HW5 Solution

Jitong

2023-04-11

Question 2

a)

```
azdiabetes <- read.table('https://www2.stat.duke.edu/courses/Fall09/sta290/datasets/Hoffdata/azdiabetes
azdiabetes <- azdiabetes[, -8]
```

```
set.seed(2)
y = azdiabetes$glu
X = azdiabetes[,c('npreg','bp','skin','bmi','ped', 'age')]
n = length(y)
X = cbind(rep(1,n), X)
X = as.matrix(X)
y = as.vector(y)
g <- dim(X)[1]
nu0 <- 2
s20 <- 1
S <- 1000

p <- dim(X)[2]
Hg <- (g / (g+1)) * X %*% solve(t(X) %*% X) %*% t(X)
SSRg <- t(y)%*%(diag(1, nrow=n) - Hg)%*%y
s2 <- 1/rgamma(S, (nu0 + n)/2, (nu0 * s20 + SSRg)/2)
Vb <- g * solve(t(X)%*%X) / (g+1)
Eb <- Vb%*%t(X)%*%y
E <- matrix(rnorm(S*p, 0, sqrt(s2)), S, p)
beta <- t(t(E%*%chol(Vb)) + c(Eb))
beta_post_ci <- apply(beta, 2, quantile, c(0.025, 0.975))
sigma_post_ci <- quantile(s2, c(0.025, 0.975))
beta_post_ci
```

```
##       rep(1, n)      npreg          bp       skin      bmi       ped       age
## 2.5%   35.27405 -1.6111930 -0.01755559 -0.1242831 0.154386  3.210932 0.4440472
## 97.5%  69.17766  0.3440675  0.44085573  0.5043966 1.110427 17.748936 1.0729981
```

```
sqrt(sigma_post_ci)
```

```
##     2.5%    97.5%
## 27.26600 30.82394
```

b)

```
lpy.X = function(y, X, g =length(y), nu0=2, s20=1){
  n = dim(X)[1]
```

```r
  p = dim(X)[2]
  if(p == 0){
    Hg=0 ; s20=mean(y^2)
  }
  if(p>0){
    Hg = (g/(g+1)) * X%*%solve(t(X)%*%X)%*%t(X)
  }
  SSRg = t(y)%*%( diag(1,nrow=n) - Hg )%*%y

  -.5*( n*log(pi)+p*log(1+g)+(nu0+n)*log(nu0*s20+SSRg)- nu0*log(nu0*s20) ) +
    lgamma( (nu0+n)/2 ) - lgamma(nu0/2)
}
# starting values and MCMC setup
set.seed(2)
z = rep(1, dim(X)[2])
lpy.c = lpy.X(y, X[, z == 1, drop=FALSE])
S = 1000
Z = matrix(NA,S,dim(X)[2])
beta_ratio = matrix(NA, S, p)
colnames(beta_ratio) = colnames(X)
# Gibbs sampler
for(s in 1:S){
  for(j in sample(1:dim(X)[2])){
    zp = z
    zp[j]=1-zp[j]
    lpy.p = lpy.X(y, X[,zp==1,drop=FALSE])
    r = (lpy.p - lpy.c)*(-1)^(zp[j]==0)
    z[j] = rbinom(1,1, prob = 1/(1 + exp(-r)))
    if(z[j]==zp[j]){
      lpy.c=lpy.p
    }
  }
  Z[s,]=z
  # generate beta based on z
  X_tmp = X[, z==1]
  p_tmp = dim(X_tmp)[2]
  Hg = (g/(g+1)) * X_tmp%*%solve(t(X_tmp)%*%X_tmp)%*%t(X_tmp)
  SSRg = t(y)%*%( diag(1,nrow=n) - Hg ) %*%y
  s2 = 1/rgamma(1, (nu0+n)/2, (nu0*s20+SSRg)/2 )
  Vb = g*solve(t(X_tmp)%*%X_tmp)/(g+1)
  Eb = Vb%*%t(X_tmp)%*%y
  E = matrix(rnorm(p_tmp, 0, sqrt(s2)), 1, p_tmp)
  beta_part = t( t(E%*%chol(Vb)) +c(Eb))
  # put it together
  indicator = z == 1
  beta_all = c()
  pnter = 1
  for(k in 1:length(indicator)){
    if(indicator[k] == TRUE){
      beta_all[k] = beta_part[pnter]
      pnter = pnter + 1
    }else{
      beta_all[k] = 0
```

```
    }
  }
  beta_ratio[s,] = beta_all
}
```

```
for(i in 1:p){
  if(i == 1){
    print("Intercept:")
  }else{
    print(paste("Variable", colnames(X)[i]))
  }
  z_tmp = Z[, i]
  pb_0 = mean(1-z_tmp)
  print(paste0("The probability the coef is 0 is:", pb_0))
  beta_tmp = beta_ratio[, i]
  qtl = quantile(beta_tmp, prob = c(0.025, 0.975))
  print(paste0("The 95% C.I. in this case is (", qtl[1], ", ", qtl[2], ")"))
}
```

```
## [1] "Intercept:"
## [1] "The probability the coef is 0 is:0"
## [1] "The 95% C.I. in this case is (42.7813780455399, 75.2742202681002)"
## [1] "Variable npreg"
## [1] "The probability the coef is 0 is:0.91"
## [1] "The 95% C.I. in this case is (-0.873688372082157, 0)"
## [1] "Variable bp"
## [1] "The probability the coef is 0 is:0.8"
## [1] "The 95% C.I. in this case is (0, 0.33955786412059)"
## [1] "Variable skin"
## [1] "The probability the coef is 0 is:0.912"
## [1] "The 95% C.I. in this case is (0, 0.346016092209648)"
## [1] "Variable bmi"
## [1] "The probability the coef is 0 is:0.014"
## [1] "The 95% C.I. in this case is (0.471099006206528, 1.32823613059596)"
## [1] "Variable ped"
## [1] "The probability the coef is 0 is:0.312"
## [1] "The 95% C.I. in this case is (0, 16.8662206586243)"
## [1] "Variable age"
## [1] "The probability the coef is 0 is:0"
## [1] "The 95% C.I. in this case is (0.478673944016649, 1.01162462692109)"
```

