Homework 6

Zach Calhoun

2/21/2022

Task 1

Load the data and run the gibbs sampling procedure

The diagnostics are provided below.

Task 2

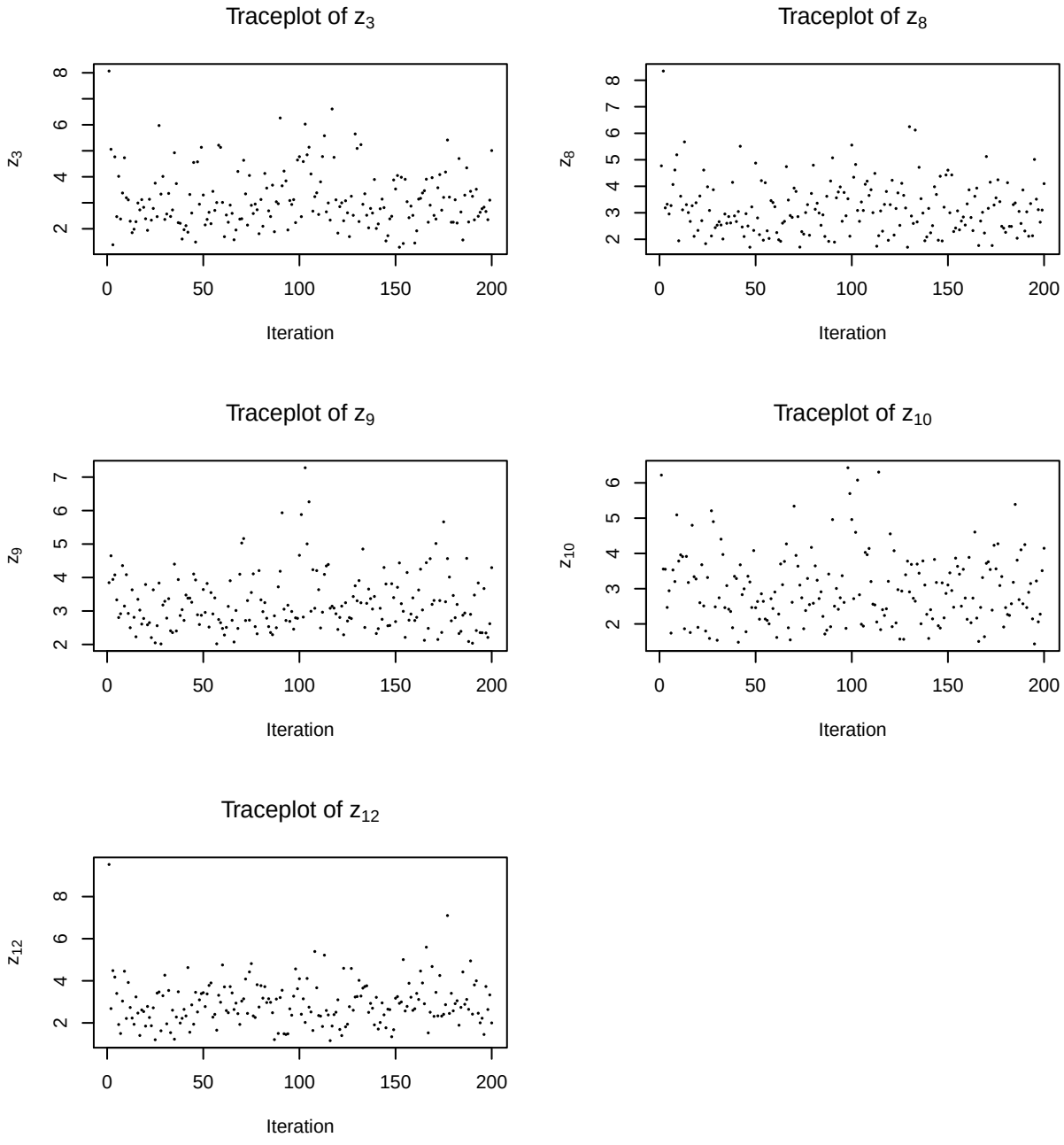
The below code is for setting up the Gibbs sampler.

```
# The code here is taken from the lectures (Gibbs Sampling, part 2)
# Set up the function
sampleTrunGamma <- function(t, a, b){
  p0 <- pgamma(t, shape = a, rate = b)
  # Use the modification of the inverse CD method
  x <- runif(1, min = p0, max = 1)
  y <- qgamma(x, shape = a, rate = b)
  return(y)
}
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.seed(5983)
# set parameter values and enter data
r <- 10
a <- 1
b <- 1
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 <- 200
init.theta <- 1
```

```r
init.missing <- rgamma(length(c), shape = r, rate = init.theta)

res <- sampleGibbs(z, c, n.iter, init.theta, init.missing, r, a, b)
```

