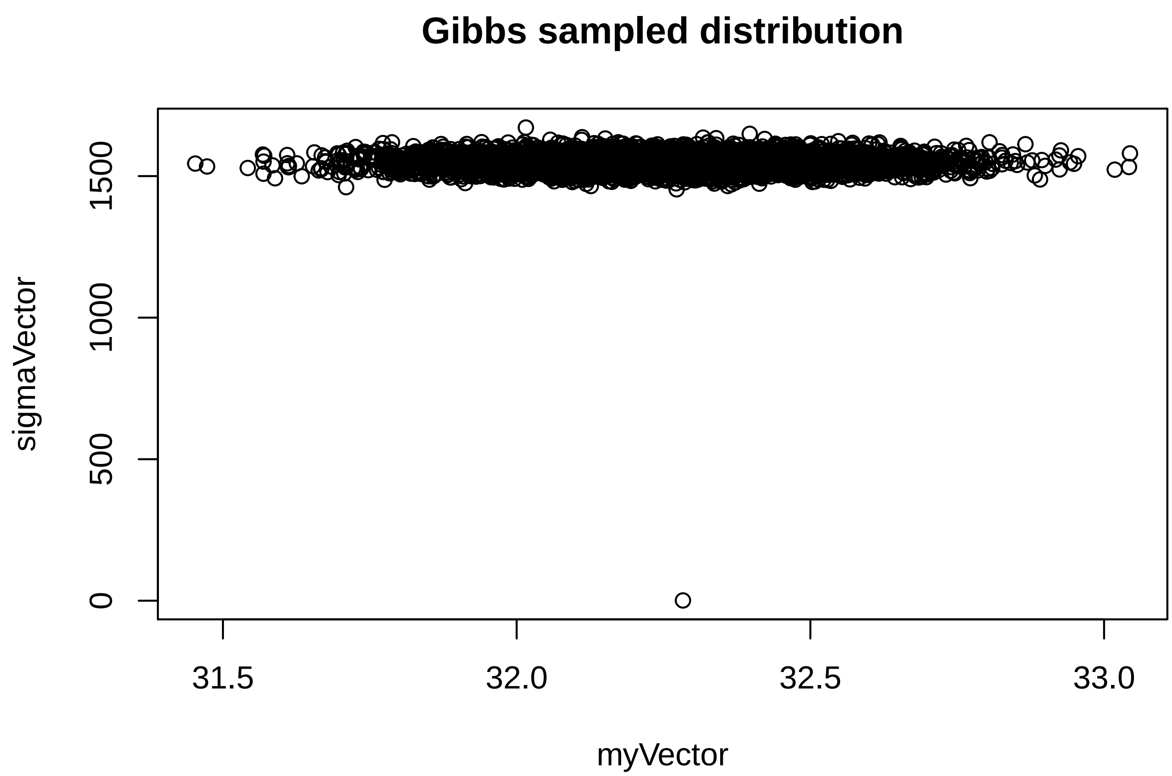
Lab Report 3

Assignment 1:

Task a) i.

