

Bayesian Learning

Lab 3

Emil K Svensson and Rasmus Holm

2017-05-15

Question 1

```
options(digits=2)

rainfall <- read.table("../data/rainfall.dat", header=F)
```

Prior

$$\begin{aligned}\mu &\sim N(\mu_0, \tau_0^2) \\ \sigma^2 &\sim \text{Inv-}\chi^2(\nu_0, \sigma_0^2)\end{aligned}$$

Likelihood

$$\mathbf{y}|\mu, \sigma^2 \sim N(\mu, \sigma^2)$$

Posterior

$$\begin{aligned}\mu|\sigma^2, \mathbf{x} &\sim N(\mu_n, \tau_n^2) \\ \sigma^2|\mu, \mathbf{x} &\sim \text{Inv-}\chi^2\left(\nu_n, \frac{\nu_0\sigma_0^2 + \sum_{i=1}^n (x_i - \mu)^2}{n + \nu_0}\right)\end{aligned}$$

where

$$\begin{aligned}\mu_n &= \frac{\frac{1}{\tau_0^2}}{\frac{1}{\tau_0^2} + \frac{n}{\sigma^2}} \mu_0 + \frac{\frac{n}{\sigma^2}}{\frac{1}{\tau_0^2} + \frac{n}{\sigma^2}} \bar{x} \\ \frac{1}{\tau_n^2} &= \frac{1}{\tau_0^2} + \frac{n}{\sigma^2} \\ \nu_n &= \nu_0 + n\end{aligned}$$