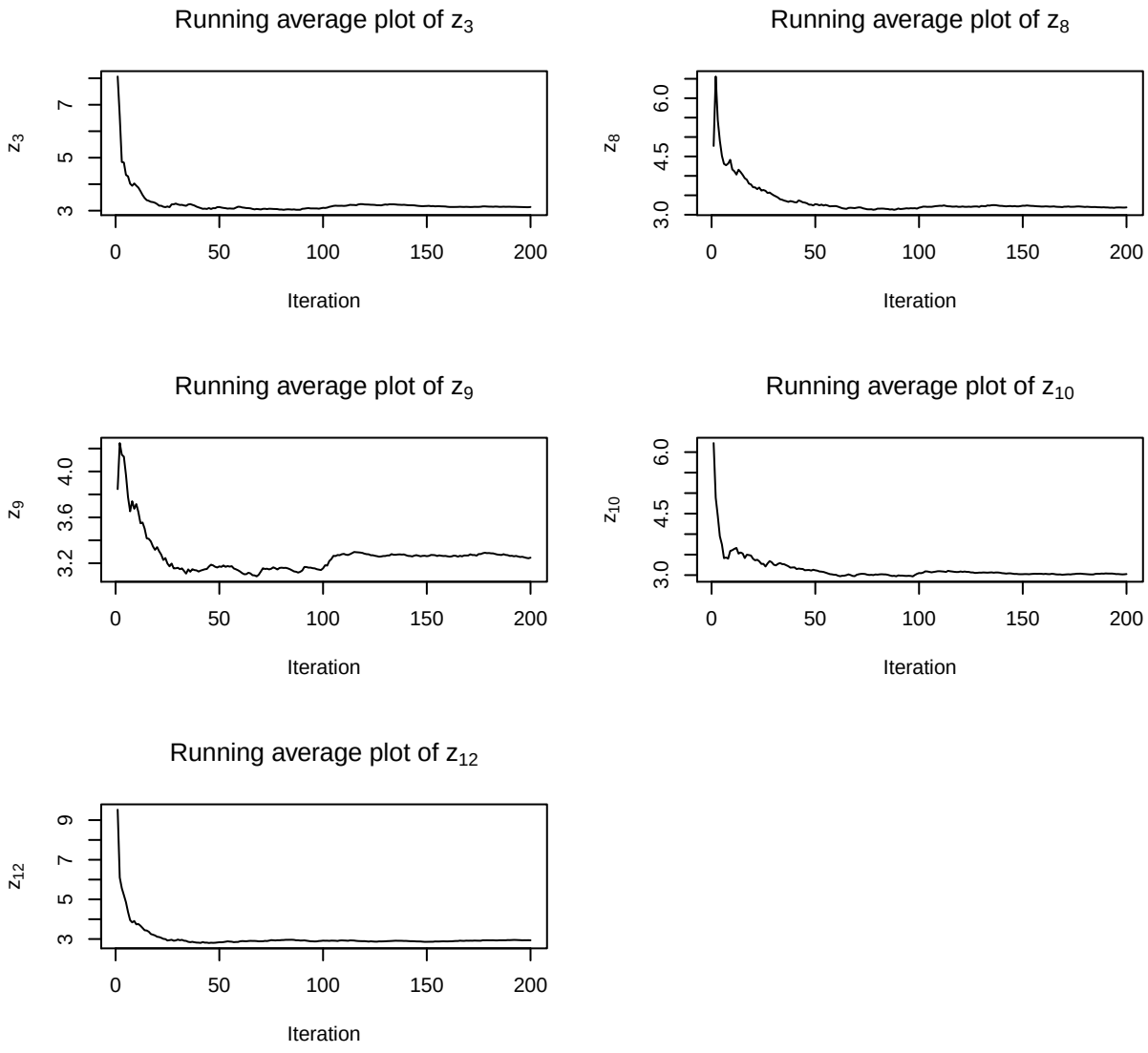
Part A

```
# Create traceplots of the censored data.
z.labs <- c(0,3,8,9,10,12)
par(mfrow = c(3,2))
for (i in 2:6) {
  # Plot each of the missing variables
  plot(1:n.iter, res[,i],pch = 16, cex=.35, xlab= "Iteration",
       ylab=bquote(z[.(z.labs[i])]),
       main=bquote("Traceplot of z"[.(z.labs[i])]))
}
```



Now, I will create the running average plots.

```
# Calculate the average for each of the columns (code taken from the lecture)
run.avg <- apply(res, 2, cumsum)/(1:n.iter)

# Iterate through the latent variables and plot the running average as a line plot
par(mfrow=c(3,2))
for(i in 2:6) {
  plot(1:n.iter, run.avg[,i], type="l", xlab="Iteration",
       ylab=bquote(z[.(z.labs[i])]),
       main=bquote("Running average plot of z"[.(z.labs[i])]))
}
```