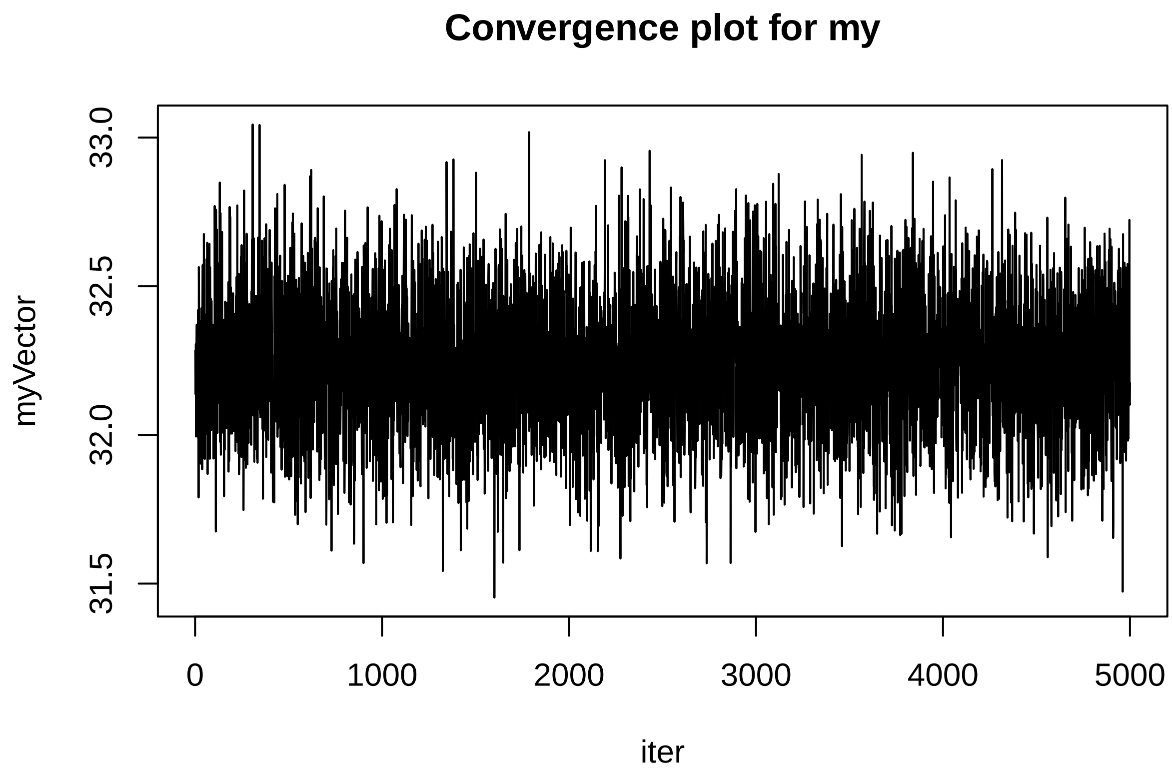
```
data <- read.table("rainfall.dat")

#Setup
v0 <- 1
tausq0 <- 100
meandata <- mean(data$V1)
sigmasq0 <- 1
n= length(data$V1)
nDraws <- 5000
mu0=14.79
vn = v0+n
calcTau <- function(n, tau, sigmasq){
  return(1/(n/sigmasq+1/tau))
}

calcMy <- function(sigmasq, n, my0,mean, tausq){

  return(calcW(n, sigmasq, tausq)*mean+((1-calcW(n,sigmasq,tausq))*my0))
}

calcW <- function(n, sigmasq, tausq){
  return((n/sigmasq)/((n/sigmasq)+(1/tausq)))
}

calcSigmaHat <- function(v0, sigmasq0, data, my, n){
  return((v0*sigmasq0+sum((data-my)^2))/(v0+n))
}

#Gibbs sampling
myVector <- c()
sigmaVector <- c(sigmasq0)
tauVector <- c(tausq0)
for( i in 1:nDraws){
  myVector <- c(myVector, rnorm(1, calcMy(sigmaVector[i],n,mu0,meandata, tausq0), calcTau(n,tausq0,sigma
  if(i<nDraws){
  drawX <- rchisq(1,vn)
  sigmaVector <- c(sigmaVector, vn*calcSigmaHat(v0,sigmasq0,data$V1,myVector[i],n)/drawX)
  }
}
```

```r
plot(myVector,sigmaVector, main="Gibbs sampled distribution")
```

