable is available.

Gibbs sampler is a special example of a Markov chain Monte Carlo algorithm. The Gibbs sampler generates an instance from the conditional distribution of each variable in turn, conditionally on the current values of the other variables. It can be shown that the sequence of samples constitutes a Markov chain, and the stationary distribution of that Markov chain is just the sought-after joint distribution (Geman & Geman, 1994).

Gibbs sampler

Suppose we want to sample from a multivariate distribution $p(\boldsymbol{\theta})$ with $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_q)$. This is often the posterior distribution for the Bayesian analysis. Usually it is difficult to sample from it directly. In Bayesian analysis, often the conditional distribution of $p(\theta_i|\boldsymbol{\theta}_{-i}), \boldsymbol{\theta}_{-i} = (\theta_1, \ldots, \theta_{i-1}, \theta_{i+1}, \ldots, \theta_q)$ can be obtained relatively easily. Using the conditional distribution as the transition kernel, we can construct a Markov chain for $\boldsymbol{\theta}$. It can be shown that the Markov chain is ergodic and thus after convergence, the generated $\boldsymbol{\theta}$ is actually from the joint distribution $p(\boldsymbol{\theta})$.

Gibbs sampling algorithm

1. Start with some initial guess $\boldsymbol{\theta}^{(0)} = (\theta_1^{(0)}, \ldots, \theta_q^{(0)})$;
2. At the jth iteration, we have $\boldsymbol{\theta}^{(j)} = (\theta_1^{(j)}, \ldots, \theta_q^{(j)})$. Then sampling $\boldsymbol{\theta}^{(j+1)}$ according to
   - 2.1 Sampling $\theta_1^{(j+1)}$ from $p(\theta_1 | \theta_2^{(j)}, \ldots, \theta_q^{(j)})$,
   - 2.2 Sampling $\theta_2^{(j+1)}$ from $p(\theta_2 | \theta_1^{(j+1)}, \theta_3^{(j)}, \ldots, \theta_q^{(j)})$,
   - 2.3 $\vdots$
   - 2.4 Sampling $\theta_q^{(j+1)}$ from $p(\theta_q | \theta_1^{(j+1)}, \ldots, \theta_{q-1}^{(j+1)})$.
3. Repeat Step 2.

After a sufficient number of iteration denoting n, n is often called burn-in period, the resulting samples $\boldsymbol{\theta}^{(n+1)}, \boldsymbol{\theta}^{(n+2)}, \ldots$ can be viewed from $p(\boldsymbol{\theta})$.

Bivariate normal random number generation

For the purpose of demonstration, we look at how to generate bivariate normal random numbers. Let

$$\boldsymbol{\theta} = \begin{pmatrix} \theta_1 \\ \theta_2 \end{pmatrix} \sim MN\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix} \right).$$

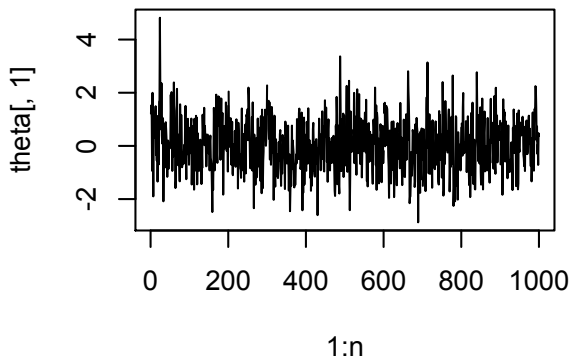
Then the conditional distributions are

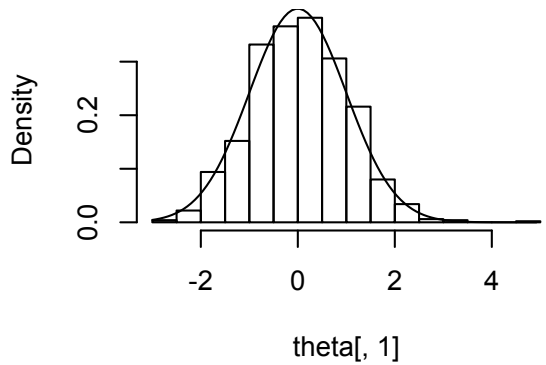
$$\begin{aligned} \theta_1|\theta_2 &\sim N(\rho\theta_2, 1-\rho^2) \\ \theta_2|\theta_1 &\sim N(\rho\theta_1, 1-\rho^2). \end{aligned}$$

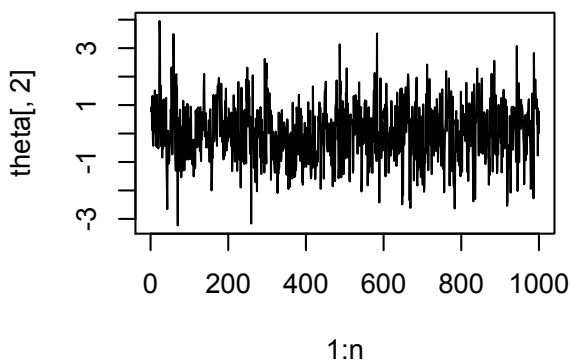
```
## Gibbs sampling for the bivariate normal
# Correlation
rho<-.5
sig<-sqrt(1-rho^2)

n<-1000
theta0<-0

theta<-array(NA,dim=c(n,2))

for (i in 1:n){
        theta[i,1]<-rnorm(1,rho*theta0,sig)
        theta[i,2]<-rnorm(1,rho*theta[i,1],
           sig)
        theta0<-theta[i,2]
        }
```

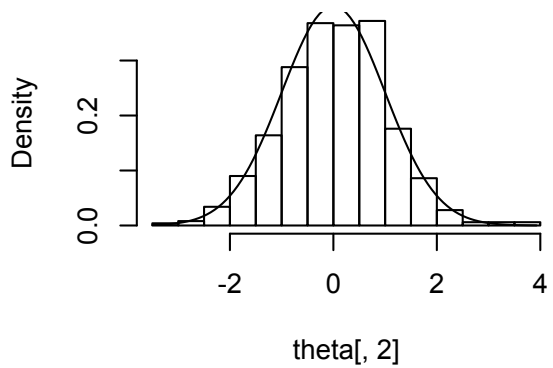
```
## Plot the history and the density
par(mfrow=c(2,2))
plot(1:n,theta[,1],type='l')
hist(theta[,1],freq=F,main='')
curve(dnorm,add=TRUE)

plot(1:n,theta[,2],type='l')
hist(theta[,2],freq=F,main='')
curve(dnorm,add=TRUE)
```

History and density plot









Summary statistics

- The estimate is calculated by

$$\hat{\theta} = \frac{1}{n}\sum \theta_i$$

- Standard deviation of θ (corresponding to the standard error of frequentist statistics) is

$$SD(\theta) = \sqrt{\frac{1}{n-1}\sum(\theta_i - \hat{\theta})}.$$

- The Monte Carlo standard error is

$$MC\ SE(\theta) = \sqrt{\frac{1}{n(n-1)}\sum(\theta_i - \hat{\theta})}.$$

- The covariance between θ_1 and θ_2 can also be calculated

$$Cov(\theta_1, \theta_2) = \frac{1}{n-1}\sum(\theta_{1i} - \hat{\theta}_1)(\theta_{2i} - \hat{\theta}_2).$$

- Credible intervals - symmetric CI and HPD CI.

```
## R function to calcualte summary statistics
sum.stat<-function(x,alpha=.95){
          n<-length(x)
          est.x<-mean(x)
          sd.x<-sd(x)
          se.x<-sd.x/sqrt(n)
          CI<-quantile(x,probs=c(1-(1+alpha)/2,
                    (1+alpha)/2))
          HPD.CI<-emp.hpd(x,alpha)
          stat<-c(n,est.x,sd.x,se.x,CI,HPD.CI)
          names(stat)<-c('n','estimate','SD','
             MC SE',
                        'CI.L','CI.U','HPD.L','HPD
                           .U')
          stat
          }

rbind(sum.stat(theta[,1]),sum.stat(theta[,2])
   )
```

```
cov(theta[100:1000,])


## Function to find HPD based on simulation
emp.hpd<-function (x, alpha = 0.95)  {
    alpha <- min(alpha, 1 - alpha)
    n <- length(x)
    L.U <- round(n * alpha)
    x <- sort(x)
    e <- x[(n - L.U + 1):n] - x[1:L.U]
    m <- min(e)
    ind <- which(e == m)[1]
    return(c(x[ind], x[n - L.U + ind]))
}
```

Bivariate normal

	estimate	SD	MC SE	CI.L	CI.U	HPD.L	HPD.U
θ_1	0.006	0.997	0.010	-1.957	1.933	-1.982	1.903
θ_2	0.011	0.993	0.010	-1.936	1.940	-1.888	1.973

Homework (Problem 4.3.3 on Page 75)

Generate random numbers from a tri-variate normal distribution using the Gibbs sampling method. Let

$$\boldsymbol{\theta} = \begin{pmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \end{pmatrix} \sim MN\left(\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & \rho & \rho \\ \rho & 1 & \rho \\ \rho & \rho & 1 \end{bmatrix} \right).$$

Then the conditional distributions are

$$\begin{aligned} \theta_1|\theta_2,\theta_3 &\sim N\left(\frac{\rho(\theta_2+\theta_3)}{1+\rho}, 1-\frac{2\rho^2}{1+\rho}\right) \\ \theta_2|\theta_1,\theta_3 &\sim N\left(\frac{\rho(\theta_1+\theta_3)}{1+\rho}, 1-\frac{2\rho^2}{1+\rho}\right). \\ \theta_3|\theta_1,\theta_2 &\sim N\left(\frac{\rho(\theta_1+\theta_2)}{1+\rho}, 1-\frac{2\rho^2}{1+\rho}\right) \end{aligned}$$

Homework (cont'd)

For this problem, do the following

- Sample θ using Gibbs sampling method
- History plot and histogram
- Summary statistics (using sum.stat() function). Compare the results from $n = 1000, 10,000, 100,000$.
- Using `set.seed(0)` to set the random number seed at 0.