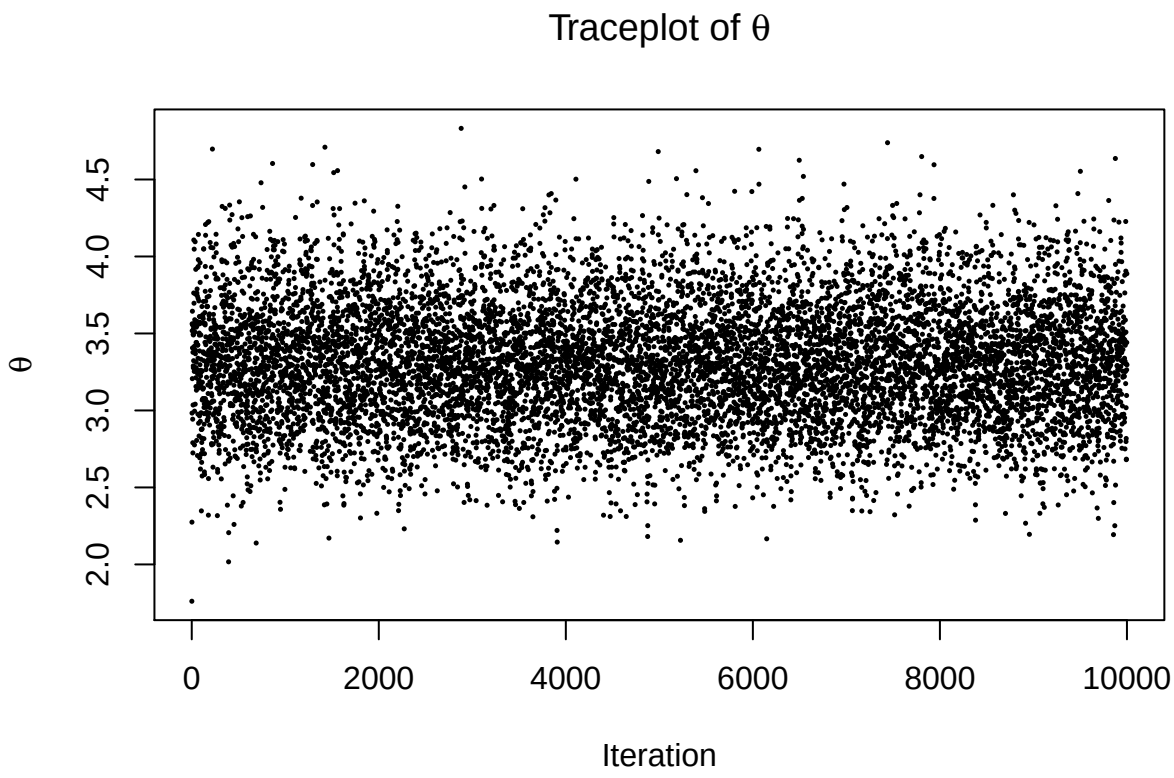


Based off of these plots, it appears that most of the latent variables are relatively stable. However, the running average plot for z_9 makes it appear that perhaps we have not converged, so we should run the sampler for longer to ensure convergence.

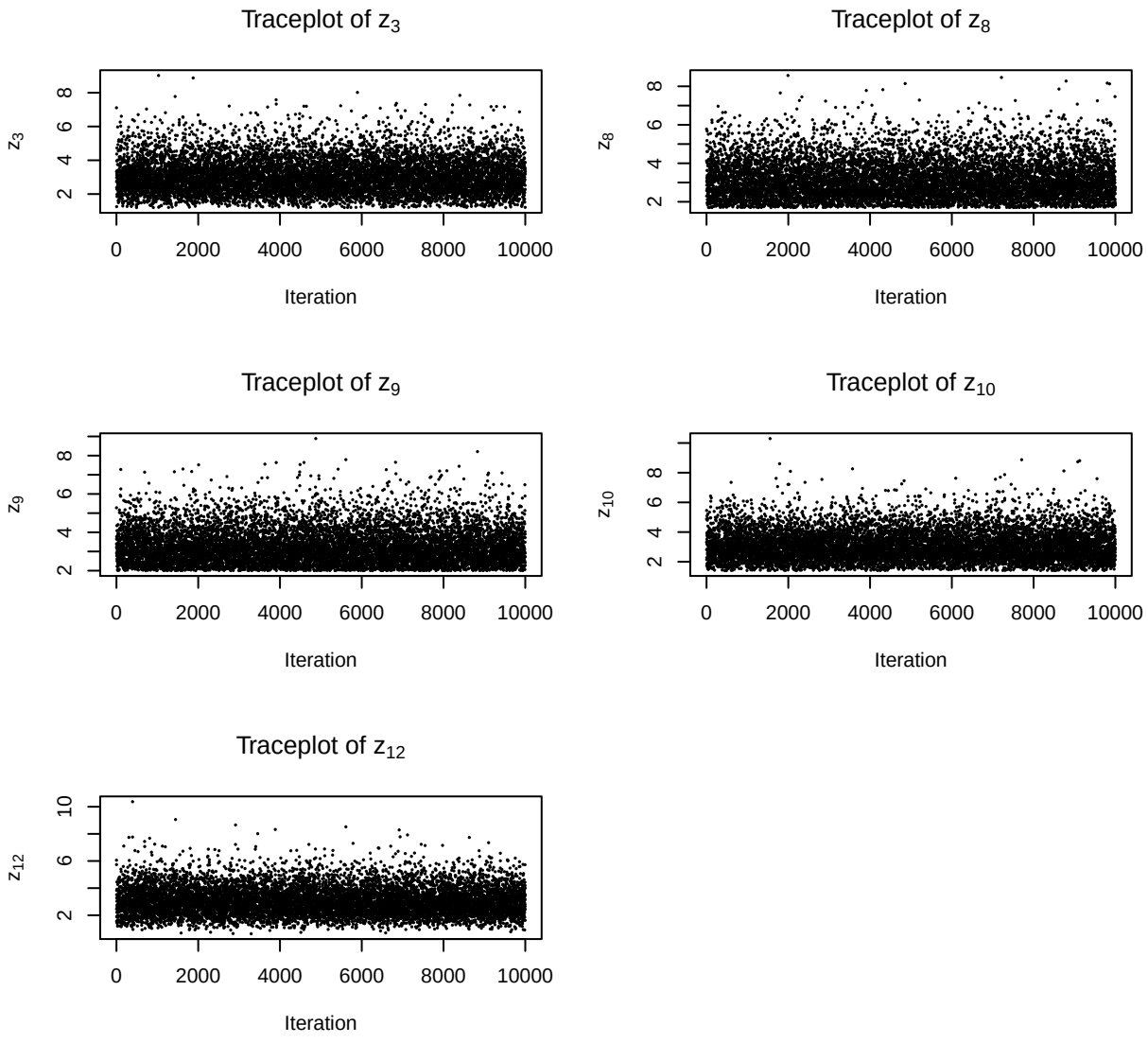
Part B

```
# Run the sampler for 10,000 iterations.
n.iter = 10000
set.seed(5983)
res <- sampleGibbs(z, c, n.iter, init.theta, init.missing, r, a, b)
```

