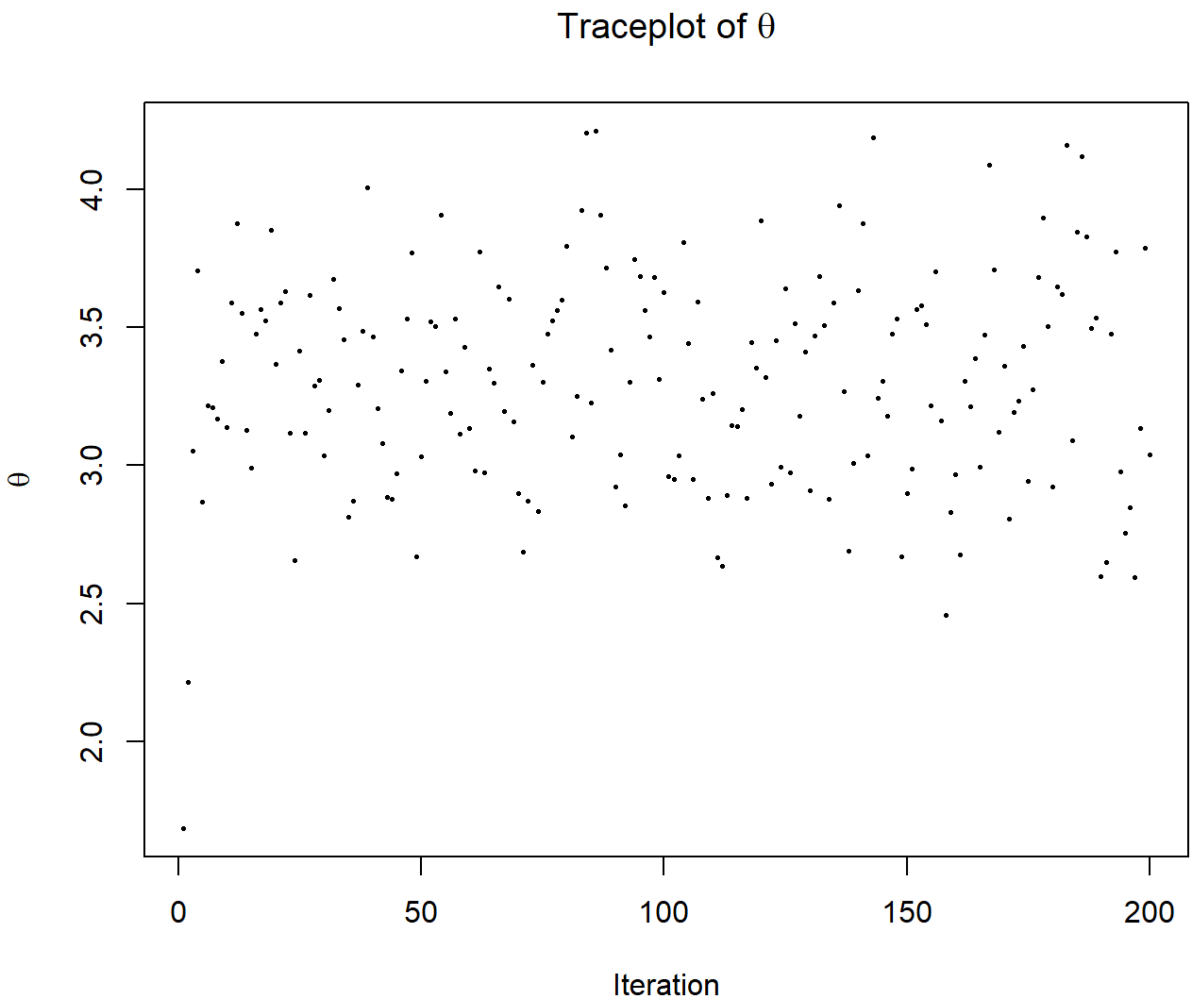
Dial-Jackson-homework6

Rebecca C. Steorts, Jackson Dial

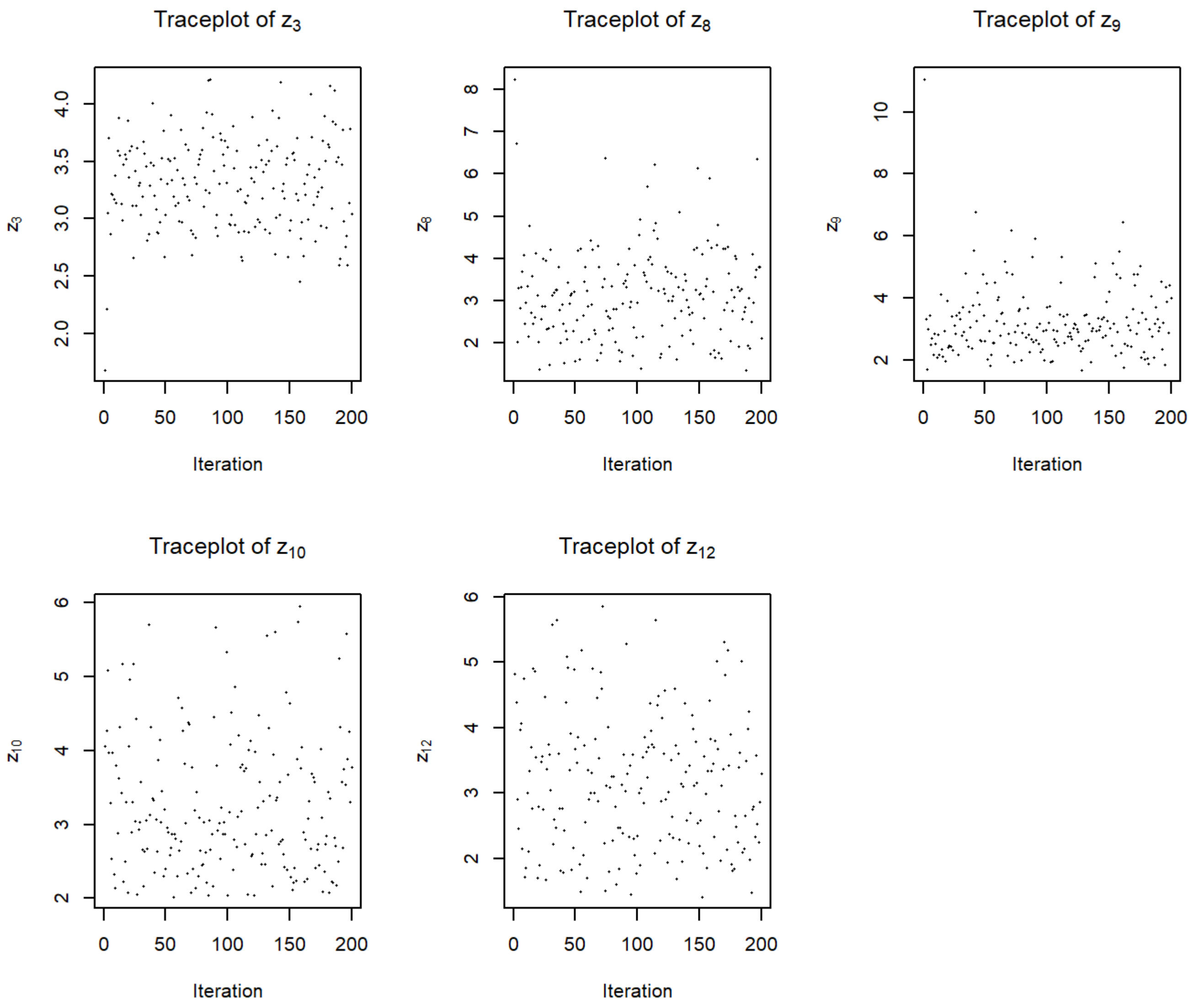
A

```
knitr::opts_chunk$set(cache=FALSE)
library(xtable)

# Samples from a truncated gamma with
# truncation (t, infty), shape a, and rate b
# Input: t,a,b
# Output: truncated Gamma(a,b)
sampleTrunGamma <- function(t, a, b){
  # This function samples from a truncated gamma with
  # truncation (t, infty), shape a, and rate b
  p0 <- pgamma(t, shape = a, rate = b)
  x <- runif(1, min = p0, max = 1)
  y <- qgamma(x, shape = a, rate = b)
  return(y)
}

# Gibbs sampler for censored data
# Inputs:
  # this function is a Gibbs sampler
  # z is the fully observe data
  # c is censored data
  # n.iter is number of iterations
  # init.theta and init.miss are initial values for sampler
  # r,a, and b are parameters
  # burnin is number of iterations to use as burnin
# Output: theta, z
sampleGibbs <- function(z, c, n.iter, init.theta, init.miss, r, a, b, burnin = 1){

  z.sum <- sum(z)
  m <- length(c)
  n <- length(z) + m
  miss.vals <- init.miss
  res <- matrix(NA, nrow = n.iter, ncol = 1 + m)
  for (i in 1:n.iter){
    var.sum <- z.sum + sum(miss.vals)
    theta <- rgamma(1, shape = a + n*r, rate = b + var.sum)
    miss.vals <- sapply(c, function(x) {sampleTrunGamma(x, r, theta)})
    res[i,] <- c(theta, miss.vals)
  }
  return(res[burnin:n.iter,])
}

# set parameter values
r <- 10
a <- 1
b <- 1
# input data
z <- c(3.4,2.9,1.4,3.2,1.8,4.6,2.8)
c <- c(1.2,1.7,2.0,1.4,0.6)
```

```
n.iter2 <- 200
init.theta <- 1
init.missing <- rgamma(length(c), shape = r, rate = init.theta)
# run sampler
res <- sampleGibbs(z, c, n.iter2, init.theta, init.missing, r, a, b)
```