a)

```
library(geoR)

mun <- function(x, sigmasq, hyperparams){
  n <- length(x)
```

```

    denom <- ((1 / hyperparams$tausq0) + (n / sigmasq))

    pt1 <- ((1 / hyperparams$tausq0) / denom) * hyperparams$mu0
    pt2 <- ((n / sigmasq) / denom) * mean(x)

    pt1 + pt2
}

taun <- function(x, sigmasq, hyperparams){
  hyperparams$tausq0 + (sigmasq / length(x))
  1 / hyperparams$tausq0
}

nun <- function(x, hyperparams){
  hyperparams$nu0 + length(x)
}

sigmasqn <- function(x, mu, hyperparams){
  (hyperparams$nu0 * hyperparams$sigmasq0 + sum((x - mu)^2 )) / (length(x) + hyperparams$nu0)
}

musampler <- function(x,sigmasq, hyperparams){
  mu <- mun(x, sigmasq, hyperparams)
  sigma <- sqrt(taun(x,sigmasq, hyperparams))
  rnorm(1, mu, sigma)
}

sigmasampler <- function(x, mu, hyperparams){
  scale <- sigmasqn(x, mu, hyperparams)
  df <- nun(x, hyperparams)
  rinvchisq(1, df, scale)
}

gibbs <- function(x, iter, init, hyperparams){
  samples <- matrix(NA, ncol = 2, nrow = iter + 1)
  samples[1,] <- init

  for (i in 2:(iter+1)){
    mu <- musampler(x, samples[i-1, 2], hyperparams)
    sigma <- sigmasampler(x, mu, hyperparams)
    samples[i,] <- c(mu, sigma)
  }

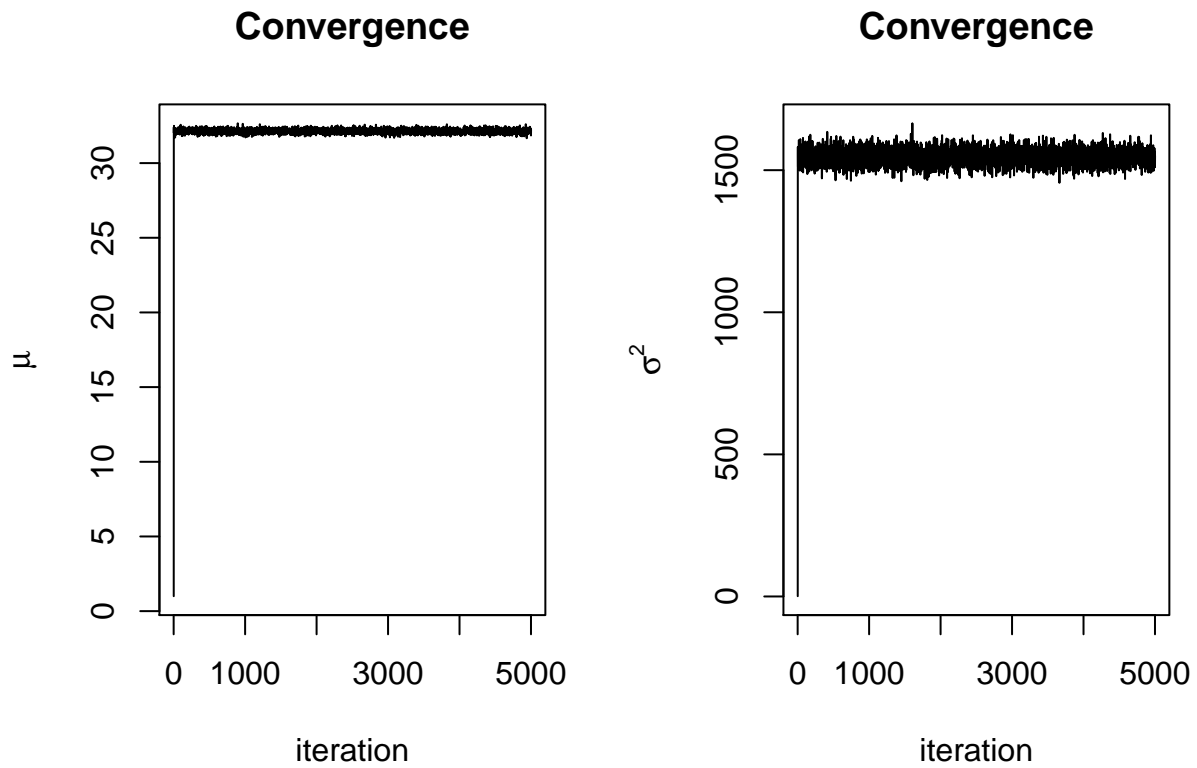
  colnames(samples) <- c("mu", "sigmasq")
  samples
}

set.seed(123456)

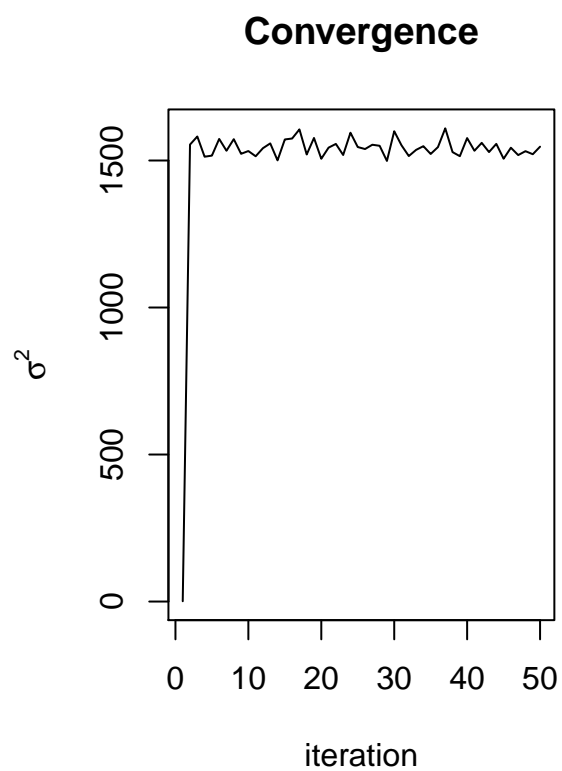
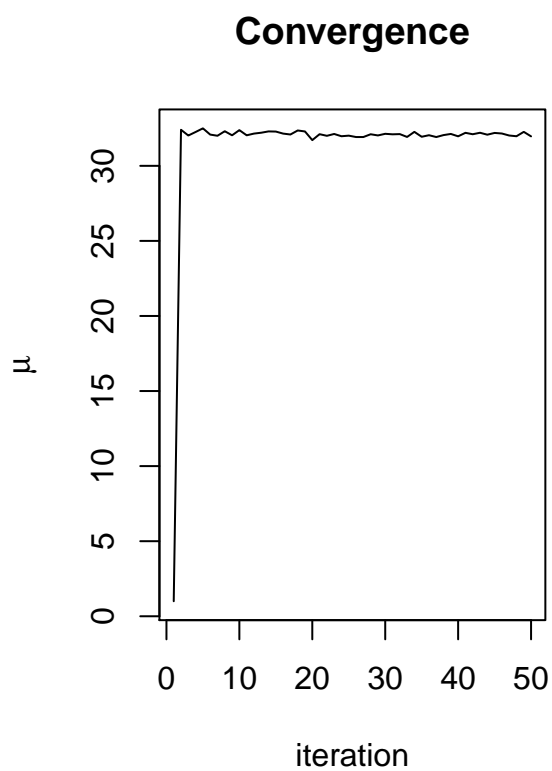
hyperparams <- list(mu0=0, tausq0=50, nu0=1, sigmasq0=50)
iter <- 5000