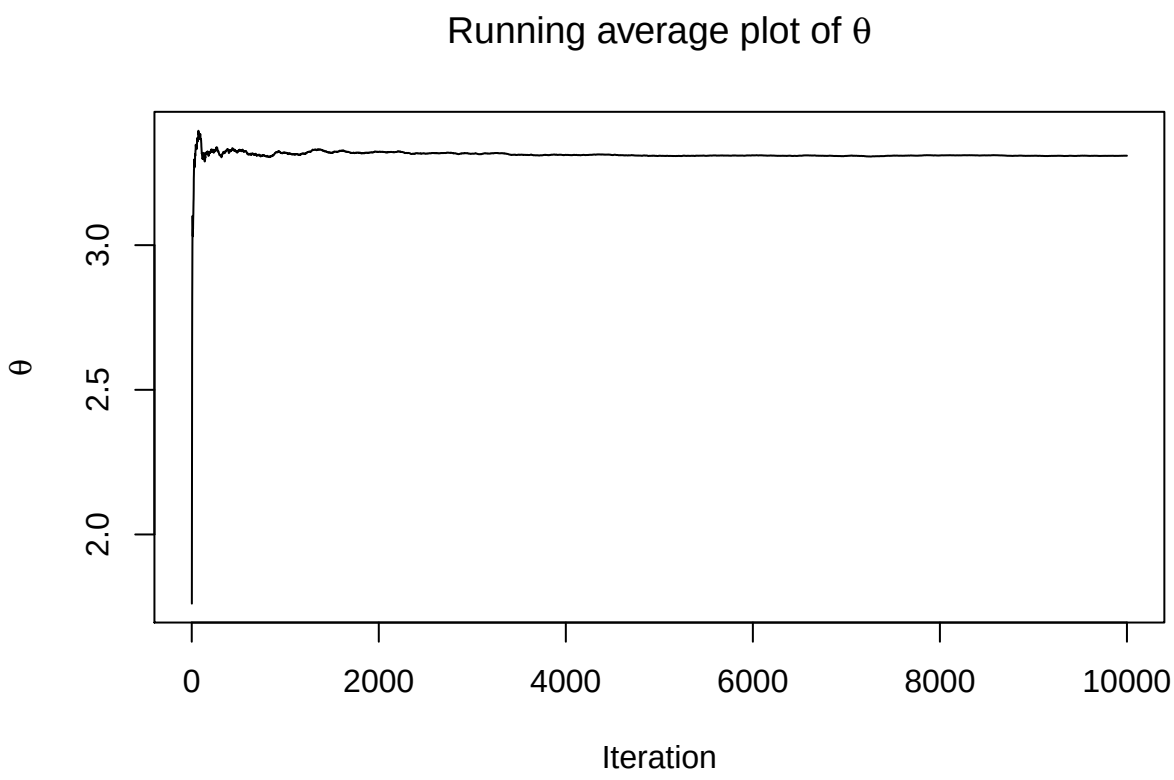
```
# Plot the traceplot of theta
# Traceplot created using code from the lecture
plot(1:n.iter, res[,1], pch = 16, cex = .35, xlab = "Iteration",
     ylab = expression(theta), main = expression(paste("Traceplot of ", theta)))
```



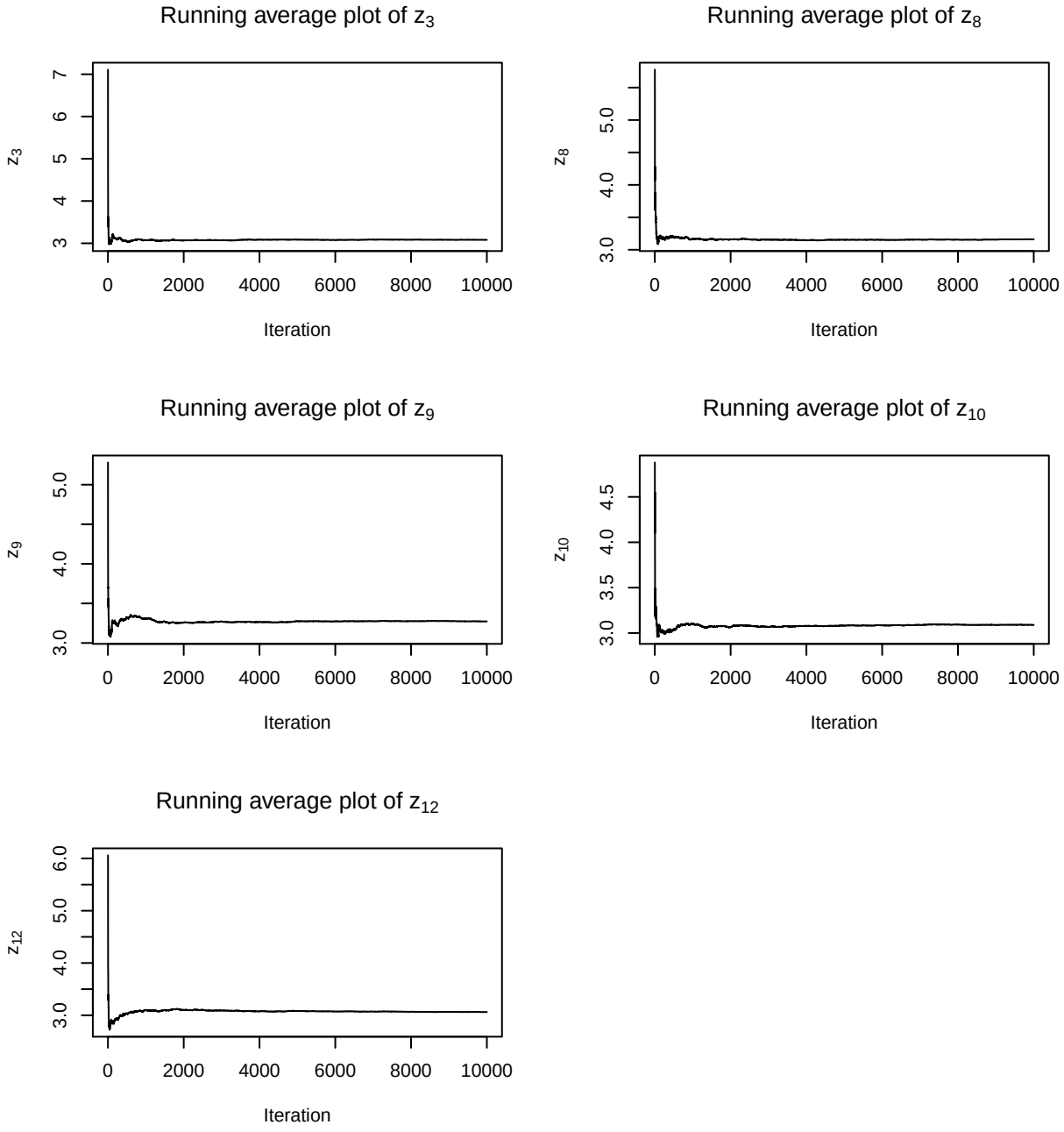
```r
# Plot the traceplot of the latent variables with 10,000 iterations
z.labs <- c(0,3,8,9,10,12)
par(mfrow = c(3,2))
for (i in 2:6) {
  # Plot each of the missing variables
  plot(1:n.iter, res[,i],pch = 16, cex=.35, xlab= "Iteration",
       ylab=bquote(z[.(z.labs[i])]),
       main=bquote("Traceplot of z"[.(z.labs[i])]))
}
```



```r
# Plot the running average of theta for 10,000 iterations
run.avg <- apply(res, 2, cumsum)/(1:n.iter)
# We just want to look at theta here.
plot(1:n.iter, run.avg[,1], type="l", xlab="Iteration",
     ylab=expression(theta),
     main=expression(paste("Running average plot of ", theta)))
```