Question 3

```
data_p3 = read.table("http://www2.stat.duke.edu/~pdh10/FCBS/Exercises/msparrownest.dat", header = FALSE
head(data_p3)
```

```
##   V1    V2
## 1  0 13.03
## 2  1 13.69
## 3  1 12.62
## 4  0 11.70
## 5  0 12.39
## 6  0 12.44
```

```
y = data_p3[, 1]
X = data_p3[, 2]
n = length(X)[1]
X = cbind(rep(1,n), X)
```

c)

```
likelihood <- function(X, y, params){
  e <- exp(X %*% params)
  px <- e/(1+e)
  return(px^y * ((1-px)^(1-y)))
}
prior <- function(params){
  return(dnorm(params[1], mean=0, sd=10)*dnorm(params[2], mean=0, sd=2))
}
# starting values
log_posterior <- function(X, y, params){
  log_prior <- log(prior(params))
  log_likelihoods = log(likelihood(X, y, params))
  return(log_prior + sum(log_likelihoods))
}
Metro_logreg <- function(X, y, n, params_start=c(0,0), jd_alpha=1, jd_beta=1){
  B <- ncol(X)
  PARAMS <- matrix(nrow=n, ncol=B)
  PARAMS[1,] <- params_start
  for(i in 2:n){
    current_betas <- PARAMS[i-1,]
    new_betas <- current_betas + c(rnorm(1, mean=0, sd=jd_alpha), rnorm(1, mean=0, sd=jd_beta))
    rr <- log_posterior(X, y, new_betas) - log_posterior(X, y, current_betas)
    if(log(runif(1)) < rr){
      PARAMS[i,] <- new_betas
    }else{
      PARAMS[i,] <- current_betas
    }

  }
  return(PARAMS)
}

N = 10000 * 20
samples = Metro_logreg(X, y, n=N, params_start = c(0, 0),
                      jd_beta = 0.25, jd_alpha = 4)

library(coda)
effectiveSize(samples[,1])
```

```
##     var1
## 1232.331
```

```
effectiveSize(samples[,2])
```

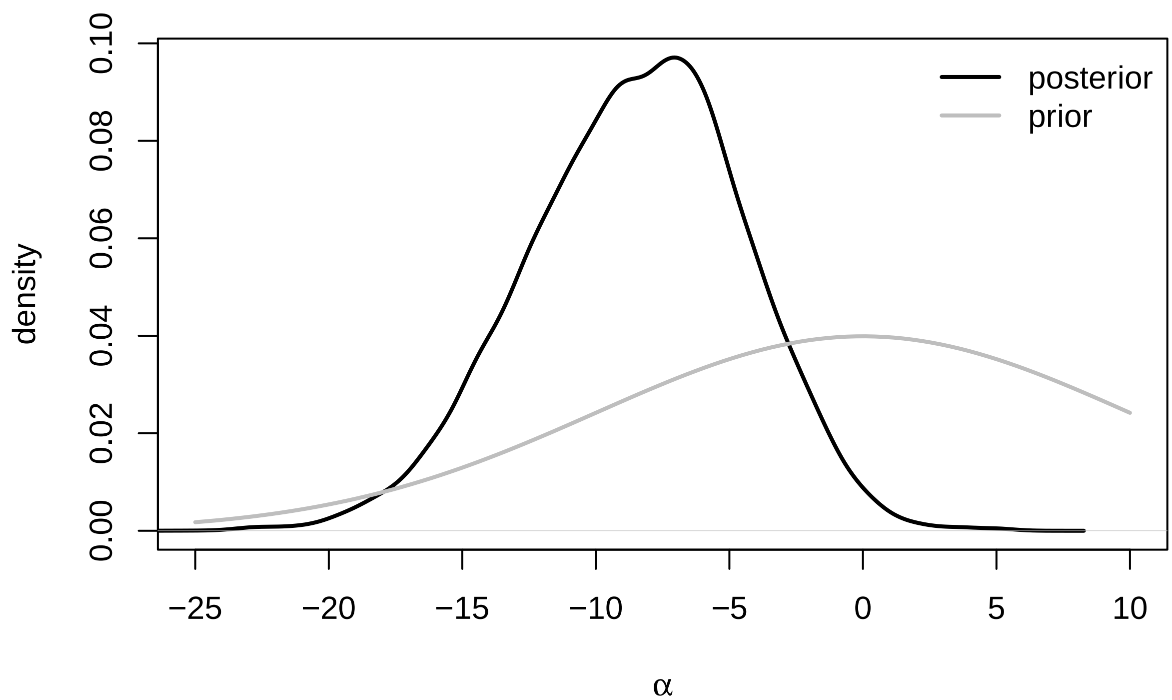
```
##     var1
## 1220.565
```

d)

