ST-540 Assignment-5

Tilekbek Zhoroev

3/4/2022

Problem 1

In Section 2.4 we compared Reggie Jackson's home run rate in the regular season and World Series. He hit 563 home runs in 2820 regular-season games and 10 home runs in 27 World Series games (a player can hit 0, 1, 2, ... home runs in a game). Assuming Uniform(0,10) priors for both home run rates, use JAGS to summarize the posterior distribution of (i) his home run rate in the regular season, (ii) his home run rate in the World Series, and (iii) the ratio of these rates. Provide trace plots for all three parameters and discuss convergence of the MCMC sampler including appropriate convergence diagnostics.

Solution Here we have given the Poisson likelihood and uniform prior.

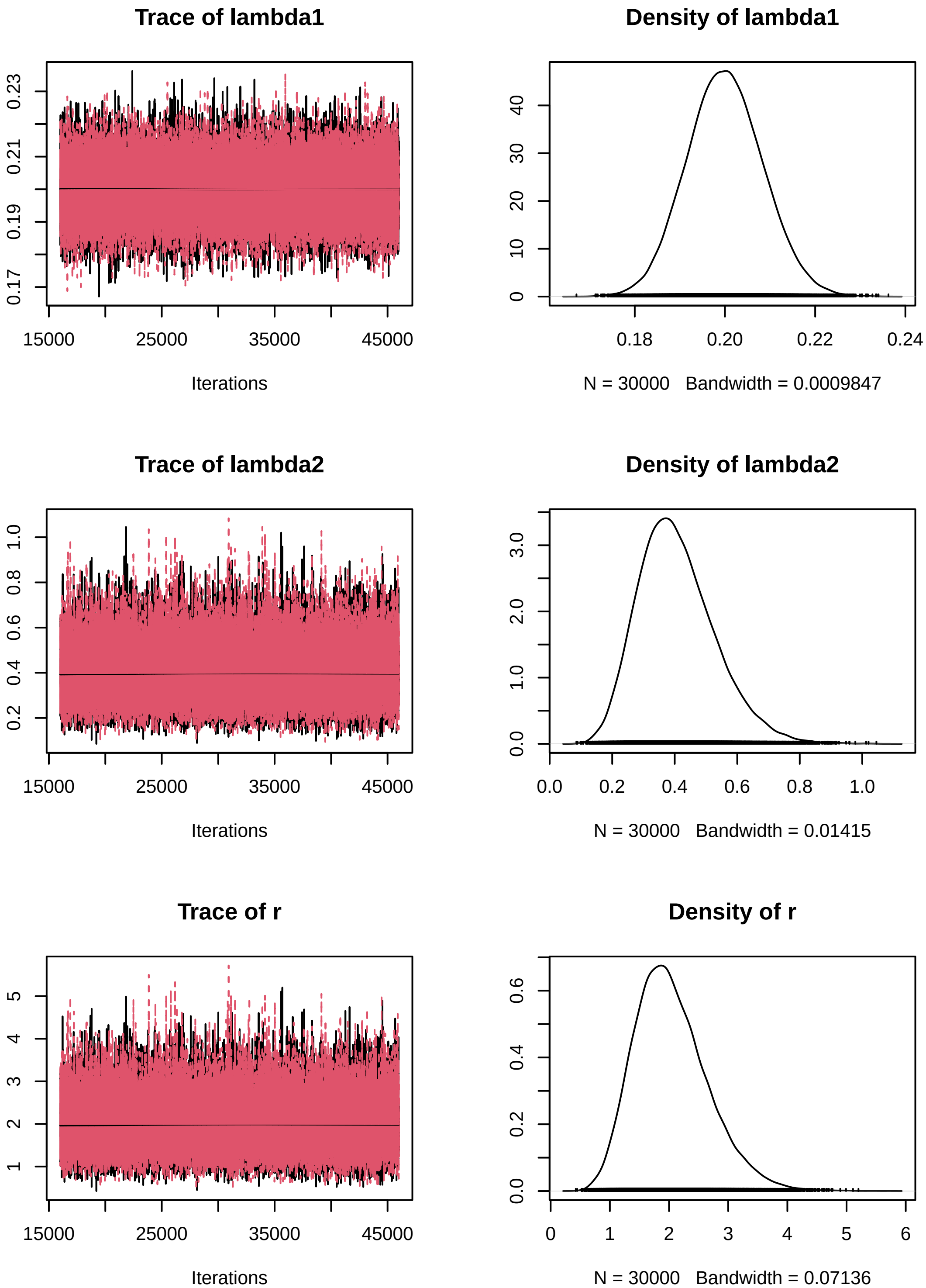
```
# Given parameters
N1 = 2820; Y1 = 563; N2 = 27; Y2 = 10
# define string model
model_string <- textConnection("model{
    # Likelihood
    Y1 ~ dpois(N1*lambda1)
    Y2 ~ dpois(N2*lambda2)
    # Priors
    lambda1 ~  dunif(0, 10)
    lambda2 ~  dunif(0, 10)
    r <- lambda2/lambda1
 }")
# initizalize the parameters
inits <- list(lambda1= Y1/N1,lambda2 = Y2/N2)
# Load the data and compile the MCMC code
data <- list(N1 = N1,Y1 = Y1,N2 = N2,Y2 = Y2)
model <- jags.model(model_string,data = data, inits=inits, n.chains=2)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 2
##    Unobserved stochastic nodes: 2
##    Total graph size: 11
##
## Initializing model
```

```
#Burn-in for 10000 samples
update(model, 15000, progress.bar="none")


 params  <- c("lambda1","lambda2","r")
 samples <- coda.samples(model,
```

```
             variable.names=params,
             n.iter=30000, progress.bar="none")
```

```
plot(samples)
```



```
summary(samples)
```

```
##
## Iterations = 16001:46000
## Thinning interval = 1
## Number of chains = 2
## Sample size per chain = 30000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##           Mean       SD  Naive SE Time-series SE
## lambda1 0.2000 0.008388 3.424e-05      0.0000435
## lambda2 0.4069 0.122367 4.996e-04      0.0006945
## r       2.0376 0.619093 2.527e-03      0.0034991
##
## 2. Quantiles for each variable:
##
##          2.5%    25%    50%    75%  97.5%
## lambda1 0.184 0.1943 0.1999 0.2056 0.2169
## lambda2 0.204 0.3195 0.3940 0.4810 0.6810
## r       1.016 1.5955 1.9699 2.4100 3.4276
```

```
effectiveSize(samples)
```

```
##  lambda1  lambda2        r
## 37195.65 31059.89 31315.36
```

```
gelman.diag(samples)
```

```
## Potential scale reduction factors:
##
##         Point est. Upper C.I.
## lambda1          1          1
## lambda2          1          1
## r                1          1
##
## Multivariate psrf
##
## 1
```

The trace plots look great, the effective sample sizes are all large (over 32000), and the Gelman-Rubin statistics are 1.0. Therefore, the chains have clearly converged.

Problem2

A clinical trial gave six subjects a placebo and six subjects a new weight loss medication. The response variable is the change in weight (pounds) from baseline (so -2.0 means the subject lost 2 pounds). The data for the 12 subjects are:

Placebo	Treatment
2.0	-3.5
-3.1	-1.6
-1.0	-4.6
0.2	-0.9
0.3	-5.1

Placebo	Treatment
0.4	0.1

Conduct a Bayesian analysis to compare the means of these two groups. Would you say the treatment is effective? Is your conclusion sensitive to the prior?