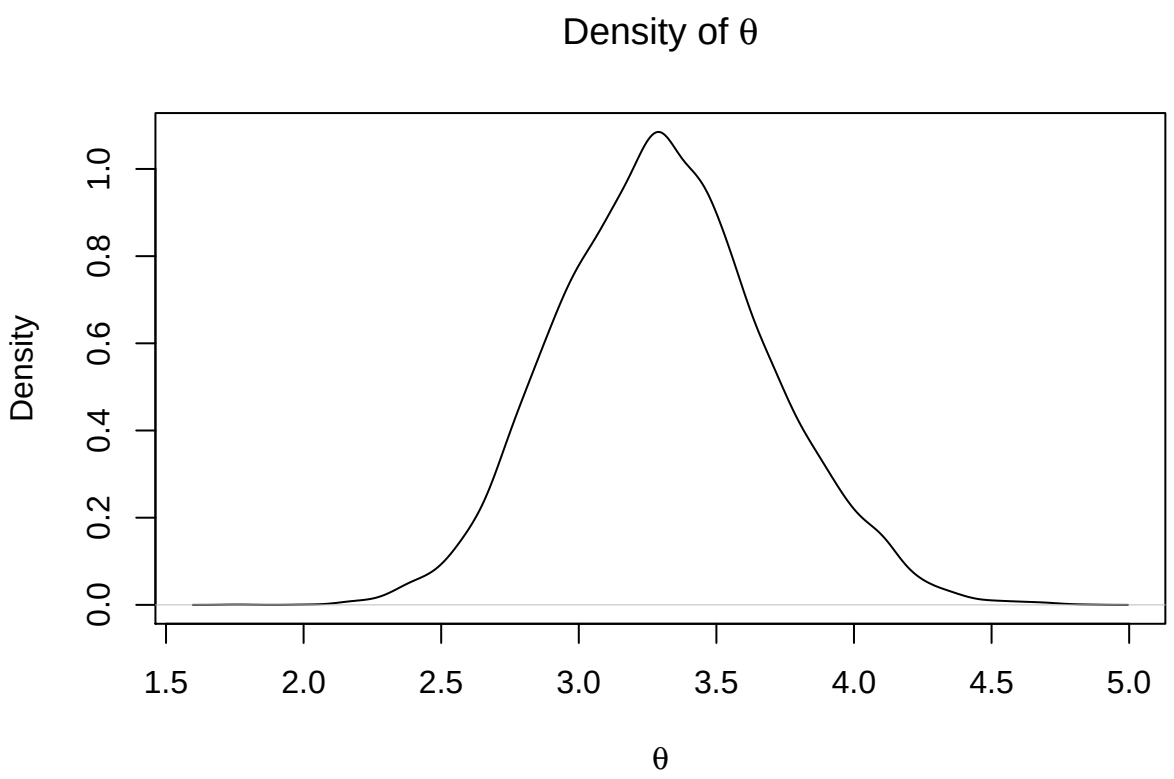
```r
# Plot the running average of the latent variables for 10,000 iterations
par(mfrow=c(3,2))
for(i in 2:6) {
  plot(1:n.iter, run.avg[,i], type="l", xlab="Iteration",
       ylab=bquote(z[.(z.labs[i])]),
       main=bquote("Running average plot of z"[.(z.labs[i])]))
}
```



