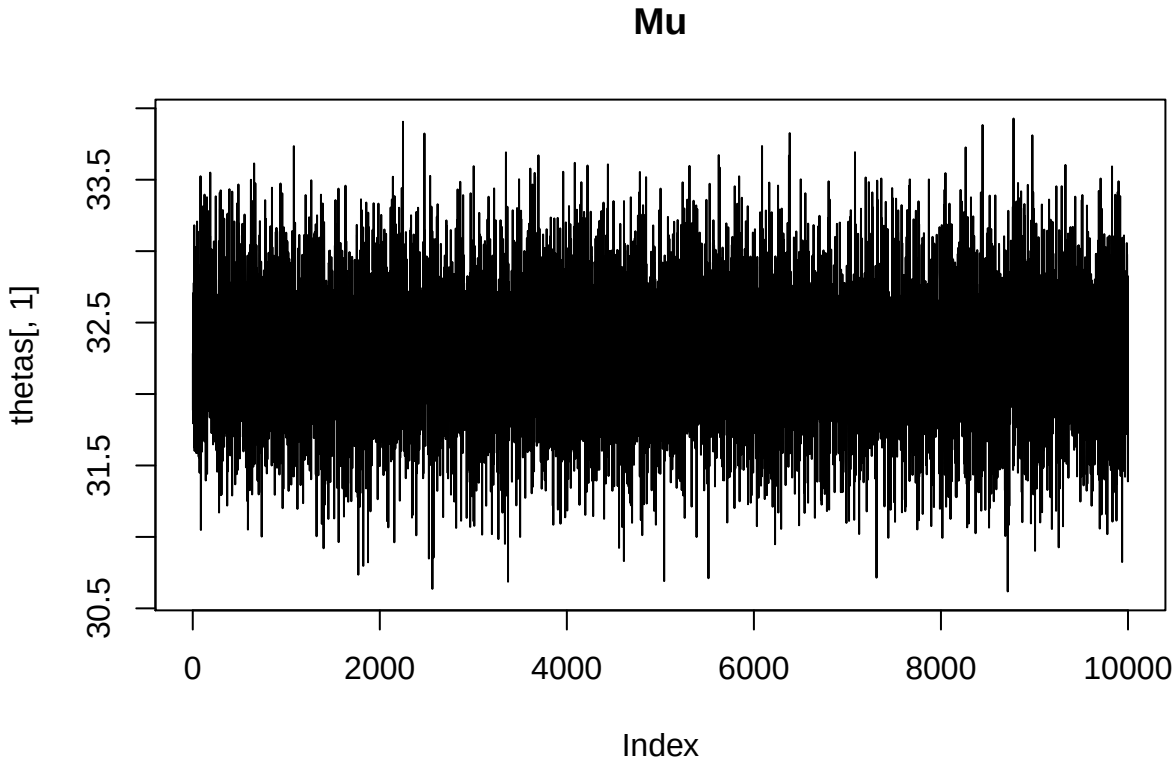
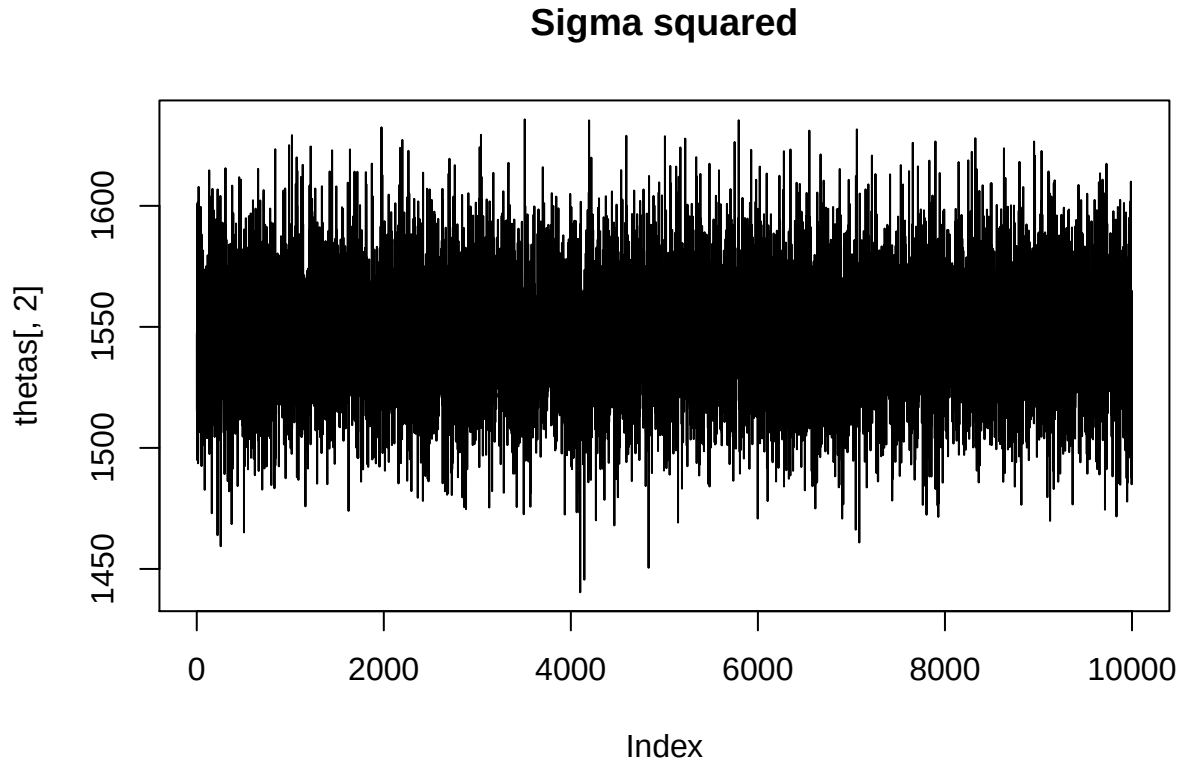
Lab_3_report

