HW7

Cheng Huang 2658312

26 October 2018

Normal Mixture revisite

Find posterior density

The likelihood function given $\delta, \mu_1, \mu_2, \sigma_1, \sigma_2$ is

$$f(x|\delta, \mu_1, \mu_2, \sigma_1, \sigma_2) = \prod_{i=1}^{n}[\delta\frac{1}{\sqrt{2\pi}\sigma_1}e^{-\frac{(x_i-\mu_1)^2}{2\sigma_1^2}} + (1-\delta)\frac{1}{\sqrt{2\pi}\sigma_2}e^{-\frac{(x_i-\mu_2)^2}{2\sigma_2^2}}]$$

The prior for μ_1, μ_2 are normal $N(0, 10^2)$, so

$$\pi(\mu_1) \propto exp(-\frac{\mu_1^2}{200})$$

$$\pi(\mu_2) \propto exp(-\frac{\mu_2^2}{200})$$

The prior for σ_1^2, σ_2^2 are $IG(0.5, 10)$, so

$$\pi(\sigma_1^2) \propto (\sigma_1^2)^{-1.5}exp(-\frac{10}{\sigma_1^2})$$

$$\pi(\sigma_2^2) \propto (\sigma_2^2)^{-1.5}exp(-\frac{10}{\sigma_2^2})$$

Therefore the posterior distribution propotional to

$$q(\delta, \mu_1, \mu_2, \sigma_1, \sigma_2|x) \propto \prod_{i=1}^{n}[\delta\frac{1}{\sqrt{2\pi}\sigma_1}e^{-\frac{(x_i-\mu_1)^2}{2\sigma_1^2}} + (1-\delta)\frac{1}{\sqrt{2\pi}\sigma_2}e^{-\frac{(x_i-\mu_2)^2}{2\sigma_2^2}}] \times exp(-\frac{\mu_1^2}{200}) \times exp(-\frac{\mu_2^2}{200})$$

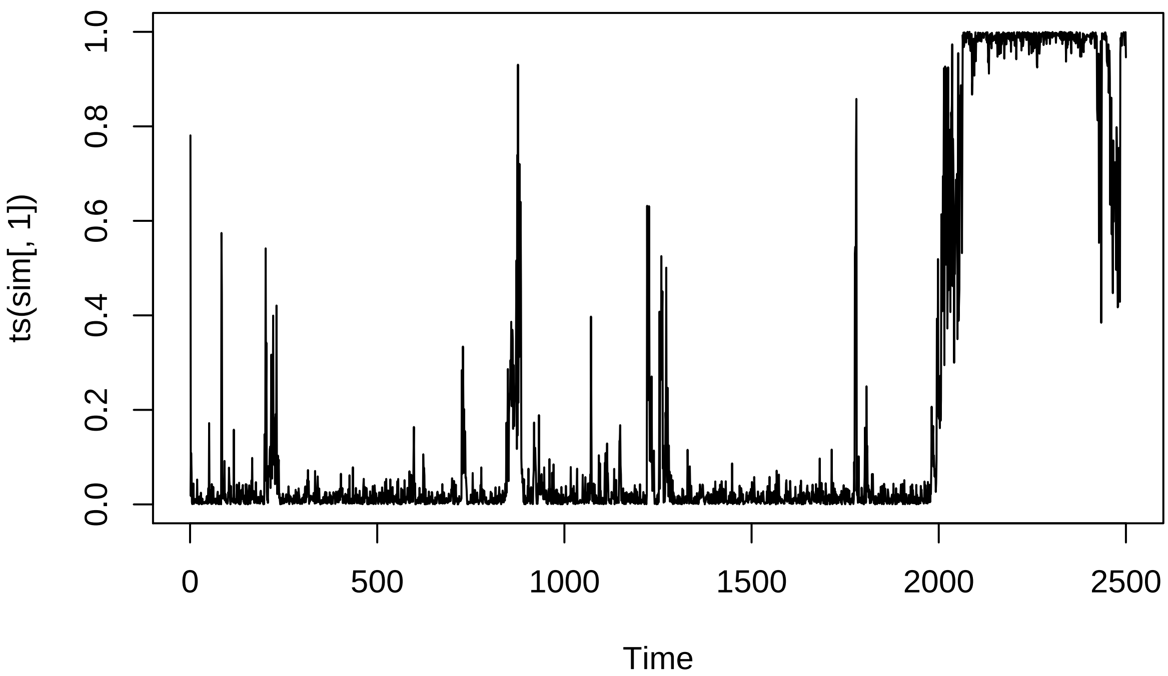
$$\times(\sigma_1^2)^{-1.5}exp(-\frac{10}{\sigma_1^2}) \times (\sigma_2^2)^{-1.5}exp(-\frac{10}{\sigma_2^2})$$

