Assignment 7

YUYU ZHANG

10/24/2020

5.3.1 Rejection sampling

1. As for the normalizing constant for g, we have

To calculate the normalizing constant for g, we have

$$\begin{aligned}&\int_0^\infty (2x^{\theta-1} + x^{\theta-1/2})e^{-x}dx \\ &= \int_0^\infty 2x^{\theta-1}e^{-x}dx + \int_0^\infty x^{\theta-1/2}e^{-x}dx \\ &= 2\Gamma(\theta) + \Gamma(\theta+\frac{1}{2})\end{aligned}$$

Thus, we can get,

$$C = \frac{1}{2\Gamma(\theta) + \Gamma(\theta+\frac{1}{2})}$$

Then, the full expression of g will be,

$$g(x) = \frac{1}{2\Gamma(\theta) + \Gamma(\theta+\frac{1}{2})}(2x^{\theta-1} + x^{\theta-1/2})e^{-x}$$

which can also be expressed as,

$$g(x) = \frac{2\Gamma(\theta)}{2\Gamma(\theta) + \Gamma(\theta+\frac{1}{2})}\frac{x^{\theta-1}e^{-x}}{\Gamma(\theta)} + \frac{\Gamma(\theta+\frac{1}{2})}{2\Gamma(\theta) + \Gamma(\theta+\frac{1}{2})}\frac{x^{(\theta+\frac{1}{2})-1}e^{-x}}{\Gamma(\theta+\frac{1}{2})}$$

Thus we can have that g is a mixture of Gamma distributions. The component distribution is $\Gamma(\theta, 1)$ with weight $w1$, and $\Gamma(\theta+\frac{1}{2}, 1)$ with weight $w2$, which can be expressed as:

$$w1 = \frac{2\Gamma(\theta)}{2\Gamma(\theta) + \Gamma(\theta+\frac{1}{2})}, w2 = \frac{\Gamma(\theta+\frac{1}{2})}{2\Gamma(\theta) + \Gamma(\theta+\frac{1}{2})}$$

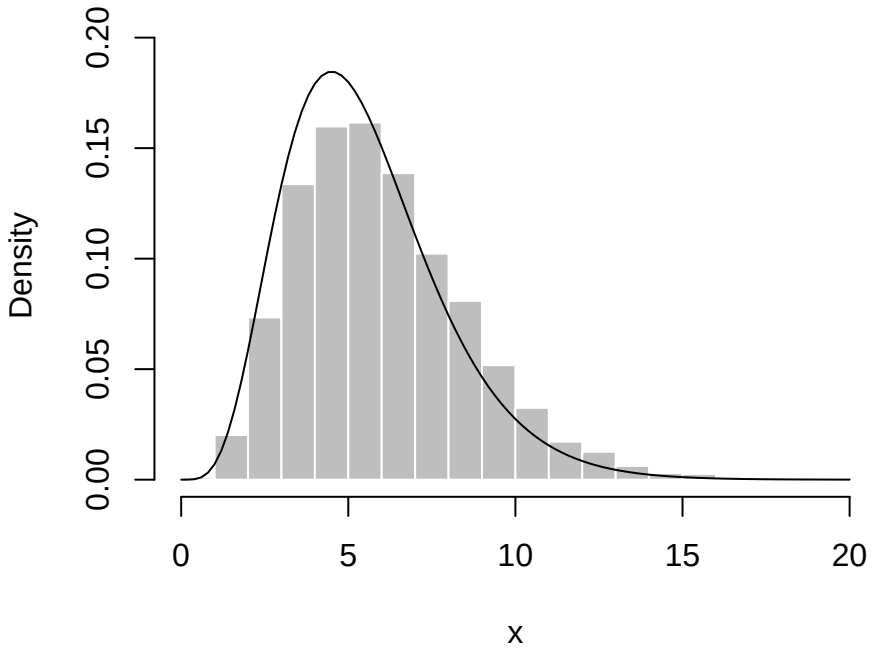
2.Design a procedure (pseudo-code) to sample from g; implement it in an R function; draw a sample of size n=10,000 using your function for at least one
θ value; plot the kernel density estimation of g from your sample and the true density in one figure

```
rgamma.ar <- function(n, shape, scale=1) {
s <- shape   #theta
s.int <- floor(s)
b <- s / s.int
M <- gamma(s.int) / gamma(s) * b**s.int * (s * exp(-1))**(s - s.int) *2
f <- function(y) dgamma(y, shape=s)
```

