RB_HW9

Robin Baldeo

March 30, 2019

```
library("rjags")
library("MASS")
```

Question 1

```
y1<- c(2,-3.1, -1, .2, .3, .4)
y2<- c(-3.5, -1.6, -4.6, -.9, -5.1, .1)

#using frequentist way
t.test(y1,y2)

##
##  Welch Two Sample t-test
##
## data:  y1 and y2
## t = 2.164, df = 9.5951, p-value = 0.05685
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.08533306  4.88533306
## sample estimates:
## mean of x mean of y
##      -0.2      -2.6

#mean
y1bar<- mean(y1)
y2bar<- mean(y2)

# baysian using formula for pop variance unkown
s21<- var(y1)

s22<- var(y2)

sp<- sqrt((s21 + s22)/ 2)

# using t table degeee of freedom n1 + n2 = 2.179 from t table
#low bound
(y1bar- y2bar) - 2.179 * sp*sqrt(1/length(y1) +1/length(y2)  )

## [1] -0.0166279

# upper bound
(y1bar- y2bar) + 2.179 * sp* sqrt(1/length(y1) + 1/length(y2) )
```

```
## [1] 4.816628
```

Based on the results I don't think the prior has too much of an effect. When compared to the conventional t- test I get a ci of [-.085, 4.8853] which is close to the Bayesian Ci [-.016, 4.82]. Both methods signify that there is no difference between the placebo and treatment at the 95% ci, because 0 is within the interval.