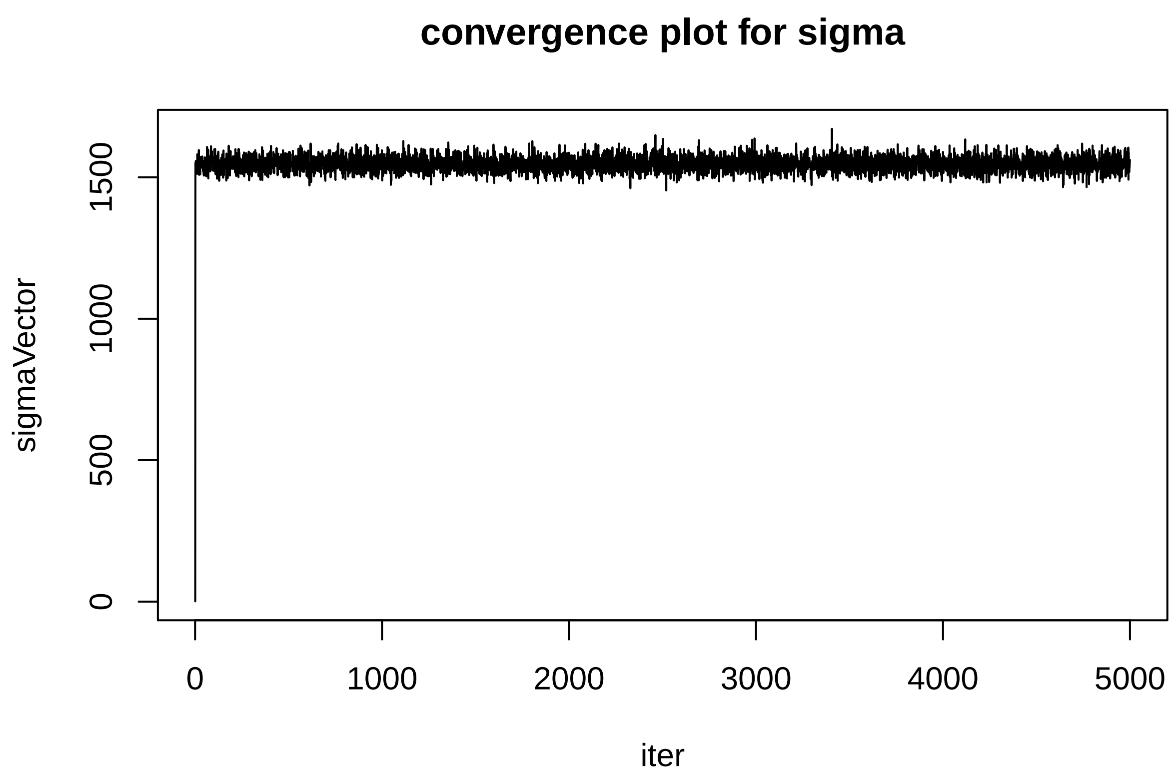


##Task a) ii.

```r
iter=seq(1,5000,1)
plot(iter, myVector, type="l", main="Convergence plot for my")