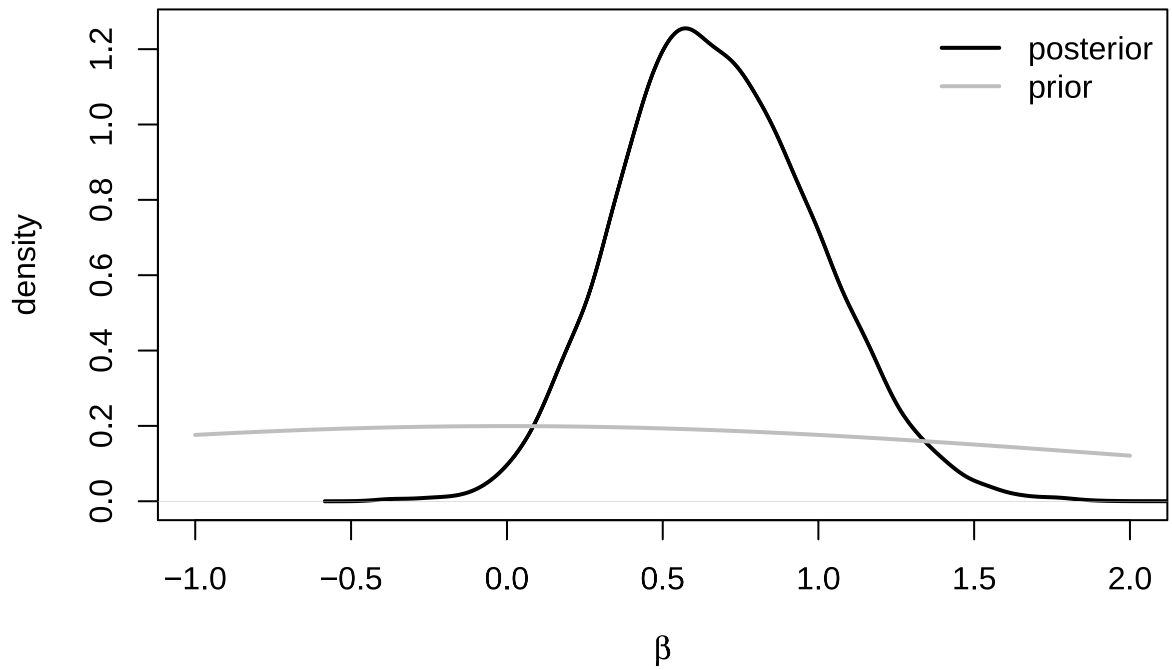
```r
results <- apply(samples, MARGIN=2, function(x){c(mean(x), quantile(x, c(0.025, 0.5, 0.975)))})
results
```

```
##             [,1]      [,2]
##        -8.539547 0.6801612
## 2.5%  -16.812108 0.1101769
## 50%    -8.352706 0.6631189
## 97.5%  -1.196766 1.3176939
```

```r
plot(density(samples[,1],adj=2), xlim = c(-25,10),
main="",xlab=expression(alpha),ylab="density",lwd=2 )
ds<-seq(-25,10,length=100)
lines(ds,dnorm(ds,0,10),lwd=2,col="gray")
legend("topright",legend=c("posterior","prior"),lwd=c(2,2),col=c("black","gray"), bty="n")
```