A simple missing data example

For the Catholic student example, suppose we asked 10 students and the results for the first 9 are $(1, 0, 1, 1, 1, 0, 1, 1, 0)$. However, for the 10th student, he/she did not respond. Thus, the datum is missing. A convenient method is to ignore the missing data and compute $\theta = 6/9 = .67$.

Gibbs sampling with missing data

Gibbs sampling can be used to deal with missing data problems. Let x_{10} denote the missing datum. Using a uniform prior $Beta(1,1)$, the posterior for the parameter θ is (conditional posterior assuming that x_{10} is k

$$\theta | x_1, \ldots, x_9, x_{10} \sim Beta(7 + x_{10}, 5 - x_{10}).$$

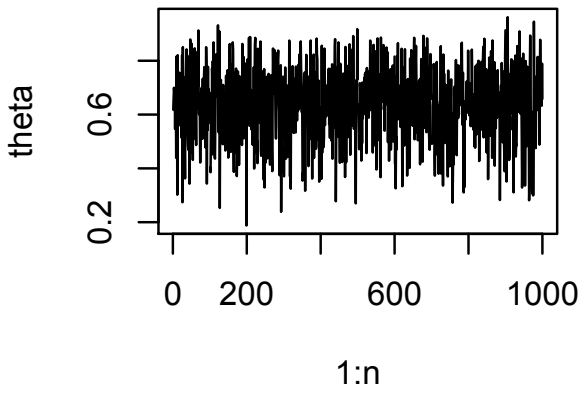
Given θ, the distribution of the missing datum is

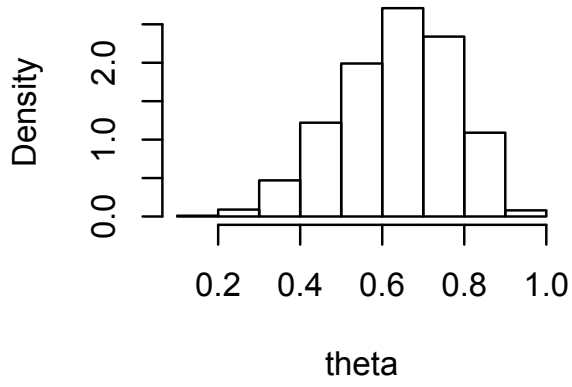
$$x_{10} | \theta \sim Bernoulli(\theta).$$

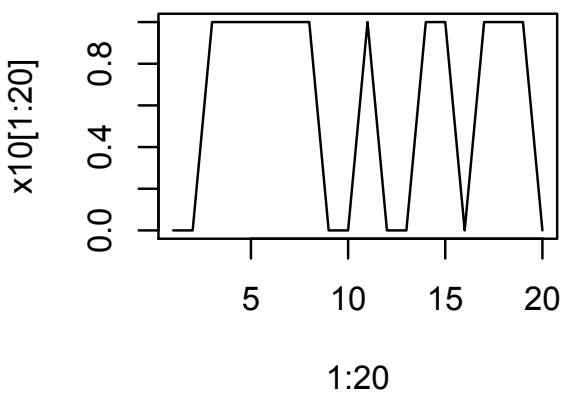
Clearly, we can implement the Gibbs sampling for the two conditional distribution to generate data for both θ and x_{10}.

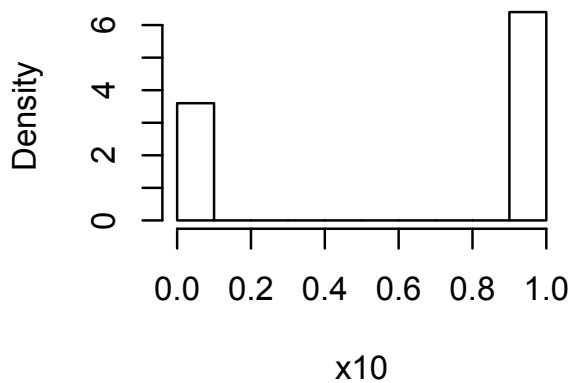
```
theta0<-.5
n<-10000
theta<-rep(NA,n)
x10<-rep(NA,n)

for (i in 1:n){
        x10[i]<-rbinom(1,1,theta0)
        theta[i]<-rbeta(1,7+x10[i],5-x10[i])
        theta0<-theta[i]
        }

par(mfrow=c(2,2))
plot(1:n,theta,type='l')
hist(theta,freq=F,main='')

plot(1:20,x10[1:20],type='l')
hist(x10,freq=F,main='')

sum.stat(theta)
```

History plots and histograms









Estimates

	estimate	SD	MC SE	CI.L	CI.U	HPD.L	HPD.U
θ	.633	.138	.00138	.349	.876	.361	.883
x_{10}	.6268		1:	6268		0:	3732

Metropolis-Hastings Sampling

The *Metropolis-Hastings sampling* (algorithm) is a more general method than Gibbs sampler for creating a Markov chain that can be used to generate a sequence of samples from a probability distribution that is difficult to sample from directly. If the conditional distributions are available, one can use Gibbs sampler. However, many times, the conditional distributions are not easy to obtain. Metropolis et al. (1953) first proposed the algorithm for a specific application and Hastings (1970) generalized it.

Algorithm

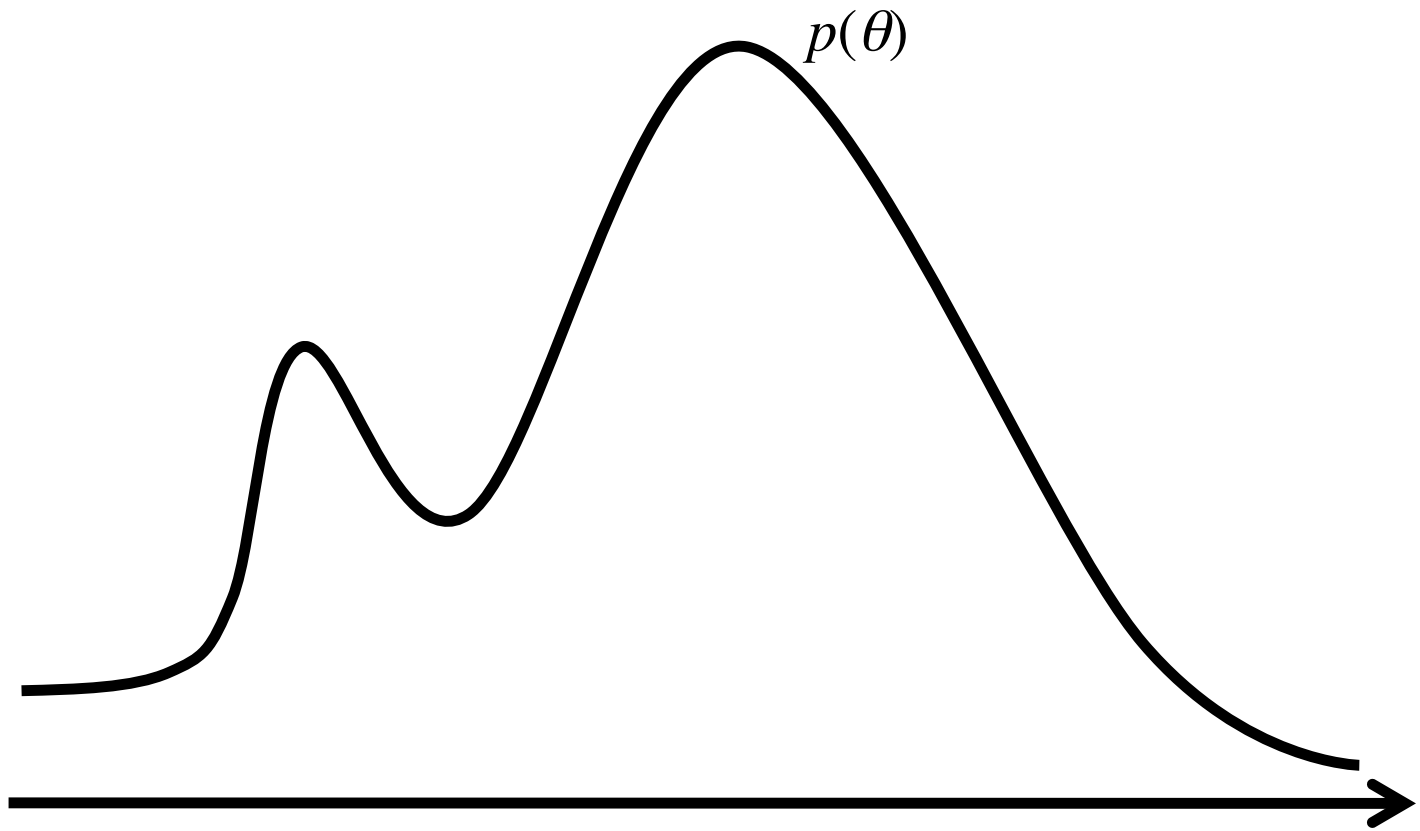
To sample from a distribution $p(\boldsymbol{\theta})$, the following Metropolis-Hastings algorithm can be followed.