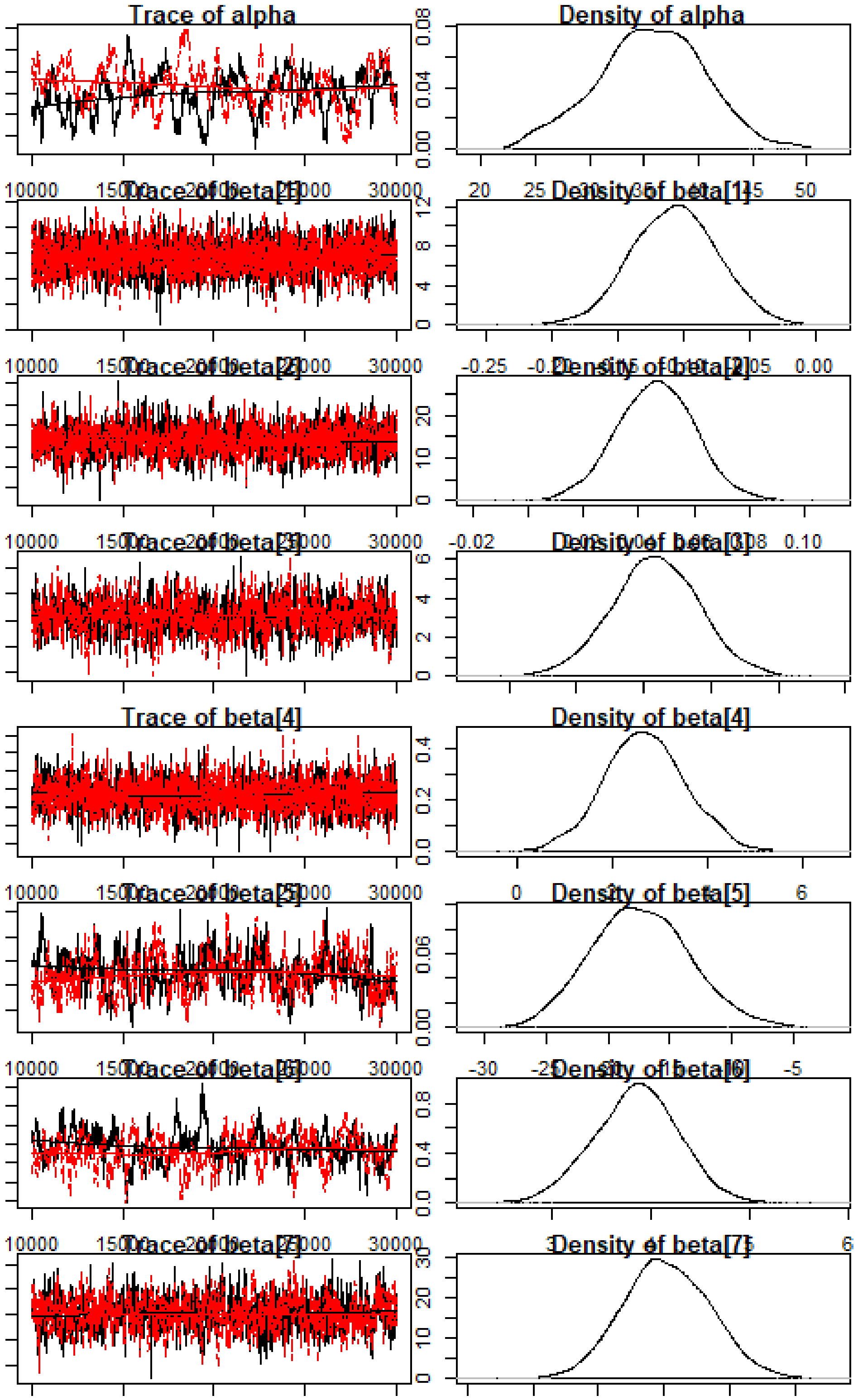
Question 2

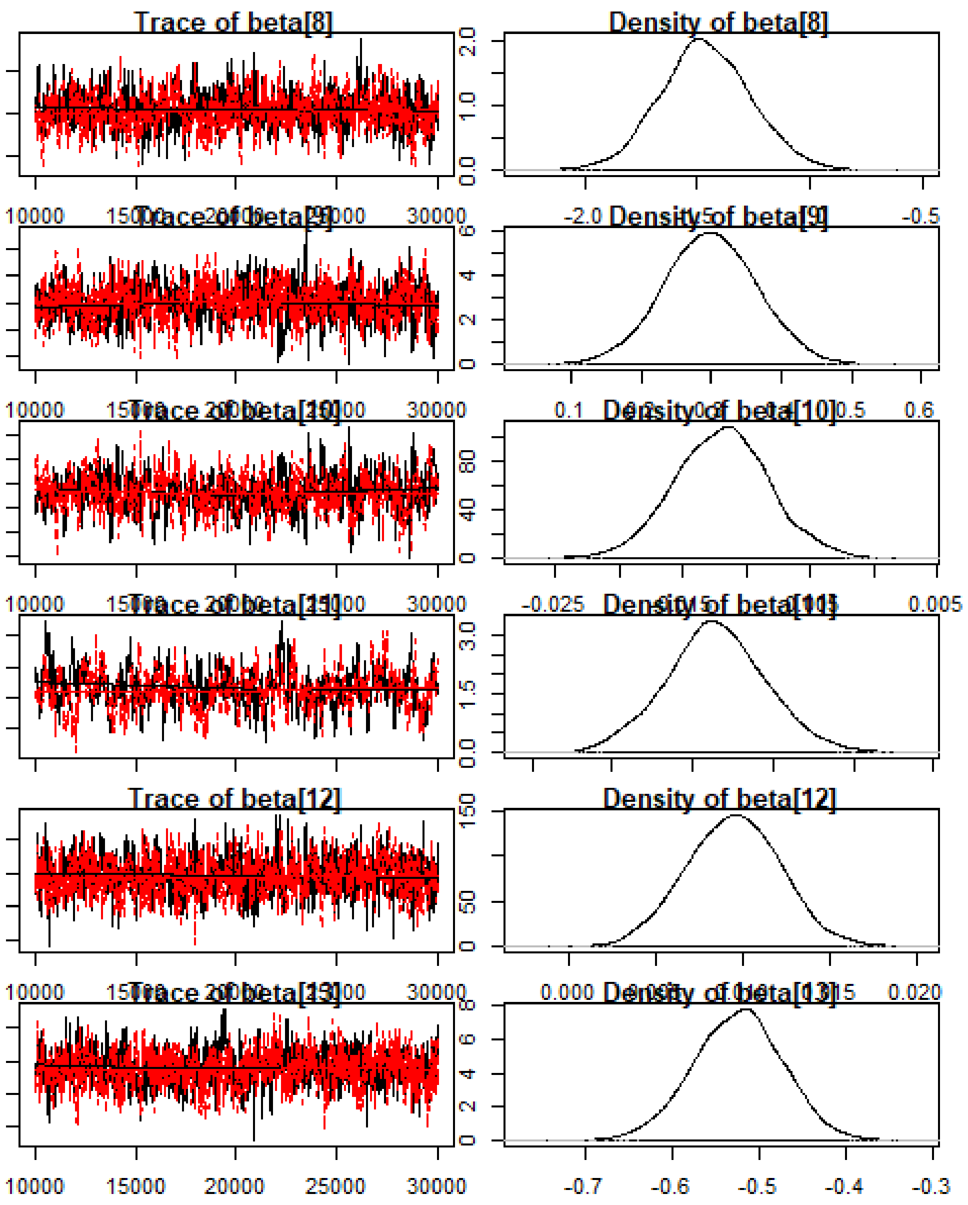
(A)

```
par(mar=c(1,1,1,1))
data(Boston)

#View(Boston)

#response
y<- Boston$medv

#covar
x<- Boston[1:13]

x<- as.matrix(x)


n <- length(y)
p <- ncol(x)

# building a list
data   <- list(Y=y,X=x,n=n,p=p)

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] ~ dnorm(0,0.001)
     }
     alpha ~ dnorm(0,0.001)
     taue  ~ dgamma(0.1, 0.1)
}")


model <- jags.model(model_string,data = data, n.chains=2,quiet=TRUE)

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