MCMC

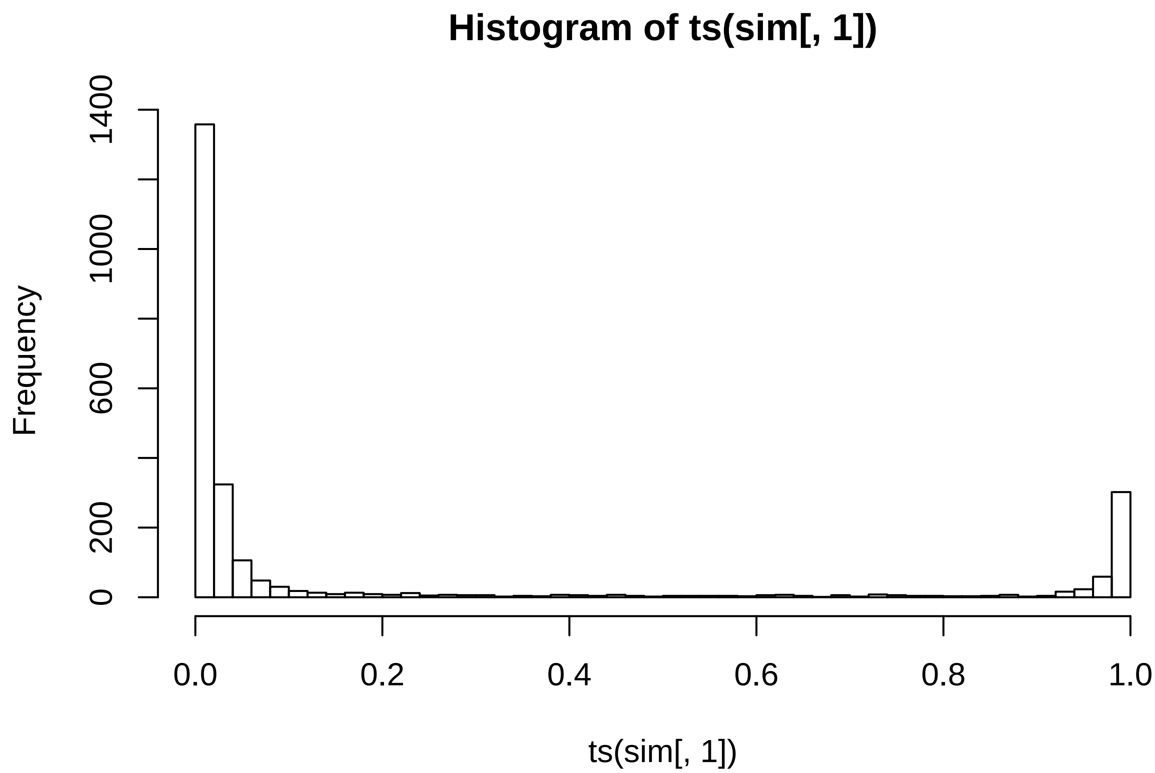
```r
library("invgamma")
library("HI")
n <- 100
## set true value of parameters
delta <- 0.7
mu1 <- 7
mu2 <- 10
```

```
sigmasq1 <- 0.5
sigmasq2 <- 0.5

## simulation
set.seed(123)
u <- rbinom(n, prob = delta, size = 1)
d <- rnorm(n, ifelse(u == 1, mu1, mu2), ifelse(u == 1, sigmasq1, sigmasq2))

## calculate the posterior distribution
logposterior <- function(delta, mu1, mu2, sigmasq1, sigmasq2, x = d) {
  logL <- sum(log(delta * dnorm(x, mu1, sqrt(sigmasq1))+(1-delta)*dnorm(x, mu2, sqrt(sigmasq2)
  logprior.mu1 <- dnorm(mu1, 0, 10, log = T)
  logprior.mu2 <- dnorm(mu2, 0, 10, log = T)
  logprior.sigma1 <- dinvgamma(sigmasq1, 0.5, 10, log = T)
  logprior.sigma2 <- dinvgamma(sigmasq2, 0.5, 10, log = T)

  return(logL + logprior.mu1 + logprior.mu2 + logprior.sigma1 + logprior.sigma2)
}

mymcmc <- function (niter, thetaInit, x = d){
  p <- length(thetaInit)
  thetaCurrent <- thetaInit
  out <- matrix(NA, niter, p)
  for (i in 1:niter) {
      ## arms algorithm
      out[i, 1] <- thetaCurrent[1] <-
      HI::arms(thetaCurrent[1], logposterior,
              function(x, ...) ((x > 0) * (x < 1)), 1, mu1 = thetaCurrent[2], mu2 = thetaCurre

      out[i, 2] <- thetaCurrent[2] <-
      HI::arms(thetaCurrent[2], logposterior,
              function(x, ...) ((x > -100) * (x < 100)), 1, delta = thetaCurrent[1], mu2 = the

      out[i, 3] <- thetaCurrent[3] <-
      HI::arms(thetaCurrent[3], logposterior,
              function(x, ...) ((x > -100) * (x < 100)), 1, delta = thetaCurrent[1], mu1 = the

      out[i, 4] <- thetaCurrent[4] <-
      HI::arms(thetaCurrent[4], logposterior,
              function(x, ...) ((x > 0) * (x < 1000)), 1, delta = thetaCurrent[1], mu1 = theta


      out[i, 5] <- thetaCurrent[5] <-
      HI::arms(thetaCurrent[5], logposterior,
              function(x, ...) ((x > 0) * (x < 1000)), 1, delta = thetaCurrent[1], mu1 = theta
      ## delta = thetaCurrent[1], mu1 = thetaCurrent[2], mu2 = thetaCurrent[3], sigmasq1 = the
```