1. Start with an initial value $\boldsymbol{\theta}^0$;
2. At step t, simulate a candidate $\boldsymbol{\theta}^*$ from a proposal or candidate distribution $q(\boldsymbol{\theta}^*|\boldsymbol{\theta}^{(t)})$;
3. Compute the ratio

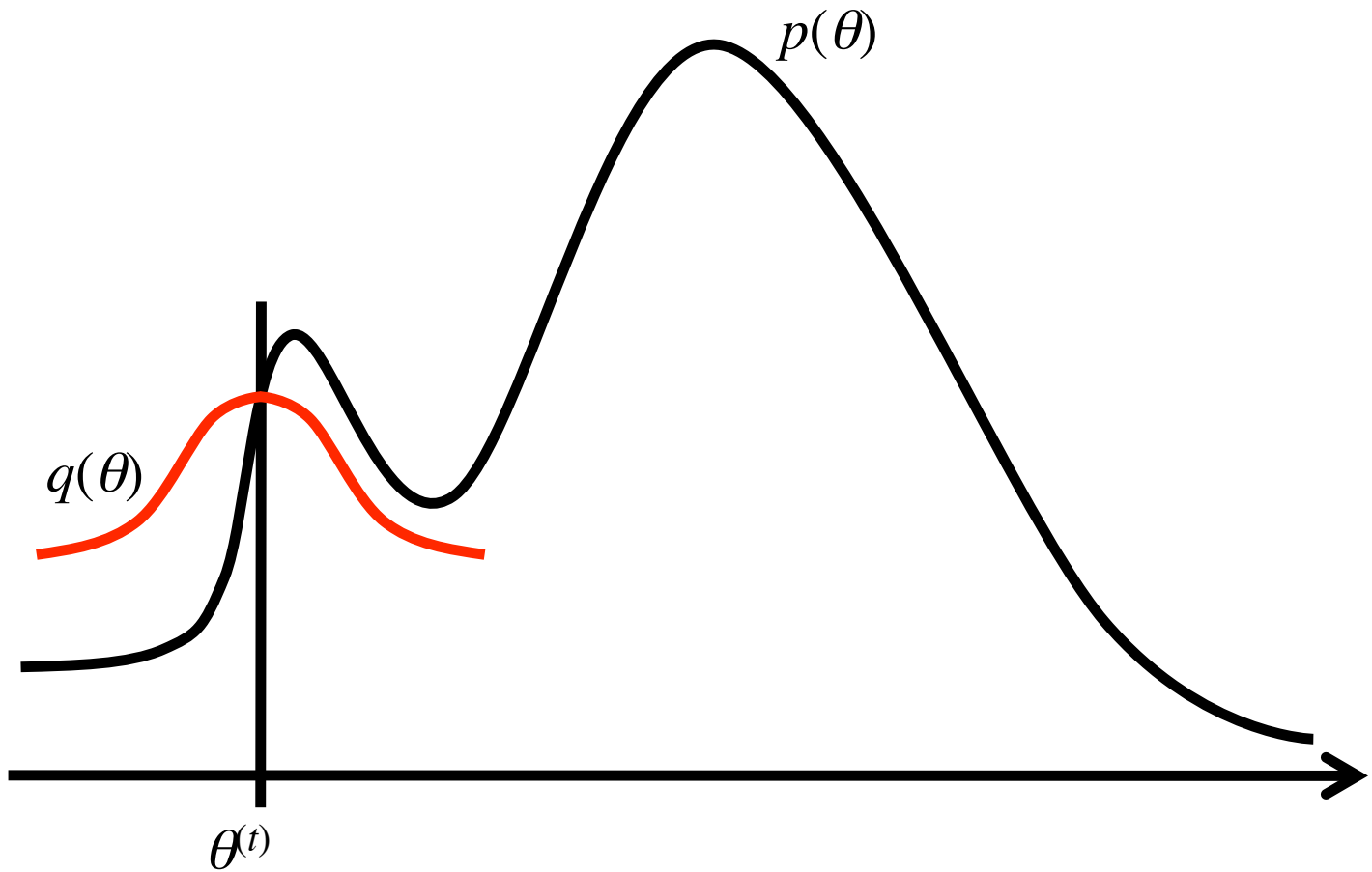
$$\alpha(\boldsymbol{\theta}^*, \boldsymbol{\theta}^{(t)}) = \frac{p(\boldsymbol{\theta}^*)}{p(\boldsymbol{\theta}^{(t)})} \frac{q(\boldsymbol{\theta}^{(t)}|\boldsymbol{\theta}^*)}{q(\boldsymbol{\theta}^*|\boldsymbol{\theta}^{(t)})};$$

4. The acceptance probability is $P = \min(\alpha, 1)$;
5. Accept $\boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}^*$ with probability P and $\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)}$ with probability $1 - P$.