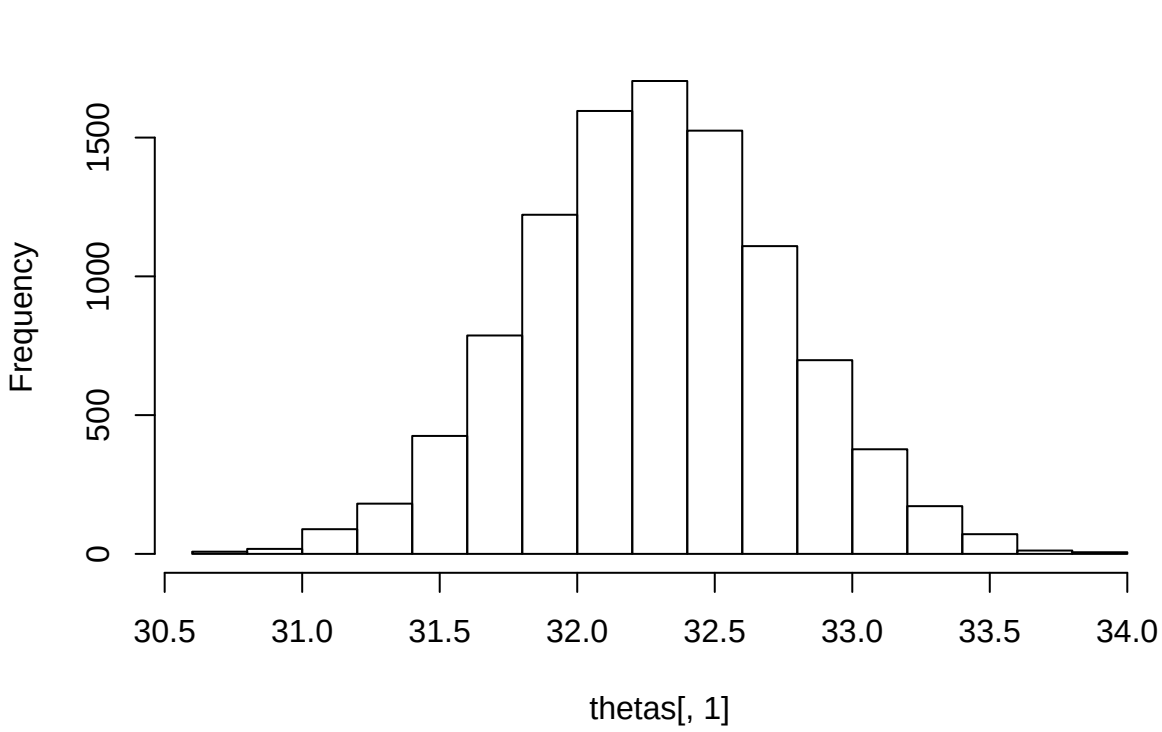
Axel Holmberg (axeho681), Wilhelm Hansson (wilha431)