Solution

Let us assume two different cases, (i) two groups have same variance and different variances. In first case, let the placebo group is $Y_i \sim^{iid} \mathcal{N}(\mu, \sigma^2)$ for $i = 1, 2, ..., n_1$ and treatment group is $Y_i \sim^{iid} \mathcal{N}(\mu + \delta, \sigma^2)$ for $i = n_1 + 1, n_1 + 2, ..., n_1 + n_2 = n$. Here we would like to analyze whether $\delta = 0$ or not. Since, the true variance of the groups are unknown we would like to use Jeffrey's prior $\pi(\mu, \delta, \sigma^2)$ the the marginal posterior distribution of δ integrating over both μ and σ^2 is

$$\delta | rest \sim t_n \left[\bar{Y}_2 - \bar{Y}_1, \hat{\sigma}^2 \left(\frac{1}{n_1} + \frac{1}{n_2} \right) \right].$$

where $\hat{Y}_1$ and $\hat{Y}_2$ are the mean of Placebo and Treatment group, respectively.

In second case we assume $Y_i \sim^{iid} \mathcal{N}(\mu, \sigma_1^2)$ for $i = 1, 2, ..., n_1$ and $Y_i \sim^{iid} \mathcal{N}(\mu + \delta, \sigma_2^2)$ for $i = n_1 + 1, n_1 + 2, ..., n_1 + n_2 = n$. Then the posterior can be approximated by MCMC.

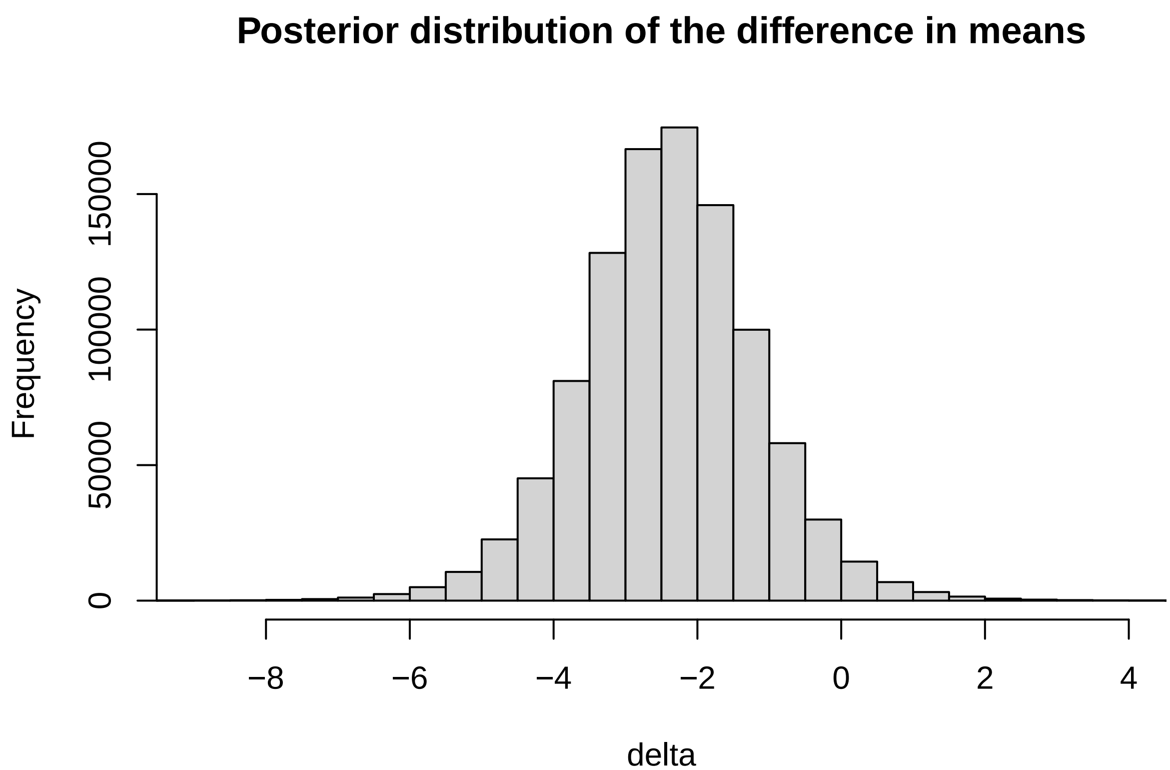
```
Y1 = c(2.0, -3.1, -1.0,0.2,0.3,0.4)
Y2 = c(-3.5, -1.6, -4.6,-0.9,-5.1,0.1)

Ybar1 <- mean(Y1)
s21 <- mean((Y1-Ybar1)^2)
n1 <- length(Y1)

 # Statistics from group 2
Ybar2 <- mean(Y2)
s22 <- mean((Y2-Ybar2)^2)
n2 <- length(Y2)

# Posterior of the difference assuming equal variance
delta_hat <- Ybar2-Ybar1
s2 <- (n1*s21 + n2*s22)/(n1+n2)
scale <- sqrt(s2)*sqrt(1/n1+1/n2)
df <- n1+n2
cred_int <- delta_hat + scale*qt(c(0.025,0.975),df=df)
delta_hat
```

```
## [1] -2.4
```

```
cred_int
```

```
## [1] -4.6058799 -0.1941201
```

```
 # Posterior of delta assuming unequal variance using MC sampling
mu1 <- Ybar1 + sqrt(s21/n1)*rt(1000000,df=n1)
mu2 <- Ybar2 + sqrt(s22/n2)*rt(1000000,df=n2)
delta <- mu2-mu1

hist(delta,main="Posterior distribution of the difference in means",xlim = c(-9,4), breaks = 100)
```



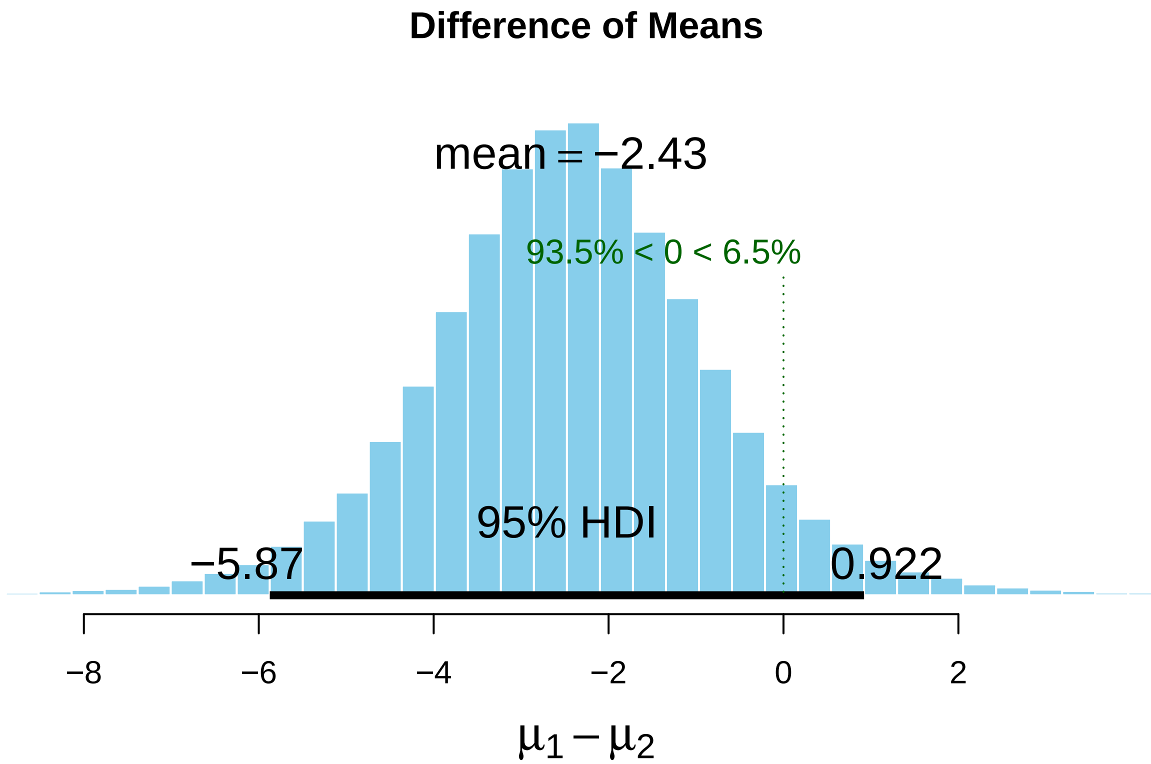

```r
quantile(delta,c(0.025,0.975)) # 95% credible set
```

```
##        2.5%       97.5%
## -4.86315294  0.06414645
```

```r
#other way of representation
Bay_model= BESTmcmc(Y2, Y1)
```

```
## Waiting for parallel processing to complete...done.
```

```r
plot(Bay_model)
```