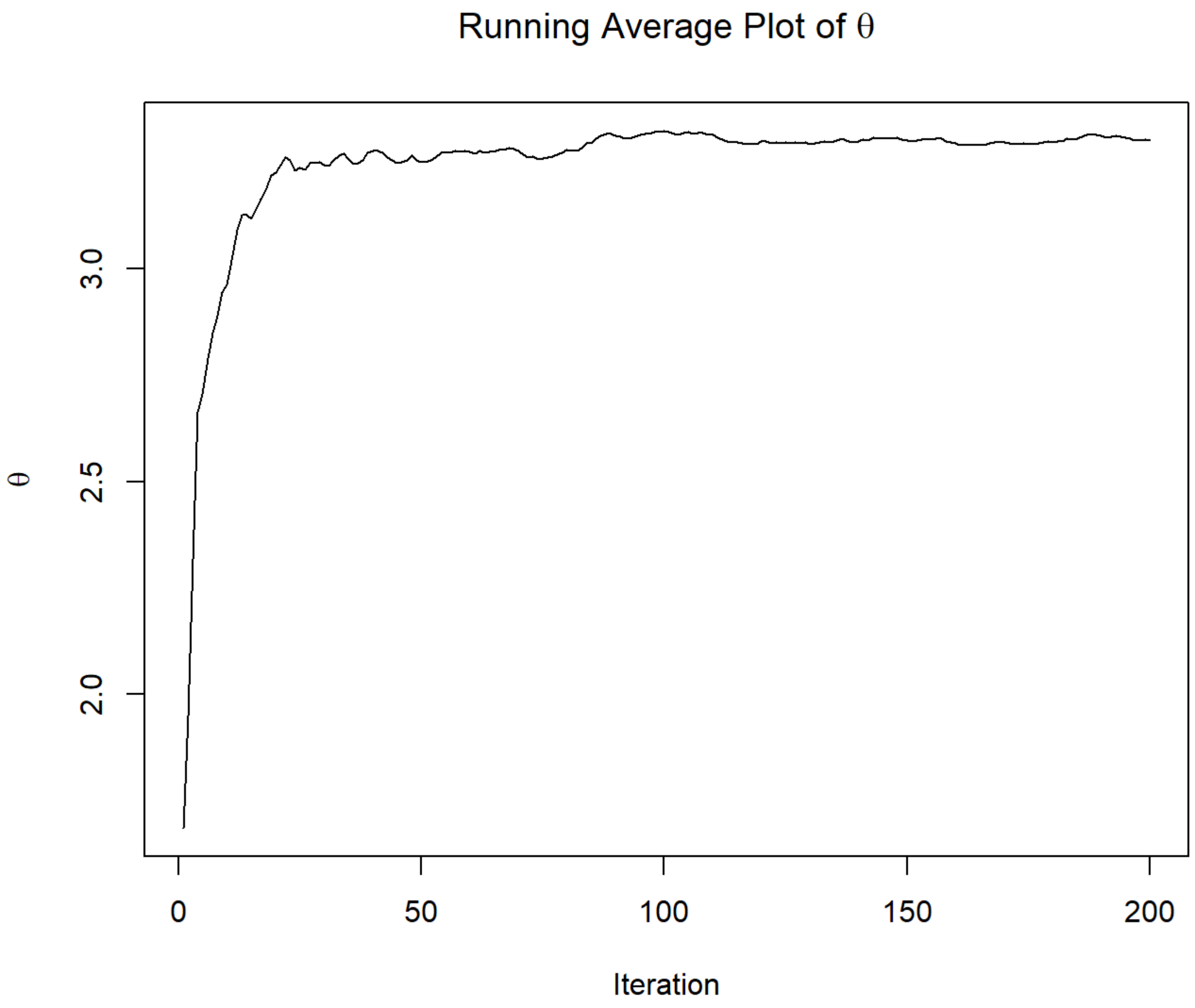


Traceplot of theta

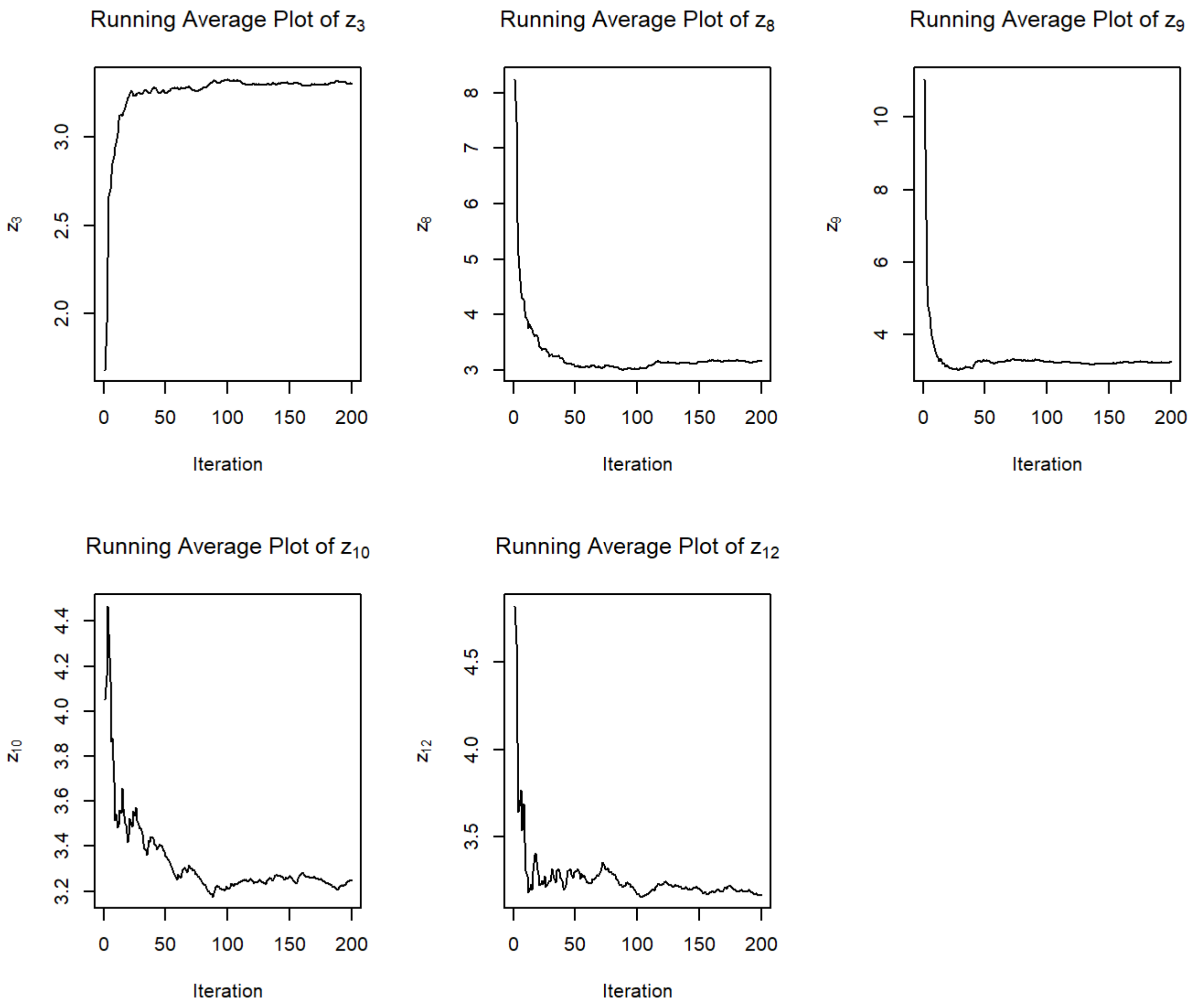


Traceplot of $z_3, z_8, z_9, z_{10}, z_{12}$.

```
# get running averages
run.avg <- apply(res, 2, cumsum)/(1:n.iter2)
```