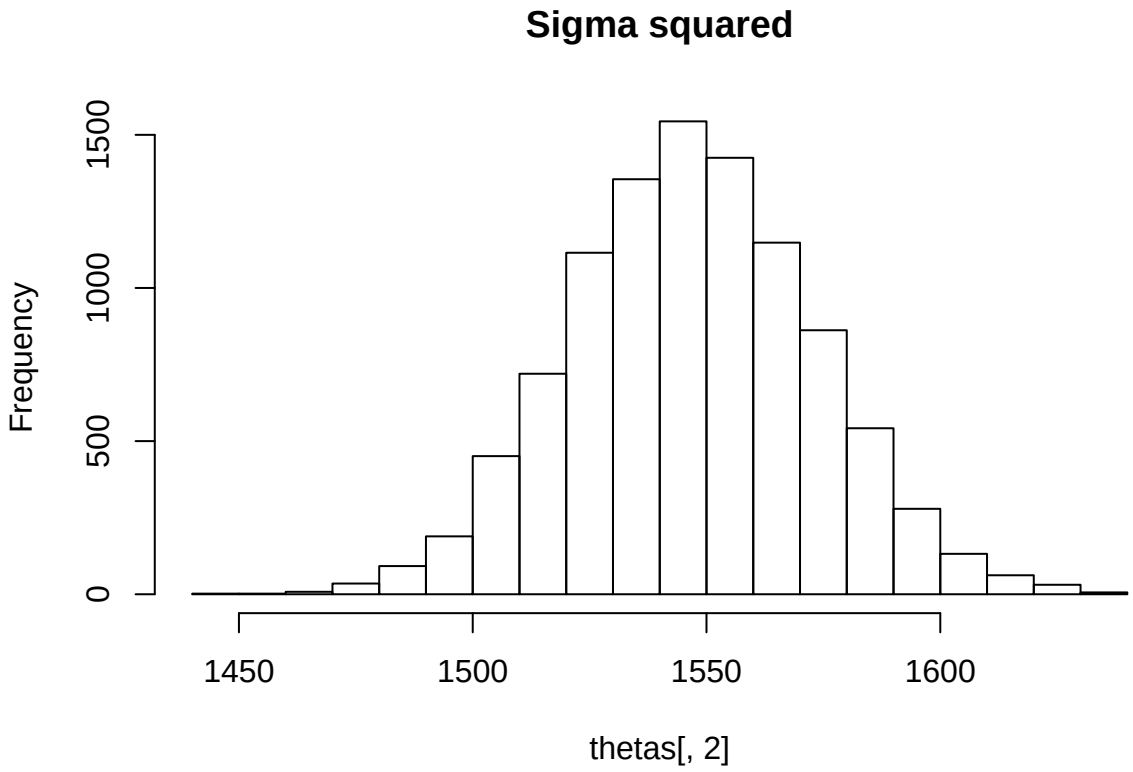
1

a)