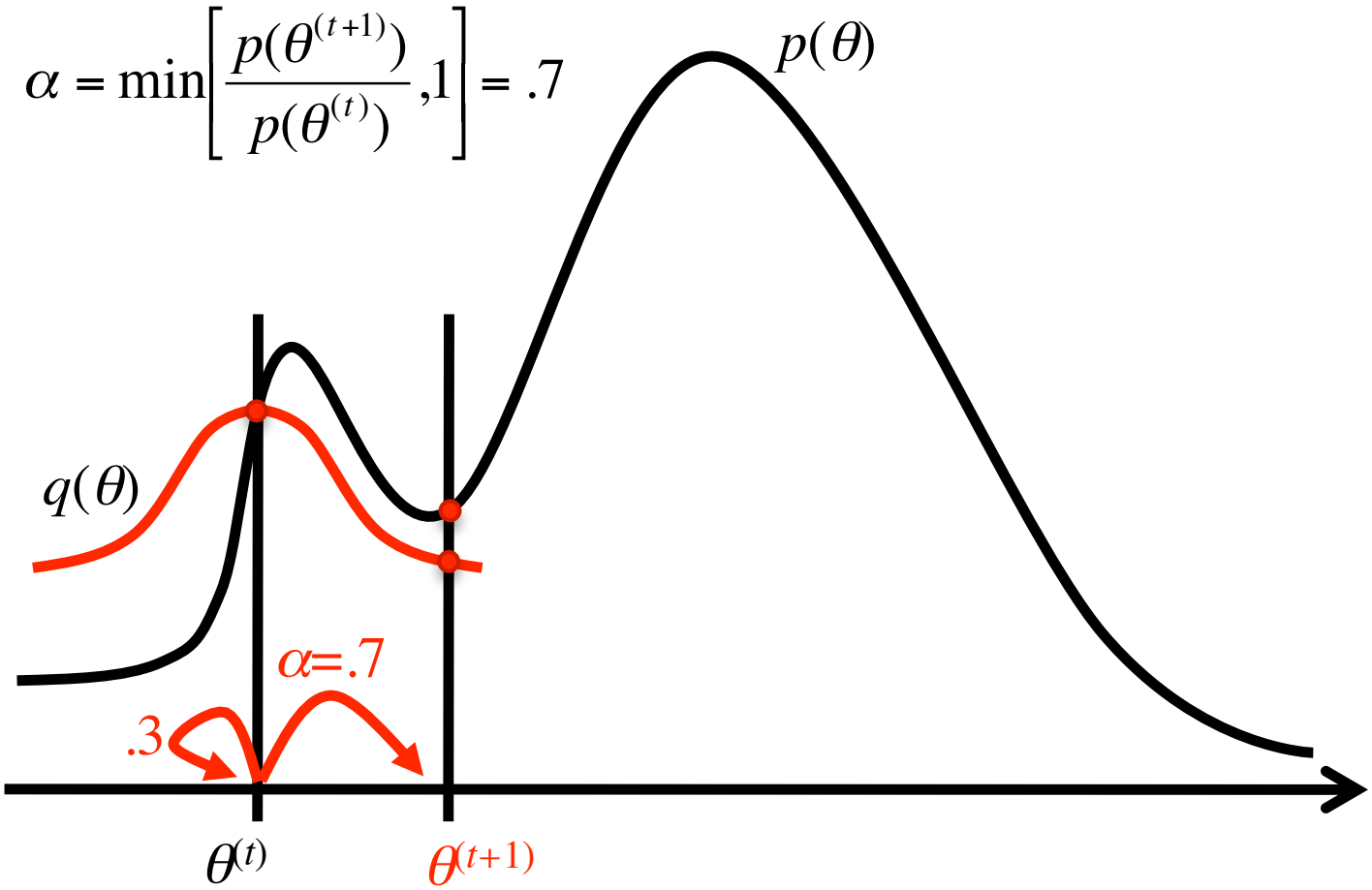
Metropolis-Hastings algorithm



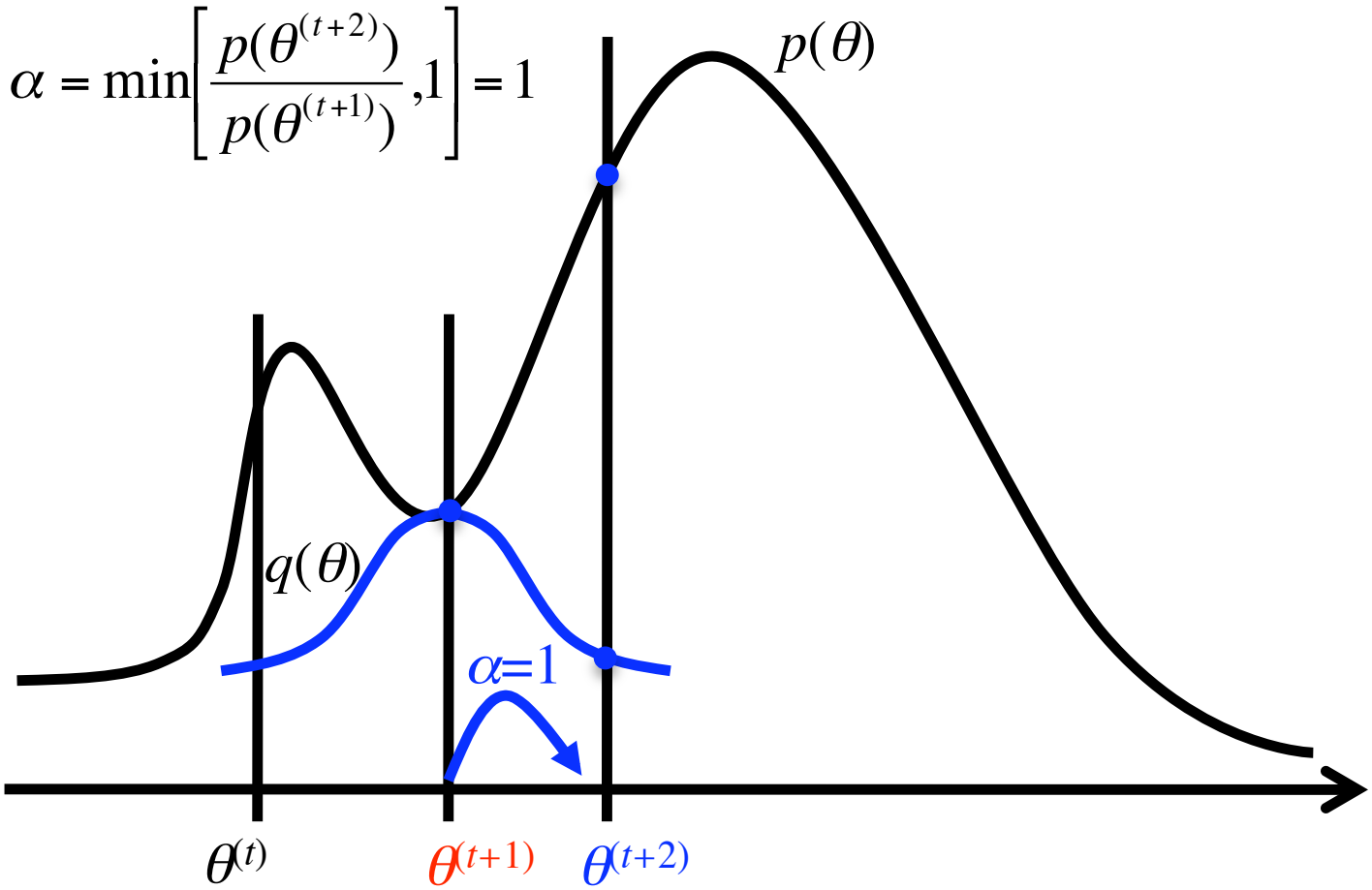
Metropolis-Hastings algorithm



Metropolis-Hastings algorithm



Metropolis-Hastings algorithm



Some general remarks

- Because we only need to define the ratio α, there is no need to know the normalized constant for $p(\boldsymbol{\theta})$.
- If the candidate distribution is symmetric so that $q(\boldsymbol{\theta}^{(t)}|\boldsymbol{\theta}^*) = q(\boldsymbol{\theta}^*|\boldsymbol{\theta}^{(t)})$, then the ratio can be simplified to the comparison of the target density $\alpha = p(\boldsymbol{\theta}^*)/p(\boldsymbol{\theta}^{(t)})$.
- The generated Markov chain can either stay in its current state or move to another state.
- Gibbs sampler is a special case of Metropolis-Hasting algorithm in which there is always a new movement, and thus usually faster.
- The choice of the candidate distribution is critical.
- It can be shown that the Markov chain generated converges to the stationary distribution $p(\boldsymbol{\theta})$. Clearly, it is necessary that the overall parameter space of the $q(\boldsymbol{\theta}^*|\boldsymbol{\theta}^{(t)})$ should be larger than that of the target distribution.

Symmetric Metropolis-Hasting sampling

If the candidate distribution is symmetric so that $q(\boldsymbol{\theta}^{(t)}|\boldsymbol{\theta}^*) = q(\boldsymbol{\theta}^*|\boldsymbol{\theta}^{(t)})$, then the ratio can be simplified to the comparison of the target density $\alpha = p(\boldsymbol{\theta}^*)/p(\boldsymbol{\theta}^{(t)})$. The multivariate normal distribution is a widely used symmetric candidate distribution.

$$q(\boldsymbol{\theta}^*, \boldsymbol{\theta}^{(t)}) \propto \exp\left[-\frac{1}{2}(\boldsymbol{\theta}^* - \boldsymbol{\theta}^{(t)})'\Sigma^{-1}(\boldsymbol{\theta}^* - \boldsymbol{\theta}^{(t)})\right]$$

Generate bivariate normal random numbers

As a (dummy) example, we demonstrate how to generate bivariate (correlated) normal data using the independent normal distribution (bivariate uncorrelated normal data) as the candidate distribution. Let

$$\boldsymbol{\theta} = \begin{pmatrix} \theta_1 \\ \theta_2 \end{pmatrix} \sim MN\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix} \right)$$

be the target distribution. The candidate distribution is

$$q(\boldsymbol{\theta}^*, \boldsymbol{\theta}) \propto \exp\left[-\frac{1}{2} (\boldsymbol{\theta}^* - \boldsymbol{\theta})^t (\boldsymbol{\theta}^* - \boldsymbol{\theta}) \right].$$

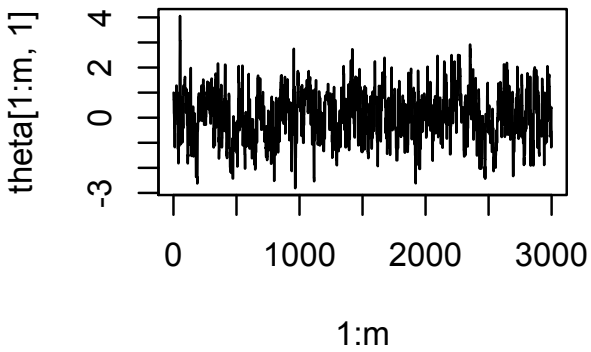
This is equivalent to two univariate normal distribution.

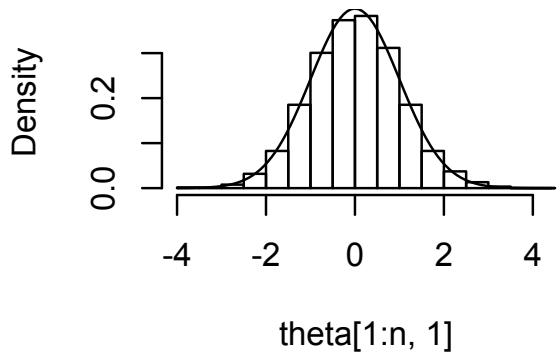
```
## Function to calculate the acceptance
   probability

alpha<-function(thetastar,thetat,rho){
 r.thetastar<--(thetastar[1]^2+thetastar[2]^2
      -2*rho*thetastar[1]*thetastar[2])
                    /(2*(1-rho^2))
 r.thetat<--(thetat[1]^2+thetat[2]^2
        -2*rho*thetat[1]*thetat[2])/(2*(1-rho
           ^2))
 exp(r.thetastar-r.thetat)
         }

rho<-.5

## starting values
theta0<-c(1,1)

n<-10000
```

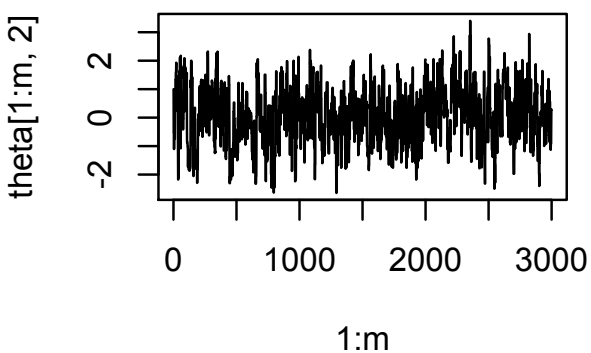
```
theta<-array(NA, dim=c(n,2))
P<-rep(NA,n)

for (i in 1:n){
 thetastar<-c(rnorm(1,theta0[1]),rnorm(1,
    theta0[2]))
 P[i]<-min(alpha(thetastar,theta0,rho),1)

 u<-runif(1)

 if (u<P[i]){ theta[i,]<-thetastar}
 else{theta[i,]<-theta0}

 theta0<-theta[i,]
}

## Plot the history and the density
par(mfrow=c(2,2))
m<-3000
```