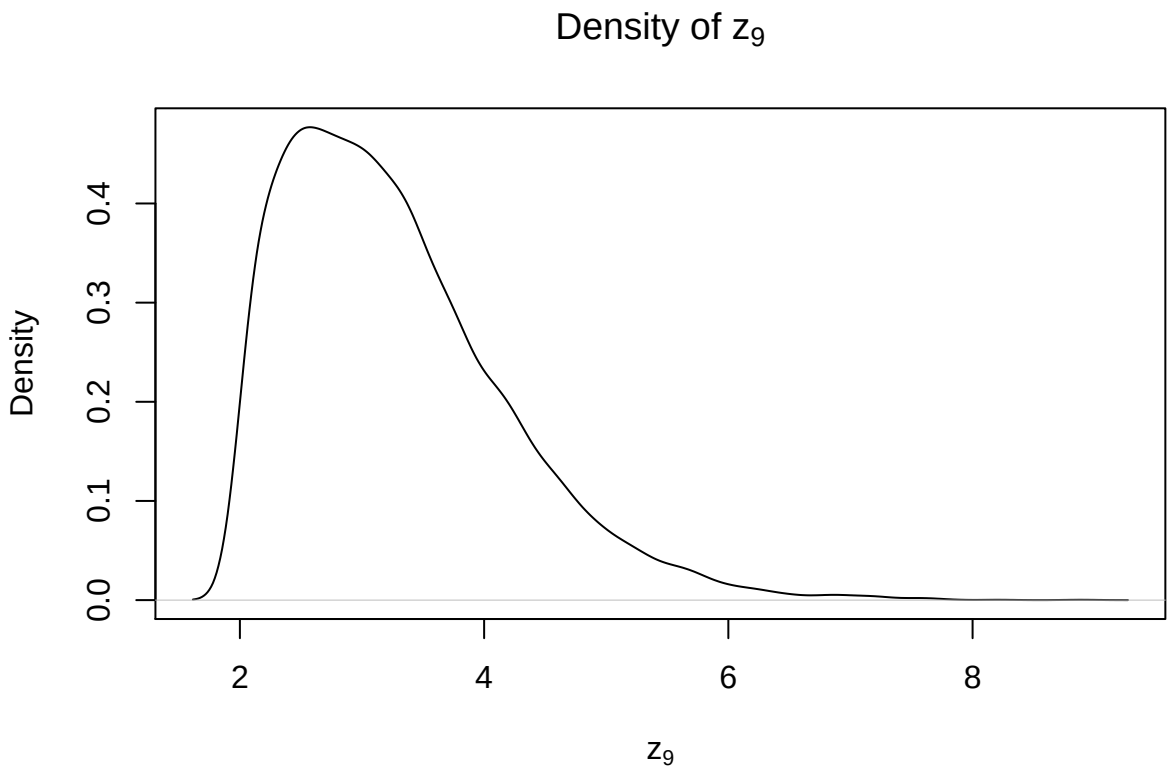
After running the sampler for 10,000 iterations, it is clear that the plots have converged. We can tell that they have converged, because the running average values do not change, and the plot has flattened out.

Part C

```
# Plot the density for theta
plot(density(res[,1]), xlab=expression(theta),
     main=expression(paste("Density of ", theta)))
```



```r
# Plot the density of z_9
plot(density(res[,4]), xlab=expression("z"[9]), main=expression("Density of z"[9]))
```

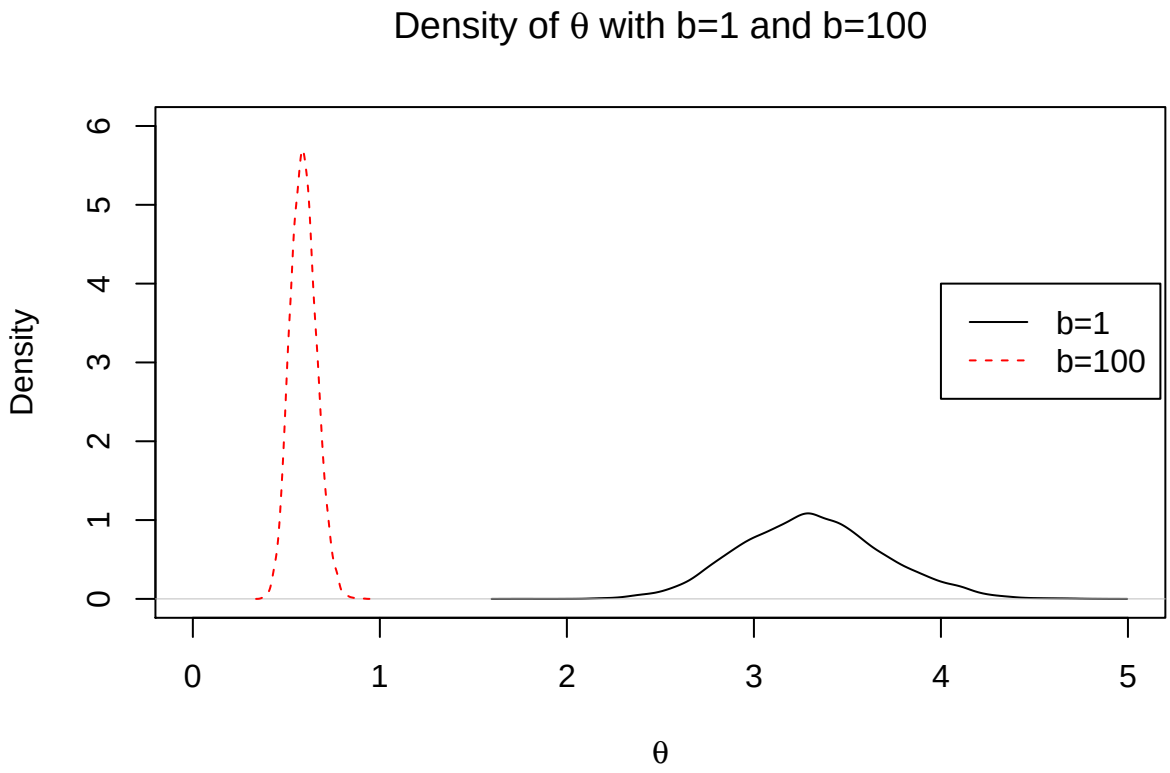


Based on the density plots above, it is even more clear that the sampler has converged. The plots are smooth, with a clear mode. Because the Gibbs sampler is subject to autocorrelation between samples, it makes sense that as we run more iterations of the sampler, we will see that the effects on the samples due to autocorrelation will mostly disappear, giving us a better understanding of the true density. For both θ and z_9, we should expect that the density plots resemble Gamma distributions, and this is clear for both density plots above, albeit less so for the density of θ.