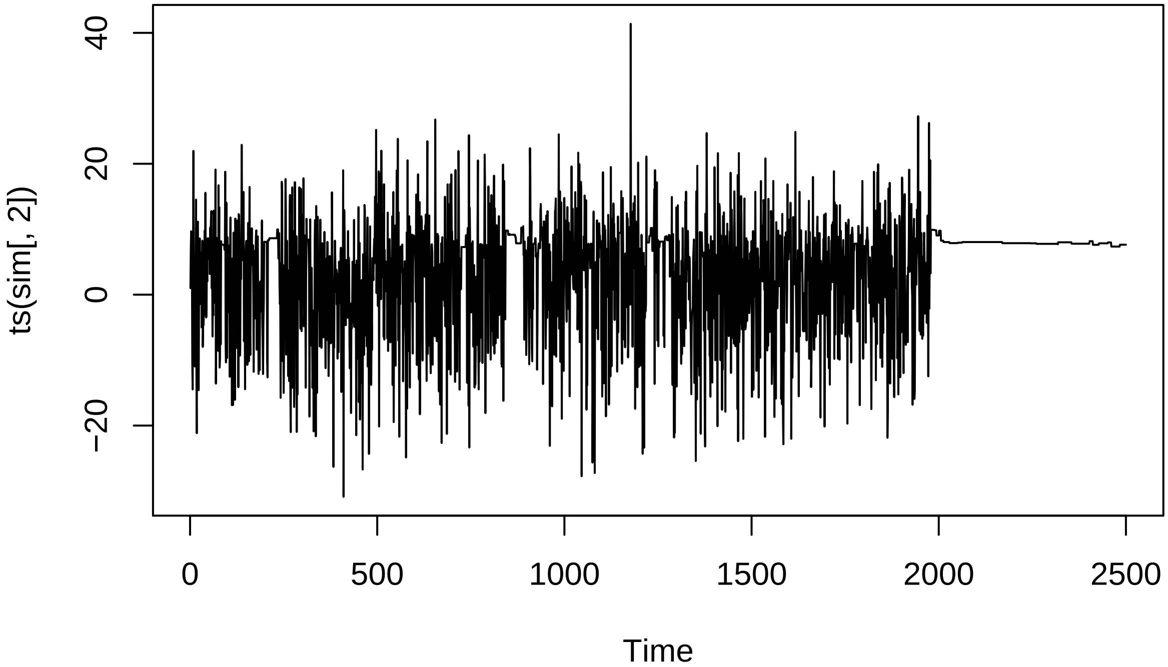
```r
  }
  out
}
niter <- 2500
thetaInit <- c(0.5, 1, 1, 10, 10)
sim <- mymcmc(niter, thetaInit, d)
plot(ts(sim[,1]))
```



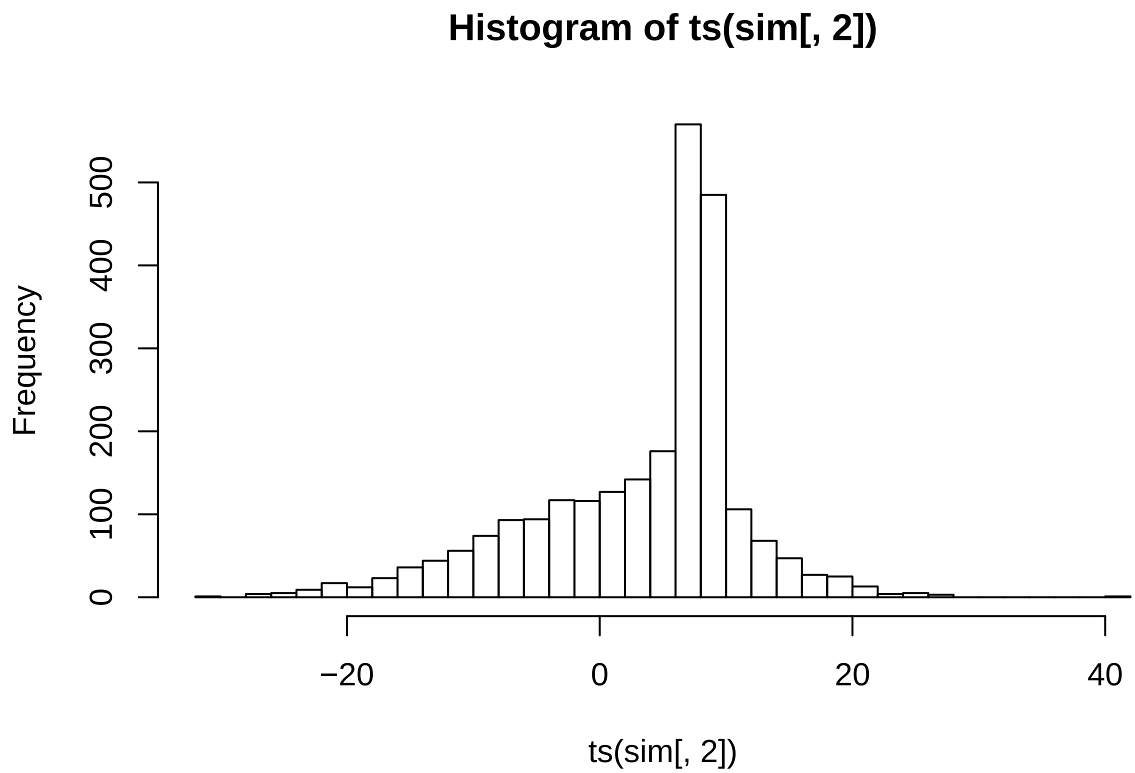

```r
hist(ts(sim[,1]), breaks = 50)
```




```r
plot(ts(sim[,2]))
```

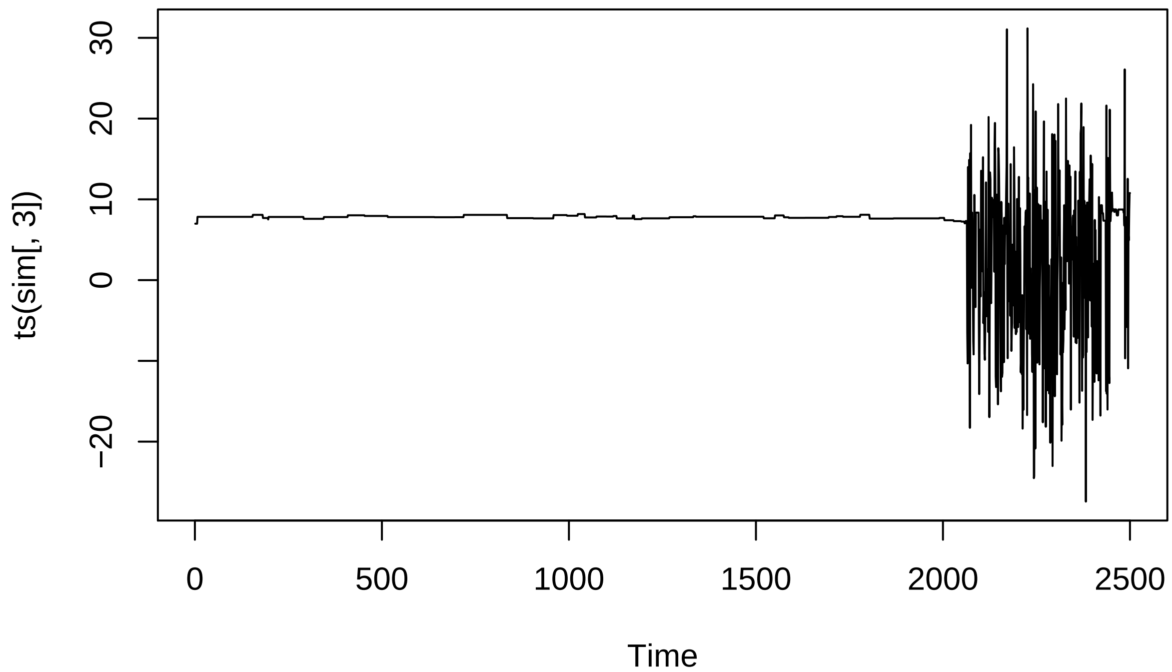



```r
hist(ts(sim[,2]), breaks = 50)
```