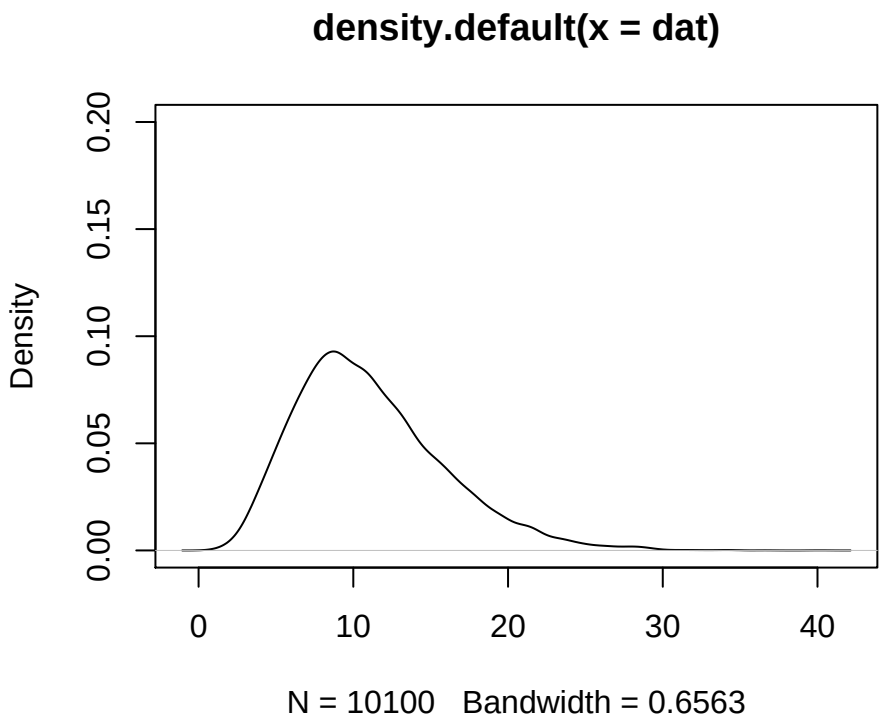
```r
Mg <- function(y) M * dgamma(y, shape=s.int, rate=1 )*2
acpt <- 0
total <- 0

X <- numeric(n)
while(acpt < n) {

  total <- total + 1
  Y <- sum(-b * log(runif(s.int)))
  if(runif(1) <= f(Y) / Mg(Y)) {
    acpt <- acpt + 1
    X[acpt] <- Y
}
}
return(list(X=scale * X, rate.true= 1/M, rate.obs=acpt / total))
}
o <- rgamma.ar(10000, shape=5.5)
print(o[-1])
```

```
## $rate.true
## [1] 0.4760121
##
## $rate.obs
## [1] 0.249663
```

```r
ylim <- c(0, 1.05 * dgamma(4.5, shape=5.5))
hist(o$X, freq=FALSE, col="gray", border="white", xlab="x", ylim=ylim, main="")
curve(dgamma(x, shape=5.5), add=TRUE)
```



```r
set.seed(10)
dat<-c(rgamma(10000,shape=5.5,scale=2),rgamma(100,shape=10,scale=2))
plot(density(dat),ylim=c(0,0.2))
```



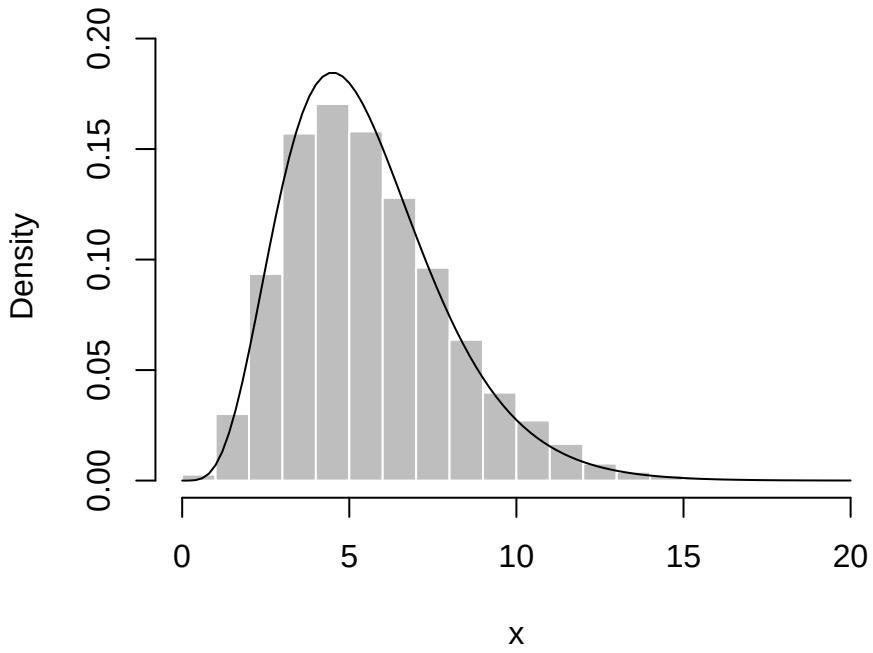
5.3.1(2)

```r
rgamma.ar <- function(n, shape, scale=1) {
s <- shape   #theta
s.int <- floor(s)
b <- s / s.int
M <- gamma(s.int) / gamma(s) * b**s.int * (s * exp(-1))**(s - s.int) *2
f <- function(y) dgamma((y+4)^0.5, shape=s)
Mg <- function(y) M * dgamma((y+4)^0.5, shape=s.int, rate=1 )*2
acpt <- 0
total <- 0

X <- numeric(n)
while(acpt < n) {

  total <- total + 1
  Y <- sum(-b * log(runif(s.int)))
  if(runif(1) <= f(Y) / Mg(Y)) {
    acpt <- acpt + 1
    X[acpt] <- Y
}
}
return(list(X=scale * X, rate.true= 1/M, rate.obs=acpt / total))
}
```