Sigma squared
thetas[, 2]
1450
1500
1550
1600
0
2000
4000
6000
8000
10000
Index


Mu
Frequency
0
500
1000
1500
30.5
31.0
31.5
32.0
32.5
33.0
33.5
34.0
thetas[, 1]


Sigma squared
Frequency
0
500
1000
1500
1450
1500
1550
1600
thetas[, 2]

b)

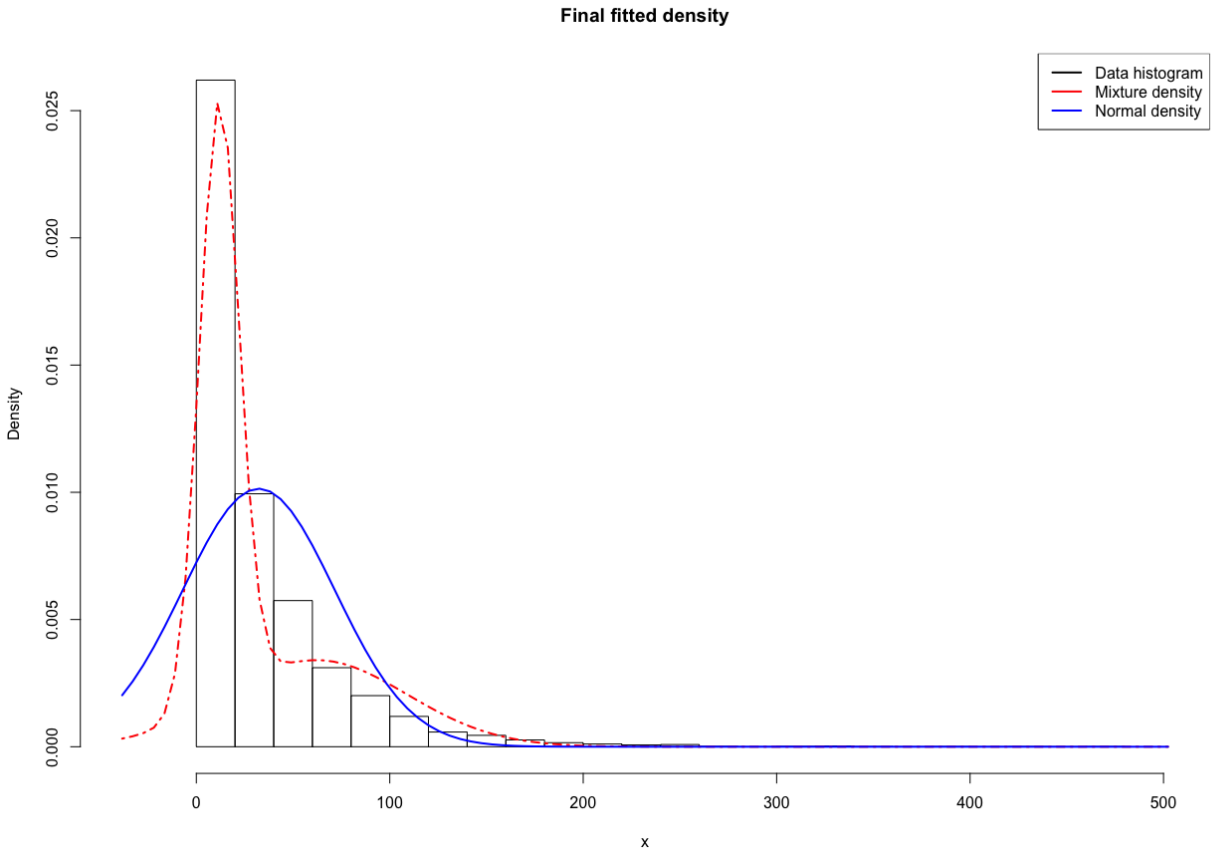


Figure 1: Final Fitted

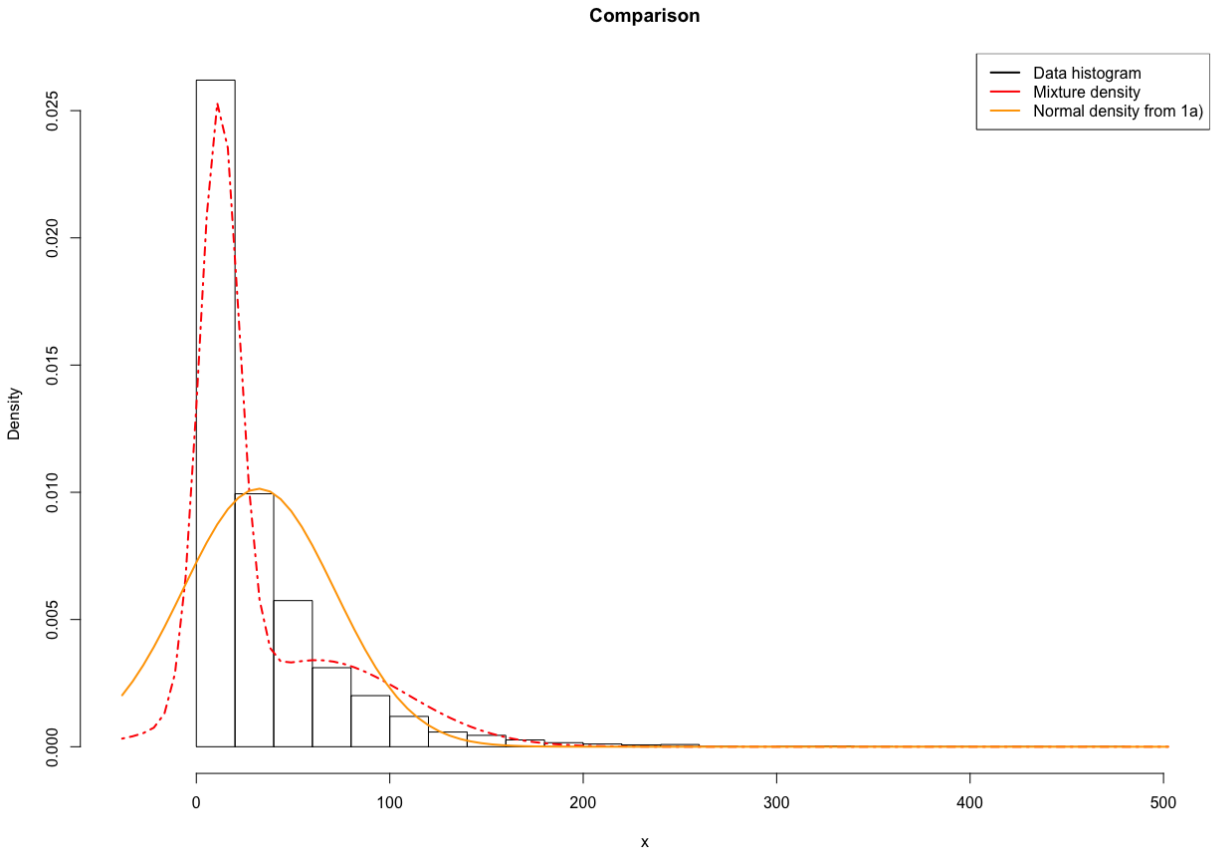


Figure 2: Comparison

Appendix for code

```
data <-read.delim("rainfall.dat", header=FALSE, sep="\n")[,1]

## 1
### a)

#Prior values
mu_0 <- mean(data)
sigma_0 <- sd(data)
tau_0 <- 5
v_0 <- 10
n <- length(data)
iterations <- 10000

#Prepping thetas for gibbs sampling
thetas_col <- c(rep(0,iterations))
thetas <- cbind(thetas_col,thetas_col)
thetas[1,] = c(mu_0, sigma_0^2)

#Function for draws for mu from the conditional posterior
mu_draw_cond_post <- function(sigma_2) {
    tau_n_2 <- 1/(n/sigma_2 + 1/tau_0^2)    #!
    raw <- (n/sigma_2) / (n/sigma_2 + 1/tau_0^2)#!
    mu_n <- raw*mu_0 + (1-raw) * mu_0
    return(rnorm(1,mu_n,sqrt(tau_n_2)))
}

#Fucntion for random inverse chi squared
randominvchisq <- function(vn,sigman,ndraw) {
    return(vn*sigman/rchisq(ndraw,vn))
}

#Function for draws for sigma from the conditional posterior
sigma_draw_cond_post <- function(mu) {
    v_n <-  n + v_0 #!
    sigma_n <- (v_0*sigma_0^2 + sum((data-mu)^2))/v_n #!
    return(randominvchisq(v_n,sigma_n,1))