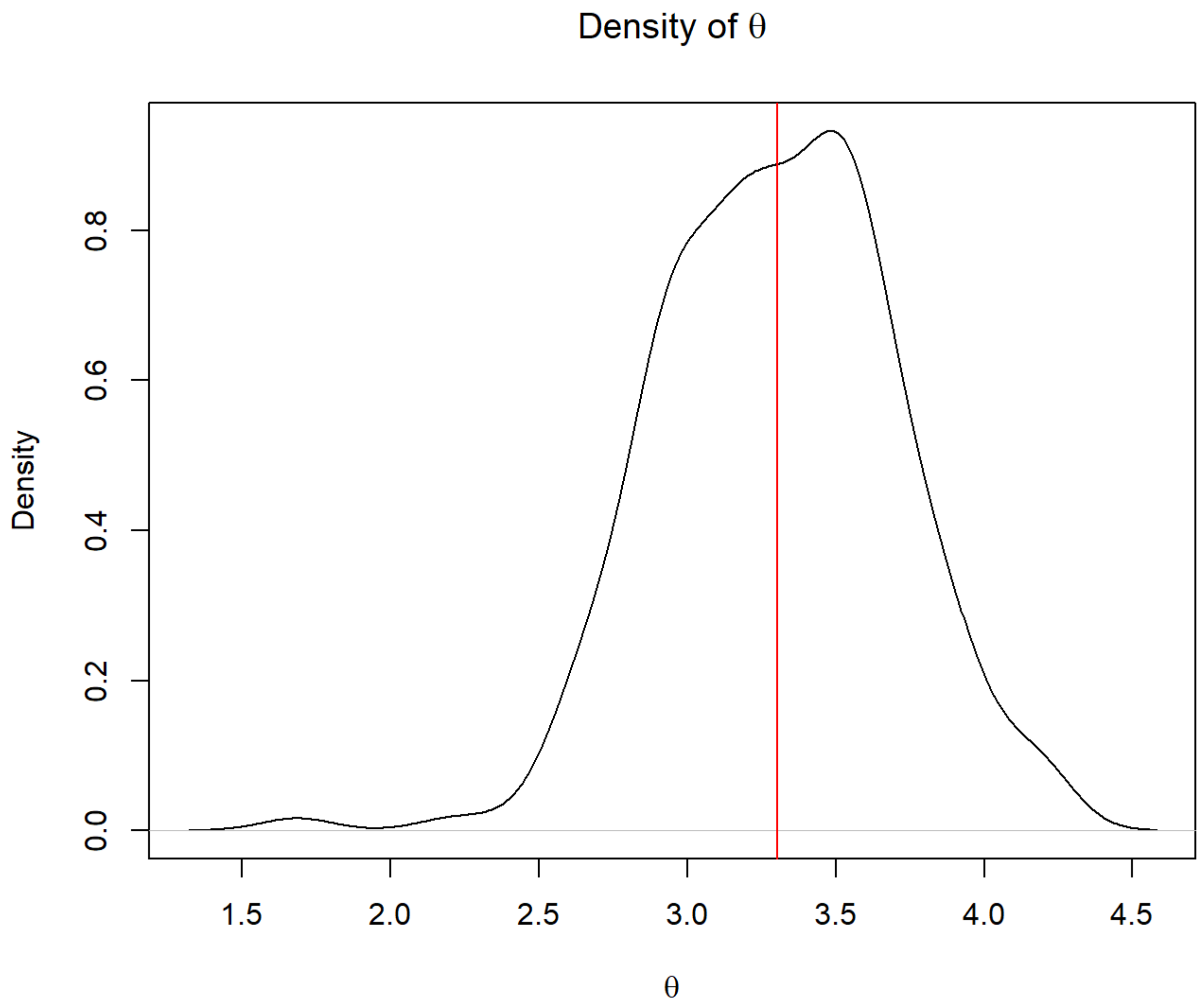


Running average plot of theta

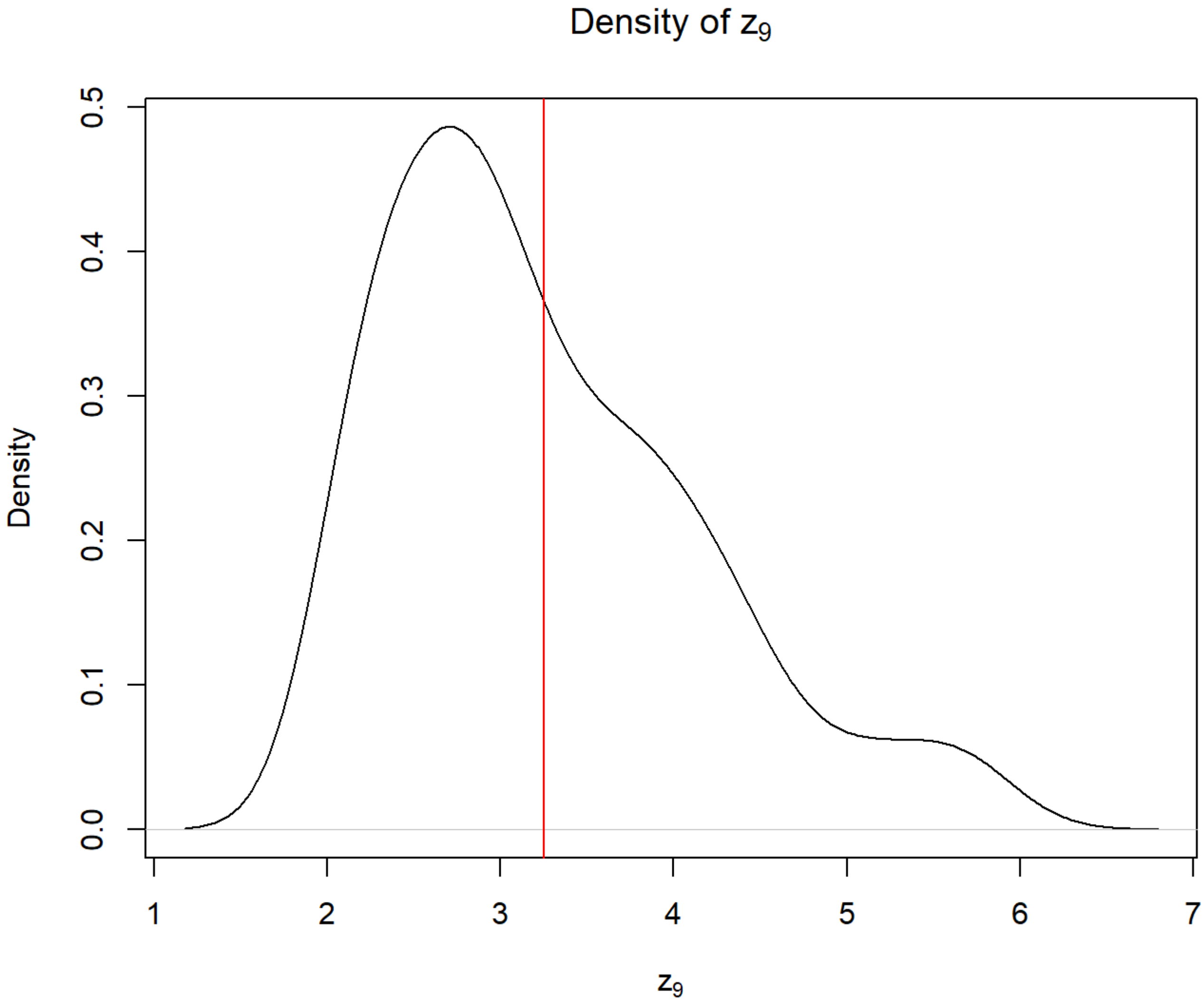


Running average plots of $z_3, z_8, z_9, z_{10}, z_{12}$.

Figures ??? and ??? do not provide meaniful inference at this point since the sampler has not been run long enough.