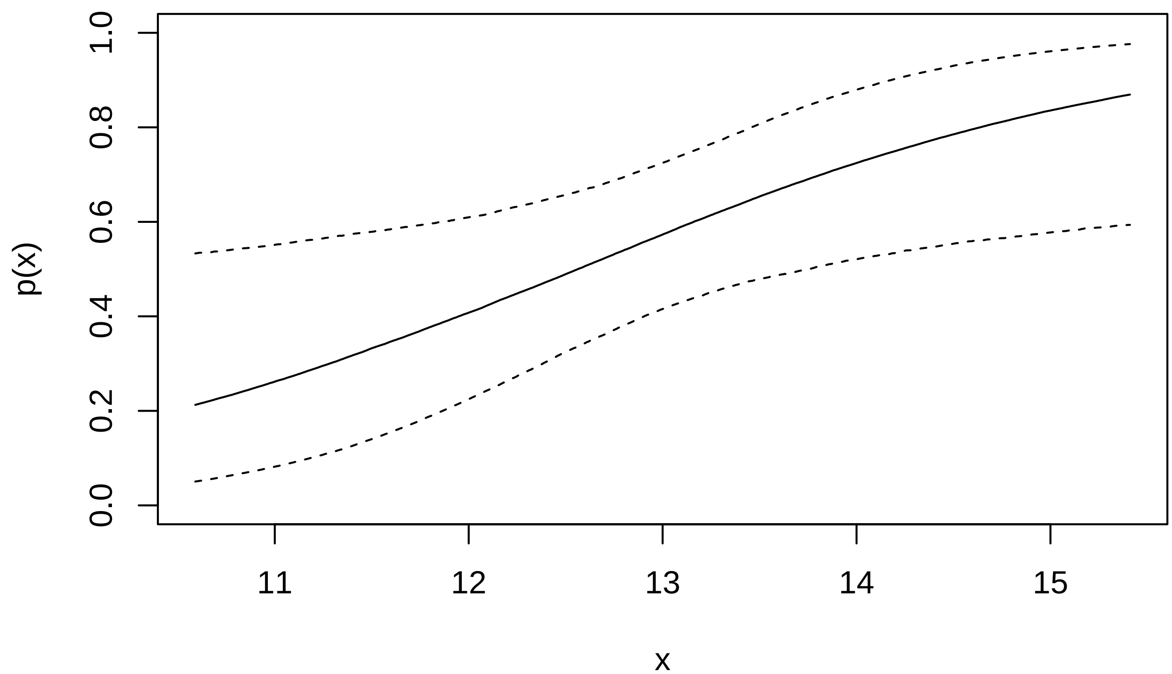



```r
plot(density(samples[,2],adj=2), xlim = c(-1,2),
main="",xlab=expression(beta),ylab="density",lwd=2 )
ds<-seq(-1,2,length=100)
lines(ds,dnorm(ds,0,2),lwd=2,col="gray" )
legend("topright",legend=c("posterior","prior"),lwd=c(2,2),col=c("black","gray"),
bty="n")
```



e)

```
x_grid = seq(min(X[,2]), max(X[,2]), length.out = 1000)
alpha = samples[,1]
beta = samples[,2]
n = length(x_grid)
confi = matrix(nrow = n, ncol= 3)
for(i in 1:length(x_grid)){
  x = x_grid[i]
  lincomb = x*beta + alpha
  p_tmp = (exp(lincomb))/(1+exp(lincomb))
  confi[i,] = quantile(p_tmp, c(0.025, 0.5, 0.975))
}
plot(x_grid, confi[,1], type = 'l', lty = 2, xlab = "x", ylab = "p(x)", ylim = c(0,1))
lines(x_grid, confi[,2])
lines(x_grid, confi[,3], lty = 2)
```



Question 4

```
data_p4 = read.table('https://data.giss.nasa.gov/gistemp/graphs/graph_data/Global_Mean_Estimates_based_o
colnames(data_p4) <- c('Year', 'No_Smoothing', 'Lowess')
summary(lm(No_Smoothing ~ Year, data = data_p4))
```

```
##
## Call:
## lm(formula = No_Smoothing ~ Year, data = data_p4)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.36134 -0.13366 -0.02528  0.12775  0.44584
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.505e+01  7.094e-01  -21.22   <2e-16 ***
## Year         7.747e-03  3.635e-04   21.31   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1795 on 141 degrees of freedom
## Multiple R-squared:  0.763,  Adjusted R-squared:  0.7614
## F-statistic:   454 on 1 and 141 DF,  p-value: < 2.2e-16
```

a)

```r
y = data_p4[,2]
X = data_p4[,1]
# With Rstan:
library(rstan)
write(
"data {
  int N; // number of observations
  // response
  vector[N] y;
  // number of columns in the design matrix X
  vector[N] X;
  real scale_alpha; // prior sd on alpha
  real scale_beta; // prior sds on betas
  real loc_sigma;
}
parameters {
  real alpha; // intercept
  real beta; // regression coefficients beta vector
  real sigma;
}
// this is a convenient way to utilize the fact the mean of Y depends on x
  transformed parameters {
  vector[N] mu; // defines the mean of each Y
  mu = alpha + X * beta; //notice * is vector multiplication
}
model {
  // priors
  alpha ~ normal(0, scale_alpha);
  beta ~ normal(0, scale_beta); // notice the beta priors are independent
  // to model correlated betas you can use lkj_corr prior
  sigma ~ exponential(loc_sigma); // this does not match the textbook
  y ~ normal(mu, sigma); // likelihood
}
", "Example4.stan")
mod = stan_model("Example4.stan")
mod_data = list(
  X = X,
  N = length(y),
  y = y
)
mod_data$scale_alpha = 150
mod_data$scale_beta <- 150
mod_data$loc_sigma <- sd(y)
mod_fit = sampling(mod, data = mod_data, iter = 3000)
```

```
## Warning: There were 1190 transitions after warmup that exceeded the maximum treedepth. Increase max_t
## https://mc-stan.org/misc/warnings.html#maximum-treedepth-exceeded
```

```
## Warning: Examine the pairs() plot to diagnose sampling problems
```

```r
summary(mod_fit, pars = c("alpha", "beta", "sigma"), probs = c(0.025, 0.975))$summary
```

```
##                mean      se_mean           sd          2.5%        97.5%
```