```



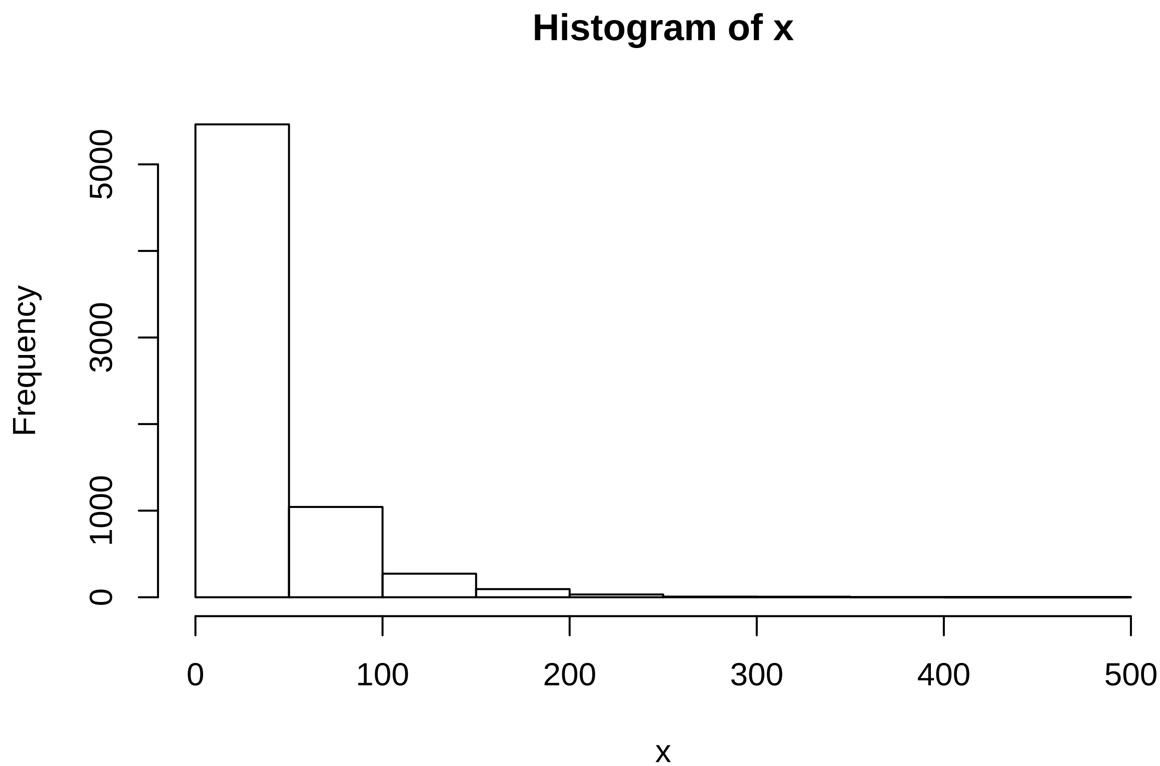

```r
plot(iter, sigmaVector, type="l", main="convergence plot for sigma")
```