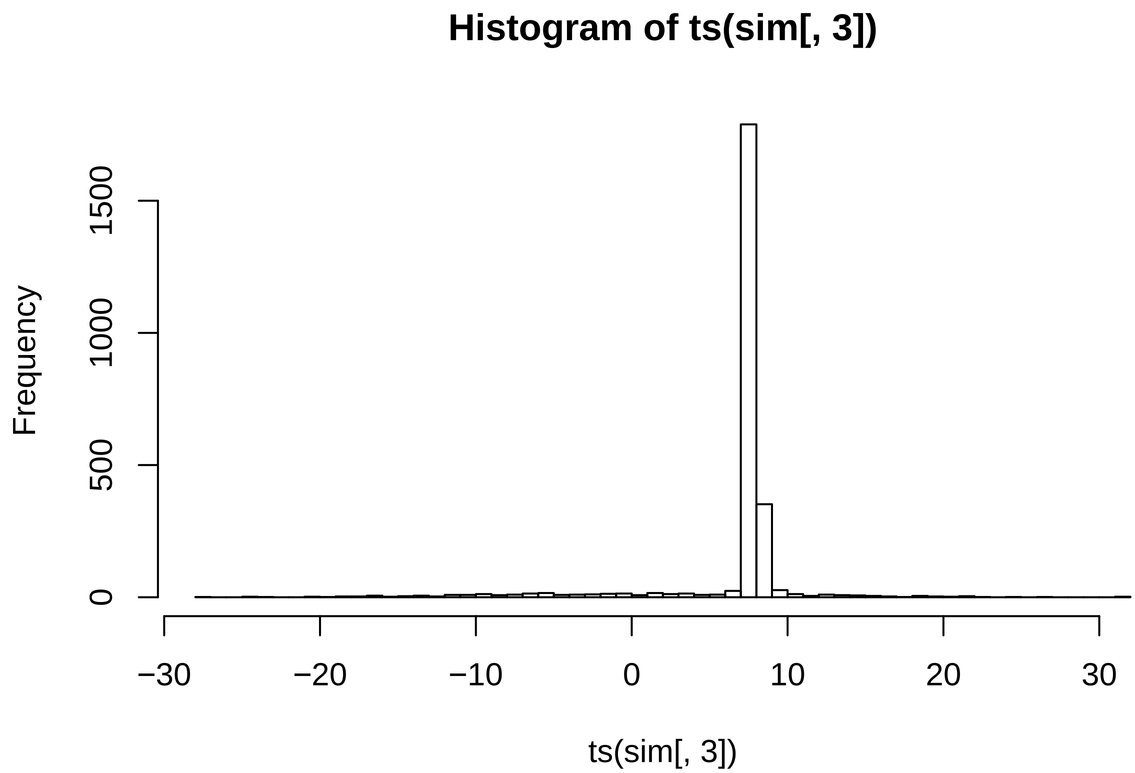


Histogram of ts(sim[, 2])
Frequency
0
100
200
300
400
500
−20
0
20
40
ts(sim[, 2])