Estimated posterior density of theta



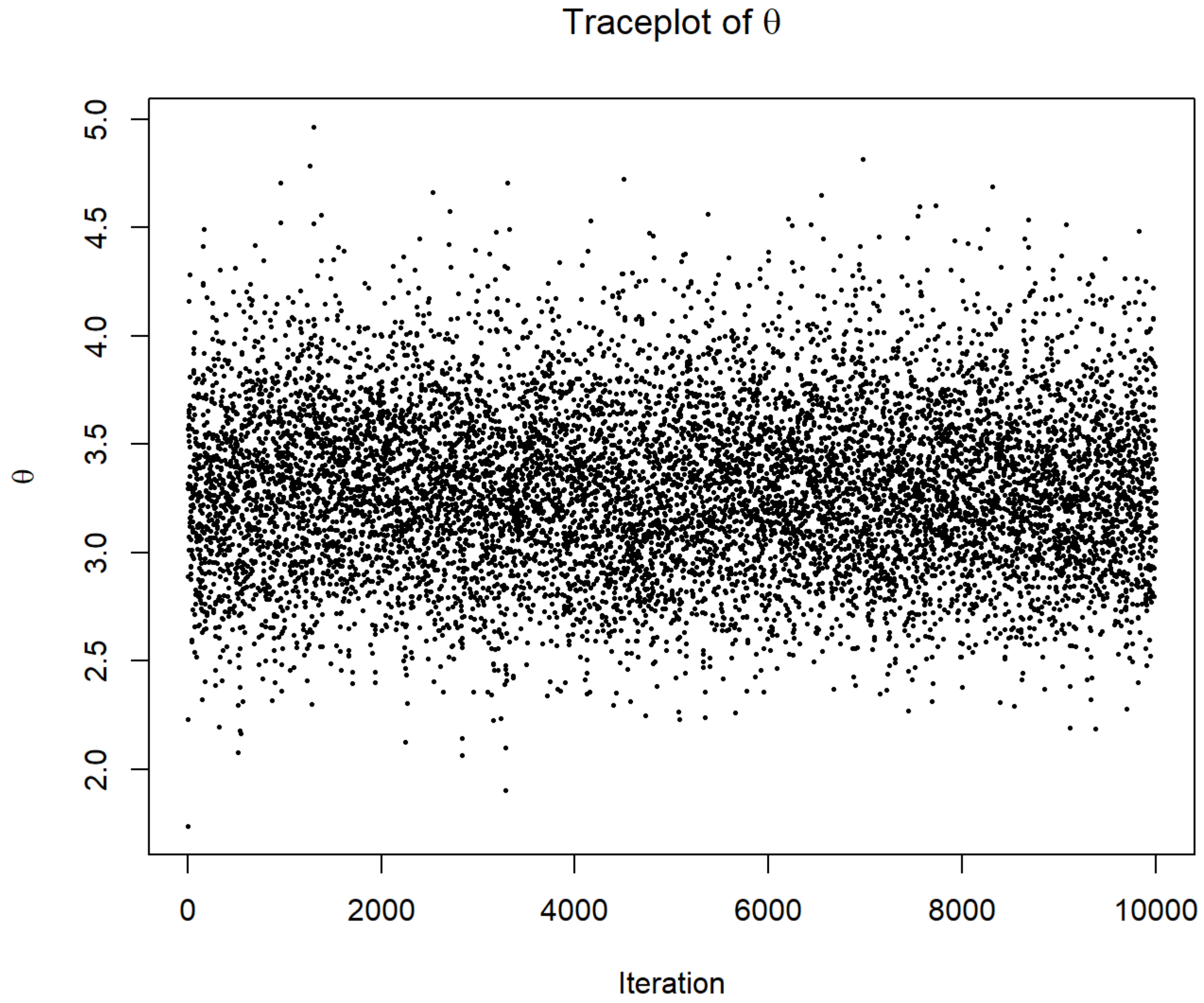
Estimated posterior density of z_9 (posterior mean in red).

These diagnostic plots do not indicate that we have run the sampler for long enough. The plot does not seem to converge.

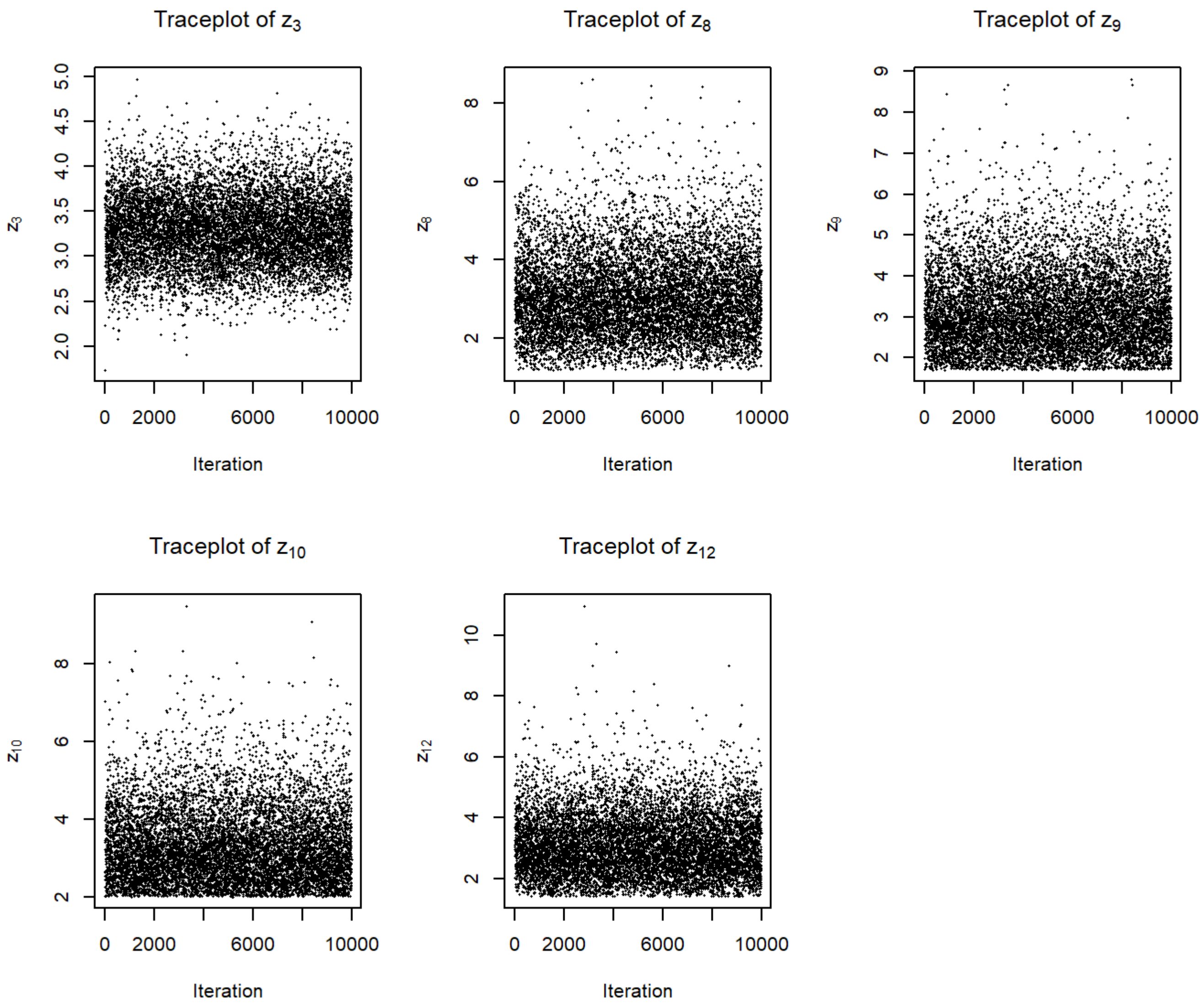
B

```
# knitr::opts_chunk$set(cache=FALSE)
# library(xtable)
#
# # Samples from a truncated gamma with
# # truncation (t, infty), shape a, and rate b
# # Input: t,a,b
# # Output: truncated Gamma(a,b)
# sampleTrunGamma <- function(t, a, b){
#   # This function samples from a truncated gamma with
#   # truncation (t, infty), shape a, and rate b
#   p0 <- pgamma(t, shape = a, rate = b)
#   x <- runif(1, min = p0, max = 1)
#   y <- qgamma(x, shape = a, rate = b)
#   return(y)
# }
#
# # Gibbs sampler for censored data
# # Inputs:
#   # this function is a Gibbs sampler
#   # z is the fully observe data
#   # c is censored data
#   # n.iter is number of iterations
#   # init.theta and init.miss are initial values for sampler
#   # r,a, and b are parameters
#   # burnin is number of iterations to use as burnin
# # Output: theta, z
# sampleGibbs <- function(z, c, n.iter, init.theta, init.miss, r, a, b, burnin = 1){
#
#   z.sum <- sum(z)
#   m <- length(c)
#   n <- length(z) + m
#   miss.vals <- init.miss
#   res <- matrix(NA, nrow = n.iter, ncol = 1 + m)
#   for (i in 1:n.iter){
#     var.sum <- z.sum + sum(miss.vals)
#     theta <- rgamma(1, shape = a + n*r, rate = b + var.sum)
#     miss.vals <- sapply(c, function(x) {sampleTrunGamma(x, r, theta)})
#     res[i,] <- c(theta, miss.vals)
#   }
#   return(res[burnin:n.iter,])
# }
#
# # set parameter values
# r <- 10
# a <- 1
# b <- 1
# # input data
# z <- c(3.4,2.9,1.4,3.2,1.8,4.6,2.8)
# c <- c(1.2,1.7,2.0,1.4,0.6)
```

```
n.iter <- 10000
init.theta <- 1
init.missing <- rgamma(length(c), shape = r, rate = init.theta)
# run sampler
res <- sampleGibbs(z, c, n.iter, init.theta, init.missing, r, a, b)
```