```

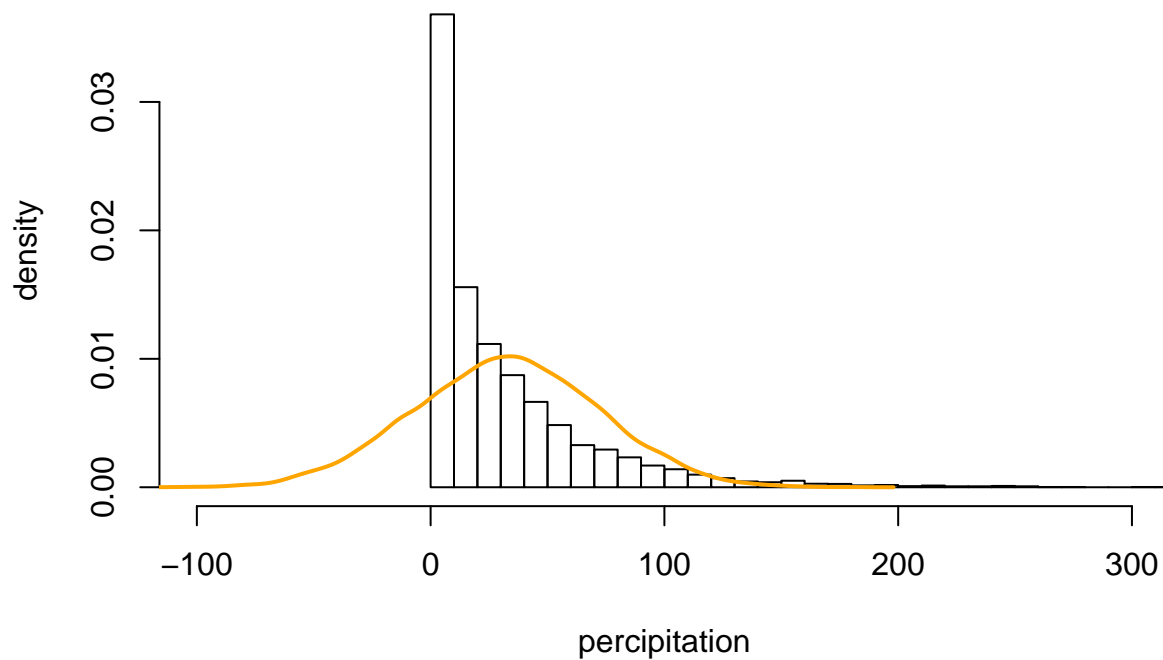
```
gibbs_params <- gibbs(x=rainfall$V1, iter = iter, init = c(1,1), hyperparams)
```



The μ converges to 32.13 which is similar to the data mean, 32.28, and σ^2 converges to 1546.15 which is similar to data variance, 1547.1.



The convergence happens after just a few iterations.



The data is obviously not normally distributed so the model is not particularly good which is seen by the orange curve.

b)

```
nn <- function(I){
  n1 <- sum(I == 1)
  n2 <- sum(I == 2)
  list(n1 = n1, n2 = n2)
}

dd <- function(x, I) {
  d1 <- x[I == 1]
  d2 <- x[I == 2]
  list(d1=d1, d2=d2)
}

pisampler <- function(x, I, hyperparams){
  n <- nn(I)
  rbeta(1, shape1 = n$n1 + hyperparams$a1, shape2 = n$n2 + hyperparams$a2)
}

sigmasq2sampler <- function(x, I, hyperparams){
  n <- nn(I)
  d <- dd(x, I)
```

```

vnsn1 <- hyperparams$nu0 * hyperparams$sigmasq0 +
  (n$n1 - 1) * var(d$d1) +
  (((1 / hyperparams$tausq0) * n$n1) / ((1 / hyperparams$tausq0) + n$n1)) *
  ((mean(d$d1) - hyperparams$mu0)^2)

vnsn2 <- hyperparams$nu0 * hyperparams$sigmasq0 +
  (n$n2 - 1) * var(d$d2) +
  (((1 / hyperparams$tausq0) * n$n2) / ((1 / hyperparams$tausq0) + n$n2)) *
  ((mean(d$d2) - hyperparams$mu0)^2)

vn1 <- hyperparams$nu0 + n$n1
vn2 <- hyperparams$nu0 + n$n2

sn1 <- vnsn1 / vn1
sn2 <- vnsn2 / vn2

sigmasq1 <- rinvchisq(1, vn1, sn1)
sigmasq2 <- rinvchisq(1, vn2, sn2)

c(sigmasq1, sigmasq2)
}

mu2sampler <- function(x, I, sigmasq, hyperparams){
  d <- dd(x, I)

  sigma1 <- sqrt(taun(d$d1, sigmasq[1], hyperparams))
  mu1 <- mun(d$d1, sigmasq[1], hyperparams)

  sigma2 <- sqrt(taun(d$d2, sigmasq[2], hyperparams))
  mu2 <- mun(d$d2, sigmasq[2], hyperparams)

  mu1 <- rnorm(1, mu1, sigma1)
  mu2 <- rnorm(1, mu2, sigma2)

  c(mu1, mu2)
}

Isampler <- function(x, pi, mu, sigmasq){
  nom <- (1 - pi) * dnorm(x, mu[2], sqrt(sigmasq[2]))
  denom <- pi * dnorm(x, mu[1], sqrt(sigmasq[1])) + nom
  theta <- nom / denom
  rbinom(length(x), prob = theta, size = 1) + 1 # to get them into 1 and 2
}

ysampler <- function(pi, mu, sigmasq){
  pi * rnorm(1, mu[1], sqrt(sigmasq[1])) +
  (1 - pi) * rnorm(1, mu[2], sqrt(sigmasq[2]))
}

mixedgibbs <- function(x, iter, init, hyperparams){
  params_samples <- matrix(NA, ncol = 5, nrow = iter)
  samples <- rep(NA, iter)
  I <- init