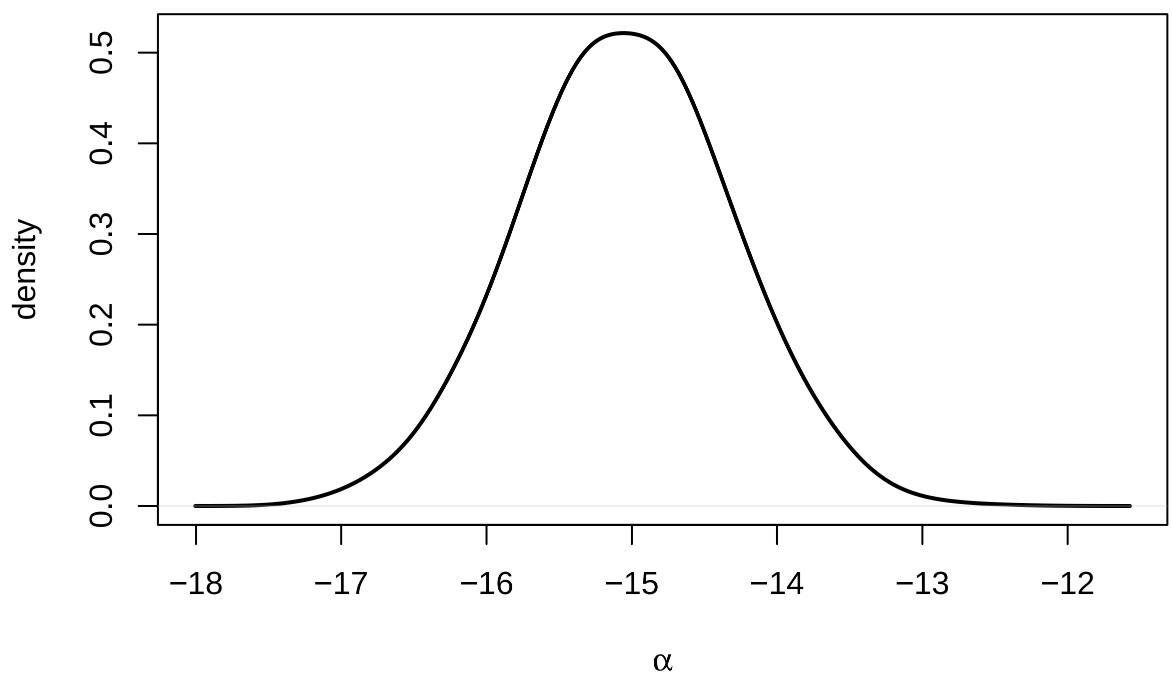
```
## alpha -15.04347359 1.612727e-02 0.7134149262 -16.452680242 -13.64695320
## beta    0.00774182 8.273269e-06 0.0003654234   0.007024605   0.00846196
## sigma   0.18100353 2.259575e-04 0.0106506728   0.162022825   0.20320568
##          n_eff     Rhat
## alpha 1956.873 1.002482
## beta  1950.915 1.002474
## sigma 2221.777 1.001060
```

b)

From the output of Rstan the mean posterior of the slope is 0.0077, which means for each year global surface temperature increase by 0.0077 degrees.

c)

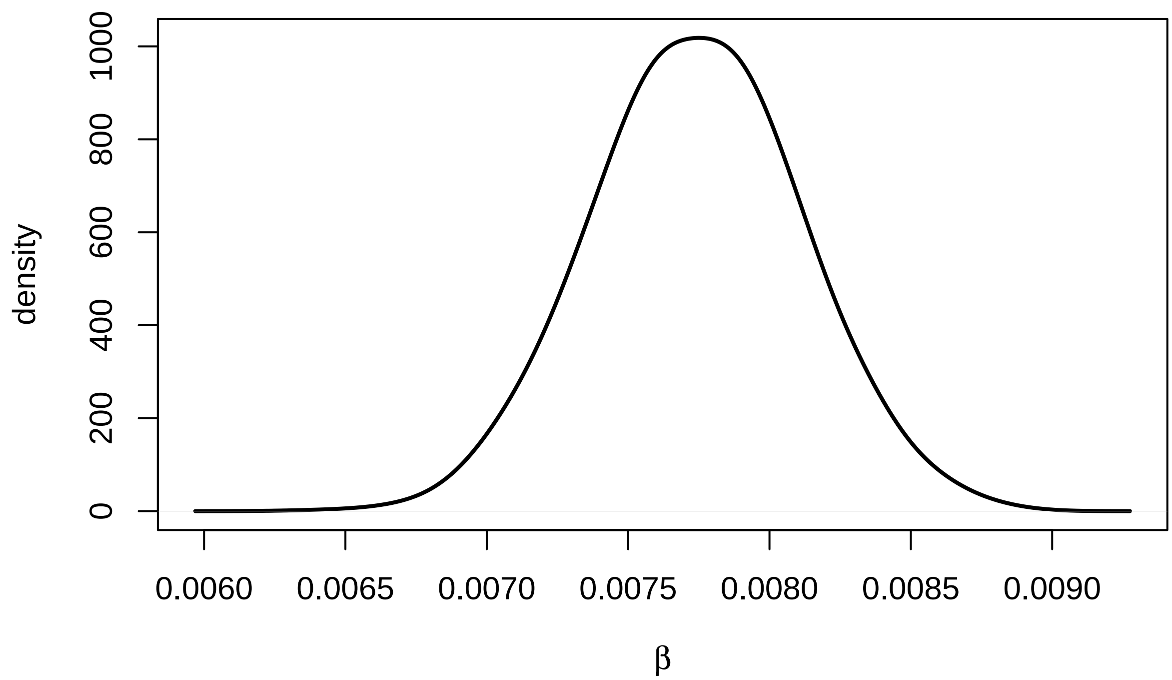
```
params = extract(mod_fit)
plot(density(params$alpha, adj=2), main="", xlab=expression(alpha),ylab="density",lwd=2)
```




```
quantile(probs = c(0.025, 0.975), params$alpha)
```

```
##      2.5%     97.5%
## -16.45268 -13.64695
```