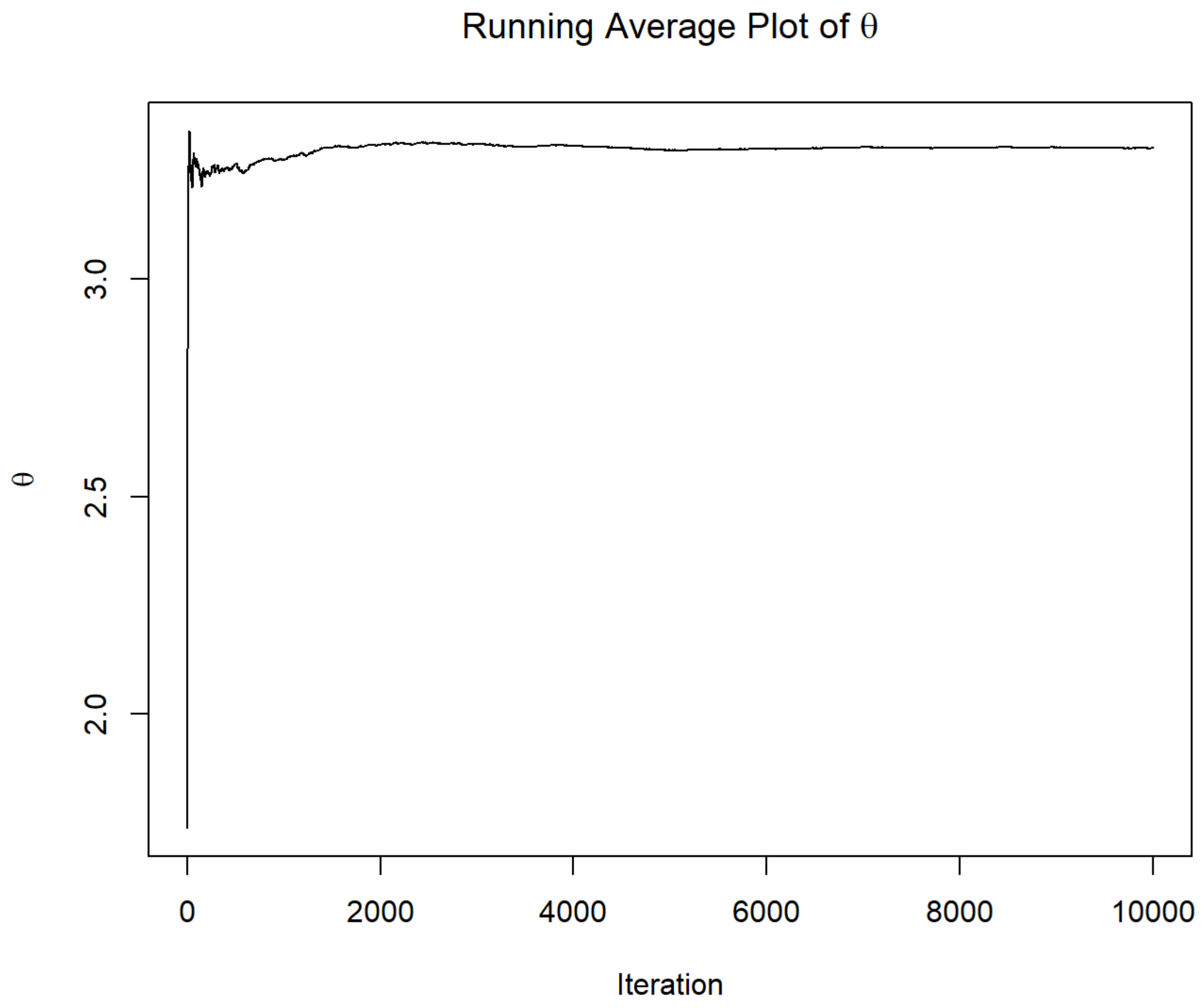


Traceplot of theta

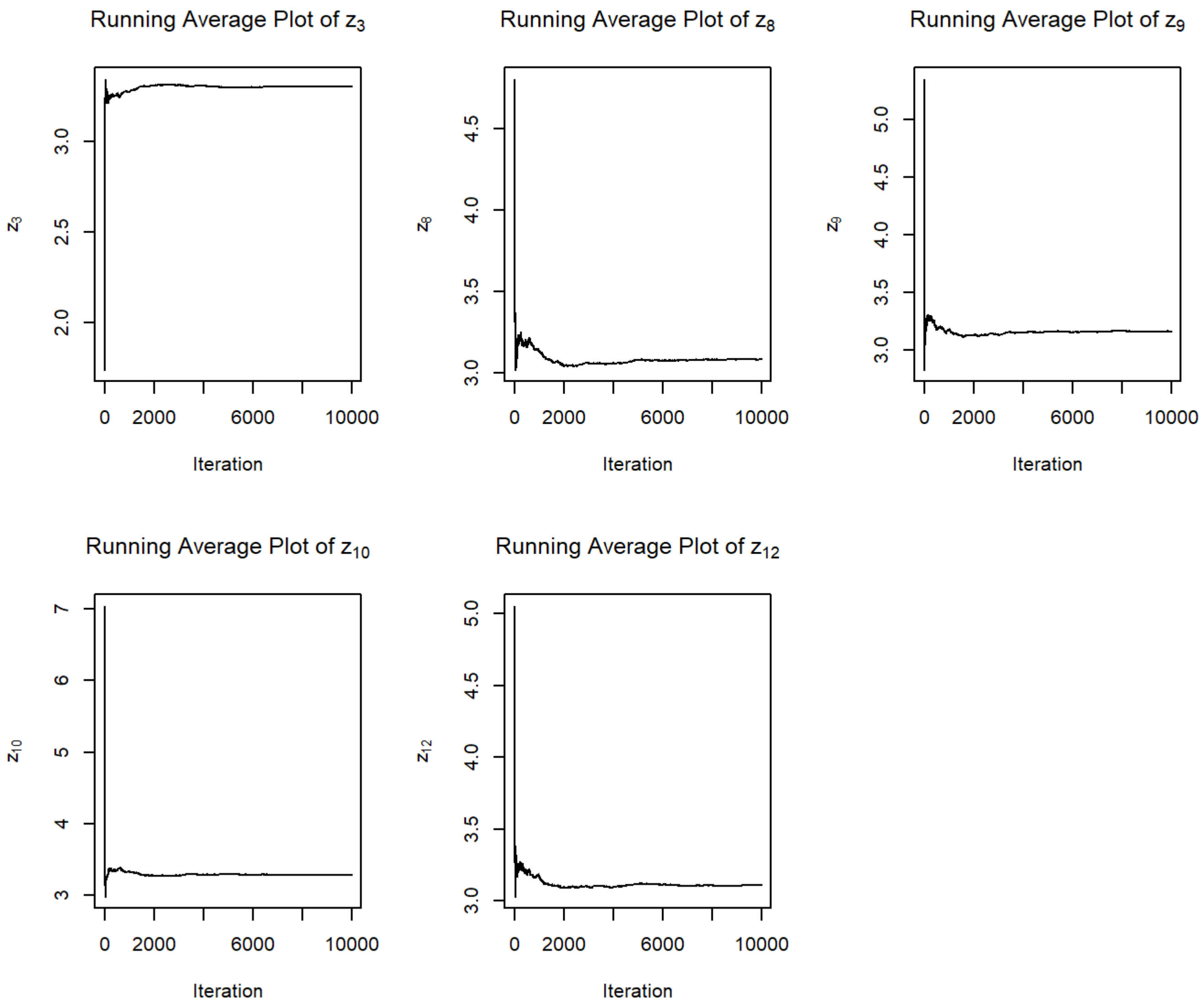


Traceplot of $z_3, z_8, z_9, z_{10}, z_{12}$.

```
# get running averages
run.avg <- apply(res, 2, cumsum)/(1:n.iter)
```