```

```

for (i in 1:iter){
  pi <- pisampler(x, I, hyperparams)
  sigmasq <- sigmasq2sampler(x, I, hyperparams)
  mu <- mu2sampler(x, I, sigmasq, hyperparams)
  I <- Isampler(x, pi, mu, sigmasq)

  samples[i] <- ysampler(pi, mu, sigmasq)
  params_samples[i,] <- c(mu, sigmasq, pi)
}

colnames(params_samples) <- c("mu1", "mu2", "sigmasq1", "sigmasq2", "pi")
list(samples=samples, params=params_samples)
}

```

```
set.seed(123456)
```

```

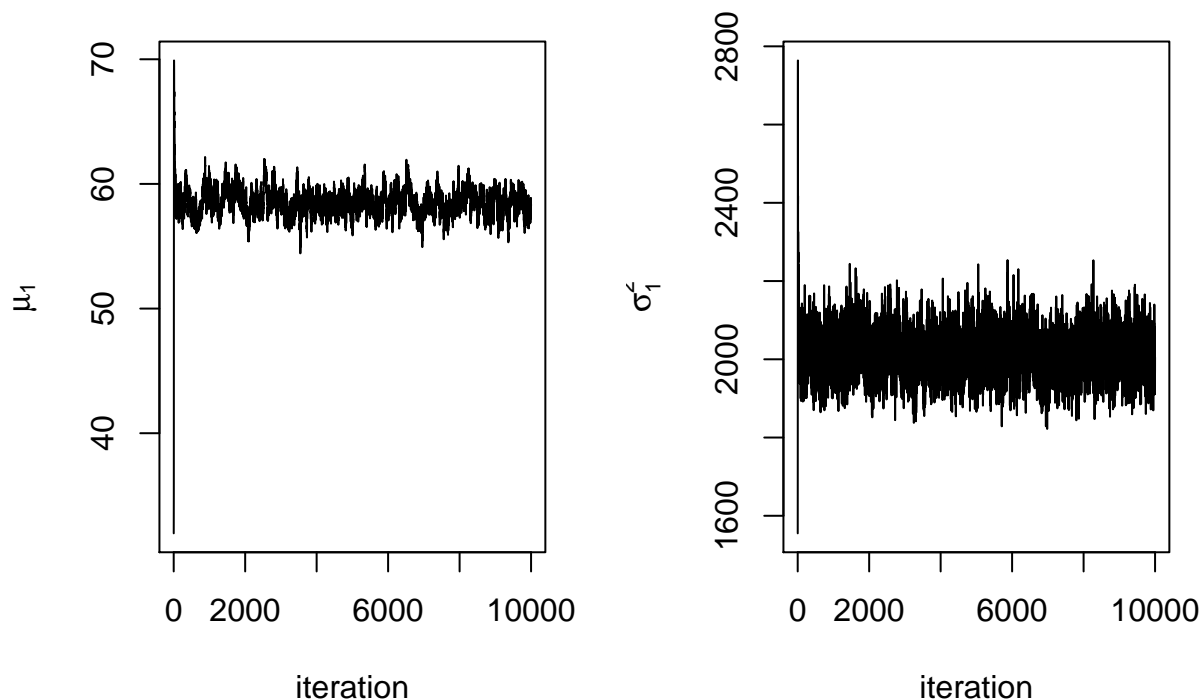
hyperparams <- list(a1=1, a2=1, mu0=1, tausq0=50, nu0=1, sigmasq0=50)
init <- sample(c(1, 2), size=length(rainfall$V1), replace=TRUE)
iter <- 10000