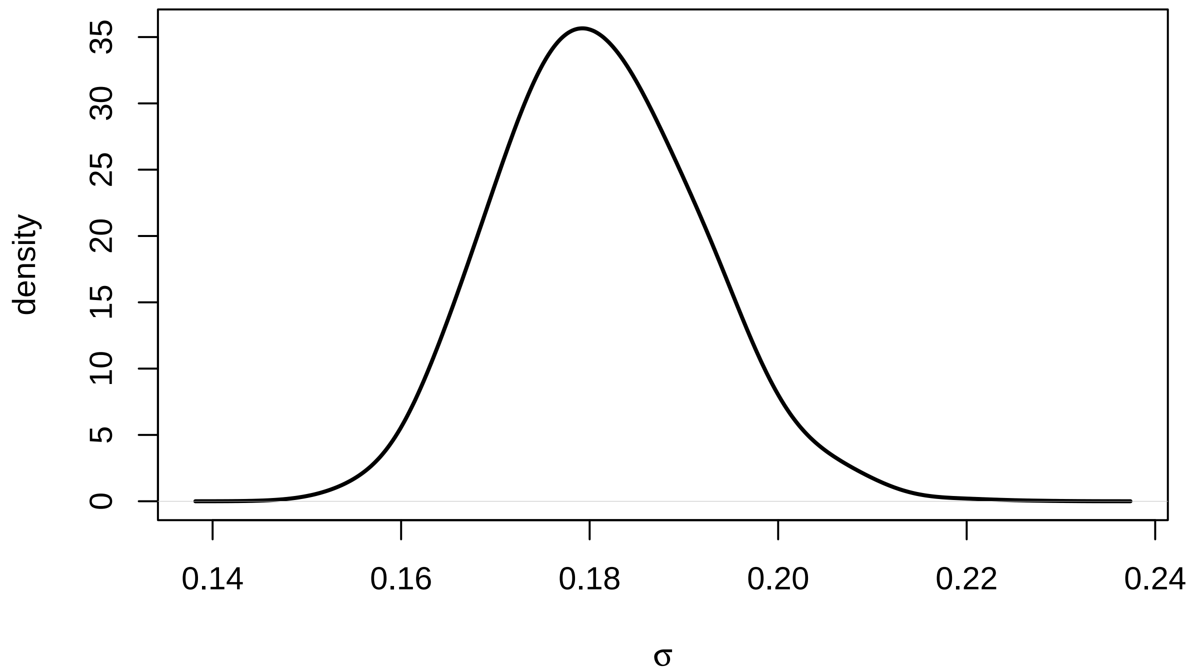
```
plot(density(params$beta, adj=2), main="", xlab=expression(beta),ylab="density",lwd=2)
```




```
quantile(probs = c(0.025, 0.975), params$beta)
```

```
##        2.5%       97.5%
## 0.007024605 0.008461960
```

```
plot(density(params$sigma, adj=2), main="", xlab=expression(sigma),ylab="density",lwd=2)
```




```
quantile(probs = c(0.025, 0.975), params$sigma)
```

```
##      2.5%     97.5%
## 0.1620228 0.2032057
```

d)

```
mean(params$beta > 0)
```

```
## [1] 1
```

e)

We change the prior to favor the value 0.

```
write(
"data {
  int N; // number of observations
  // response
  vector[N] y;
  // number of columns in the design matrix X
  vector[N] X;
  real scale_alpha; // prior sd on alpha
  real scale_beta; // prior sds on betas
  real loc_sigma;
}
parameters {
```

```
  real alpha; // intercept
  real beta; // regression coefficients beta vector
  real sigma;
}
// this is a convenient way to utilize the fact the mean of Y depends on x
transformed parameters {
  vector[N] mu; // defines the mean of each Y
  mu = alpha + X * beta; //notice * is vector multiplication
}
model {
  // priors
  alpha ~ normal(0, scale_alpha);
  beta ~ normal(0, scale_beta); // notice the beta priors are independent
  // to model correlated betas you can use lkj_corr prior
  sigma ~ exponential(loc_sigma); // this does not match the textbook
  y ~ normal(mu, sigma); // likelihood
}
", "Example5.stan")
mod = stan_model("Example5.stan")
mod_data = list(
  X = X,
  N = length(y),
  y = y
)
mod_data$scale_alpha = 150
mod_data$scale_beta <- 0.001
mod_data$loc_sigma <- sd(y)
mod_fit = sampling(mod, data = mod_data, iter = 3000)
```