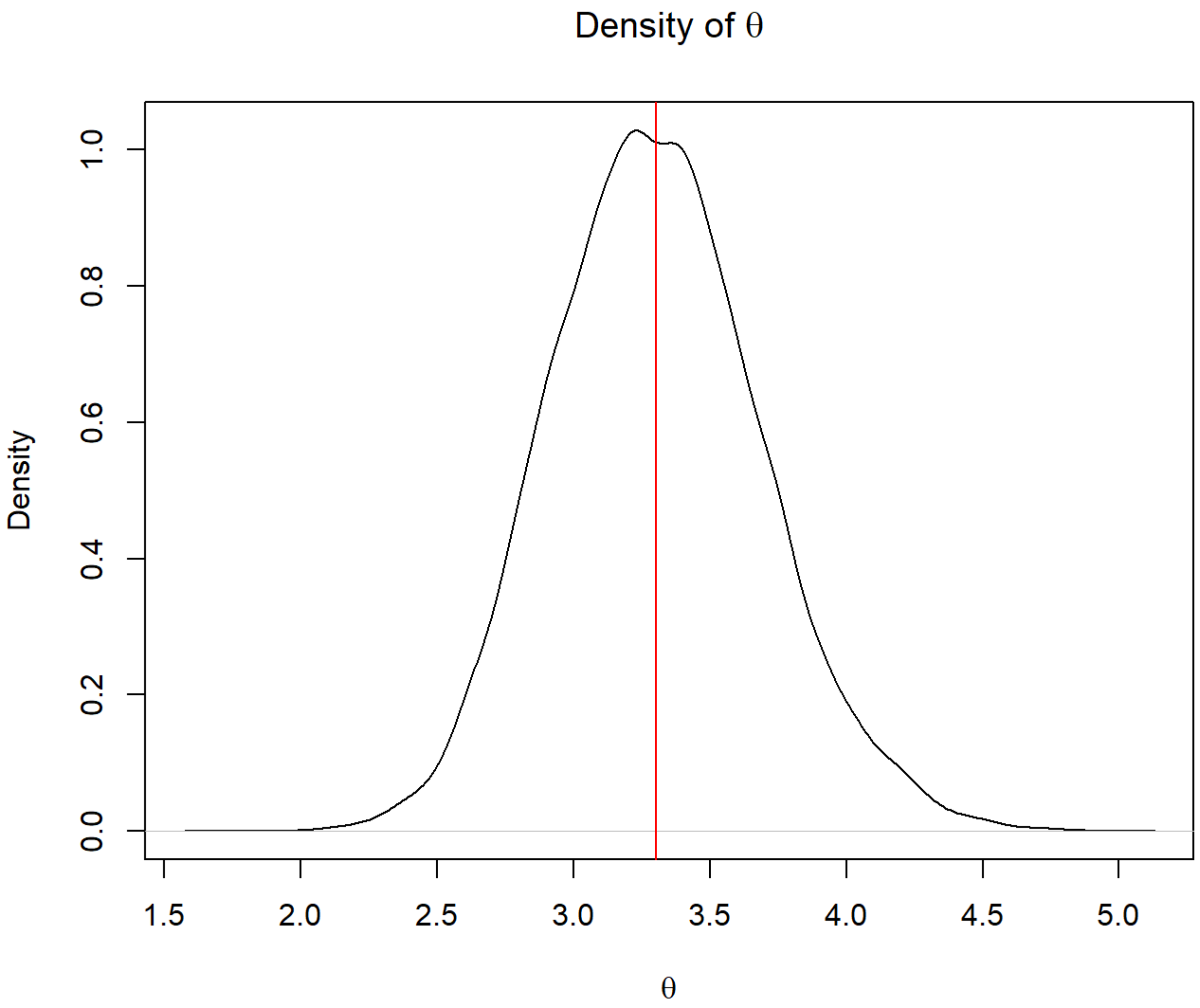
Running average plot of theta



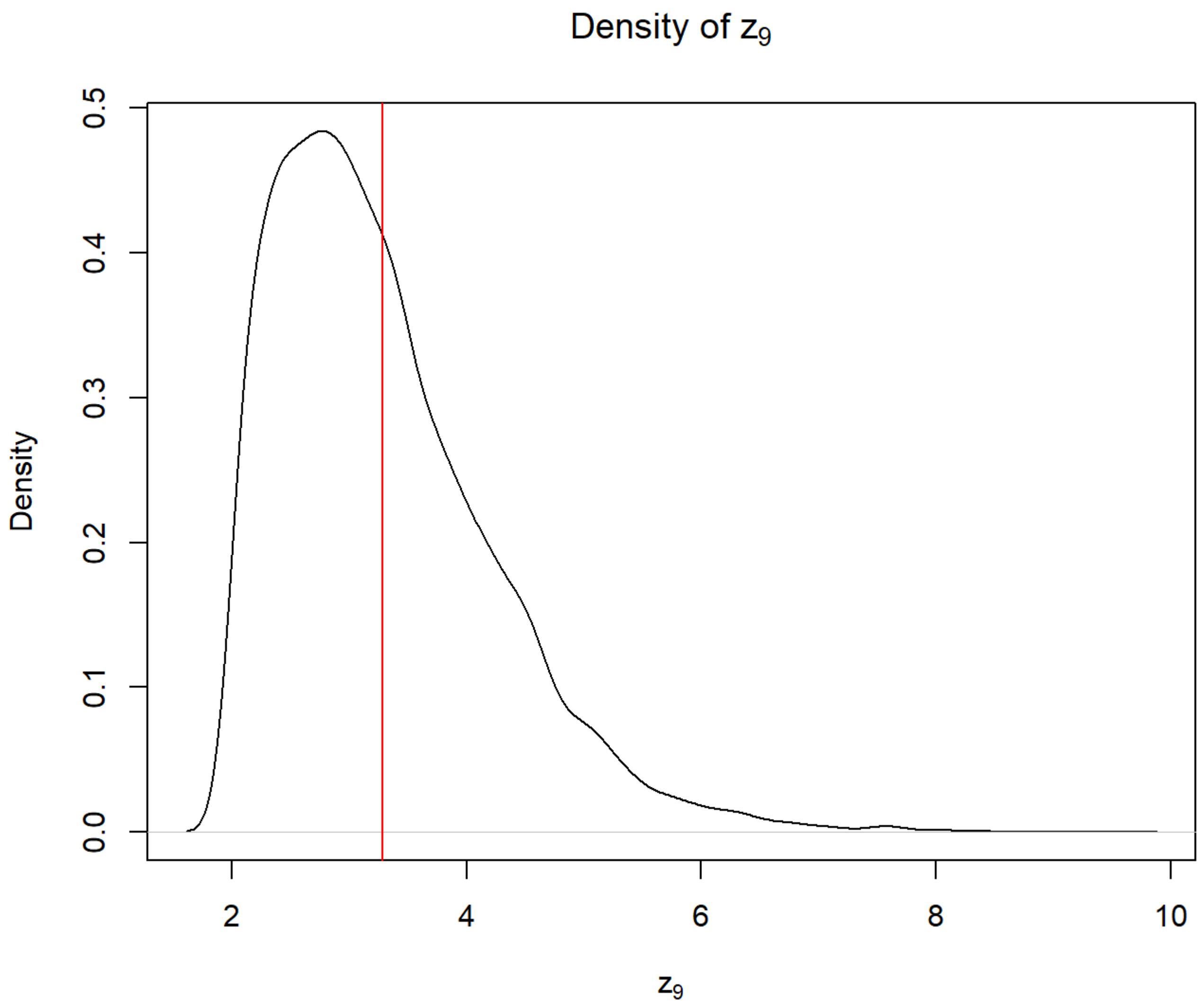
Running average plots of $z_3, z_8, z_9, z_{10}, z_{12}$.

does seem to converge in this example due to the 10,000 iterations.The other diagnostic plots also help to indicate convergence.