From here we observe that if we take same variance that 0 is not included in credible interval. We can say that the treatment is effective, but we would like to check sensitivity of these results. The next case show us that the credible interval is includes the 0. Hence, it's sensitive to the choice of priors.

Problem3

The response variable is `medv`, the median value of owner-occupied homes (in $1,000s), and the other 13 variables are covariates that describe the neighborhood.

(a) Fit a Bayesian linear regression model with uninformative Gaussian priors for the regression coefficients. Verify the MCMC sampler has converged, and summarize the posterior distribution of all regression coefficients.

(b) Perform a classic least squares analysis (e.g., using the lm function in R). Compare the results numerically and conceptually with the Bayesian results.

(c) Refit the Bayesian model with double exponential priors for the regression coefficients, and discuss how the results differ from the analysis with uninformative priors.

(d) Fit a Bayesian linear regression model in (a) using only the first 500 observations and compute the posterior predictive distribution for the final 6 observations. Plot the posterior predictive distribution versus the actual value for these 6 observations and comment on whether the predictions are reasonable.

Solution

(a) Before we start let us explore the given data is there any missing terms?

```
library(MASS)
```

```
##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select
```

```r
data(Boston)
summary(Boston)
```

```
##       crim                zn             indus            chas        
##  Min.   : 0.00632   Min.   :  0.00   Min.   : 0.46   Min.   :0.00000  
##  1st Qu.: 0.08205   1st Qu.:  0.00   1st Qu.: 5.19   1st Qu.:0.00000  
##  Median : 0.25651   Median :  0.00   Median : 9.69   Median :0.00000  
##  Mean   : 3.61352   Mean   : 11.36   Mean   :11.14   Mean   :0.06917  
##  3rd Qu.: 3.67708   3rd Qu.: 12.50   3rd Qu.:18.10   3rd Qu.:0.00000  
##  Max.   :88.97620   Max.   :100.00   Max.   :27.74   Max.   :1.00000  
##       nox               rm             age              dis        
##  Min.   :0.3850   Min.   :3.561   Min.   :  2.90   Min.   : 1.130  
##  1st Qu.:0.4490   1st Qu.:5.886   1st Qu.: 45.02   1st Qu.: 2.100  
##  Median :0.5380   Median :6.208   Median : 77.50   Median : 3.207  
##  Mean   :0.5547   Mean   :6.285   Mean   : 68.57   Mean   : 3.795  
##  3rd Qu.:0.6240   3rd Qu.:6.623   3rd Qu.: 94.08   3rd Qu.: 5.188  
##  Max.   :0.8710   Max.   :8.780   Max.   :100.00   Max.   :12.127  
##       rad              tax           ptratio          black       
##  Min.   : 1.000   Min.   :187.0   Min.   :12.60   Min.   :  0.32  
##  1st Qu.: 4.000   1st Qu.:279.0   1st Qu.:17.40   1st Qu.:375.38  
##  Median : 5.000   Median :330.0   Median :19.05   Median :391.44  
##  Mean   : 9.549   Mean   :408.2   Mean   :18.46   Mean   :356.67  
##  3rd Qu.:24.000   3rd Qu.:666.0   3rd Qu.:20.20   3rd Qu.:396.23  
##  Max.   :24.000   Max.   :711.0   Max.   :22.00   Max.   :396.90  
##      lstat            medv      
##  Min.   : 1.73   Min.   : 5.00  
##  1st Qu.: 6.95   1st Qu.:17.02  
##  Median :11.36   Median :21.20  
##  Mean   :12.65   Mean   :22.53  
##  3rd Qu.:16.95   3rd Qu.:25.00  
##  Max.   :37.97   Max.   :50.00
```

We observe that all entries are filled. Next, we would like to construct Bayesian model with uninformative Gaussian prior.

```r
Y = Boston%>%
  dplyr::select(medv)
Y = as.matrix(Y)
X = Boston%>%
  dplyr::select(-medv)

X <- scale(X) # standardize covariates
X <- cbind(1,X) # add intercept
colnames(X)[1] = "Intercept"
names = colnames(X)

#load given data
data <- list(n=length(Y),p=ncol(X),Y=Y,X=X)

# define model string
model_string <- textConnection("model{
# Likelihood
for(i in 1:n){
Y[i,] ~ dnorm(inprod(X[i,],beta[]),tau)
}
```

```r
# Priors
for(j in 1:p){beta[j] ~ dnorm(0, 0.0001)}
tau ~ dgamma(0.01,0.01)
}")

model <- jags.model(model_string, data = data, n.chains=2,quiet=TRUE)
update(model, 10000, progress.bar="none")
params <- c("beta")
samples <- coda.samples(model, variable.names=params, n.iter=10000,progress.bar="none")
```

```r
effectiveSize(samples)
```

```
##   beta[1]   beta[2]   beta[3]   beta[4]   beta[5]   beta[6]   beta[7]   beta[8]
## 20000.000  7177.998  4948.493  2830.755 14598.177  3306.599  5066.582  4812.181
##   beta[9]  beta[10]  beta[11]  beta[12]  beta[13]  beta[14]
##  3402.395  1384.307  1302.666  6547.066 11926.715  5069.139
```

```r
gelman.diag(samples)
```

```
## Potential scale reduction factors:
##
##          Point est. Upper C.I.
## beta[1]           1       1.00
## beta[2]           1       1.00
## beta[3]           1       1.00
## beta[4]           1       1.00
## beta[5]           1       1.00
## beta[6]           1       1.00
## beta[7]           1       1.00
## beta[8]           1       1.00
## beta[9]           1       1.00
## beta[10]          1       1.01
## beta[11]          1       1.01
## beta[12]          1       1.00
## beta[13]          1       1.00
## beta[14]          1       1.00
##
## Multivariate psrf
##
## 1
```

```r
sum                        <- summary(samples)
rownames(sum$statistics) <- names
rownames(sum$quantiles)  <- names
sum$statistics           <- round(sum$statistics,4)
sum$quantiles            <- round(sum$quantiles,4)
sum
```

```
##
## Iterations = 10001:20000
## Thinning interval = 1
## Number of chains = 2
## Sample size per chain = 10000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
```

```
##
##              Mean     SD Naive SE Time-series SE
## Intercept 22.5319 0.2124   0.0015         0.0015
## crim      -0.9337 0.2817   0.0020         0.0033
## zn         1.0872 0.3149   0.0022         0.0045
## indus      0.1340 0.4226   0.0030         0.0079
## chas       0.6808 0.2184   0.0015         0.0018
## nox       -2.0479 0.4466   0.0032         0.0078
## rm         2.6737 0.2910   0.0021         0.0041
## age        0.0196 0.3713   0.0026         0.0054
## dis       -3.1104 0.4149   0.0029         0.0071
## rad        2.6667 0.5893   0.0042         0.0158
## tax       -2.0794 0.6433   0.0045         0.0178
## ptratio   -2.0589 0.2837   0.0020         0.0035
## black      0.8499 0.2463   0.0017         0.0023
## lstat     -3.7538 0.3560   0.0025         0.0050
##
## 2. Quantiles for each variable:
##
##              2.5%     25%     50%     75%   97.5%
## Intercept 22.1150 22.3901 22.5305 22.6752 22.9454
## crim      -1.4873 -1.1228 -0.9338 -0.7447 -0.3773
## zn         0.4637  0.8790  1.0853  1.2981  1.7021
## indus     -0.6916 -0.1512  0.1375  0.4162  0.9600
## chas       0.2524  0.5345  0.6792  0.8269  1.1129
## nox       -2.9095 -2.3458 -2.0527 -1.7546 -1.1659
## rm         2.1011  2.4769  2.6756  2.8684  3.2452
## age       -0.7072 -0.2327  0.0179  0.2730  0.7467
## dis       -3.9104 -3.3893 -3.1135 -2.8349 -2.2853
## rad        1.4902  2.2779  2.6813  3.0628  3.8007
## tax       -3.3240 -2.5110 -2.0804 -1.6463 -0.8026
## ptratio   -2.6113 -2.2520 -2.0622 -1.8680 -1.5010
## black      0.3650  0.6829  0.8502  1.0155  1.3265
## lstat     -4.4575 -3.9935 -3.7549 -3.5105 -3.0634
```