```

```

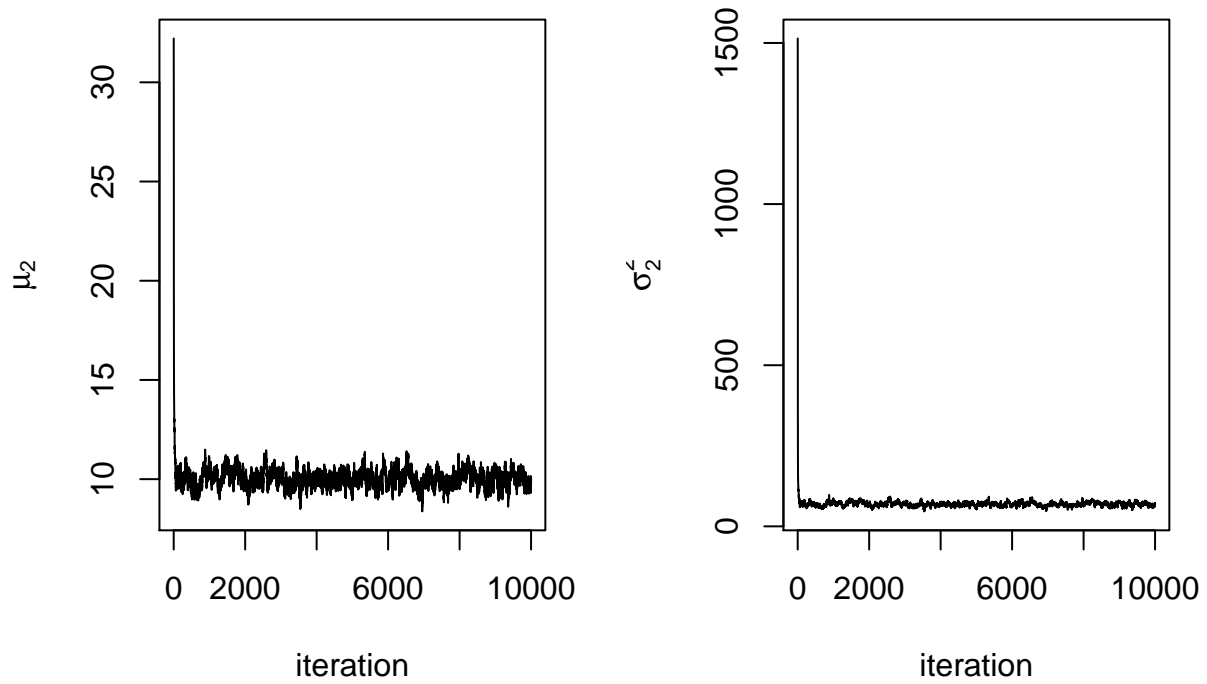
mixed <- mixedgibbs(rainfall$V1, iter, init, hyperparams)
mixed_params <- mixed$params
mixed_samples <- mixed$samples

```

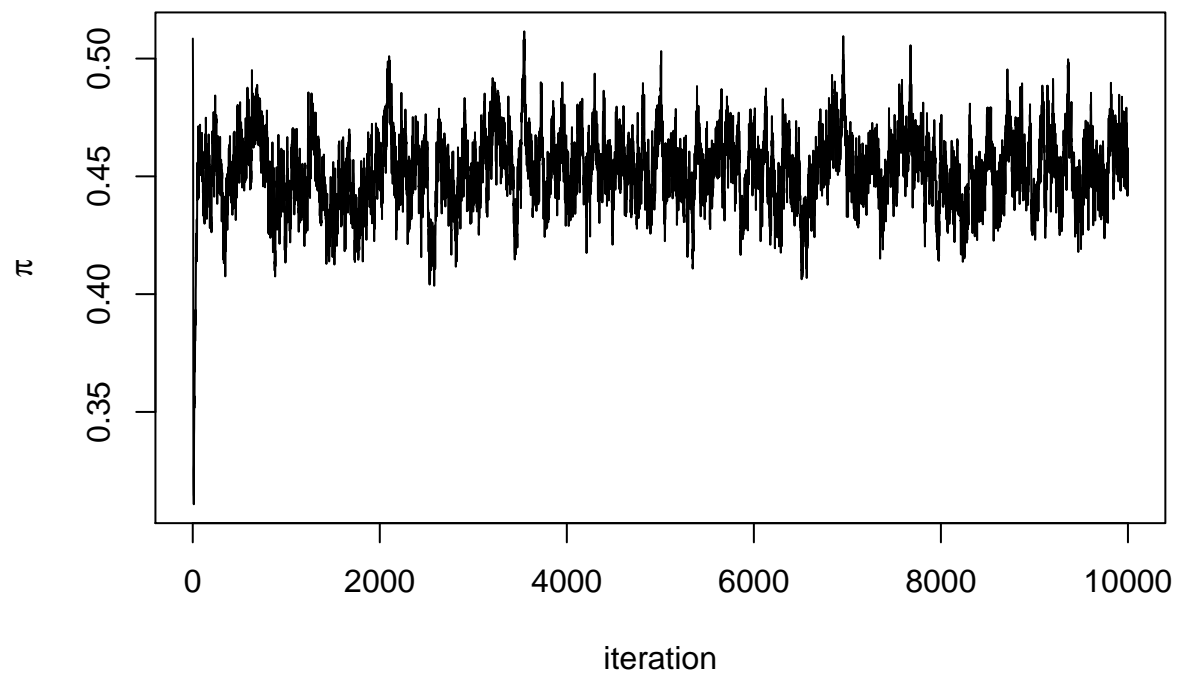


The first normal distribution have settled at mean of around 60 and variance of 2000. We can see that it has larger variance in the parameters than the previous model and for the mean it is noticeable autocorrelation

between iterations.