C



Estimated posterior density of theta



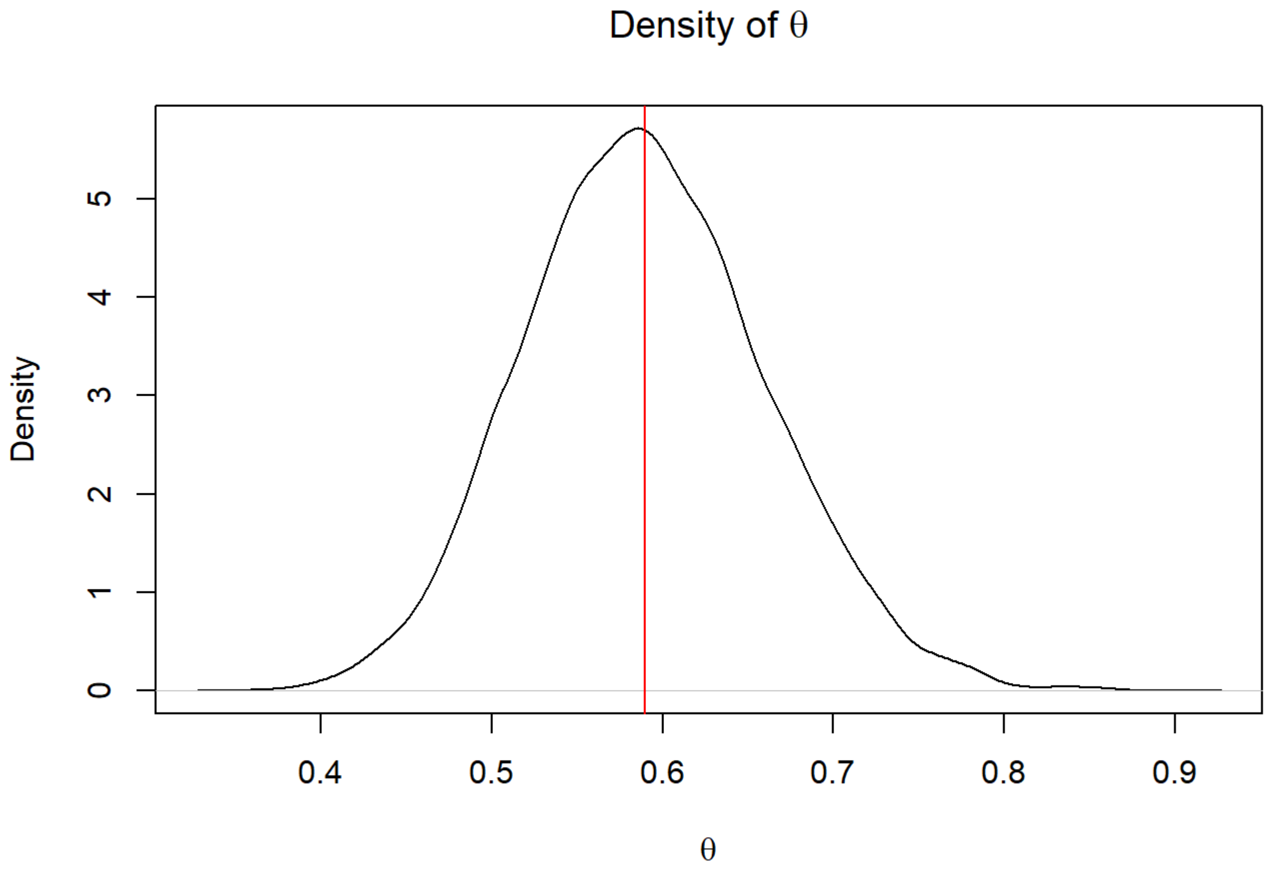
Estimated posterior density of z_9 (posterior mean in red).

These plots show much better distributions of the variables and z_9. Though z_9 is still skewed, it is not as volatile as in 200 iterations. The plot is much more normally distributed.

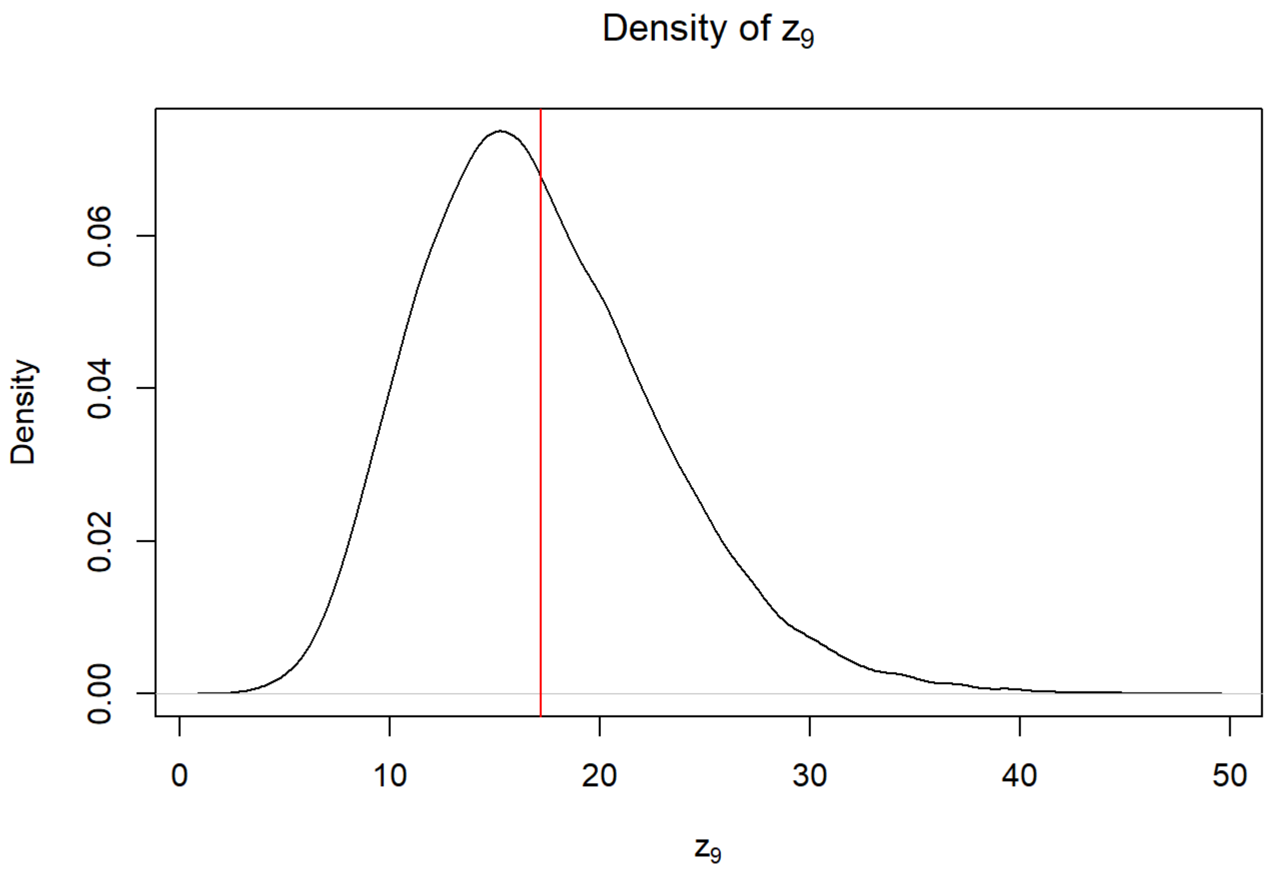
D

```
r <- 10
a <- 1
b <- 100
# input data
z <- c(3.4,2.9,1.4,3.2,1.8,4.6,2.8)
c <- c(1.2,1.7,2.0,1.4,0.6)


n.iter3 <- 10000
init.theta <- 1
init.missing <- rgamma(length(c), shape = r, rate = init.theta)
# run sampler
res3 <- sampleGibbs(z, c, n.iter3, init.theta, init.missing, r, a, b)
```

```
par(mfrow = c(1,1))
plot(density(res3[,1]), xlab = expression(theta),
     main = expression(paste("Density of ", theta)))
abline(v = mean(res3[,1]), col = "red")
```