```r
params <- c("beta", "alpha")
samples <- coda.samples(model,variable.names=params,n.iter=20000,thin=10,  pr
ogress.bar="none")

#summary of beta and alpha
summary(samples)

##
## Iterations = 10010:30000
## Thinning interval = 10
## Number of chains = 2
## Sample size per chain = 2000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##                 Mean       SD  Naive SE Time-series SE
## alpha      3.560e+01 4.867957 7.697e-02      6.323e-01
## beta[1]   -1.072e-01 0.032169 5.086e-04      5.264e-04
## beta[2]    4.613e-02 0.013987 2.212e-04      3.828e-04
## beta[3]    1.714e-02 0.064434 1.019e-03      2.135e-03
## beta[4]    2.684e+00 0.859901 1.360e-02      1.516e-02
## beta[5]   -1.736e+01 3.899600 6.166e-02      2.765e-01
## beta[6]    3.852e+00 0.429029 6.784e-03      3.745e-02
## beta[7]    7.588e-04 0.013161 2.081e-04      4.760e-04
## beta[8]   -1.462e+00 0.202221 3.197e-03      8.279e-03
## beta[9]    2.989e-01 0.066205 1.047e-03      2.865e-03
## beta[10] -1.197e-02 0.003744 5.920e-05      1.930e-04
## beta[11] -9.413e-01 0.123762 1.957e-03      8.345e-03
## beta[12]  9.469e-03 0.002698 4.266e-05      9.244e-05
## beta[13] -5.241e-01 0.051842 8.197e-04      2.154e-03
##
## 2. Quantiles for each variable:
##
##                2.5%        25%        50%        75%     97.5%
## alpha     25.47738  32.432465  3.571e+01  38.971992 44.735563
## beta[1]   -0.16931  -0.129230 -1.070e-01  -0.085091 -0.044584
## beta[2]    0.01785   0.036586  4.632e-02   0.055562  0.073544
## beta[3]   -0.11172  -0.024989  1.764e-02   0.060718  0.144375
## beta[4]    0.98291   2.103714  2.668e+00   3.248996  4.360080
## beta[5]  -24.65141 -20.076636 -1.746e+01 -14.775249 -9.390641
## beta[6]    2.99703   3.568384  3.864e+00   4.134161  4.683054
## beta[7]   -0.02438  -0.008209  3.763e-04   0.009916  0.026580
## beta[8]   -1.85341  -1.593139 -1.468e+00  -1.330152 -1.056005
## beta[9]    0.16672   0.254361  2.989e-01   0.343577  0.427336
## beta[10]  -0.01936  -0.014476 -1.192e-02  -0.009492 -0.004499
## beta[11]  -1.18458  -1.023139 -9.440e-01  -0.860659 -0.688820
## beta[12]   0.00417   0.007648  9.497e-03   0.011304  0.014795
## beta[13]  -0.62739  -0.558808 -5.229e-01  -0.489942 -0.425297
```

```r
# convergence diagnostics
# plots
plot(samples)
```





```
# Low ESS indicates poor convergence, beta10 and the intercept have low sampl
e size
effectiveSize(samples)
```