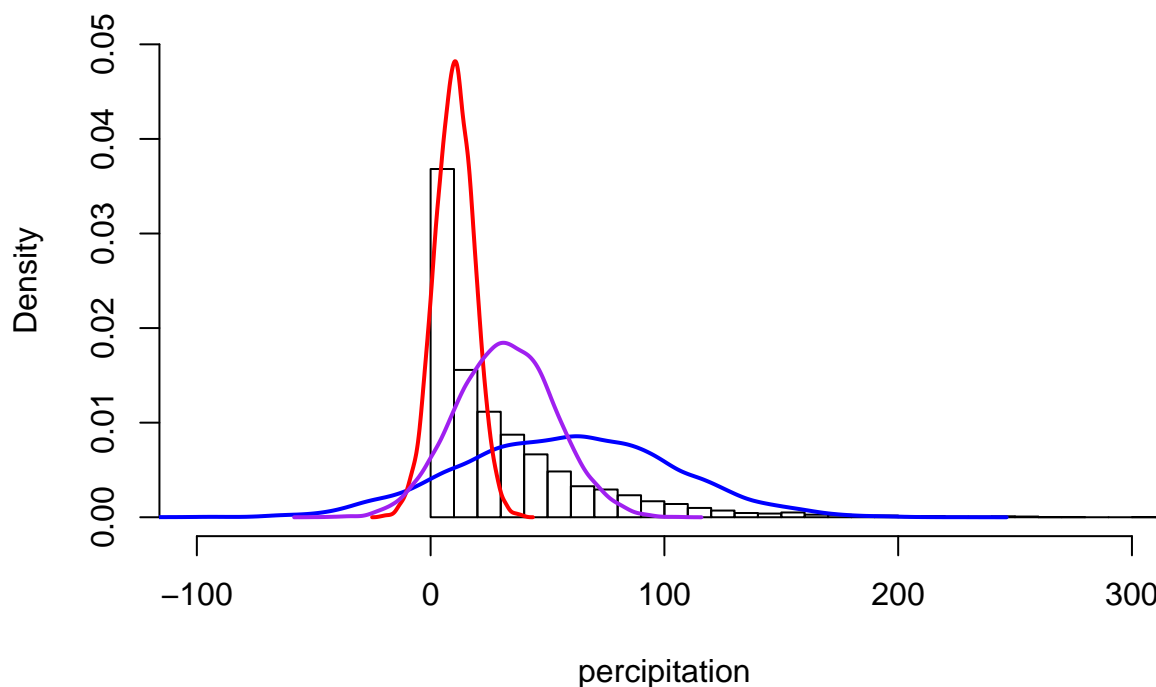


The second normal distribution have a mean of approximately 10 and variance of around 50. Similarly as before we can see autocorrelation which indicate that it would probably be beneficial to run the algorithm for longer to get a larger effective sample size that is comparable to the simple normal model in *1a*.



The π is the probability of the observation to be from the first normal distribution and we can see that it is around 45%.

Densities of Normals



We can see that the two Gaussian distributions in the mixture model represent low values with high density, the red curve, and those with higher values with a wider distribution, the blue curve. Then the model, the purple curve, is an average of those aforementioned distributions. However, as mention before the data is not normally distributed so the final model is far from perfect.