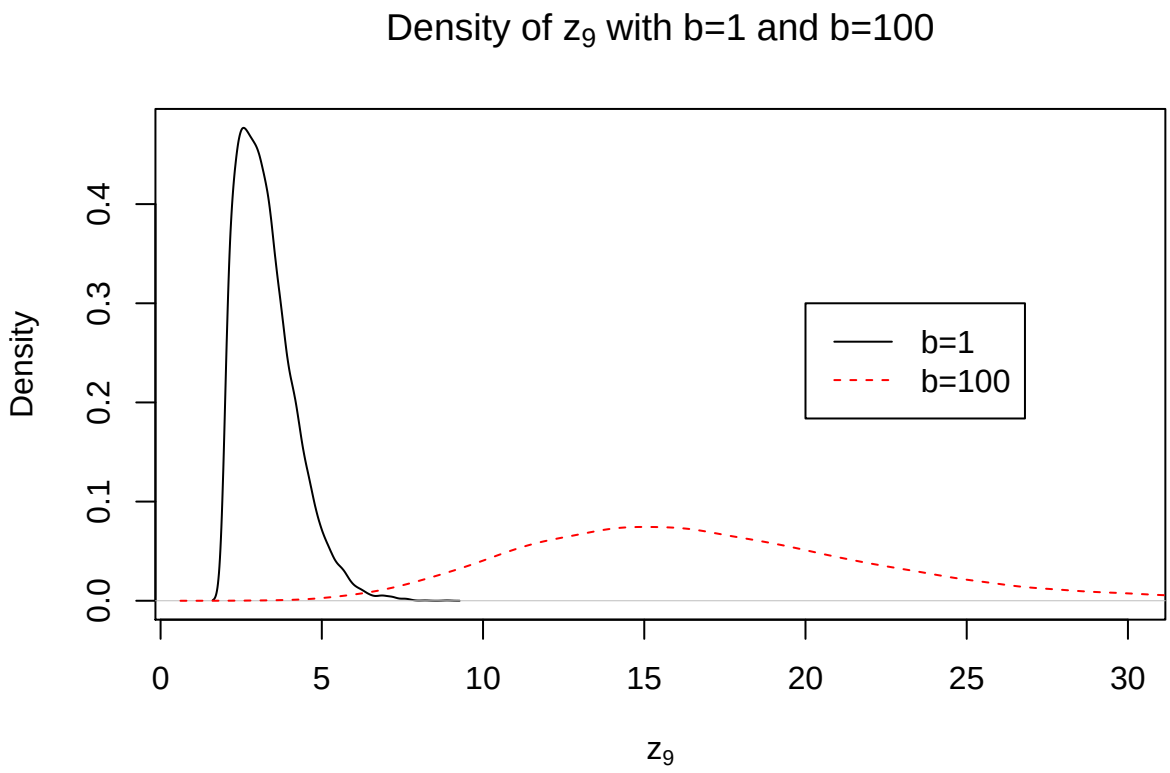
Part D

```
# Look at r = 10, a = 1, b = 100
r <- 10
a <- 1
b <- 100
n.iter = 10000
# Save into a new matrix, so that we can plot against the original data.
res.d <- sampleGibbs(z, c, n.iter, init.theta, init.missing, r, a, b)
```

```
# Again, plot the density of theta
plot(density(res[,1]), xlim=c(0,5), ylim=c(0,6), xlab=expression(theta), main=expression(paste("Density
lines(density(res.d[,1]), col="red", lty=2)
legend(4,4, legend=c("b=1", "b=100"), col=c("black", "red"), lty=1:2)
```



```r
# Plot the two densities of z9 using the different values of b
plot(density(res[,4]), xlim=c(1, 30), xlab=expression("z"[9]),
     main=expression(paste("Density of z"[9], " with b=1 and b=100")))
lines(density(res.d[,4]), col="red", lty=2)
legend(20,0.3, legend=c("b=1", "b=100"), col=c("black", "red"), lty=1:2)
```