Since the effective size is more than 1400 and and the Gelman-Rubin statistics are 1.0. Therefore, the chains have clearly converged.

(b)

```
ols_data = cbind(Y,X[,2:14])
ols_data = data.frame(ols_data)
ols_model = lm(medv~.-medv, data = ols_data)
#ols_model$coefficients
tidy(ols_model)
```

```
## # A tibble: 14 x 5
##    term        estimate std.error statistic  p.value
##    <chr>          <dbl>     <dbl>     <dbl>    <dbl>
##  1 (Intercept)  22.5        0.211   107.     0
##  2 crim         -0.929      0.283    -3.29   1.09e- 3
##  3 zn            1.08       0.320     3.38   7.78e- 4
##  4 indus         0.141      0.422     0.334  7.38e- 1
##  5 chas          0.682      0.219     3.12   1.93e- 3
##  6 nox          -2.06       0.443    -4.65   4.25e- 6
##  7 rm            2.68       0.294     9.12   1.98e-18
```

```
##  8 age           0.0195     0.372    0.0524 9.58e- 1
##  9 dis          -3.11       0.420   -7.40   6.01e-13
## 10 rad           2.66       0.578    4.61   5.07e- 6
## 11 tax          -2.08       0.634   -3.28   1.11e- 3
## 12 ptratio      -2.06       0.283   -7.28   1.31e-12
## 13 black         0.850      0.245    3.47   5.73e- 4
## 14 lstat        -3.75       0.362  -10.3    7.78e-23
```

The are numerically almost equivalent. However, in Bayesian approach this is parameter rather than fixed number. Hence, all coefficients have distributions. Even though, their representation are same, their interpratations are completely different.

(c)

```
 model_string <- textConnection("model{
   # Likelihood
    for(i in 1:n){
      Y[i,] ~ dnorm(alpha+inprod(X[i,],beta[]),taue)
    }
   # Priors
    for(j in 1:p){
      beta[j] ~ ddexp(0,taue*taub)
    }
    alpha ~ dnorm(0,0.001)
    taue  ~ dgamma(0.1, 0.1)
    taub  ~ dgamma(0.1, 0.1)
 }")

 model <- jags.model(model_string,data = data, n.chains = 2,quiet=TRUE)
update(model, 10000, progress.bar="none")
 samples2 <- coda.samples(model, variable.names=params, n.iter=10000,progress.bar="none")
```

```
effectiveSize(samples2)
```

```
##     beta[1]     beta[2]     beta[3]     beta[4]     beta[5]     beta[6]
##    40.71476  4855.87844  3384.04232  2341.89118 10153.11727  1850.73164
##     beta[7]     beta[8]     beta[9]    beta[10]    beta[11]    beta[12]
##  3652.10870  3138.45357  2236.27803   917.73155   859.19104  4262.52840
##    beta[13]    beta[14]
##  8122.85357  3048.25726
```

```
gelman.diag(samples2)
```

```
## Potential scale reduction factors:
##
##          Point est. Upper C.I.
## beta[1]        1.03       1.14
## beta[2]        1.00       1.00
## beta[3]        1.00       1.00
## beta[4]        1.00       1.00
## beta[5]        1.00       1.00
## beta[6]        1.00       1.00
## beta[7]        1.00       1.01
## beta[8]        1.00       1.01
## beta[9]        1.00       1.00
## beta[10]       1.00       1.01
## beta[11]       1.00       1.01
```

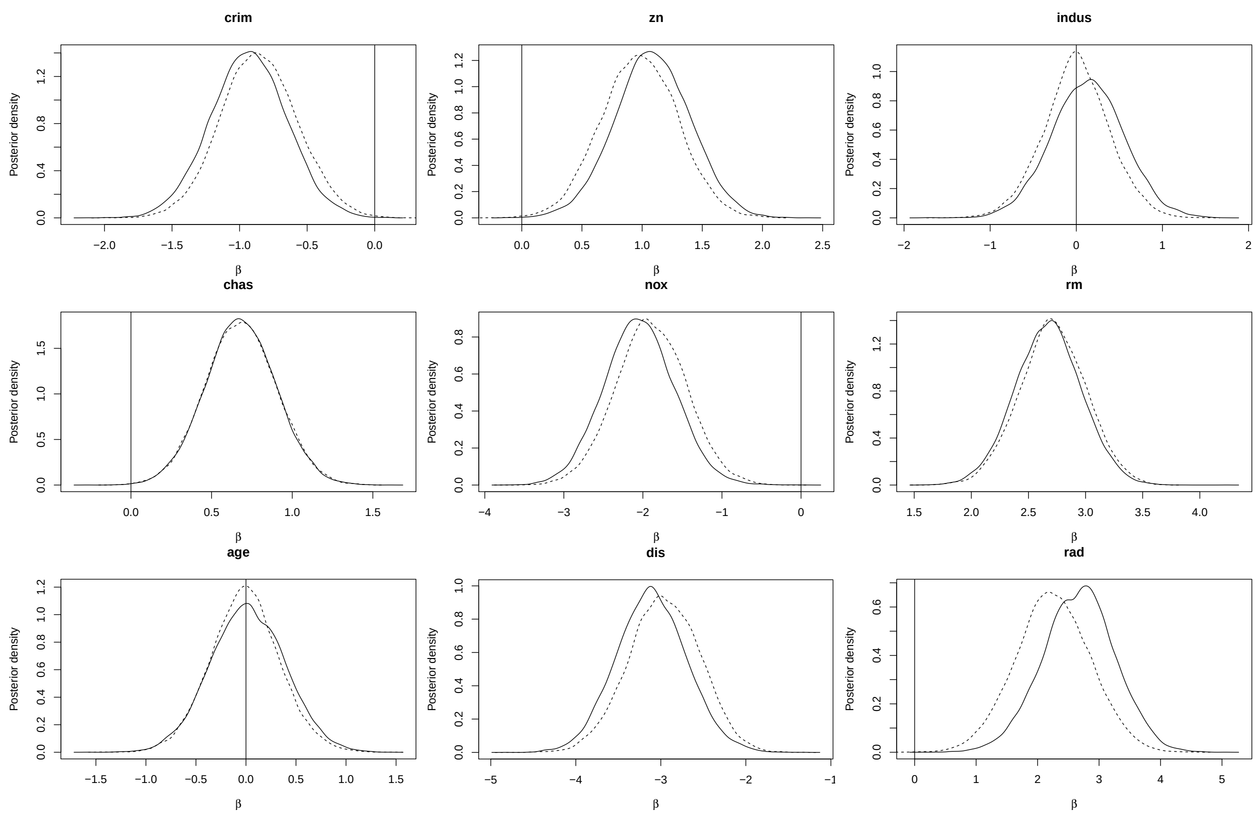
```
## beta[12]       1.00       1.00
## beta[13]       1.00       1.00
## beta[14]       1.00       1.00
##
## Multivariate psrf
##
## 1.02
```