```
##      alpha    beta[1]    beta[2]    beta[3]    beta[4]    beta[5] 
##   57.80722 3749.15313 1337.63562  940.72272 3266.49737  197.99810 
##    beta[6]    beta[7]    beta[8]    beta[9]   beta[10]   beta[11] 
##  128.59837  806.96170  597.18940  546.15103  376.87984  220.28229 
##   beta[12]   beta[13] 
##  850.80605  579.33063
```

```
# R greater than 1.1 indicates poor convergence, therefore we have good conve
rgence.
gelman.diag(samples)
```

```
## Potential scale reduction factors:
## 
##          Point est. Upper C.I.
## alpha          1.01       1.02
## beta[1]        1.00       1.00
## 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]        1.01       1.03
## beta[7]        1.00       1.02
## beta[8]        1.00       1.01
## beta[9]        1.00       1.00
## beta[10]       1.00       1.00
## beta[11]       1.00       1.00
## beta[12]       1.00       1.00
## beta[13]       1.00       1.00
## 
## Multivariate psrf
## 
## 1.01
```

My sample size for ptratio appear a bit low, but according to the plots and the gelman test there appears to be good convergence.

(B)

```
#liner model
lsModel<- lm(medv~., data= Boston)
summary(lsModel)
```

```
## 
## Call:
## lm(formula = medv ~ ., data = Boston)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -15.595  -2.730  -0.518   1.777  26.199 
## 
## Coefficients:
```

```
##               Estimate Std. Error t value Pr(>|t|)    
## (Intercept)  3.646e+01  5.103e+00   7.144 3.28e-12 ***
## crim        -1.080e-01  3.286e-02  -3.287 0.001087 ** 
## zn           4.642e-02  1.373e-02   3.382 0.000778 ***
## indus        2.056e-02  6.150e-02   0.334 0.738288    
## chas         2.687e+00  8.616e-01   3.118 0.001925 ** 
## nox         -1.777e+01  3.820e+00  -4.651 4.25e-06 ***
## rm           3.810e+00  4.179e-01   9.116  < 2e-16 ***
## age          6.922e-04  1.321e-02   0.052 0.958229    
## dis         -1.476e+00  1.995e-01  -7.398 6.01e-13 ***
## rad          3.060e-01  6.635e-02   4.613 5.07e-06 ***
## tax         -1.233e-02  3.760e-03  -3.280 0.001112 ** 
## ptratio     -9.527e-01  1.308e-01  -7.283 1.31e-12 ***
## black        9.312e-03  2.686e-03   3.467 0.000573 ***
## lstat       -5.248e-01  5.072e-02 -10.347  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 4.745 on 492 degrees of freedom
## Multiple R-squared:  0.7406, Adjusted R-squared:  0.7338 
## F-statistic: 108.1 on 13 and 492 DF,  p-value: < 2.2e-16
```

```
confint(lsModel)
```

```
##                     2.5 %       97.5 %
## (Intercept)  26.432226009  46.486750761
## crim         -0.172584412  -0.043438304
## zn            0.019448778   0.073392139
## indus        -0.100267941   0.141385193
## chas          0.993904193   4.379563446
## nox         -25.271633564 -10.261588893
## rm            2.988726773   4.631003640
## age          -0.025262320   0.026646769
## dis          -1.867454981  -1.083678710
## rad           0.175692169   0.436406789
## tax          -0.019723286  -0.004945902
## ptratio      -1.209795296  -0.695699168
## black         0.004034306   0.014589060
## lstat        -0.624403622  -0.425113133
```

Comparing the Bayesian linear regression model in part a (uninformative prior) to the frequentist linear regression model. I don't see in major difference in the parameter means , sd and ci. It appears that both methods produce near identical results.

(C)

```
#model with double exponential prior
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] ~ dnorm(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")


params <- c("beta", "alpha")
samples <- coda.samples(model,variable.names=params,n.iter=20000,thin=10,  pr
ogress.bar="none")

#summary data
summary(samples)