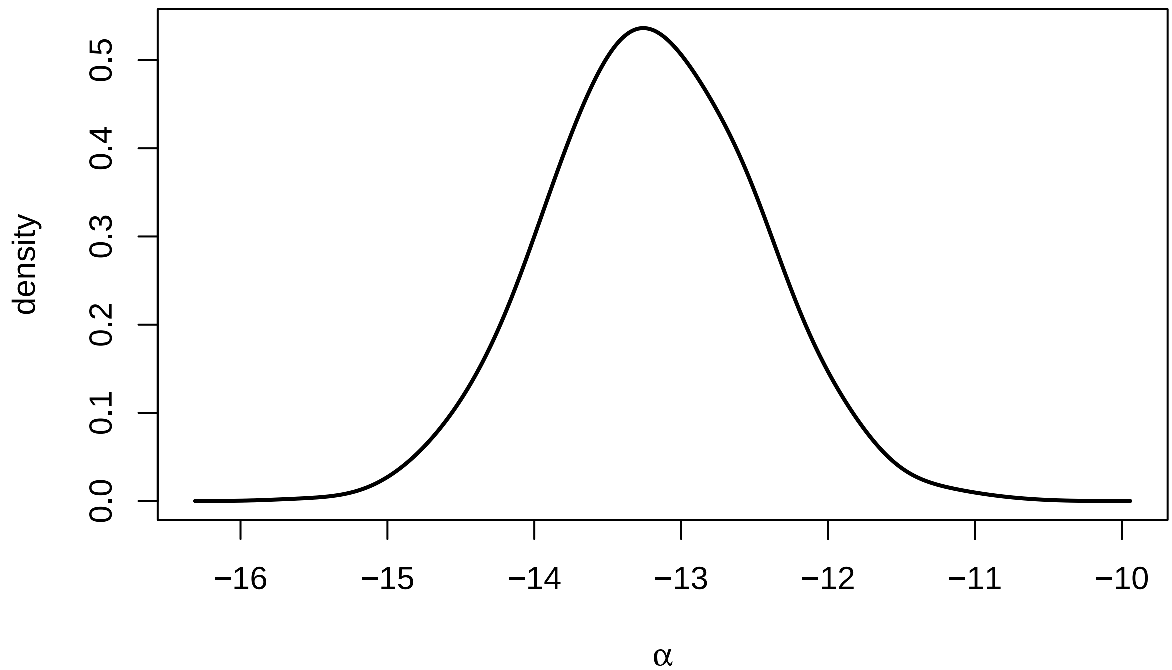
```
## Warning: There were 887 transitions after warmup that exceeded the maximum treedepth. Increase max_t
## https://mc-stan.org/misc/warnings.html#maximum-treedepth-exceeded
```

```
## Warning: Examine the pairs() plot to diagnose sampling problems
```

```
summary(mod_fit, pars = c("alpha", "beta", "sigma"), probs = c(0.025, 0.975))$summary
```

```
##               mean      se_mean           sd         2.5%         97.5%
## alpha -13.19447487 1.723898e-02 0.7099015623 -14.58380707 -11.803802684
## beta    0.00679407 8.831938e-06 0.0003637743   0.00608625   0.007509559
## sigma   0.18508423 2.843842e-04 0.0116189062   0.16402556   0.209550656
##          n_eff     Rhat
## alpha 1695.793 1.001514
## beta  1696.493 1.001540
## sigma 1669.243 1.000441
```