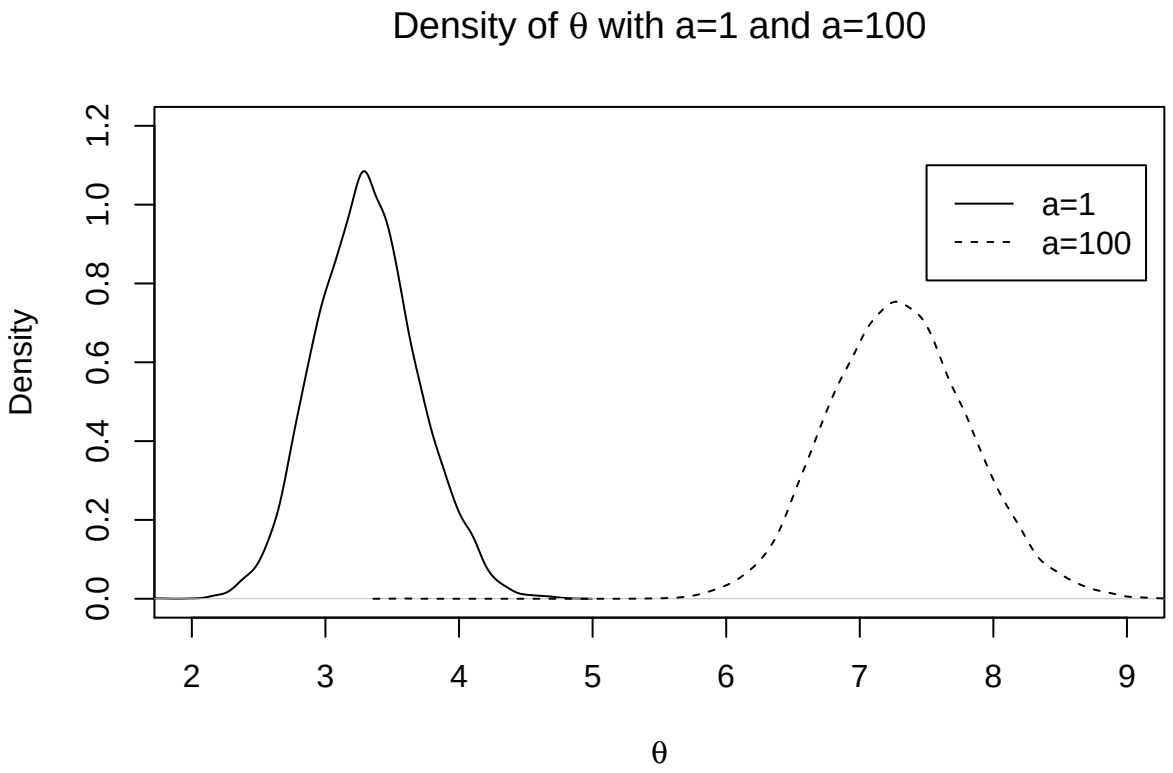


As shown in the two plots above, both of the posterior densities change significantly when the value of b changes. This makes sense, because we do not have a lot of data, and $b = 100$ is quite a difference from the prior parameter when $b = 1$.

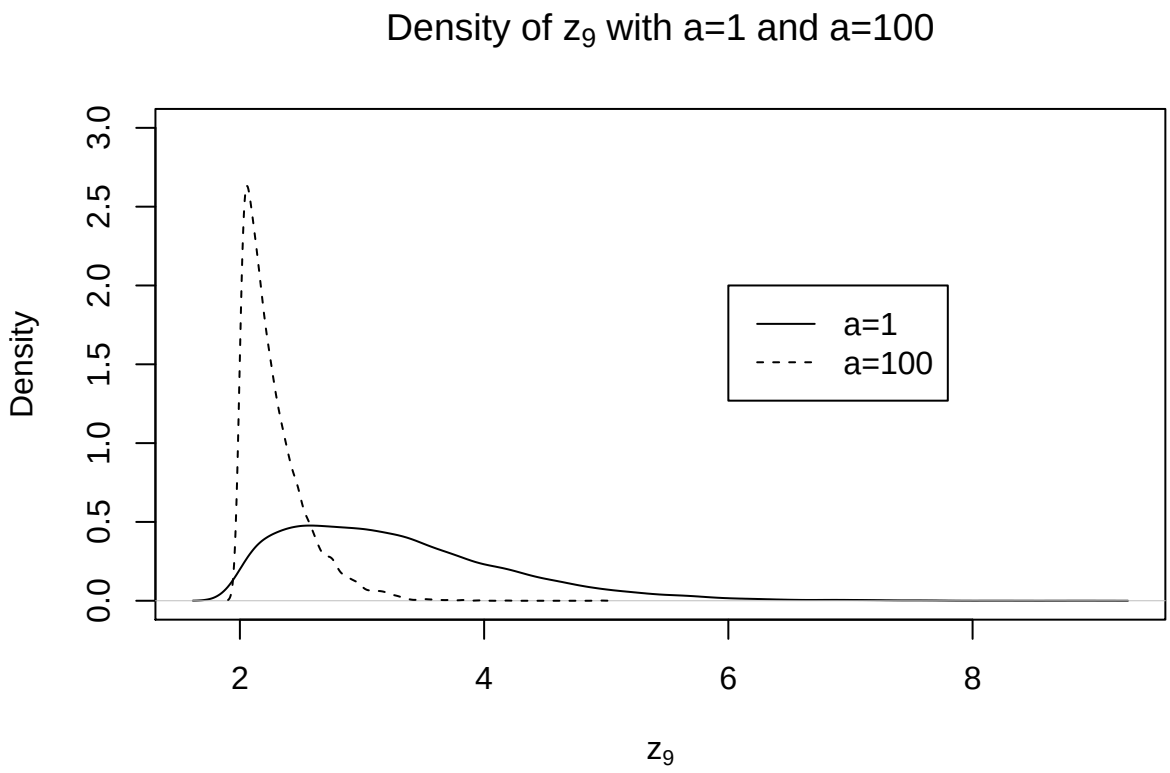
Let's look at what happens when we change the a parameter from $a = 1$ to $a = 100$.

```
# Look at 4 = 10, a = 100, b = 1
r <- 10
a <- 100
b <- 1
n.iter = 10000
# Save into a new matrix, so that we can plot against the original data.
res.d <- sampleGibbs(z, c, n.iter, init.theta, init.missing, r, a, b)
```

```
# plot the densities for different values of a
plot(density(res[,1]), xlim=c(2,9), ylim=c(0,1.2), xlab=expression(theta), main=expression(paste("Densi
lines(density(res.d[,1]), lty=2)
legend(7.5,1.1, legend=c("a=1", "a=100"), lty=1:2)
```



```r
# Plot the two densities of z9 using the different values of b
plot(density(res[,4]), ylim=c(0,3),xlab=expression("z"[9]),
     main=expression(paste("Density of z"[9], " with a=1 and a=100")))
lines(density(res.d[,4]), lty=2)
legend(6,2, legend=c("a=1", "a=100"), lty=1:2)
```



Interestingly, when we change $a = 1$ to $a = 100$, the shape does not change that significantly for θ, although the density shifts its mean. However, for z_9, the shape changes to be closer to a spike around $z_9 = 2.3$. Again, these changes make sense. We do not have a lot of data points, so placing a prior value of $a = 100$ would represent a significant change to the model, and we simply don't have the data set to overwhelm the prior. Without a large data set, we should expect that significant changes to our priors will impact the outcome of our posteriors significantly.