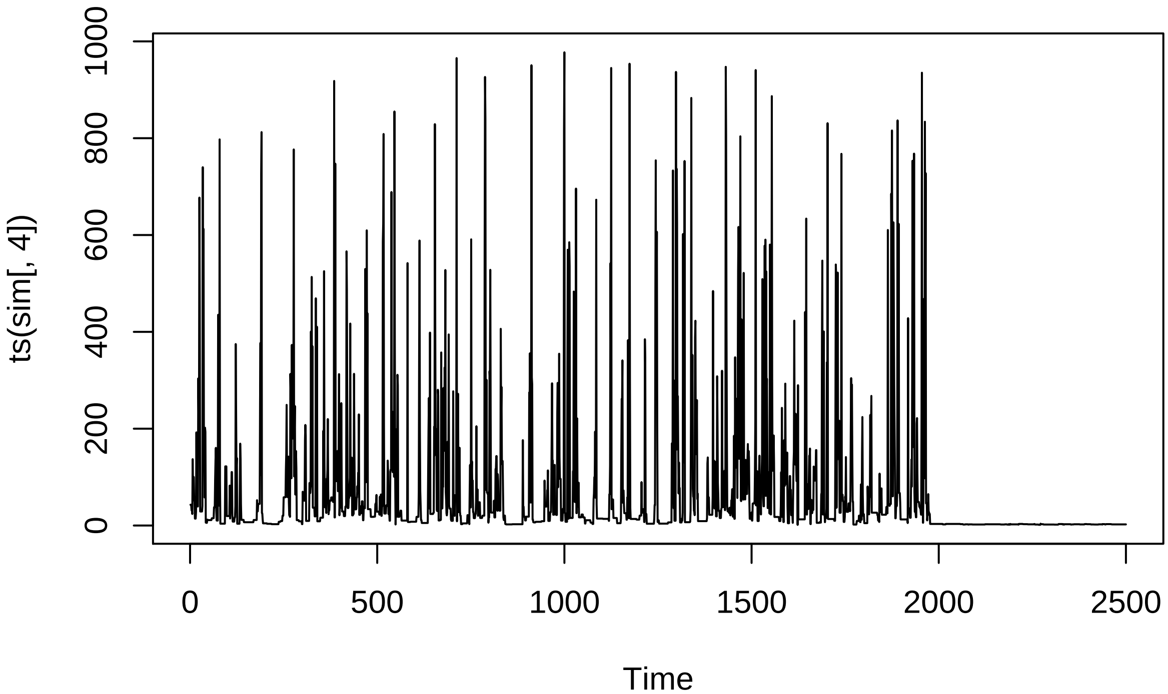

```
plot(ts(sim[,3]))
```



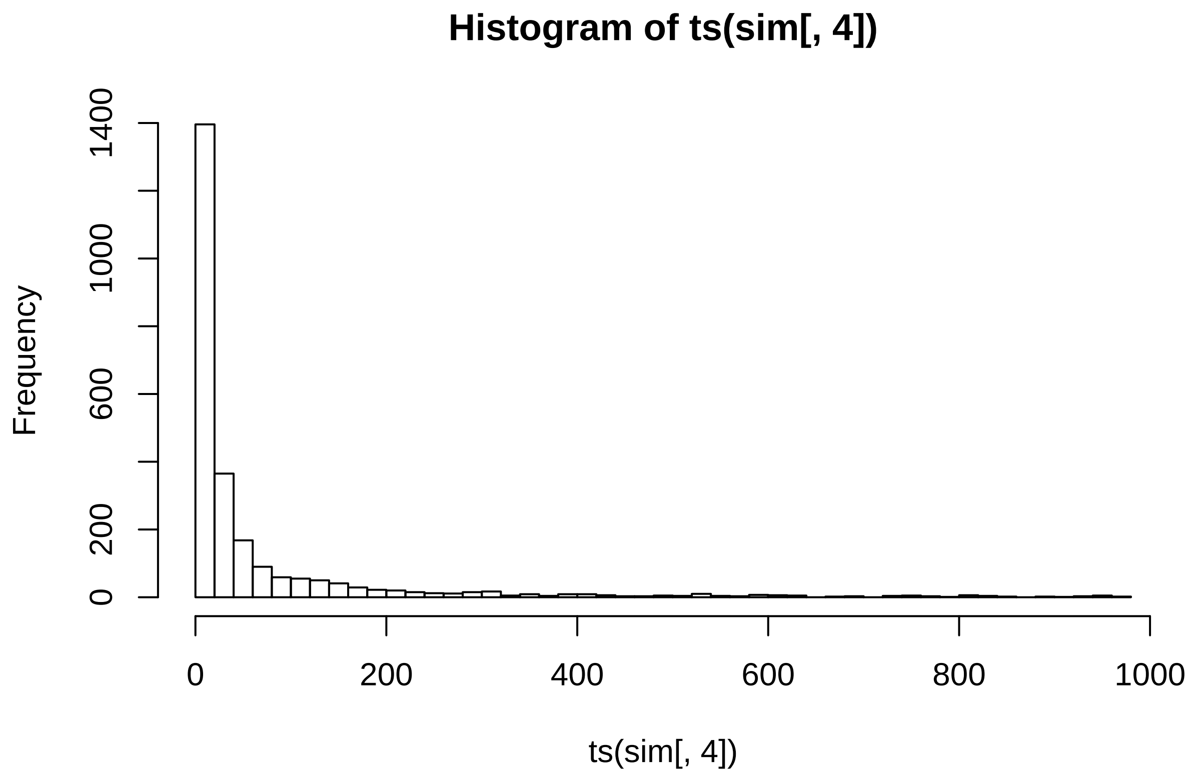

```r
hist(ts(sim[,3]), breaks = 50)
```


Histogram of ts(sim[, 3])
Frequency
0
500
1000
1500
−30
−20
−10
0
10
20
30
ts(sim[, 3])