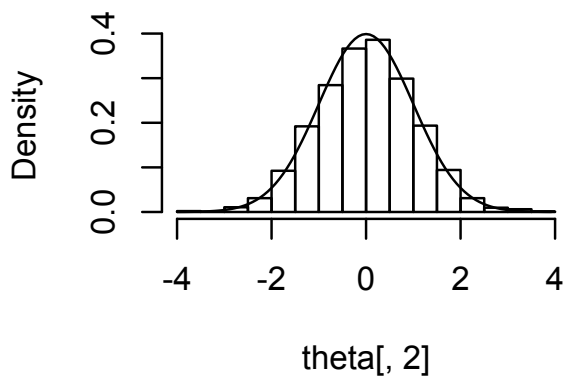
```
plot(1:m,theta[1:m,1],type='l')
hist(theta[1:n,1],freq=F,main='')
curve(dnorm,add=TRUE)

plot(1:m,theta[1:m,2],type='l')
hist(theta[,2],freq=F,main='')
curve(dnorm,add=TRUE)

## summary statistics
rbind(sum.stat(theta[,1]),sum.stat(theta[,2])
   )
cov(theta)

## The acceptance probability
length(unique(theta[,1]))/n
```

History plots and histograms







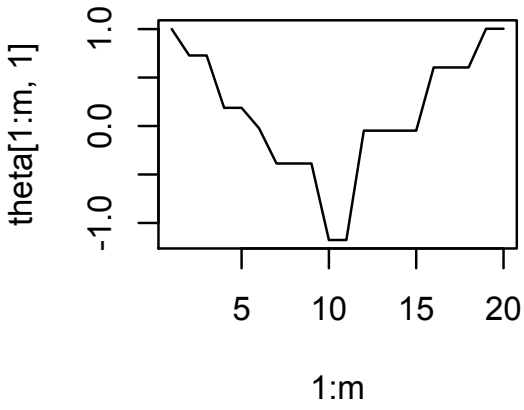


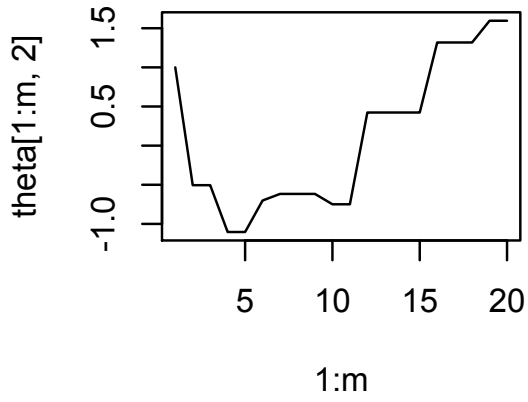
Summary statistics

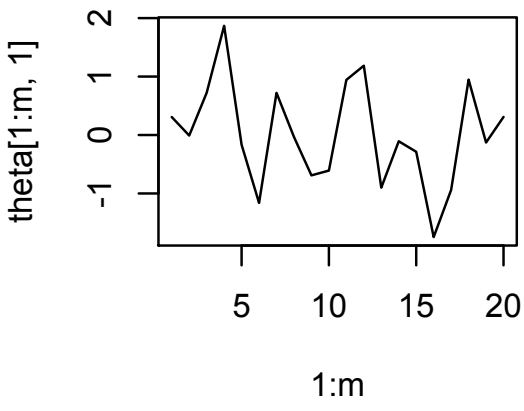
	estimate	SD	MC SE	CI.L	CI.U	HPD.L	HPD.U
θ_1	0.018	1.003	0.010	-1.946	2.013	-2.160	1.789
θ_2	0.013	1.023	0.010	-1.927	1.981	-1.921	1.983

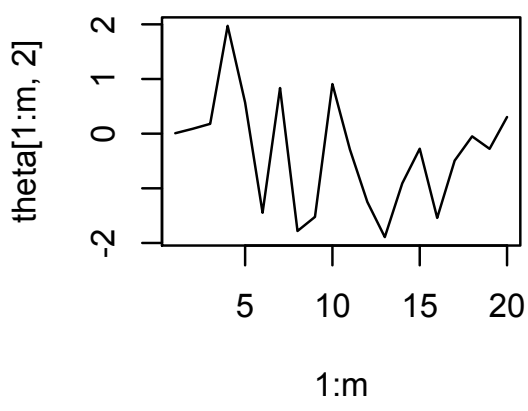
[,1] [,2]
[1,] 1.023346 0.480596
[2,] 0.480596 0.957540

Gibbs sampler vs. M-H algorithm









Homework

Problem 4.4.4. Generate random numbers from

$$\boldsymbol{\theta} = \begin{pmatrix} \theta_1 \\ \theta_2 \end{pmatrix} \sim MN\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix} \right)$$

using both the Gibbs sampling and Metropolis-Hastings sampling methods and compare the mean and covariance matrix of $\boldsymbol{\theta}$ based on generated data. Use the random number seed 0 (set.seed(0)) and use two difference number of iterations, $n = 1,000$ and $n = 10,000$.

Independent Metropolis-Hastings Algorithm

If $q(\boldsymbol{\theta}^*|\boldsymbol{\theta}^{(t)}) = q(\boldsymbol{\theta}^*)$ so that the proposal distribution is independent of the previous state, the algorithm is called independent M-H algorithm. The algorithm can be summarized as Given $\boldsymbol{\theta}^{(t)}$,

1. Generate $\boldsymbol{\theta}^*$ from $q(\boldsymbol{\theta}^*)$;
2. Compute the probability

$$P = \min\left(1, \frac{p(\boldsymbol{\theta}^*)}{p(\boldsymbol{\theta}^{(t)})} \frac{q(\boldsymbol{\theta}^{(t)})}{q(\boldsymbol{\theta}^*)}\right),$$

3. Take

$$\boldsymbol{\theta}^{(t+1)} = \begin{cases} \boldsymbol{\theta}^* & \text{with probability } P \\ \boldsymbol{\theta}^{(t)} & \text{with probability } 1-P \end{cases}.$$

Bivariate random numbers

Let

$$\boldsymbol{\theta} = \begin{pmatrix} \theta_1 \\ \theta_2 \end{pmatrix} \sim MN\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix} \right)$$

be the target distribution. The candidate distribution is

$$q(\boldsymbol{\theta}^*, \boldsymbol{\theta}) \propto \exp\left[-\frac{1}{2} (\boldsymbol{\theta}^*)^t (\boldsymbol{\theta}^*) \right].$$

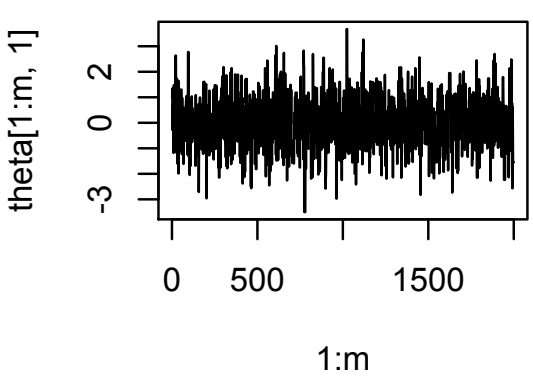
This is equivalent to two independent univariate normal distribution.

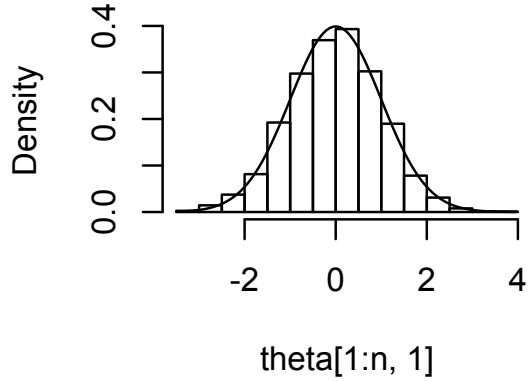
```
## Function to calculate the acceptance
   probability

alpha<-function(thetastar,thetat,rho){
 r.thetastar.t<--(thetastar[1]^2+thetastar
     [2]^2
    -2*rho*thetastar[1]*thetastar[2])/(2*(1-
       rho^2))
 r.thetat.t<--(thetat[1]^2+thetat[2]^2
     -2*rho*thetat[1]*thetat[2])/(2*(1-rho^2))
 r.thetastar.c<--(thetastar[1]^2+thetastar
     [2]^2)/2
 r.thetat.c<--(thetat[1]^2+thetat[2]^2)/2
 exp(r.thetastar.t+r.thetat.c
      -r.thetat.t-r.thetastar.c)
 }

rho<-.5
## starting values
```

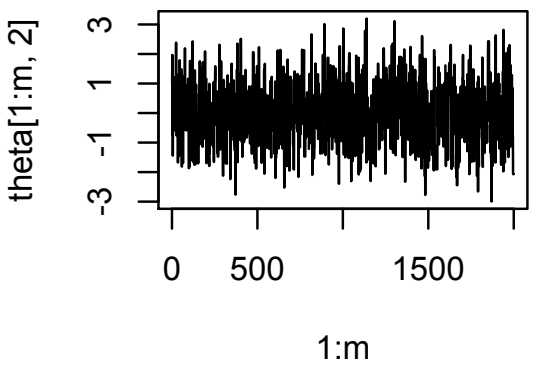
```
theta0<-c(1,1)
n<-10000
theta<-array(NA, dim=c(n,2))
P<-rep(NA,n)

for (i in 1:n){
 thetastar<-rnorm(2)
 P[i]<-min(alpha(thetastar,theta0,rho),1)
 u<-runif(1)

 if (u<P[i]){ theta[i,]<-thetastar}
 else{theta[i,]<-theta0}

 theta0<-theta[i,]
}

cov(theta)

## Plot the history and the density
```