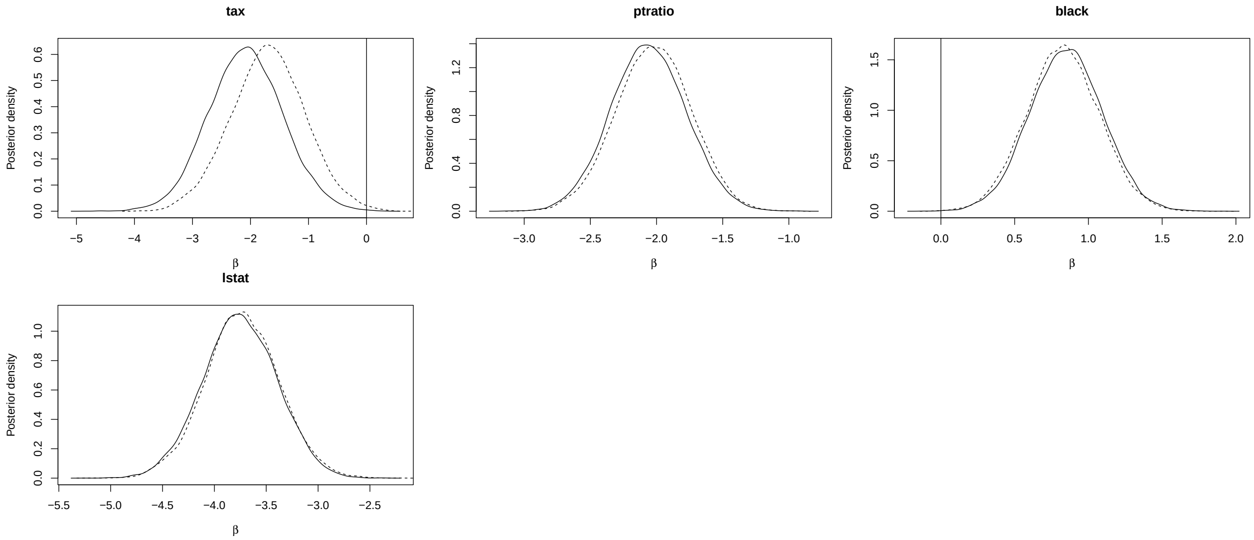
```
sum                      <- summary(samples2)
rownames(sum$statistics) <- names
rownames(sum$quantiles)  <- names
sum$statistics           <- round(sum$statistics,4)
sum$quantiles            <- round(sum$quantiles,4)
sum
```

```
##
## Iterations = 11001:21000
## Thinning interval = 1
## Number of chains = 2
## Sample size per chain = 10000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##              Mean     SD Naive SE Time-series SE
## Intercept  0.2668 2.9135   0.0206         0.4541
## crim      -0.8542 0.2831   0.0020         0.0041
## zn         0.9769 0.3171   0.0022         0.0054
## indus      0.0023 0.3814   0.0027         0.0079
## chas       0.6813 0.2202   0.0016         0.0022
## nox       -1.8954 0.4475   0.0032         0.0104
## rm         2.7135 0.2910   0.0021         0.0048
## age       -0.0088 0.3448   0.0024         0.0062
## dis       -2.9590 0.4177   0.0030         0.0088
## rad        2.2476 0.6068   0.0043         0.0200
## tax       -1.6929 0.6406   0.0045         0.0218
## ptratio   -2.0217 0.2826   0.0020         0.0043
## black      0.8274 0.2455   0.0017         0.0027
## lstat     -3.7306 0.3562   0.0025         0.0065
##
## 2. Quantiles for each variable:
##
##              2.5%     25%     50%     75%   97.5%
## Intercept -5.3969 -1.4336  0.1210  1.6912  7.5625
## crim      -1.4026 -1.0457 -0.8593 -0.6644 -0.2898
## zn         0.3605  0.7630  0.9772  1.1939  1.5957
## indus     -0.7659 -0.2425  0.0018  0.2517  0.7625
## chas       0.2525  0.5314  0.6810  0.8294  1.1125
## nox       -2.7707 -2.1943 -1.9013 -1.5922 -1.0136
## rm         2.1378  2.5221  2.7125  2.9107  3.2838
## age       -0.6801 -0.2372 -0.0089  0.2142  0.6858
## dis       -3.7900 -3.2395 -2.9597 -2.6682 -2.1640
## rad        1.0518  1.8483  2.2425  2.6556  3.4334
## tax       -2.9687 -2.1194 -1.6911 -1.2639 -0.4165
## ptratio   -2.5812 -2.2125 -2.0211 -1.8299 -1.4765
```

```
## black      0.3496  0.6630  0.8264  0.9912  1.3136
## lstat     -4.4402 -3.9666 -3.7333 -3.4931 -3.0239
```

```r
for(j in 2:14){

 # Collect the MCMC iteration from both chains for the three priors

 s1 <- c(samples[[1]][,j],samples[[2]][,j])
 s2 <- c(samples2[[1]][,j],samples2[[2]][,j])

 # Get smooth density estimate for each prior

 d1 <- density(s1)
 d2 <- density(s2)


 # Plot the density estimates

 mx <- max(c(d1$y,d2$y))

 plot(d1$x,d1$y,type="l",ylim=c(0,mx),xlab=expression(beta),ylab="Posterior density",main=names[j])
 lines(d2$x,d2$y,lty=2)
 abline(v=0)
 legend(1, 95, legend=c("Uninformative Gaussian", "Bayesian LASSO"),
       col=c("red", "blue"), lty=1:2, cex=0.8)
}
```





Since we have enough data points the choice of prior have only minor affect. It also shown in figures.