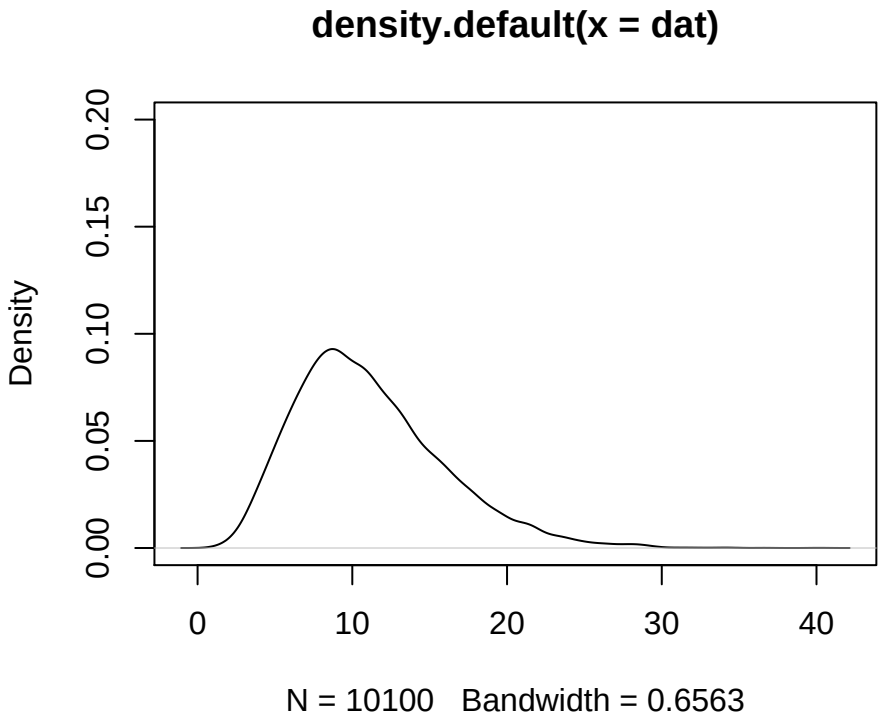
```r
o <- rgamma.ar(10000, shape=5.5)
print(o[-1])
```

```
## $rate.true
## [1] 0.4760121
##
## $rate.obs
## [1] 0.1918649
```

```r
ylim <- c(0, 1.05 * dgamma(4.5, shape=5.5))
hist(o$X, freq=FALSE, col="gray", border="white", xlab="x", ylim=ylim, main="")
curve(dgamma(x, shape=5.5), add=TRUE)
```