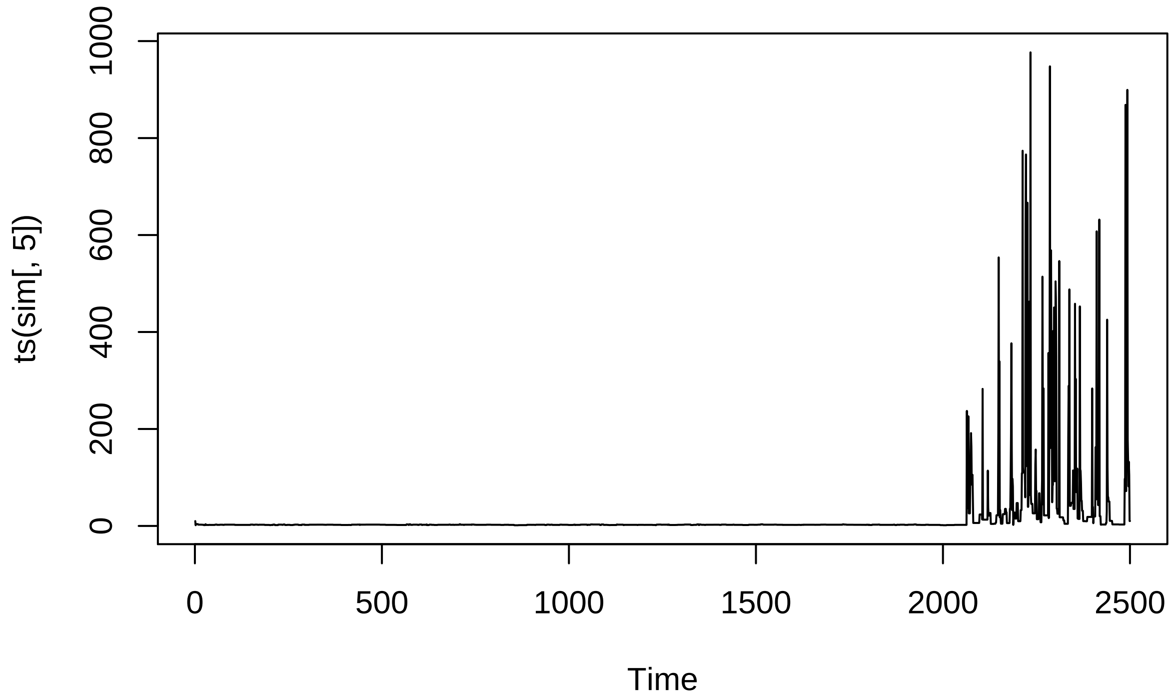
```
plot(ts(sim[,4]))
```



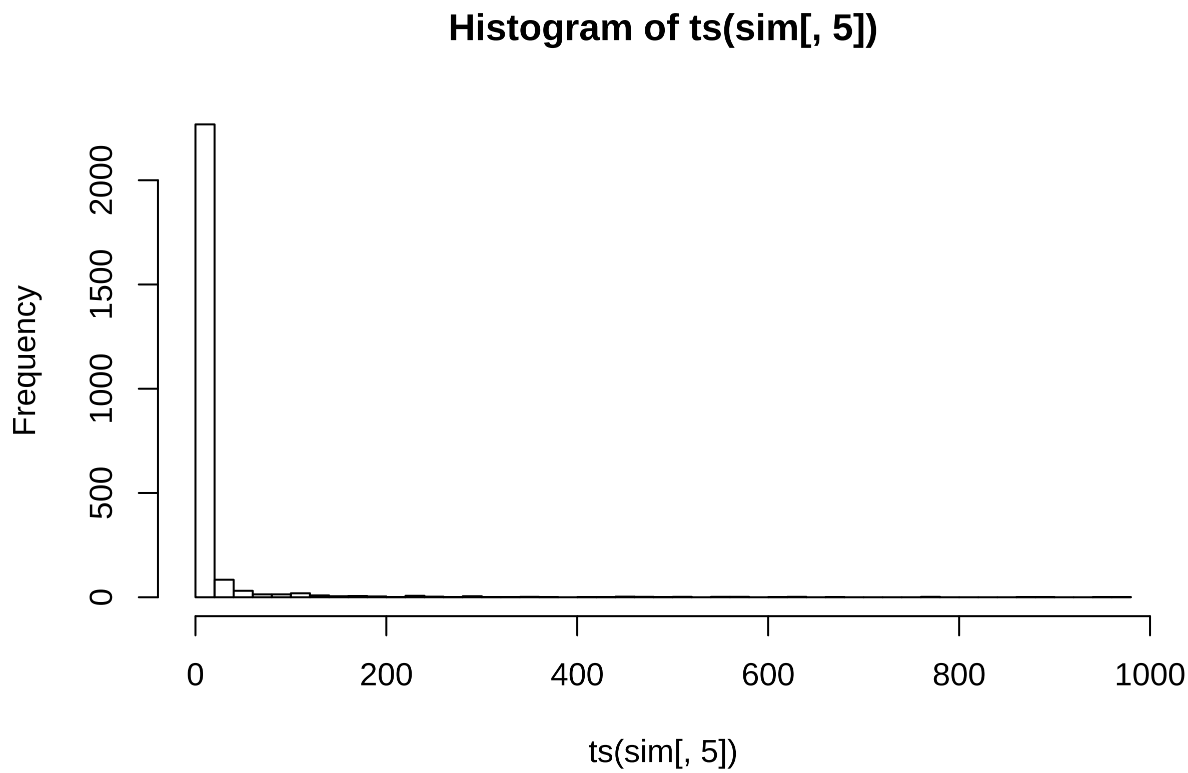

```
hist(ts(sim[,4]), breaks = 50)
```




```r
plot(ts(sim[,5]))
```

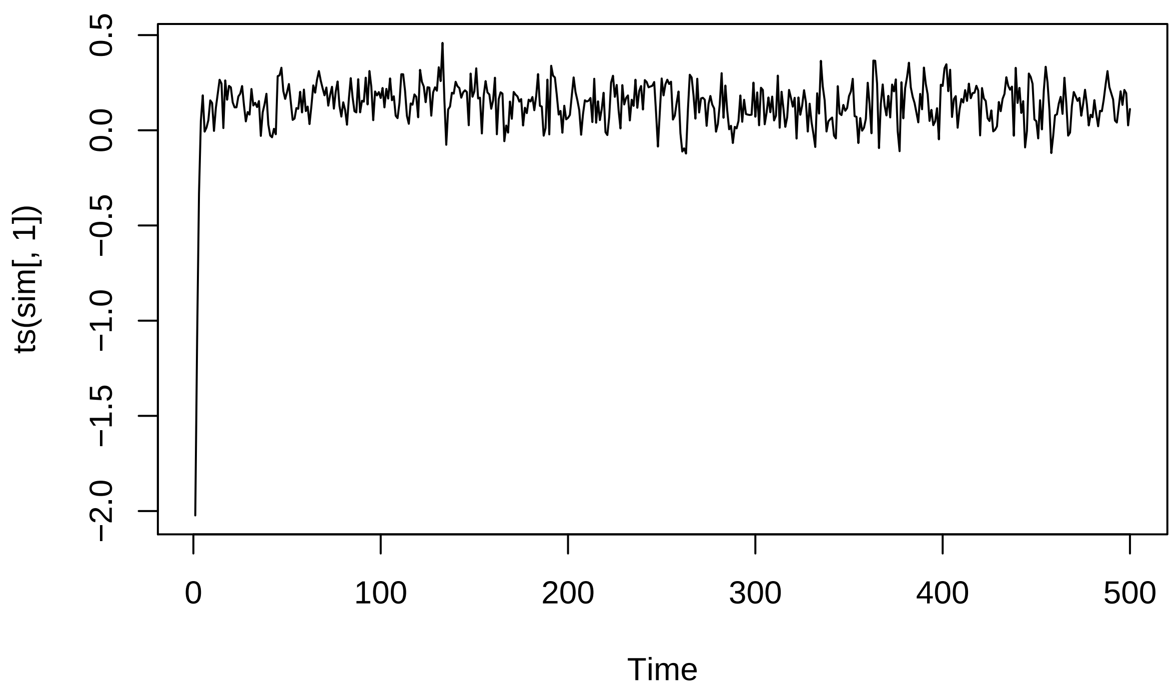



```r
hist(ts(sim[,5]), breaks = 50)
```