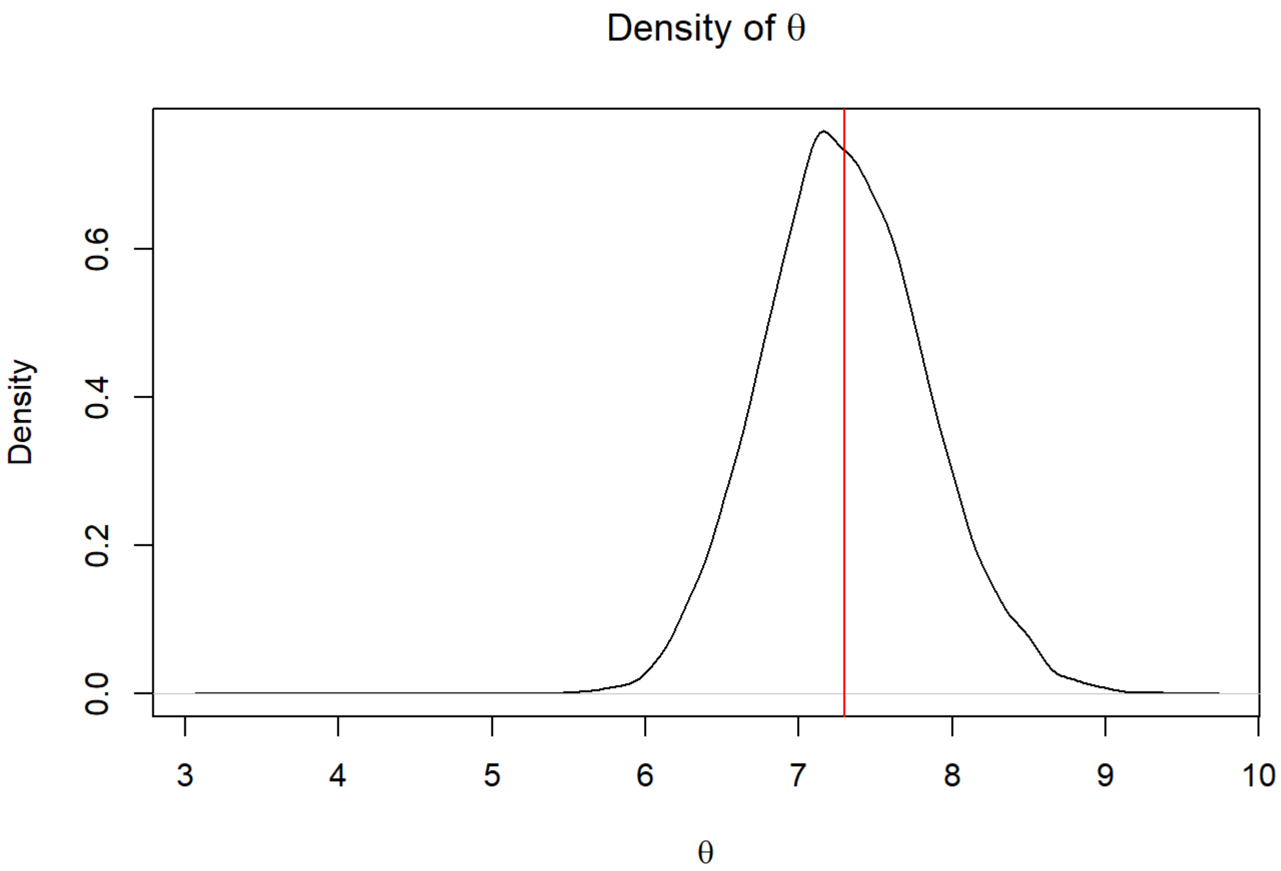


```
plot(density(res3[,4]), xlab = expression(z[9]),
     main = expression(paste("Density of ", z[9])))
abline(v = mean(res3[,4]), col = "red")
```

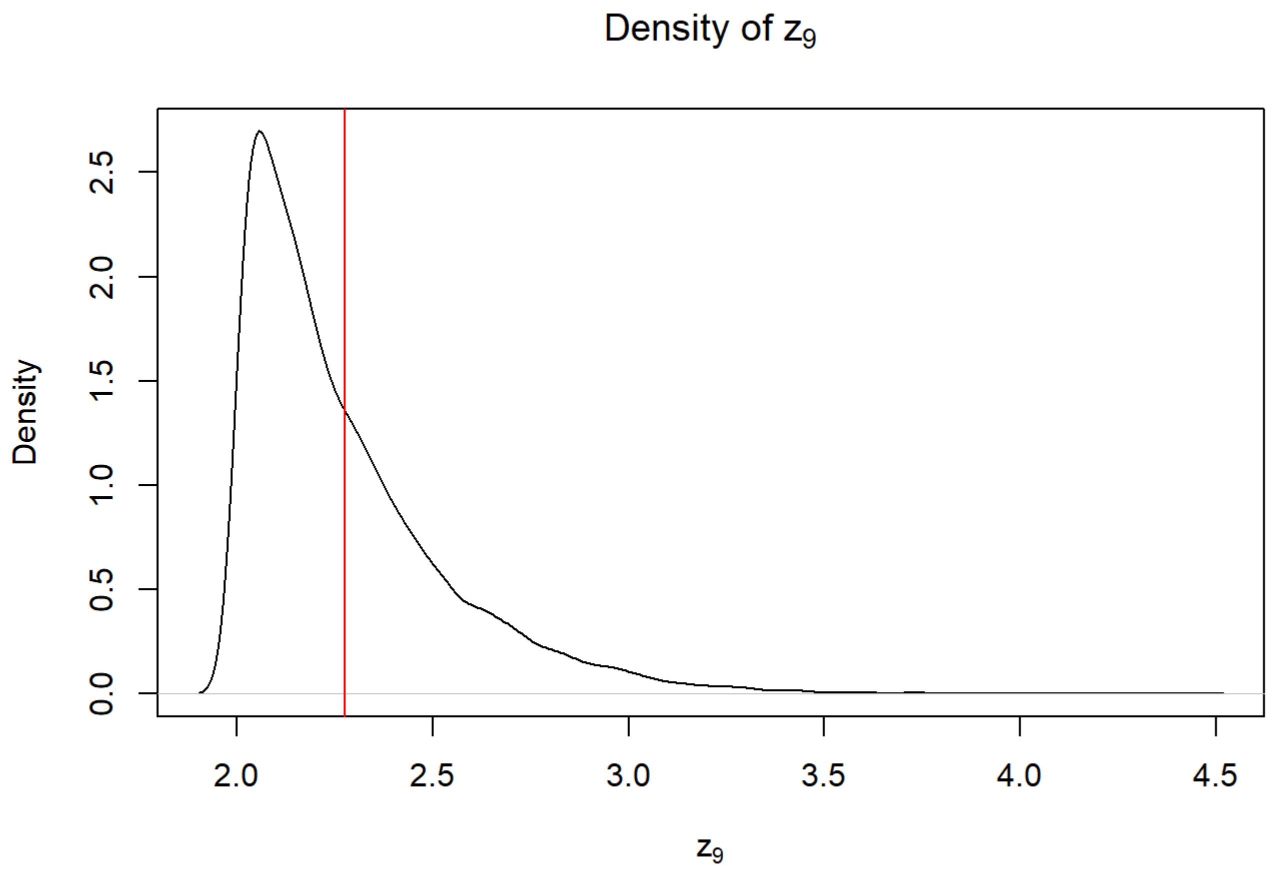


```
r <- 10
a <- 100
b <- 1
# input data
z <- c(3.4,2.9,1.4,3.2,1.8,4.6,2.8)
c <- c(1.2,1.7,2.0,1.4,0.6)


n.iter4 <- 10000
init.theta <- 1
init.missing <- rgamma(length(c), shape = r, rate = init.theta)
# run sampler
res4 <- sampleGibbs(z, c, n.iter4, init.theta, init.missing, r, a, b)
```

```
par(mfrow = c(1,1))
plot(density(res4[,1]), xlab = expression(theta),
     main = expression(paste("Density of ", theta)))
abline(v = mean(res4[,1]), col = "red")
```



```
plot(density(res4[,4]), xlab = expression(z[9]),
     main = expression(paste("Density of ", z[9])))
abline(v = mean(res4[,4]), col = "red")
```



I noticed that when we changed the parameter values from (10,1,1) to (10,1,100), decreased and z_9 increased. But, when we choose the parameter values to (10, 100, 1), the opposite occurs where increases and z_9 decreases.