c)

```
n <- 10000
burnins <- 1000 + 1

gibbs_params <- gibbs_params[-(1:burnins),]
mixed_params <- mixed_params[-(1:burnins),]

hist(rainfall$V1, freq=FALSE, breaks=50, xlim=c(-100, 200), main="", xlab="percipitation")

## Model A
mu <- mean(gibbs_params[, "mu"])
sigmasq <- mean(gibbs_params[, "sigmasq"])

samples <- rnorm(n, mu, sqrt(sigmasq))

lines(density(samples), lwd=2, col="orange")

## Model B
mu1 <- mean(mixed_params[, "mu1"])
```

```

sigmasq1 <- mean(mixed_params[, "sigmasq1"])

mu2 <- mean(mixed_params[, "mu2"])
sigmasq2 <- mean(mixed_params[, "sigmasq2"])

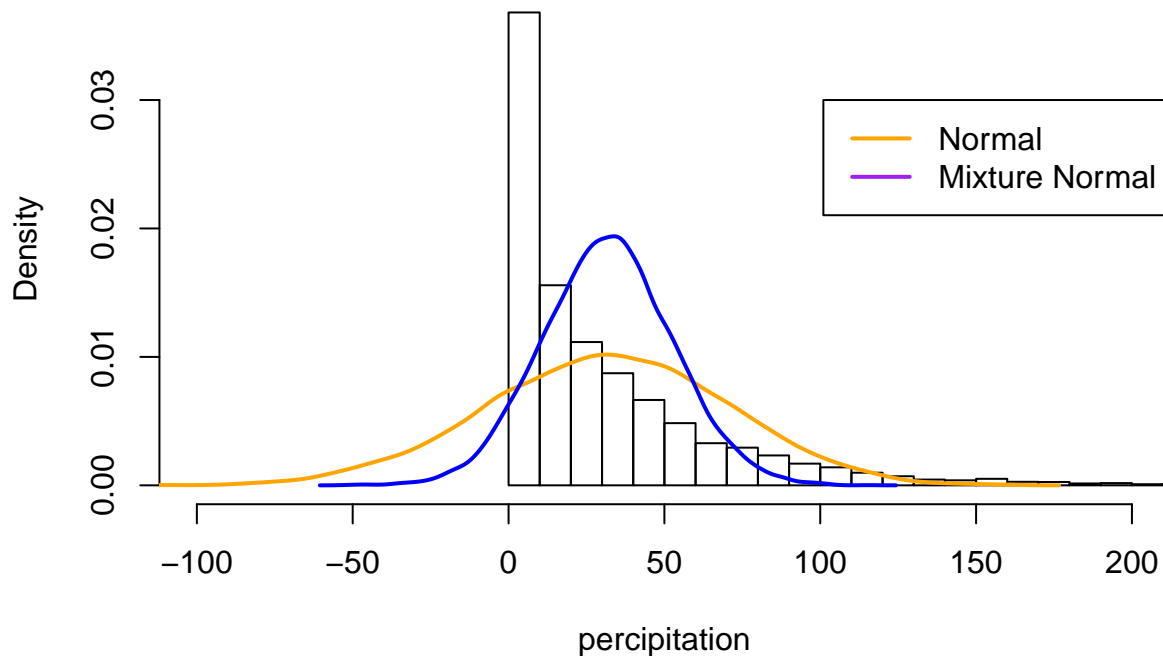
pi <- mean(mixed_params[, "pi"])

samples <- pi * rnorm(n, mu1, sqrt(sigmasq1)) + (1 - pi) * rnorm(n, mu2, sqrt(sigmasq2))

lines(density(samples), lwd=2, col="blue")

legend(101, 0.03, lty=c(1, 1), lwd=c(2, 2), col=c("orange", "purple"),
      legend=c("Normal", "Mixture Normal"))

```



Comparing the normal model and the mixture of normals we conclude that the latter is better in this case but it would probably have been better to choose either a truncated Normal or exponential distribution as the model since we know that precipitation cannot be negative.

Question 2

```
women <- read.table("../data/WomenWork.dat", header = T)
women <- as.matrix(women)
```

a)

```
library(msm)
library(mvtnorm)

betahat <- function(XX, X, y){
  solve(XX) %*% t(X) %*% y
}

mun <- function(XX, betahat, mu0, omega0){
  solve( XX + omega0 ) %*% (XX %*% betahat - omega0 %*% mu0)
}

probitgibbs <- function(X, Y, mu0, omega0, beta0, iter){
  samples <- matrix(ncol = ncol(X), nrow = iter)
  y0 <- Y == 0
  y0n <- sum(y0)
  y1n <- sum(!y0)
  XX <- t(X) %*% X
  omegan <- XX + omega0
  u <- c(rep(1, times = length(Y)))
  beta <- beta0

  for (i in 1:iter){

    xB <- X %*% beta

    u[y0] <- rtnorm(n = y0n, mean = xB[y0], sd = 1, lower = -Inf, upper = 0)
    u[!y0] <- rtnorm(n = y1n, mean = xB[!y0], sd = 1, lower = 0, upper = Inf)

    baehat <- betahat(XX, X, u)
    myn <- mun(XX, baehat, mu0, omega0)

    beta <- as.vector(rmvnorm(1, mean = myn, sigma = 1 * solve(omegan) ))
    samples[i,] <- beta
  }
  samples
}
```

b)

```
betas <- probitgibbs(X = women[, -1], Y = women[, 1],
                    mu0 = rep(0, times = ncol(women[, -1])),
                    omega0 = 100 * diag(ncol(women[, -1])),
```

```

        beta = rep(0, times = ncol(women[,-1])), iter = 10000)
probitbetas <- colMeans(betas)