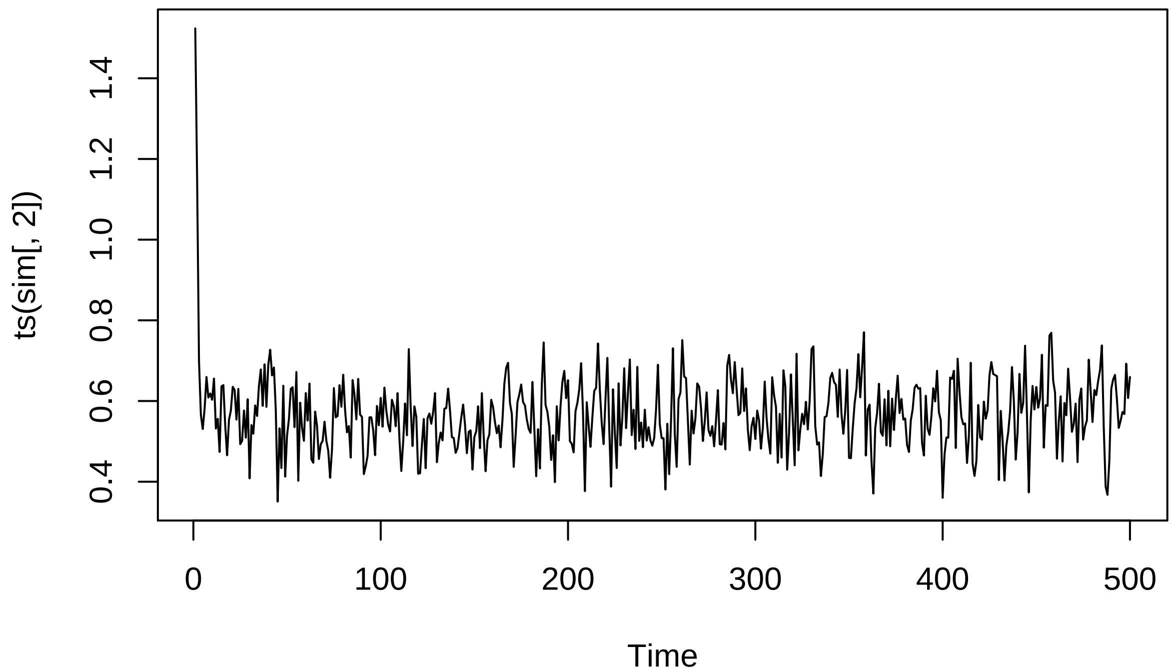



```
n <- 100
a <- 0.0; b <- 0.5
x <- rnorm(n)
y <- rpois(n, exp(a + b * x))
mydata <- data.frame(y = y, x = x)

logpost <- function(theta, data, sigma2, tau2) {
  a <- theta[1]; b <- theta[2]
  x <- data$x; y <- data$y
  return(a * sum(y) + b * sum(x * y) - exp(a) * sum(exp(b * x))
         - a^2 / 2 / sigma2 - b^2 / 2 / tau2)
}

mymcmc <- function(niter, thetaInit, data, sigma2, tau2) {
  p <- length(thetaInit)
  thetaCurrent <- thetaInit
  out <- matrix(NA, niter, p)
  for (i in 1:niter) {
    for (j in 1:p) {
      logFC <- function(thj) {
        theta <- thetaCurrent
        theta[j] <- thj
        logpost(theta, data, sigma2, tau2)
```

```
      }
      ## arms algorithm
      out[i, j] <- thetaCurrent[j] <-
        HI::arms(thetaCurrent[j], logFC,
                 function(x) ((x > -10) * (x < 10)), 1)
    }
  }
  out
}

niter <- 500
thetaInit <- c(2, 2)
sigma2 <- tau2 <- 100
sim <- mymcmc(niter, thetaInit, mydata, sigma2, tau2)
plot(ts(sim[,1]))
```