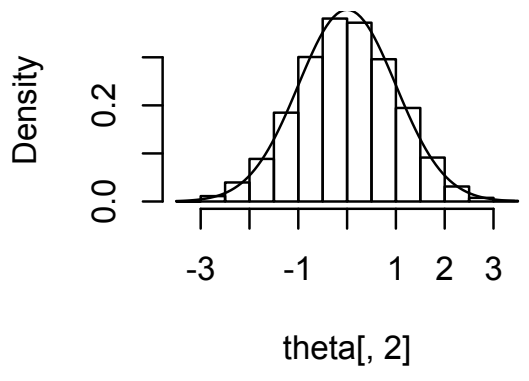
```
par(mfrow=c(2,2))
m<-2000
plot(1:m,theta[1:m,1],type='l')
hist(theta[1:n,1],freq=F,main='')
curve(dnorm,add=TRUE)

plot(1:m,theta[1:m,2],type='l')
hist(theta[,2],freq=F,main='')
curve(dnorm,add=TRUE)

## Summary statistics
rbind(sum.stat(theta[,1]),sum.stat(theta[,2])
   )
length(unique(theta[,1]))/n
```

History plots and histograms









M-H algorithm and accept-reject sampling

- The independent M-H method looks like a generalized accept-reject method.
- In general, the independent M-H method is more efficient than the accept-reject method because it can accept more proposed values.
- To produce an ergodic Markov chain, accept-reject method requires there exists a constant M so that $p(x) \leq Mq(x)$. However, for the M-H method, we don't need to know the constant exactly.

Random Walk M-H algorithm

Random walk M-H algorithm generates a candidate value by exploring the neighborhood of the current state so that

$$\boldsymbol{\theta}^* = \boldsymbol{\theta}^{(t)} + \boldsymbol{\epsilon}_t$$

where $\boldsymbol{\epsilon}_t$ is a random perturbation with distribution q. $\boldsymbol{\epsilon}_t$ itself is a random walk Markov chain. Clearly, the distribution q is of the form

$$q(\boldsymbol{\theta}^* | \boldsymbol{\theta}^{(t)}) = q(\boldsymbol{\theta}^* - \boldsymbol{\theta}^{(t)}).$$

- The widely used q functions include multivariate normal distribution and the multivariate t-distribution.
- SAS PROC MCMC procedure uses the random walk M-H algorithm and the basic proposal function is the multivariate normal distribution.

Optimizing the M-H algorithm

- Choosing appropriate candidate function. For example, for the independent M-H, p/q should be bounded. It will be better to choose a q that has the similar shape with p.
- Controlling the acceptance rate. It is not always good to have a higher acceptance rate. For the random walk M-H, the acceptance rate between .15 and .5 is good. If both the target and proposal densities are normal, the optimal acceptance rate for uni-dimensional case is about .45 and .234 for higher dimensions.
- For the random walk M-H, if the multivariate normal distribution is used as the proposal function, the covariance matrix can be tuned for better acceptance rate.
- In general, trying difference starting values with shorter chains can be used as a strategy to find a better initial value for longer simulations.

Slice sampler

- The Metropolis-Hastings algorithm is a very general and generic MCMC method.
- However, it can be slow, less efficient.
- There are special methods that are less generic but can be efficient for specific problems.
- One of the methods is called *slice sampler / sampling*

Basic idea

Recall the accept-reject sampling method. Sampling from $f(x)$ is equivalent to generating data from the joint uniform distribution

$$(X, U) \sim U\{(x, u) : 0 \leq u \leq f(x)\}.$$

The joint density of (X, U) is

$$p(x, u) = 1/c$$

where $\int f(x)dx = c$. Then the marginal density for X is

$$p(x) = \int_0^{f(x)} 1/c du = f(x)/c.$$

So to sample x, we can sample from the joint distribution uniformly and only keep x. The variable U is often called auxiliary variable.

Slice sampling algorithm

Given the conditional distributions

$$\begin{aligned} U|(X &= x) \sim U\{u:\ 0 \le u \le f(x)\} \\ X|(U &= u) \sim U\{x:\ f(x) \ge u\}. \end{aligned}$$

the slice sampler algorithm can be summarized as in the following. At iteration t,

1. Generate $u^{(t+1)} \sim U(0, f(x^{(t)}))$;
2. Generate $x^{(t+1)} \sim U(x \in A^{(t+1)})$ where $A^{(t+1)} = \{x:\ f(x) \ge u^{(t+1)}\}$.

An example

Generate data from $p(x) = \exp(-\sqrt{x})/2, x > 0$. To apply slice sampler, we can generate (U, X) from

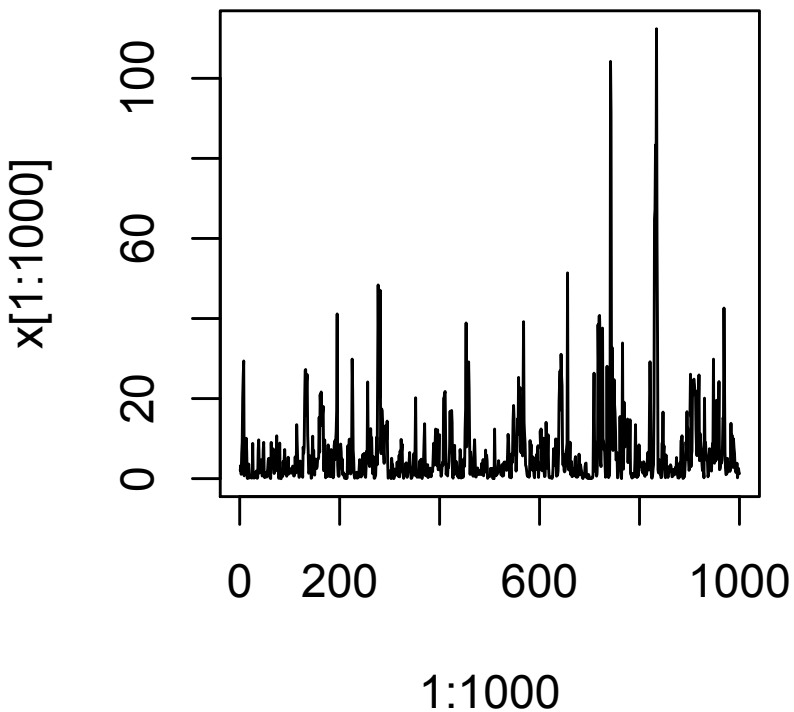
$$\begin{aligned} U|x &\sim U(0, \exp(-\sqrt{x})/2) \\ X|u &\sim U(0, [\log(2u)]^2). \end{aligned}$$

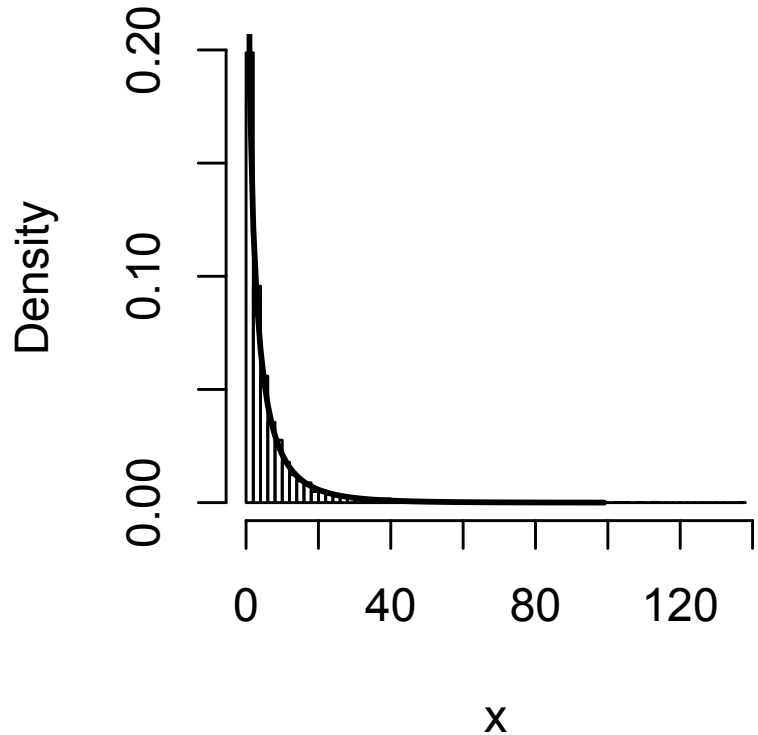
In this example,

$$A^{(t+1)} = \{x : \exp(-\sqrt{x})/2 > u \,\&\, x > 0\} = \{x : \log^2(2u) > x > 0\}.$$

```
## slice sampler
n<-10000
u<-rep(NA, n)
x<-rep(NA,n)
x0<-1
for (i in 1:n){
        u[i]<-runif(1,0,exp(-sqrt(x0))/2)
        x[i]<-runif(1,0,(log(2*u[i]))^2)
        x0<-x[i]
        }

par(mfrow=c(1,2))
plot(1:1000,x[1:1000],type='l')

hist(x,breaks=50,probability=T,main='')
lines(0:100,exp(-sqrt(0:100))/2,lwd=2)
```

History plot and Histogram





A general slice sampler

Sometimes, the uniform region A may not be easy to calculate. If the target function can be written as the product of k functions that are easy to calculate the uniform regions,

$$p(x) \propto f_1(x) \dots f_k(x),$$

then the following slice sampling algorithm can be used

- Generate $u_i^{(t+1)} \sim U(0, f_i(x^{(t)}))$
- Generate $x^{(t+1)} \sim U_{A^{(t+1)}}$ where

$$A^{(t+1)} = \cap_{i=1}^{k} \{x : f_i(x) \geq u_i^{(t+1)}\}.$$

Note that the function $f_i(x)$ may not be a density function.