}

#Function for draws from the h
gibbs_draw <- function(theta_n) {

    mu <- mu_draw_cond_post(theta_n[2])
    sigma_2 <- sigma_draw_cond_post(mu)
    return(c(mu,sigma_2))
}


#Draws for the set number of observations. First is already set above, hence we start at 2
for (i in 2:iterations) {
    thetas[i,] <- gibbs_draw(thetas[i-1,])
}
```

```r
plot(thetas[,1], type="l", main="Mu")
plot(thetas[,2], type="l", main="Sigma squared")

hist(thetas[,1], main="Mu")
hist(thetas[,2], main="Sigma squared")


### b)

##########    BEGIN USER INPUT #################
# Data options

x <- as.matrix(read.delim("rainfall.dat", header=FALSE, sep="\n"))

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

# Prior options
alpha <- 10*rep(1,nComp) # Dirichlet(alpha)
muPrior <- rep(mu_0,nComp) # Prior mean of mu
tau2Prior <- rep(tau_0^2,nComp) # Prior std of mu
sigma2_0 <- rep(var(x),nComp) # s20 (best guess of sigma2)
nu0 <- rep(v_0,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')
sleepTime <- 0.05 # Adding sleep time between iterations for plotting
################   END USER INPUT ################

###### 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
    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(x)
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){
    message(paste('Iteration number:',k))