We see that the gibbs sampling results in a distribution where μ converges to 32 and σ^2 converges to 1500. ##Task b)

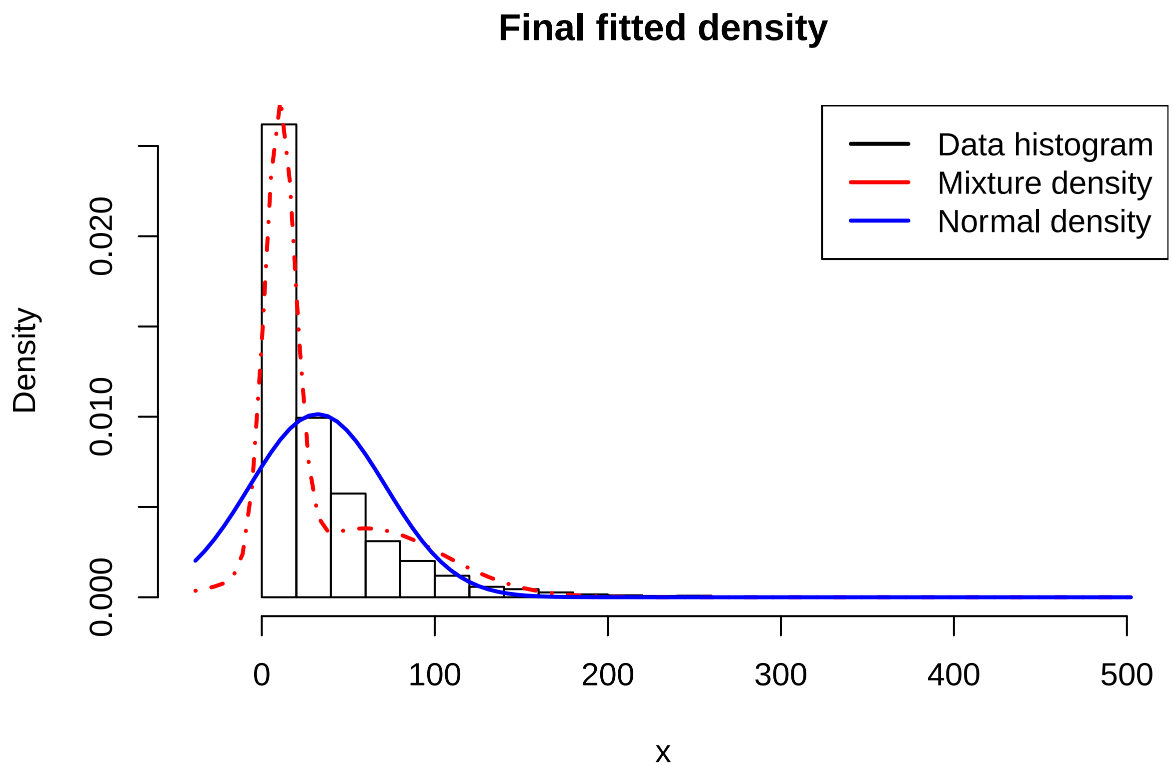
```
##########    BEGIN USER INPUT #################
x <- as.matrix(data$V1)

# Model options
nComp <- 2    # Number of mixture components

# Prior options
alpha <- 1*rep(1,nComp) # Dirichlet(alpha)
muPrior <- rep(14.79,nComp) # Prior mean of mu
tau2Prior <- rep(100,nComp) # Prior std of mu
sigma2_0 <- rep(var(x),nComp) # s20 (best guess of sigma2)
nu0 <- rep(1,nComp) # degrees of freedom for prior on sigma2

# MCMC options
nIter <- 1000 # Number of Gibbs sampling draws


# Plotting options
plotFit <- TRUE
lineColors <- c("blue", "green", "magenta", 'yellow')
# Adding sleep time between iterations for plotting
################   END USER INPUT ###############
```

```r
###### Defining a function that simulates from the
rScaledInvChi2 <- function(n, df, scale){
  return((df*scale)/rchisq(n,df=df))
}

####### Defining a function that simulates from a Dirichlet distribution
rDirichlet <- function(param){
  nCat <- length(param)
  piDraws <- matrix(NA,nCat,1)
  for (j in 1:nCat){
    piDraws[j] <- rgamma(1,param[j],1)
  }
  piDraws = piDraws/sum(piDraws) # Diving every column of piDraws by the sum of the elements in that co
  return(piDraws)
}

# Simple function that converts between two different representations of the mixture allocation
S2alloc <- function(S){
  n <- dim(S)[1]
  alloc <- rep(0,n)
  for (i in 1:n){
    alloc[i] <- which(S[i,] == 1)
  }
  return(alloc)
}

# Initial value for the MCMC
nObs <- length(data$V1)
S <- t(rmultinom(nObs, size = 1 , prob = rep(1/nComp,nComp))) # nObs-by-nComp matrix with component all
mu <- quantile(x, probs = seq(0,1,length = nComp))
sigma2 <- rep(var(x),nComp)
probObsInComp <- rep(NA, nComp)


# Setting up the plot
xGrid <- seq(min(x)-1*apply(x,2,sd),max(x)+1*apply(x,2,sd),length = 100)
xGridMin <- min(xGrid)
xGridMax <- max(xGrid)
mixDensMean <- rep(0,length(xGrid))
effIterCount <- 0
ylim <- c(0,2*max(hist(x)$density))
```