An example

Generate random numbers from the density that is proportional to

$$p(x) \propto \exp(-|x|)\exp(-x^2/2).$$

We can use

$$\begin{aligned} f_1(x) &= \exp(-|x|) \\ f_2(x) &= \exp(-x^2/2). \end{aligned}$$

The uniform region to sample from is

$$A = \{x : |x| \leq -\log u_1\} \cap \{x : |x| \leq \sqrt{-2\log u_2}\}.$$

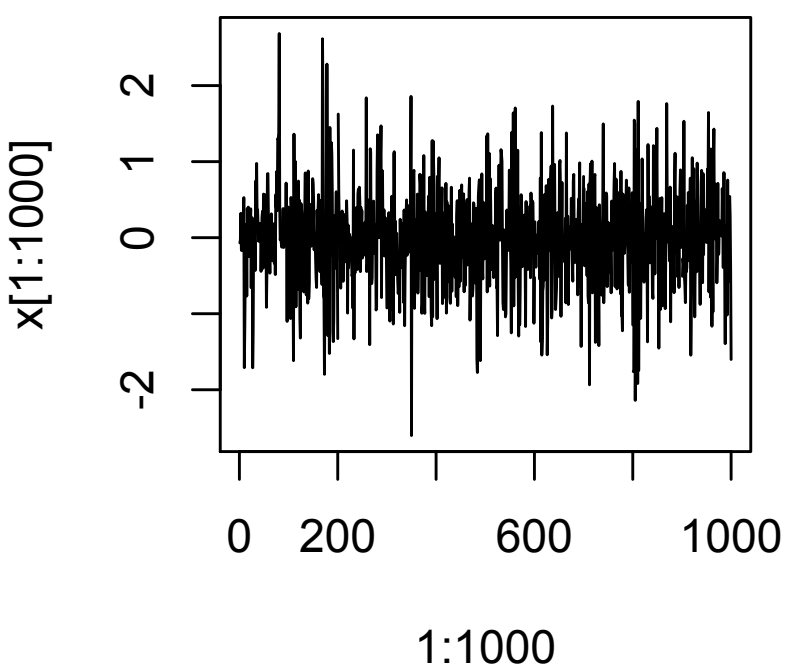
In this case, we generate u_1 and u_2 from

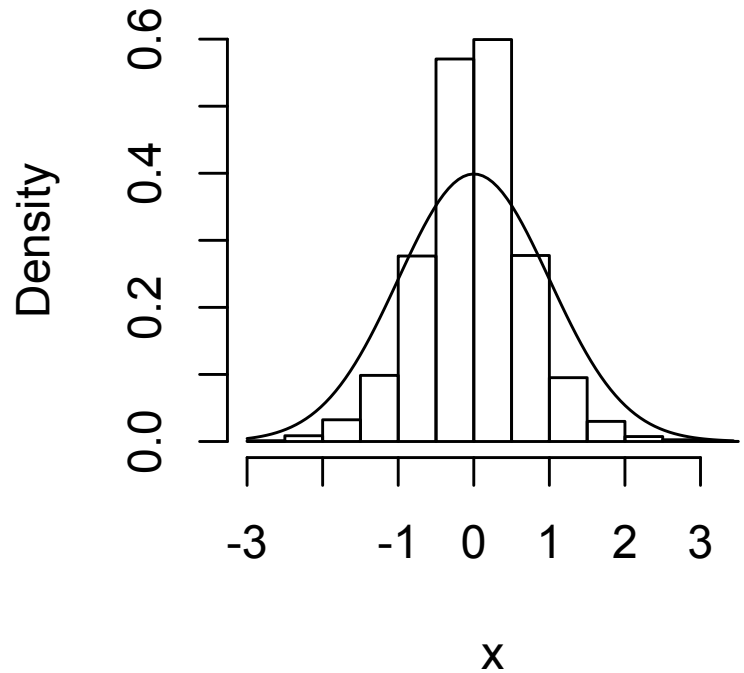
$$\begin{aligned} u_1 &\sim U(0, f_1(x)) \\ u_2 &\sim U(0, f_2(x)). \end{aligned}$$

```
## slice sampler
n<-10000
x0<-0

x<-rep(NA,n)

for (i in 1:n){
        u1<-runif(1,0,exp(-abs(x0)))
        u2<-runif(1,0,exp(-x0^2/2))

        a.l.1<-log(u1)
        a.l.2<--sqrt(-2*log(u2))
        a.u.1<--log(u1)
        a.u.2<-sqrt(-2*log(u2))

        a.l<-max(a.l.1,a.l.2)
        a.u<-min(a.u.1,a.u.2)

        x[i]<-runif(1,a.l,a.u)
```

```
        x0<-x[i]
        }

par(mfrow=c(1,2))
plot(1:1000,x[1:1000],type='l')
hist(x,freq=F,main='')
curve(dnorm,add=T)
```

History plot and histogram





Data augmentation method

The *data augmentation* algorithm was first proposed by Tanner and Wong (1987) to deal with missing data or latent variables. Let $\boldsymbol{x} = (\boldsymbol{x}_{obs}, \boldsymbol{x}_{mis})$ where $\boldsymbol{x}_{obs}$ is observed but $\boldsymbol{x}_{mis}$ is missing. With missing data or latent variables, the posterior distribution of interests is

$$p(\theta|\boldsymbol{x}_{obs}) = \int_{\boldsymbol{x}_{mis}} p(\theta|\boldsymbol{x}_{obs}, \boldsymbol{x}_{mis}) p(\boldsymbol{x}_{mis}|\boldsymbol{x}_{obs}) d\boldsymbol{x}_{mis}$$

where the distribution $p(\boldsymbol{x}_{mis}|\boldsymbol{x}_{obs})$ is the predictive distribution. If we can generate $\boldsymbol{x}_{mis}$ from the predictive distribution, the posterior distribution can be approximated by

$$p(\theta|\boldsymbol{x}_{obs}) = \frac{1}{m} \sum_{j=1}^{m} p(\theta|\boldsymbol{x}_{obs}, \boldsymbol{x}_{mis,j})$$

based on Monte Carlo integration.

Data augmentation algorithm

The basic data augmentation algorithm is

1. Generate m different values for the missing data vector $\boldsymbol{x}_{mis,j},\ j = 1, \ldots, m$ from $p(\boldsymbol{x}_{mis}|\boldsymbol{x}_{obs})$;
2. Calculate the posterior distribution of θ using

$$p(\theta|\boldsymbol{x}_{obs}) = \frac{1}{m}\sum_{j=1}^{m} p(\theta|\boldsymbol{x}_{obs}, \boldsymbol{x}_{mis,j}).$$

Note that the first step is actually a data imputation step in which we impute the missing data m times.

Alternative algorithm

1. Imputation step
   - 1.1 Generate θ from $\theta \sim p(\theta|\boldsymbol{x}_{obs})$
   - 1.2 Generate missing data $\boldsymbol{x}_{mis}$, from $p(\boldsymbol{x}_{mis}|\theta, \boldsymbol{x}_{obs})$
   - 1.3 Repeat (1.1) and (1.2) for m times to get $\boldsymbol{x}_{mis,j}$ $j = 1, \ldots m$
2. Posterior step. Calculate the posterior distribution of θ using

$$p(\theta|\boldsymbol{x}_{obs}) = \frac{1}{m}\sum_{j=1}^{m} p(\theta|\boldsymbol{x}_{obs}, \boldsymbol{x}_{mis,j}).$$

EM algorithm and data augmentation

The basic EM algorithm consists of two steps - the expectation step (E-step) and the maximization step (M-step). In the E-step, we calculate

$$\hat{\boldsymbol{x}}_{mis} = E(\boldsymbol{x}_{mis}) = \int \boldsymbol{x}_{mis} p(\boldsymbol{x}_{mis}|\hat{\theta}, \boldsymbol{x}_{obs}) d\boldsymbol{x}_{mis}.$$

Then in the M-step, we estimate the unknown parameters by

$$\hat{\theta} = \max_{\theta} p(\theta|\boldsymbol{x}_{obs}, \hat{\boldsymbol{x}}_{mis}).$$

Data augmentation and Gibbs sampler

Imputing multiple missing data in data augmentation algorithm can increase the accuracy of $p(\theta|\boldsymbol{x}_{obs})$ but not required. Furthermore, there is no need to estimate $p(\theta|\boldsymbol{x}_{obs})$. Recall from the properties of Gibbs sampling, if we can sample iteratively

$$\theta \sim p(\theta|\boldsymbol{x}_{mis}, \boldsymbol{x}_{obs})$$

$$\boldsymbol{x}_{mis} \sim p(\boldsymbol{x}_{mis}|\theta, \boldsymbol{x}_{obs}),$$

the generated Markov chain for $\boldsymbol{x}_{mis}$ can be viewed from the distribution of $p(\boldsymbol{x}_{mis}|\boldsymbol{x}_{obs})$ and the generated θ can be viewed from $p(\theta|\boldsymbol{x}_{obs})$. Thus, the data augmentation algorithm can be converted to a purely Gibbs sampling procedure as

1. Generate θ from $\theta \sim p(\theta|\boldsymbol{x}_{mis}, \boldsymbol{x}_{obs})$
2. Generate missing data $\boldsymbol{x}_{mis}$, from $p(\boldsymbol{x}_{mis}|\theta, \boldsymbol{x}_{obs})$.