(d)

```
Y_train = Y[1:500,]
Y_test = Y[501:506,]
X_train = X[1:500,]
X_test = X[501:506,]
n_train    <- length(Y_train)
n_test    <- length(Y_test)
p     <- ncol(X_train)

model_string <- textConnection("model{

  # Likelihood
  for(i in 1:no){
    Yo[i]   ~ dnorm(muo[i],inv.var)
    muo[i] <- alpha + inprod(Xo[i,],beta[])
  }

  # Prediction
  for(i in 1:np){
    Y_test[i]  ~ dnorm(mup[i],inv.var)
    mup[i] <- alpha + inprod(Xp[i,],beta[])
  }

  # Priors
  for(j in 1:p){
    beta[j] ~ dnorm(0,0.0001)
  }
  alpha     ~ dnorm(0, 0.01)
  inv.var   ~ dgamma(0.01, 0.01)
  sigma     <- 1/sqrt(inv.var)
}")

data = list(Yo=Y_train,no=n_train,np=n_test,p=p,Xo=X_train,Xp=X_test)
```

```r
model <- jags.model(model_string, data = data)
```

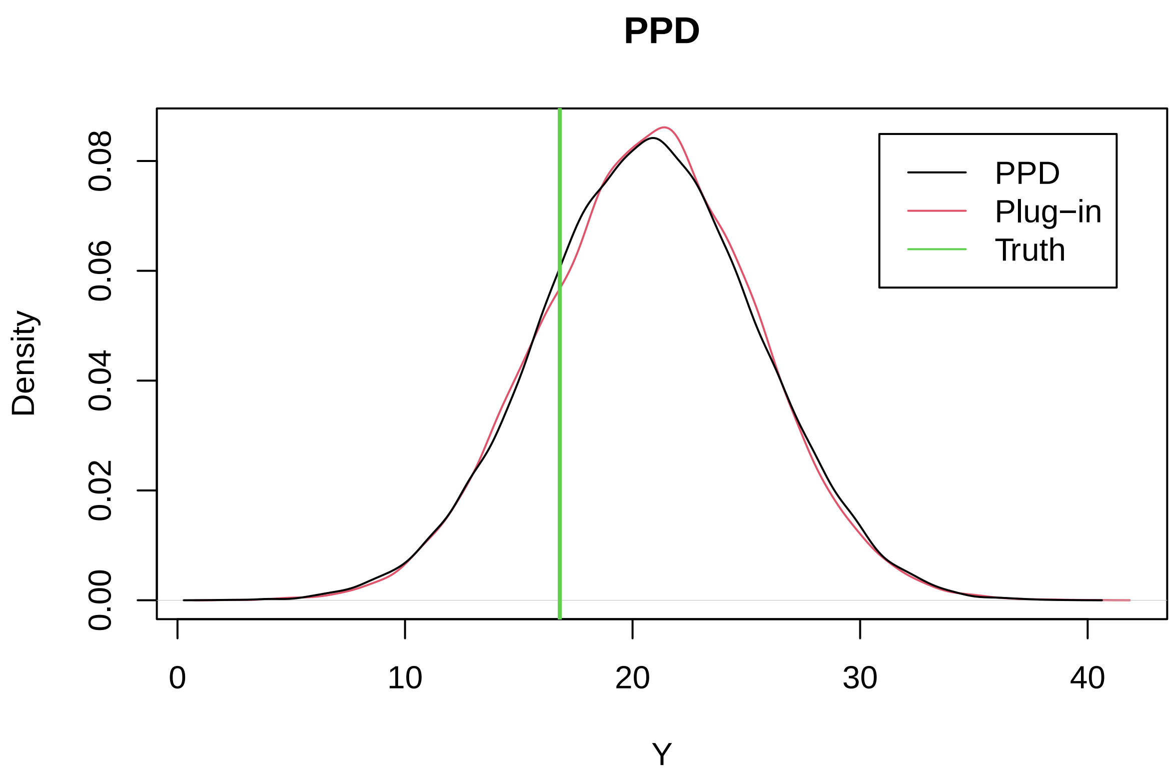
```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 500
##    Unobserved stochastic nodes: 22
##    Total graph size: 9134
##
## Initializing model
```

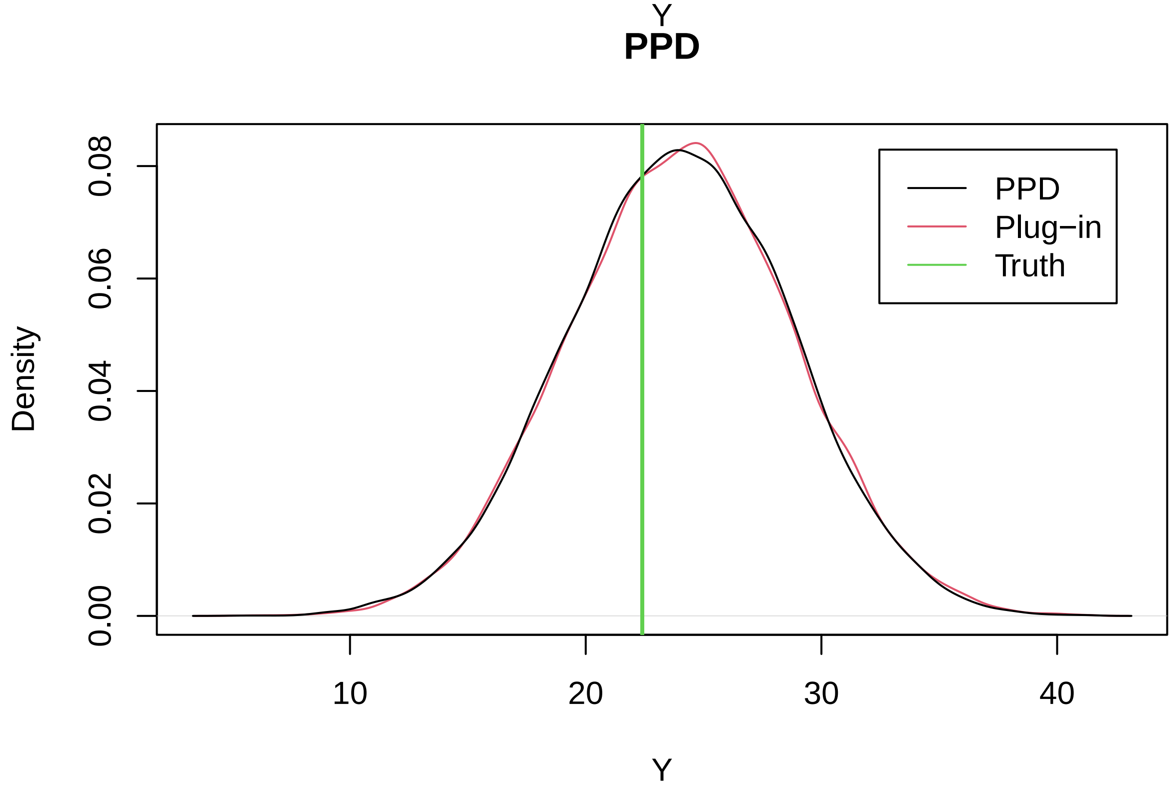
```r
update(model, 10000, progress.bar="none")

samp <- coda.samples(model,
        variable.names=c("beta","sigma","Y_test","alpha"),
        n.iter=20000, progress.bar="none")

summary(samp[,-c(1:n_test)])
```