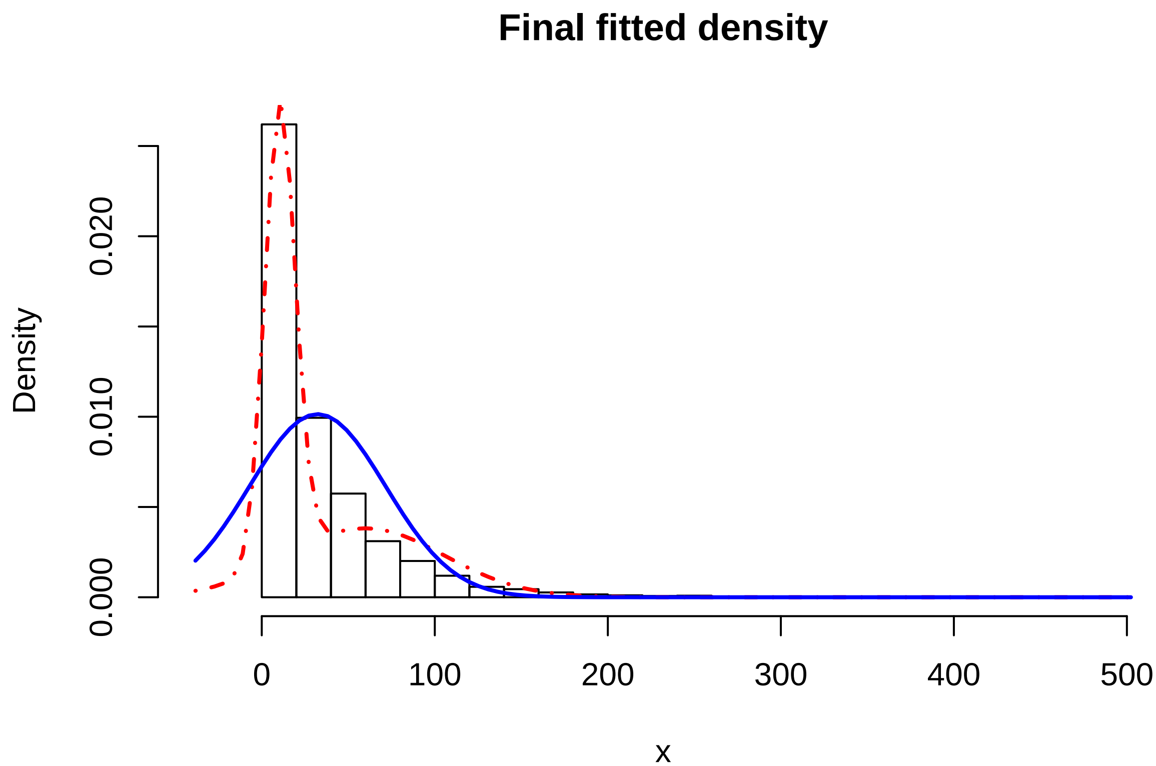

```
for (k in 1:nIter){

  alloc <- S2alloc(S) # Just a function that converts between different representations of the group al
  nAlloc <- colSums(S)

  # Update components probabilities
  pi <- rDirichlet(alpha + nAlloc)

  # Update mu's
  for (j in 1:nComp){
    precPrior <- 1/tau2Prior[j]
    precData <- nAlloc[j]/sigma2[j]
    precPost <- precPrior + precData
    wPrior <- precPrior/precPost
    muPost <- wPrior*muPrior + (1-wPrior)*mean(x[alloc == j])
    tau2Post <- 1/precPost
    mu[j] <- rnorm(1, mean = muPost, sd = sqrt(tau2Post))
  }

  # Update sigma2's
  for (j in 1:nComp){
    sigma2[j] <- rScaledInvChi2(1, df = nu0[j] + nAlloc[j], scale = (nu0[j]*sigma2_0[j] + sum((x[alloc =
  }

  # Update allocation
  for (i in 1:nObs){
```

```
    for (j in 1:nComp){
      probObsInComp[j] <- pi[j]*dnorm(x[i], mean = mu[j], sd = sqrt(sigma2[j]))
    }
    S[i,] <- t(rmultinom(1, size = 1 , prob = probObsInComp/sum(probObsInComp)))
  }

  # Printing the fitted density against data histogram
  if (plotFit && (k%%1 ==0)){
    effIterCount <- effIterCount + 1

    mixDens <- rep(0,length(xGrid))
    components <- c()
    for (j in 1:nComp){
      compDens <- dnorm(xGrid,mu[j],sd = sqrt(sigma2[j]))
      mixDens <- mixDens + pi[j]*compDens

      components[j] <- paste("Component ",j)
    }
    mixDensMean <- ((effIterCount-1)*mixDensMean + mixDens)/effIterCount



  }

}

hist(x, breaks = 20, freq = FALSE, xlim = c(xGridMin,xGridMax), main = "Final fitted density")
lines(xGrid, mixDensMean, type = "l", lwd = 2, lty = 4, col = "red")
lines(xGrid, dnorm(xGrid, mean = mean(x), sd = apply(x,2,sd)), type = "l", lwd = 2, col = "blue")
legend("topright", box.lty = 1, legend = c("Data histogram","Mixture density","Normal density"), col=c(
```



With prior $\alpha = 1$, $\mu = 14.79$, τ^2=100, $\nu_0 = 1$, the final fitted density follows the histogram quite accurately with a mixture distribution with a mode slightly below 20.

```
hist(x, breaks = 20, freq = FALSE, xlim = c(xGridMin,xGridMax), main = "Final fitted density")
lines(xGrid, mixDensMean, type = "l", lwd = 2, lty = 4, col = "red")
lines(xGrid, dnorm(xGrid, mean = mean(myVector), sd = mean(sqrt(sigmaVector))), type = "l", lwd = 2, col
```



As seen in the plot, the blue line, representing the gibbs sampled distibribution is very similar to the true distribution derived from the data in the previous plot (also the blue line).

#Assignment 2 ##Task a