```

c)

```

logprior <- function(beta, mean, sigma){
  dmvmnorm(beta, mean = mean, sigma = sigma, log = TRUE)
}

loglikelihood <- function(beta, X, Y){
  linear_prediction <- t(X) %*% beta
  P <- pnorm(linear_prediction, mean = 0, sd = 1)
  probabilities <- Y * log(P) + (1 - Y)*log(1 - P)
  loglike <- sum(probabilities)

  ## if (abs(loglike) == Inf)
  ##   loglike = -20000

  loglike
}

logposterior <- function(beta, X, Y, mean, sigma){
  loglikelihood(beta, X, Y) + logprior(beta, mean, sigma)
}

tau <- 10
mu <- rep(0,8)
sigma <- tau^2 * diag(8)

womenX <- as.matrix(women[,2:ncol(women)])
womenY <- as.matrix(women[,1])

quadapprox <- optim(par = matrix(rep(0, 8), ncol = 1),
  fn = logposterior, method = "BFGS", hessian = TRUE,
  X = t(womenX), Y = womenY,
  mean = mu, sigma = sigma,
  control=list(fnscale=-1))

mu <- quadapprox$par
sigma <- -solve(quadapprox$hessian)
quadbetas <- rmvnorm(nrow(betas), mean=mu, sigma=sigma)

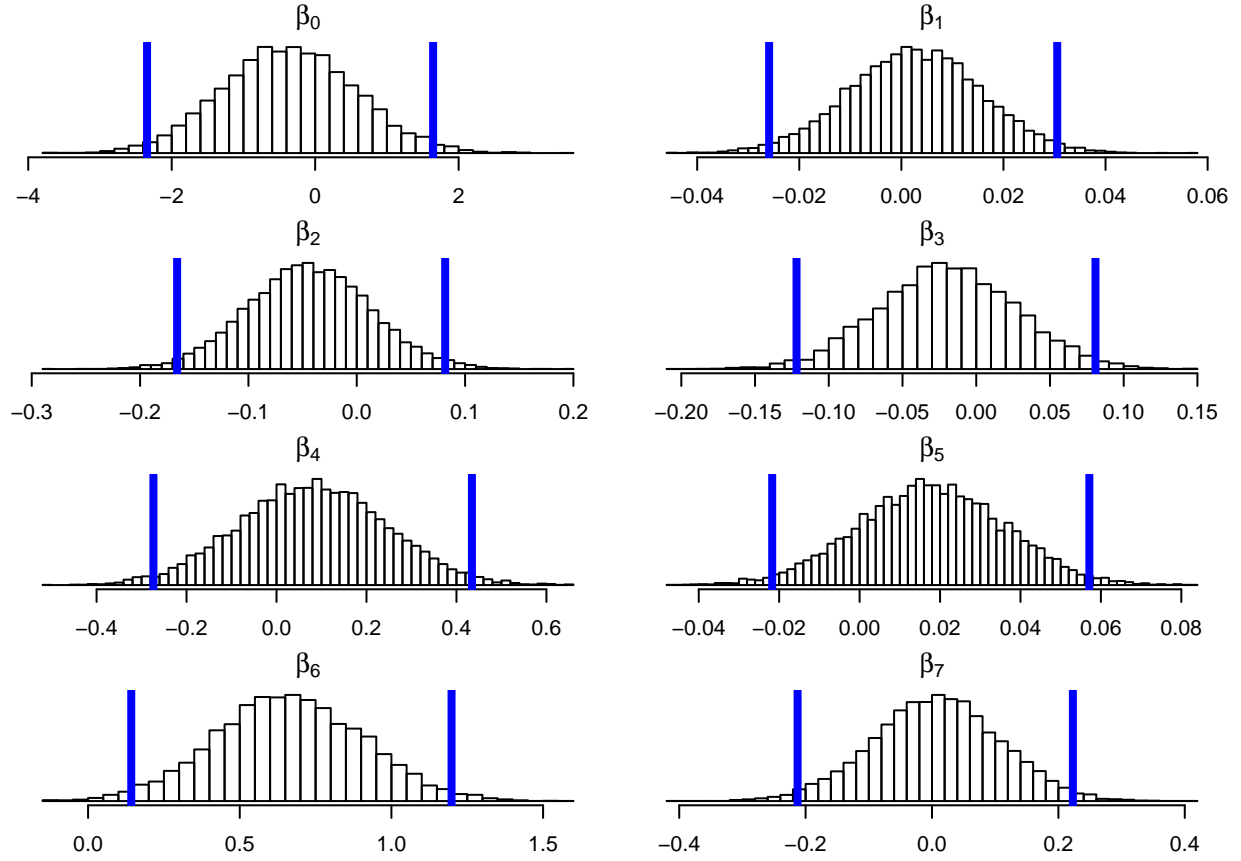
options(digits=3)

probitbetas

## [1] -0.00572 -0.00944  0.06510  0.08245 -0.00918 -0.03144 -0.15049 -0.00455
as.vector(quadapprox$par)

## [1]  0.3845 -0.0121  0.1084  0.1022 -0.0902 -0.0496 -0.8157 -0.0124

```



The histograms above show the distributions for the difference, $\beta_i^{\text{Gibbs}} - \beta_i^{\text{Norm}}$ for $i = 0, \dots, 7$, and the blue lines are the 97% credibility intervals. We can see that only in the β_6 parameter that 0 is not within the credibility interval which indicate that there is a difference in that parameter but not by much. However, the rest of the parameters do not differ statistically so we would conclude that the quadratic (normal) approximation is a decent approximation in this case.