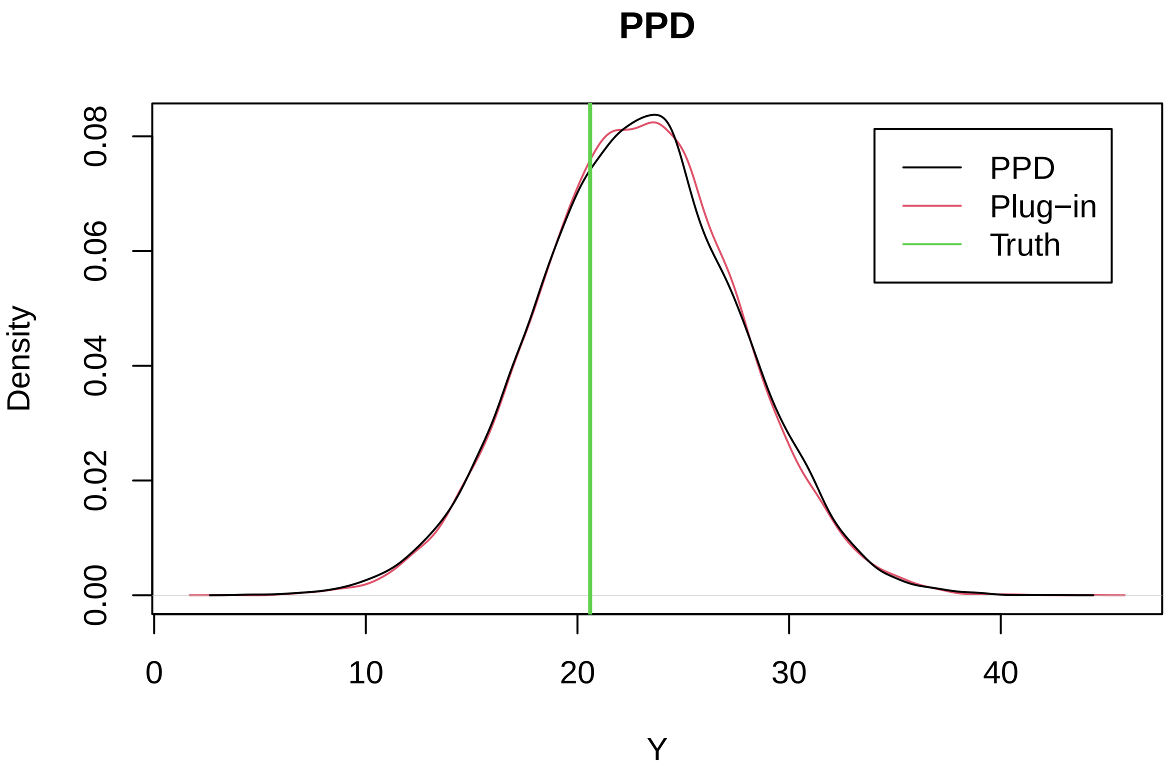
```
##
## Iterations = 10001:30000
## Thinning interval = 1
## Number of chains = 1
## Sample size per chain = 20000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##              Mean     SD Naive SE Time-series SE
## alpha     2.61156 9.0234 0.063805       3.601168
## beta[1]  19.97389 9.0264 0.063826       3.557065
## beta[2]  -0.91082 0.2849 0.002015       0.003295
## beta[3]   1.13556 0.3226 0.002281       0.004698
## beta[4]   0.12147 0.4236 0.002996       0.007787
## beta[5]   0.66519 0.2194 0.001552       0.001836
## beta[6]  -1.96348 0.4443 0.003142       0.007516
## beta[7]   2.66878 0.2968 0.002099       0.004010
## beta[8]   0.05326 0.3713 0.002626       0.005649
## beta[9]  -3.18666 0.4210 0.002977       0.007046
## beta[10]  2.54256 0.5746 0.004063       0.015110
## beta[11] -2.12498 0.6200 0.004384       0.017020
## beta[12] -1.91381 0.2917 0.002063       0.003734
## beta[13]  0.85204 0.2462 0.001741       0.002183
## beta[14] -3.84833 0.3694 0.002612       0.005373
## sigma     4.74355 0.1516 0.001072       0.001097
##
## 2. Quantiles for each variable:
##
##              2.5%     25%      50%     75%   97.5%
## alpha    -15.2887 -4.2430  3.67616  9.3539 17.5460
## beta[1]    5.0517 13.2240 18.89658 26.8466 37.8743
## beta[2]   -1.4673 -1.1008 -0.90972 -0.7203 -0.3561
## beta[3]    0.4960  0.9197  1.13801  1.3545  1.7619
```

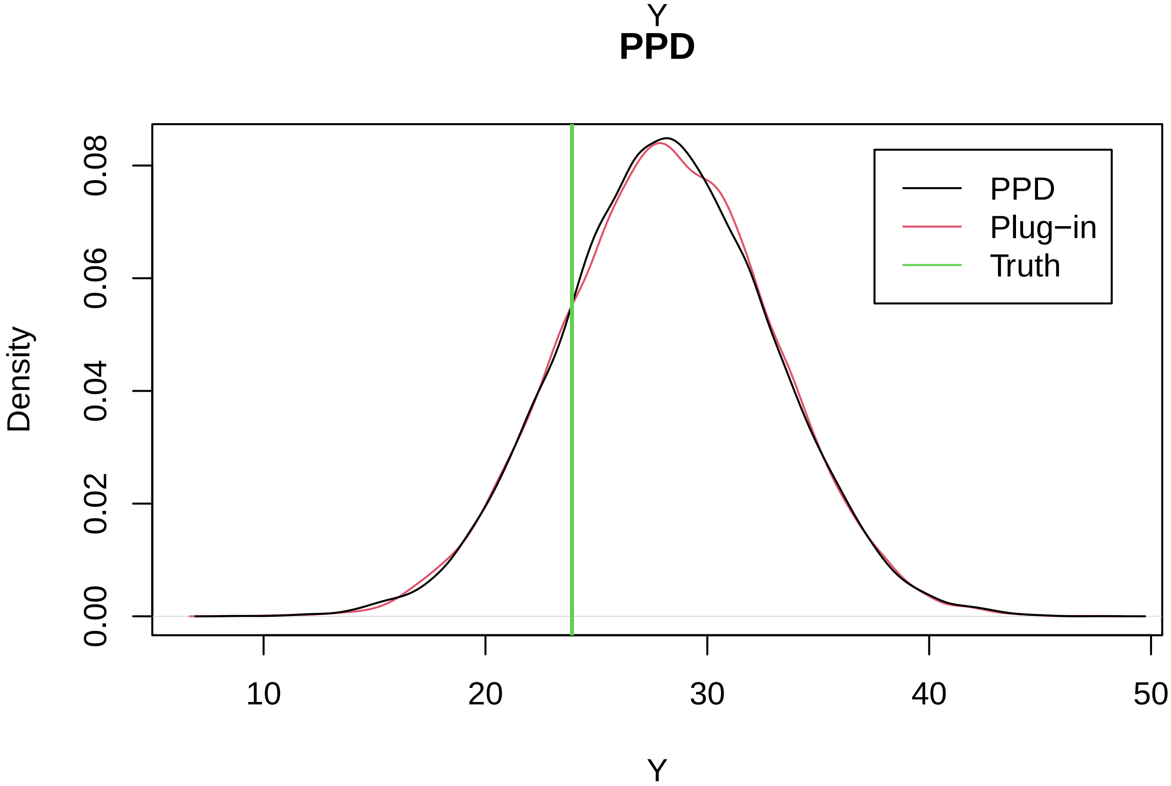
```
## beta[4]   -0.7166 -0.1648  0.12585  0.4115  0.9313
## beta[5]    0.2316  0.5181  0.66376  0.8151  1.0892
## beta[6]   -2.8499 -2.2600 -1.96029 -1.6612 -1.1069
## beta[7]    2.0898  2.4665  2.67337  2.8699  3.2439
## beta[8]   -0.6666 -0.2003  0.05323  0.3056  0.7788
## beta[9]   -4.0195 -3.4653 -3.18352 -2.9054 -2.3596
## beta[10]   1.4392  2.1520  2.53235  2.9281  3.6797
## beta[11]  -3.3559 -2.5402 -2.11879 -1.6997 -0.9414
## beta[12]  -2.4825 -2.1112 -1.91482 -1.7193 -1.3400
## beta[13]   0.3710  0.6874  0.85069  1.0194  1.3332
## beta[14]  -4.5702 -4.0985 -3.84948 -3.6012 -3.1243
## sigma      4.4573  4.6402  4.73874  4.8412  5.0553
```