    alloc <- S2alloc(S) # Just a function that converts between different representations of the group
    nAlloc <- colSums(S)
    #print(nAlloc)
    # 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[al
    }

    # 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

        hist(x, breaks = 20, freq = FALSE, xlim = c(xGridMin,xGridMax), main = paste("Iteration number"
        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
            lines(xGrid, compDens, type = "l", lwd = 2, col = lineColors[j])
            components[j] <- paste("Component ",j)
        }
        mixDensMean <- ((effIterCount-1)*mixDensMean + mixDens)/effIterCount

        lines(xGrid, mixDens, type = "l", lty = 2, lwd = 3, col = 'red')
        legend("topright", box.lty = 1, legend = c("Data histogram",components, 'Mixture'),
                    col = c("black",lineColors[1:nComp], 'red'), lwd = 2)
    #   Sys.sleep(sleepTime)
    }

}

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(



#c)
par(mfrow = c(1, 1))

hist(x, breaks = 20, freq = FALSE, xlim = c(xGridMin,xGridMax), main = "Comparison")
lines(xGrid, mixDensMean, type = "l", lwd = 2, lty = 4, col = "red")
lines(xGrid,dnorm(xGrid, mean(thetas[,1]), sd=mean(sqrt(thetas[,2]))), type = "l", lwd = 2, col = "orang
legend("topright", box.lty = 1, legend = c("Data histogram","Mixture density","Normal density from 1a)"
```