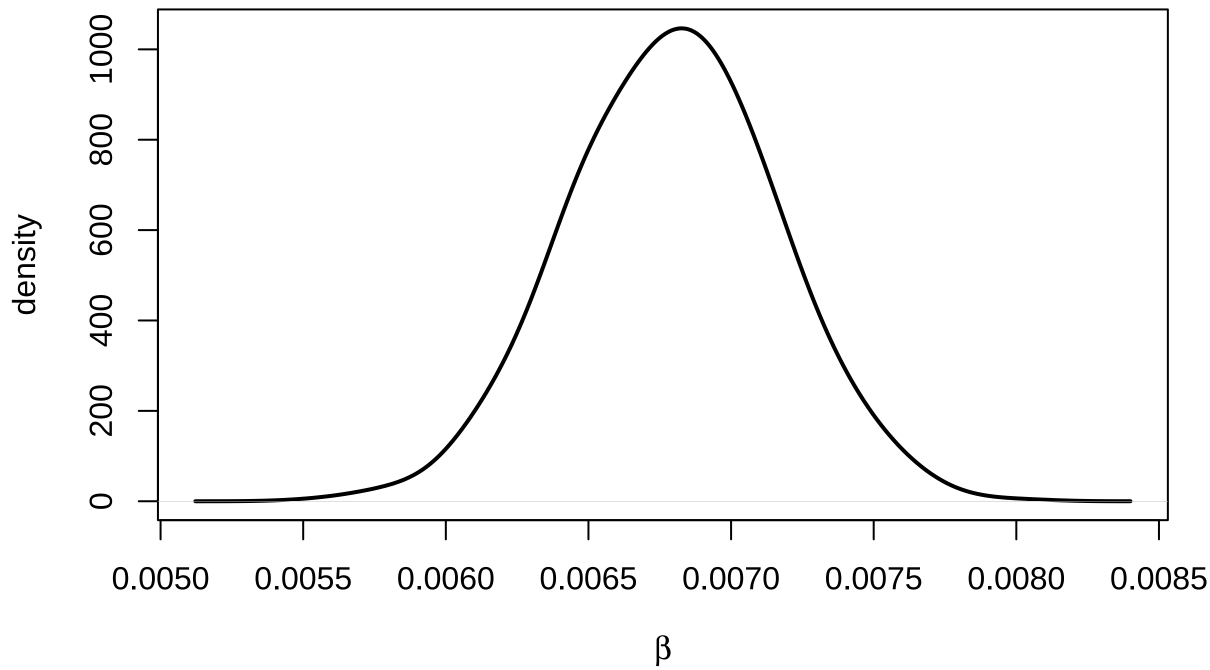
```
params = extract(mod_fit)
plot(density(params$alpha, adj=2), main="", xlab=expression(alpha),ylab="density",lwd=2)
```




```
quantile(probs = c(0.025, 0.975), params$alpha)
```

```
##      2.5%     97.5%
## -14.58381 -11.80380
```

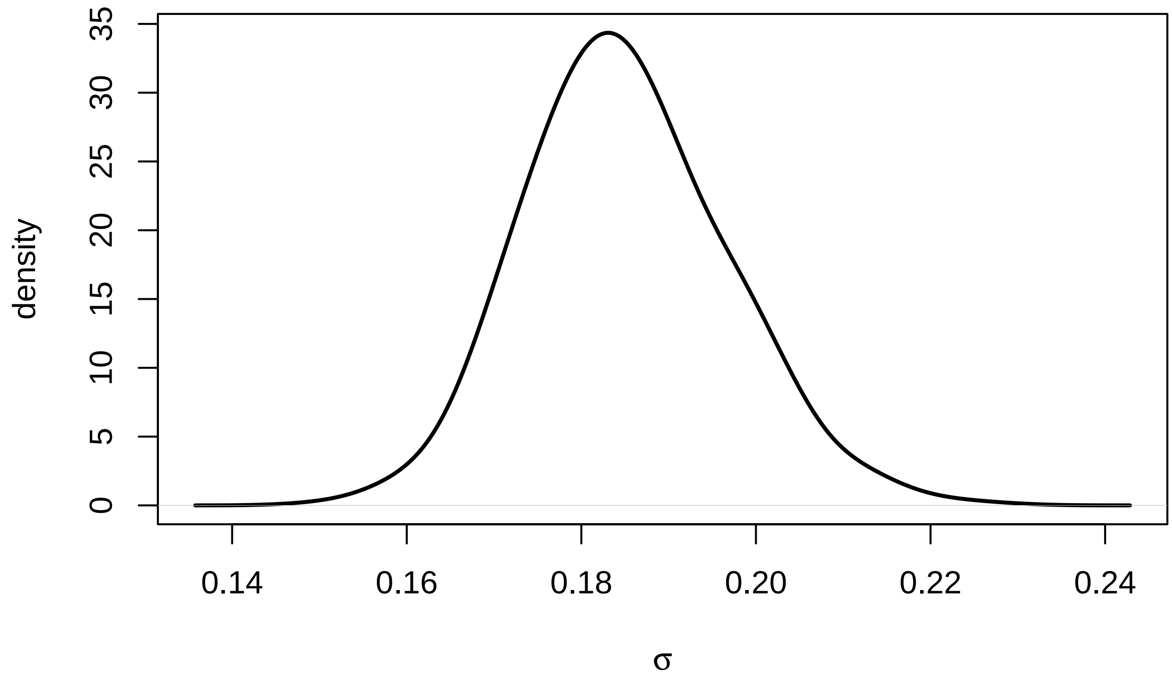
```
plot(density(params$beta, adj=2), main="", xlab=expression(beta),ylab="density",lwd=2)
```




```r
quantile(probs = c(0.025, 0.975), params$beta)
```

```
##        2.5%       97.5%
## 0.006086250 0.007509559
```

```r
plot(density(params$sigma, adj=2), main="", xlab=expression(sigma),ylab="density",lwd=2)
```




```r
quantile(probs = c(0.025, 0.975), params$sigma)
```

```
##      2.5%     97.5%
## 0.1640256 0.2095507
```

```r
mean(params$beta > 0)
```

```
## [1] 1
```

e)

The parameter for year is still significant, which means the global temperature is increasing.