General data augmentation algorithm based on Gibbs sampling

For a general case, let z denote the missing data, latent variables, or the other auxiliary variables so that the distributions of $p(\theta|\boldsymbol{x}, z)$ and $p(z|\theta, \boldsymbol{x})$ can be obtained relatively easily. The following algorithm can be used to sample for θ.

1. Sample $z \sim p(z|\theta, \boldsymbol{x})$;
2. Sample $\theta \sim p(\theta|\boldsymbol{x}, z)$.

Genetic linkage (Rao, 1965).

A classic example of data augmentation problem is the genetics problem that has been analyzed using the EM algorithm. In the example, 197 animals are distributed multinomially as

$$\begin{aligned} \boldsymbol{x} &= (x_1, x_2, x_3, x_4) = (125, 18, 20, 34) \\ &\sim \mathit{Multi}\left(197;\ \frac{1}{2} + \frac{\theta}{4}, \frac{1-\theta}{4}, \frac{1-\theta}{4}, \frac{\theta}{4}\right) \end{aligned}$$

The direct MLE estimation of θ is not very easy given the following likelihood function

$$L(\theta, \boldsymbol{x}) = \frac{197!}{125!18!20!34!}\left(\frac{1}{2} + \frac{\theta}{4}\right)^{125}\left(\frac{1-\theta}{4}\right)^{38}\left(\frac{\theta}{4}\right)^{34}.$$

EM algorithm

Dempster et al. (1977) use the augmented method (EM algorithm) by splitting the first cell to (z_1, z_2) so that

$$\boldsymbol{x} = (z_1, z_2, x_2, x_3, x_4) \sim \mathit{Multi}\left(197;\ \frac{1}{2}, \frac{\theta}{4}, \frac{1-\theta}{4}, \frac{1-\theta}{4}, \frac{\theta}{4}\right).$$

The likelihood function is

$$L(\theta|z, \boldsymbol{x}) \propto \theta^{z_2+x_4}(1-\theta)^{x_2+x_3} = \theta^{z_2+34}(1-\theta)^{38}.$$

Using EM algorithm, in M-step, one has

$$\hat{\theta} = \frac{z_2 + 34}{z_2 + 72}.$$

From the E-step,

$$E(z_2) = \frac{125\theta}{\theta + 2}.$$

The estimate of $\hat{\theta} = .6268$.

Data augmentation

Using the improper prior $p(\theta) \propto 1$, the posterior is

$$p(\theta|z, x) \propto \frac{1}{(125 - z_2)! z_2!} \theta^{z_2+34} (1-\theta)^{38}.$$

Thus,

$$\theta|z, x \sim Beta(z_2 + 35, 39).$$

For z_2, it has a binomial distribution

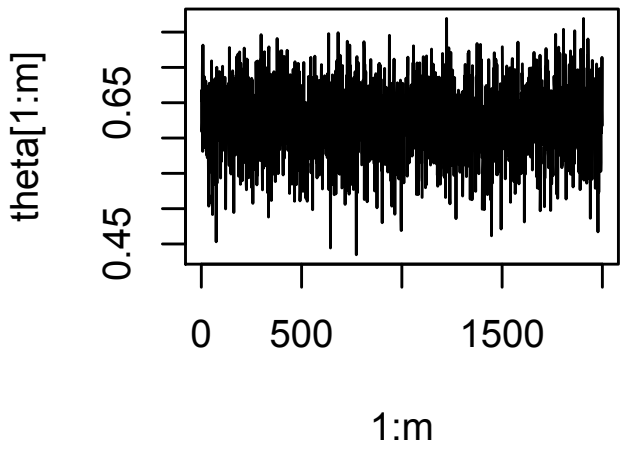
$$z_2|\theta, x \sim Binomial(125, \frac{\theta}{2+\theta})$$

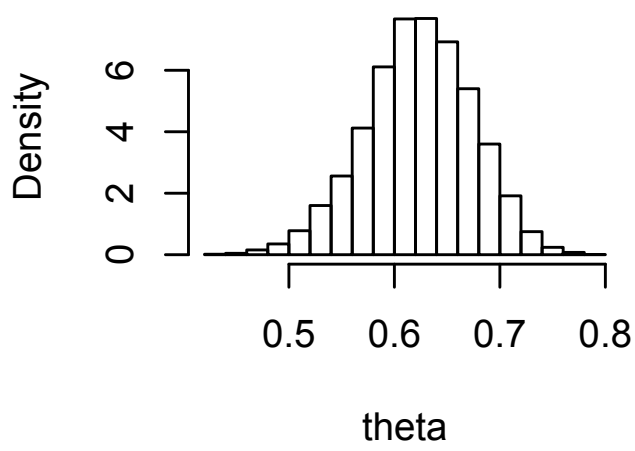
because

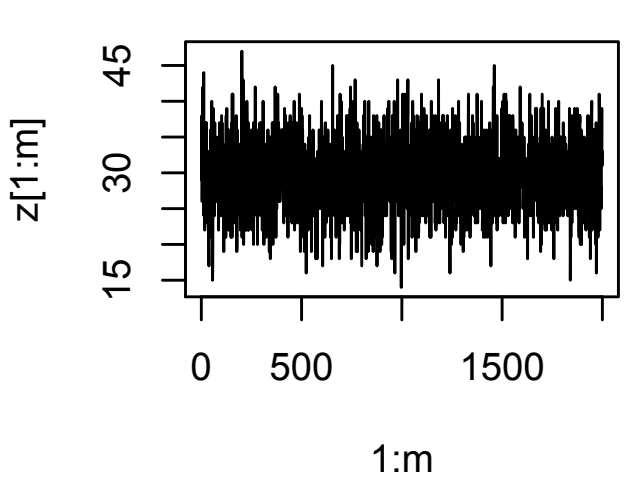
$$\begin{aligned} p(z_2|\theta, x) &\propto \frac{1}{(125 - z_2)! z_2!} \left(\frac{1}{2}\right)^{125-z_2} \left(\frac{\theta}{4}\right)^{z_2} \\ &\propto \frac{1}{(125 - z_2)! z_2!} \left(\frac{1}{2}\right)^{125-z_2} \left(\frac{\theta}{4}\right)^{z_2} / \left(\frac{1}{2} + \frac{\theta}{4}\right)^{125} \\ &= \frac{125!}{(125 - z_2)! z_2!} \left(\frac{2}{2+\theta}\right)^{125-z_2} \left(\frac{\theta}{2+\theta}\right)^{z_2}. \end{aligned}$$

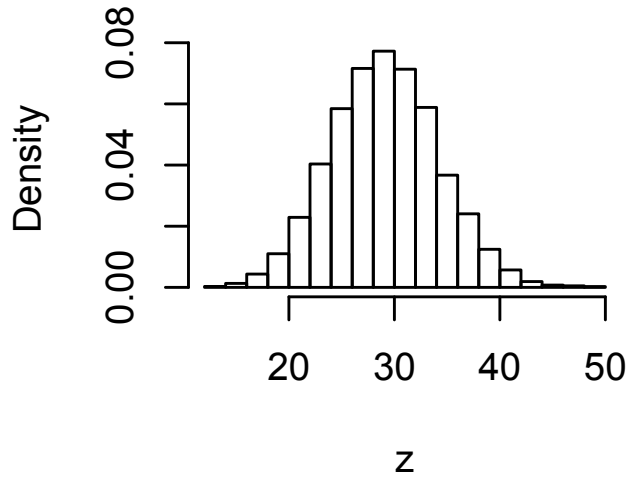
```
## data augmentation
n<-1000
theta<-rep(NA, n)
z<-rep(NA,n)
theta0<-.5
for (i in 1:n){
        z[i]<-rbinom(1,125,theta0/(2+theta0))
        theta[i]<-rbeta(1,z[i]+35,39)
        theta0<-theta[i]
        }
```

Plots


theta[1:m]
0.45
0.65
0
500
1500
1:m


Density
0
2
4
6
0.5
0.6
0.7
0.8
theta


z[1:m]
15
30
45
0
500
1500
1:m


Density
0.00
0.04
0.08
20
30
40
50
z

Estimate

	n	estimate	SD	MC SE	CI.L	CI.U	HPD.L	HPD
θ	10000	0.62	0.05	0.00	0.52	0.72	0.52	0.72
z_2	10000	29.60	5.10	0.05	20	40	18	38

Applications of data augmentation methods

- Missing data
- Latent variables - Factor analysis, SEM
- Categorical data - Logistic regression, Tobit regression, mixture model