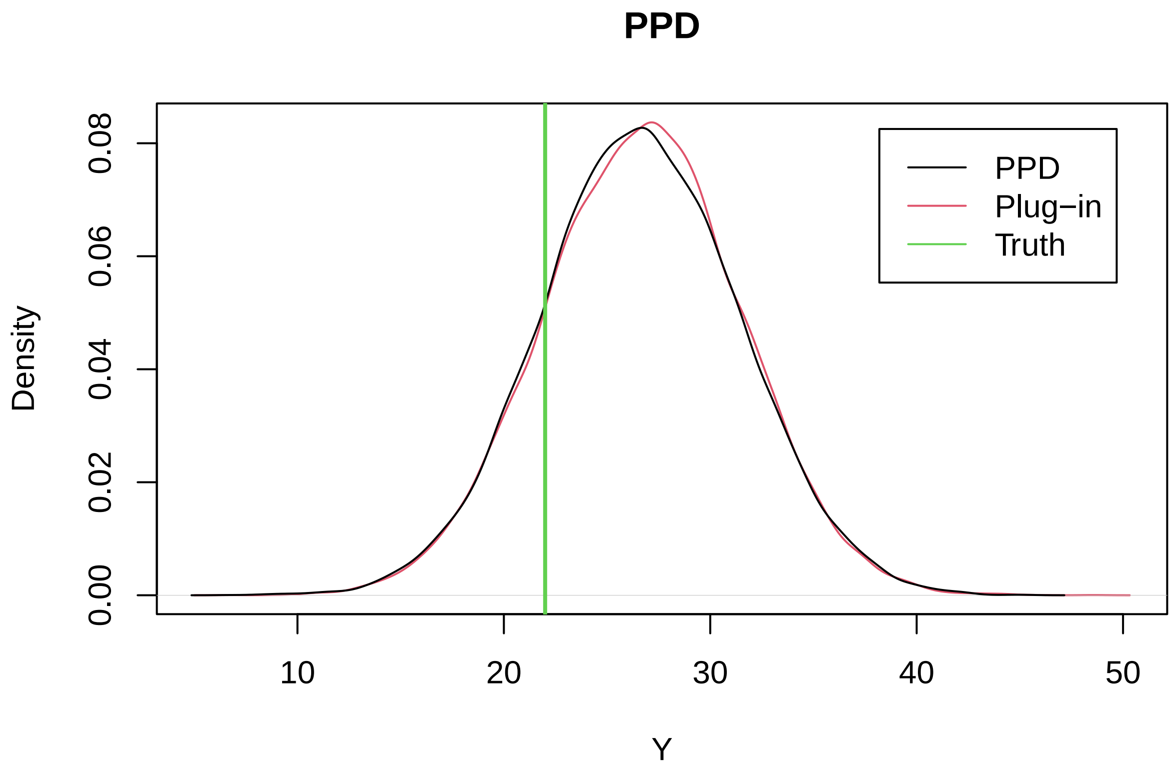
```
samps       <- samp[[1]]
 Y_test.samps    <- samps[,1:n_test]
 alpha.samps <- samps[,n_test+1]
 beta.samps  <- samps[,n_test+1+1:p]
 sigma.samps <- samps[,ncol(samps)]

# Compute the posterior mean for the plug-in predictions

 beta.mn  <- colMeans(beta.samps)
 sigma.mn <- mean(sigma.samps)
 alpha.mn <- mean(alpha.samps)


# Plot the PPD and plug-in

 for(j in 1:6){

    # Plug-in
    mu <- alpha.mn+sum(X_test[j,]*beta.mn)
    y  <- rnorm(20000,mu,sigma.mn)
    plot(density(y),col=2,xlab="Y",main="PPD")

    # PPD
    lines(density(Y_test.samps[,j]))

    # Truth
    abline(v=Y_test[j],col=3,lwd=2)

    legend("topright",c("PPD","Plug-in","Truth"),col=1:3,lty=1,inset=0.05)
 }
```

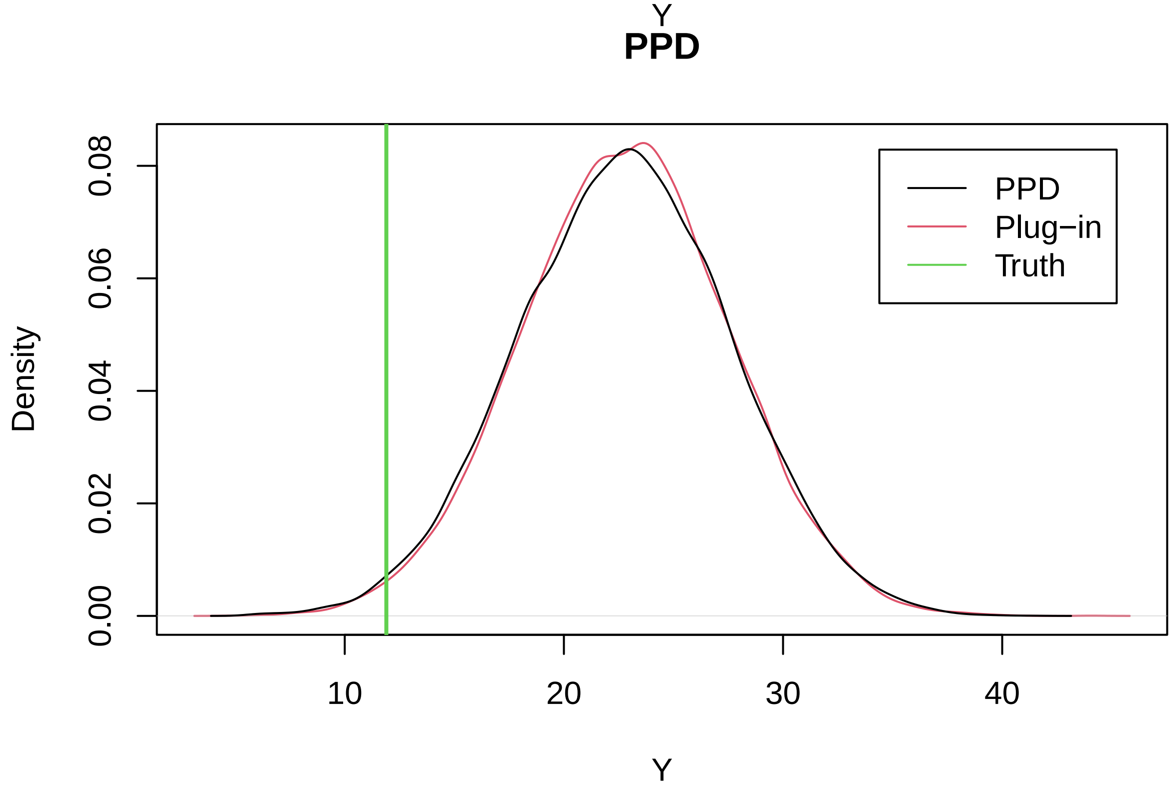

PPD
Density
0.00
0.02
0.04
0.06
0.08
0
10
20
30
40
Y
PPD
Plug−in
Truth


PPD
Density
0.00
0.02
0.04
0.06
0.08
10
20
30
40
Y
PPD
Plug−in
Truth


PPD
Density
0.00
0.02
0.04
0.06
0.08
0
10
20
30
40
Y
PPD
Plug−in
Truth


PPD
Density
0.00
0.02
0.04
0.06
0.08
10
20
30
40
50
Y
PPD
Plug−in
Truth





From plots we observe that both plug-in prediction and PPD give reasonable predictions.