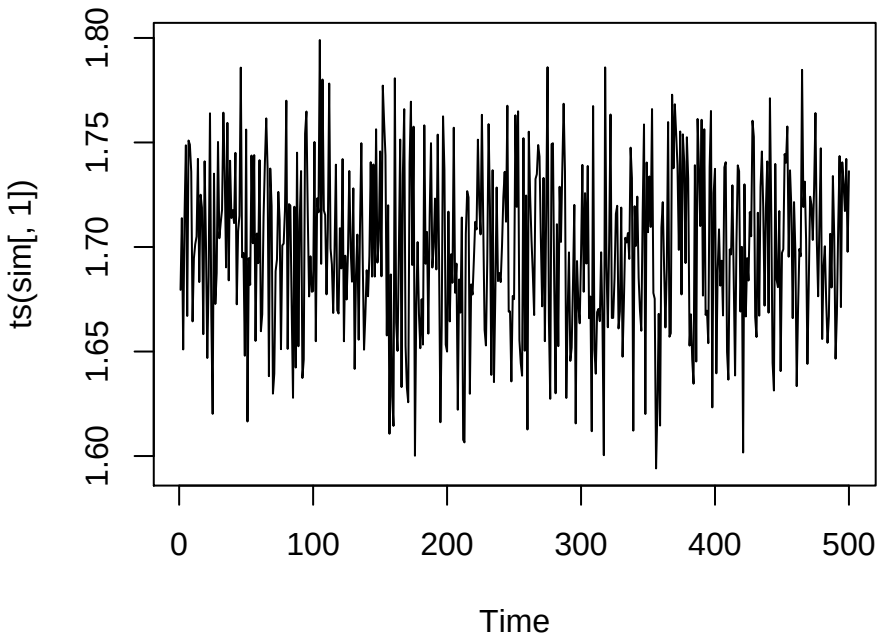



```r
set.seed(10)
dat<-c(rgamma(10000,shape=5.5,scale=2),rgamma(100,shape=10,scale=2))
plot(density(dat),ylim=c(0,0.2))
```



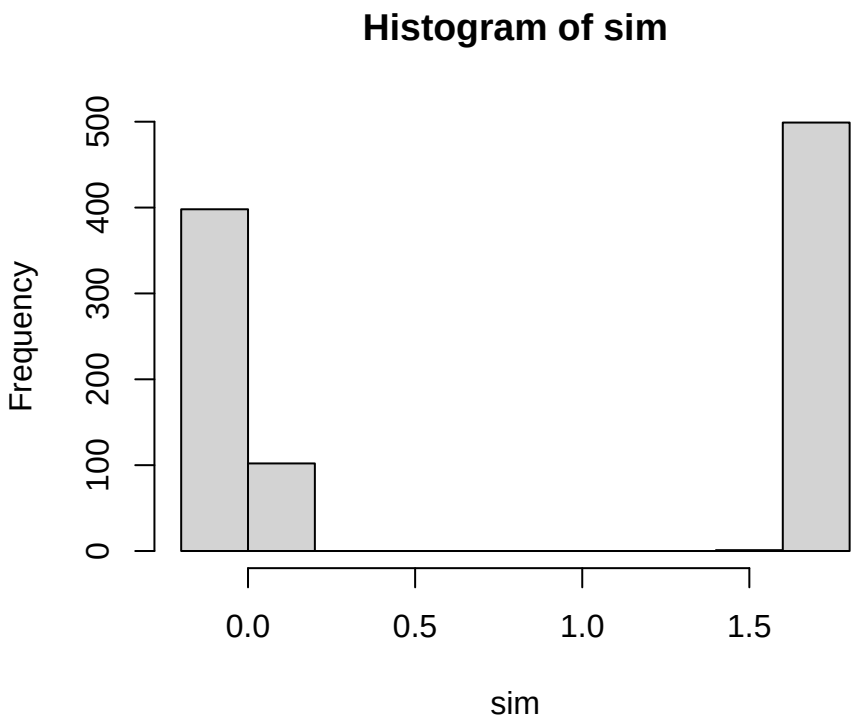

```
n <- 100
a <- 0.0; b <- 0.5
x <- rnorm(n)
y <- rgamma(n, shape=.5, scale=10)
mydata <- data.frame(y = y, x = x)


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, nburn= 100) {
  p <- length(thetaInit)
  thetaCurrent <- thetaInit
  ## define a function for full conditional sampling
  logFC <- function(th, idx) {
    theta <- thetaCurrent
    theta[idx] <- th
    logpost(theta, data, sigma2, tau2)
  }
  out <- matrix(thetaInit, niter, p, byrow = TRUE)
  ## Gibbs sampling
```

```
  for (i in 2:niter) {
    for (j in 1:p) {
      ## general-purpose arms algorithm
      out[i, j] <- thetaCurrent[j] <-
          HI::arms(thetaCurrent[j], logFC,
                   function(x, idx) ((x > -10) * (x < 10)),
                   1, idx = j)
    }
  }
  out[-(1:nburn), ]
}

niter <- 600; nburn <- 100
thetaInit <- c(2, 2)
sigma2 <- tau2 <- 100
sim <- mymcmc(niter, thetaInit, mydata, sigma2, tau2)
plot(ts(sim[,1]))
```



6.3.1 Normal mixture revisited

```
hist(sim)
```



The parameter of the μ_1 and μ_2 and the parameter of the σ_1 and σ_2 are estimated usisng An MCMC based the Gibbs sampler from R package HI.