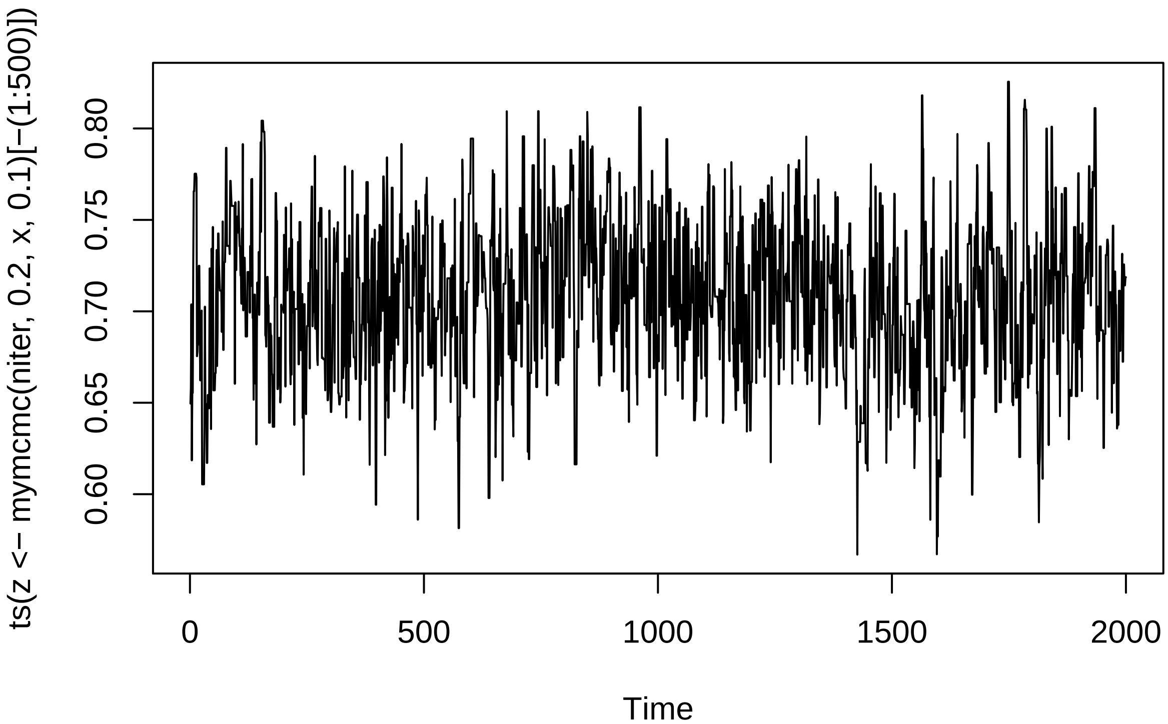



```
plot(ts(sim[,2]))
```

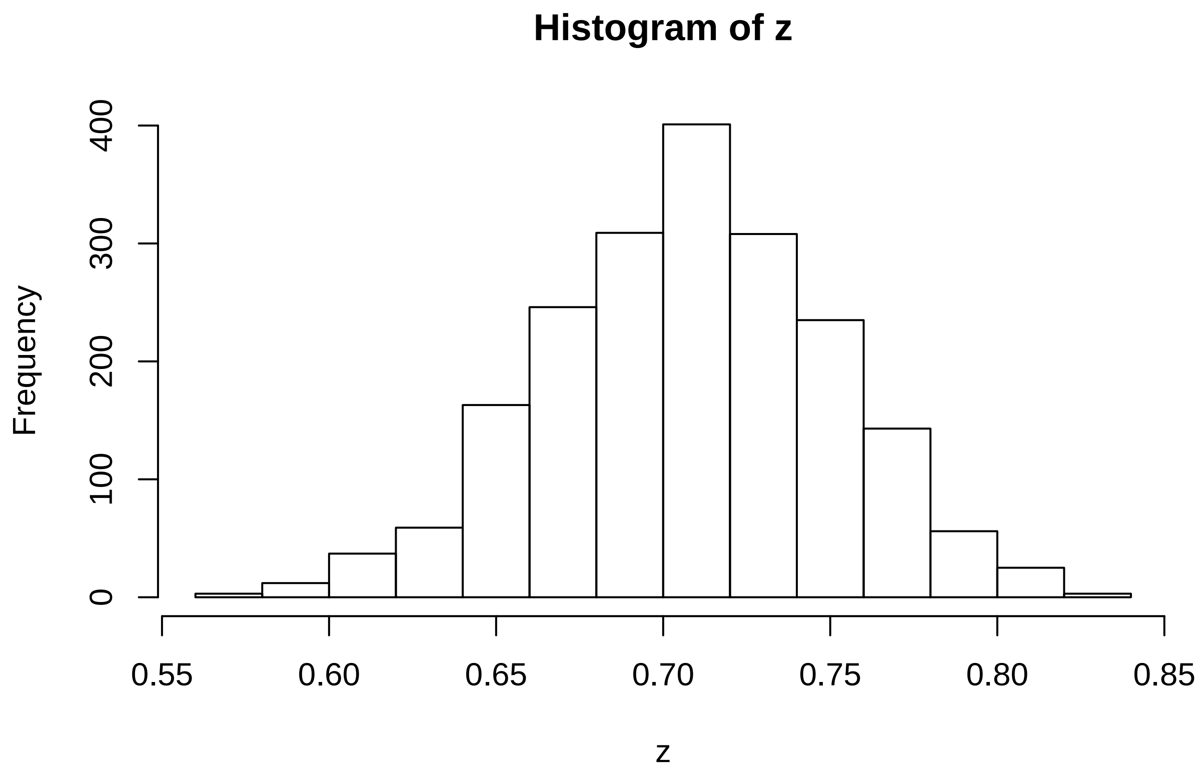



```
delta <- 0.7 # true value to be estimated based on the data
n <- 100
set.seed(123)
u <- rbinom(n, prob = delta, size = 1)
x <- rnorm(n, ifelse(u == 1, 7, 10), 0.5)

mylike <- function(delta, x) {
    prod(delta * dnorm(x, 7, 0.5) + (1 - delta) * dnorm(x, 10, 0.5))
}

## simple random walk chain
myRange <- function(v, width) {
    min(1, v + width) - max(0, v - width)
}

mymcmc <- function(niter, init, x, width) {
    v <- double(niter)
    for (i in 1:niter) {
        cand <- runif(1, max(0, init - width), min(1, init + width))
        ratio <- mylike(cand, x) / myRange(cand, width) /
            mylike(init, x) * myRange(init, width)
        if (runif(1) < min(ratio, 1)) {
            v[i] <- init <- cand
```

```
        } else v[i] <- init
    }
    v
}
niter <- 2500
plot(ts(z <- mymcmc(niter, .2, x, .1)[-(1:500)]))
```




```
hist(z)
```



Reference

[jun-yan/stat-5361]https://github.com/jun-yan/stat-5361