##
## Iterations = 10010:30000
## Thinning interval = 10
## Number of chains = 2
## Sample size per chain = 2000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##                 Mean       SD  Naive SE Time-series SE
## alpha    27.831367 4.250489 6.721e-02      5.318e-01
## beta[1]  -0.101683 0.032434 5.128e-04      5.230e-04
## beta[2]   0.049216 0.013777 2.178e-04      3.737e-04
## beta[3]  -0.036235 0.059961 9.481e-04      2.072e-03
## beta[4]   2.086782 0.814635 1.288e-02      1.480e-02
## beta[5]  -3.579076 2.501056 3.955e-02      1.260e-01
## beta[6]   3.746640 0.400827 6.338e-03      3.295e-02
## beta[7]  -0.009966 0.012987 2.053e-04      4.595e-04
## beta[8]  -1.261064 0.194179 3.070e-03      7.544e-03
## beta[9]   0.281388 0.064072 1.013e-03      2.664e-03
## beta[10] -0.013873 0.003768 5.958e-05      1.895e-04
## beta[11] -0.807703 0.123789 1.957e-03      8.516e-03
## beta[12]  0.010007 0.002792 4.415e-05      9.886e-05
## beta[13] -0.554019 0.051443 8.134e-04      1.873e-03
##
```

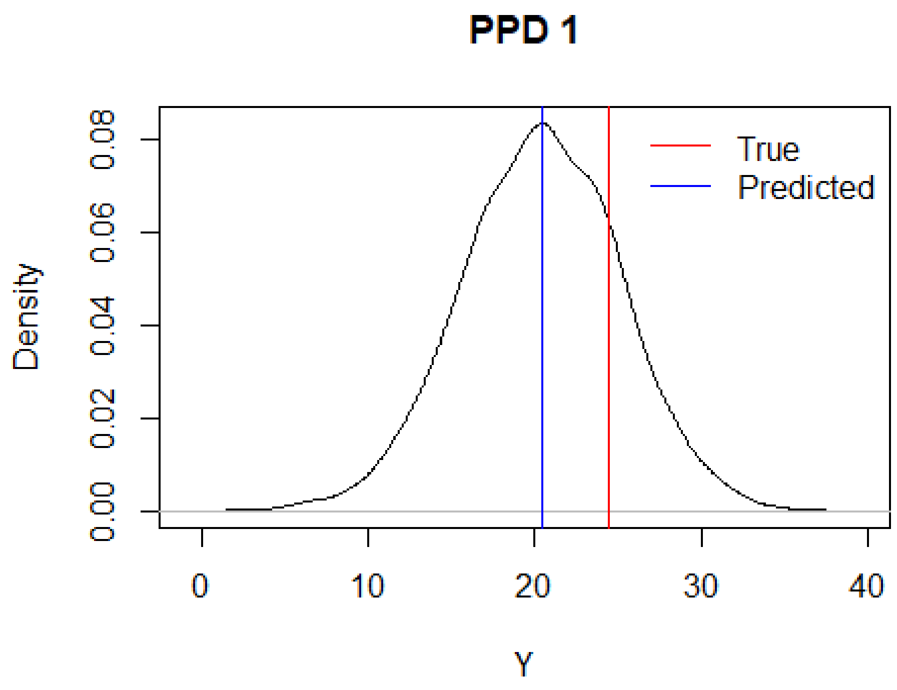
```
## 2. Quantiles for each variable:
##
##                2.5%       25%       50%       75%     97.5%
## alpha    20.217494 24.670462 27.678692 30.846340 36.349122
## beta[1]  -0.165496 -0.123039 -0.101379 -0.079803 -0.040353
## beta[2]   0.023006  0.039930  0.049094  0.058270  0.075835
## beta[3]  -0.154627 -0.078353 -0.036328  0.005273  0.080614
## beta[4]   0.540570  1.525112  2.076819  2.632478  3.718103
## beta[5]  -9.790344 -4.907032 -3.216946 -1.802903  0.289694
## beta[6]   2.946193  3.469627  3.750413  4.021908  4.523951
## beta[7]  -0.035712 -0.018676 -0.009768 -0.001108  0.015120
## beta[8]  -1.651951 -1.392012 -1.260056 -1.131954 -0.877538
## beta[9]   0.157248  0.237572  0.280705  0.322946  0.411491
## beta[10] -0.021446 -0.016334 -0.013850 -0.011401 -0.006524
## beta[11] -1.069384 -0.889713 -0.801395 -0.721662 -0.577755
## beta[12]  0.004386  0.008163  0.009995  0.011899  0.015507
## beta[13] -0.657355 -0.587783 -0.553368 -0.519917 -0.455526
```

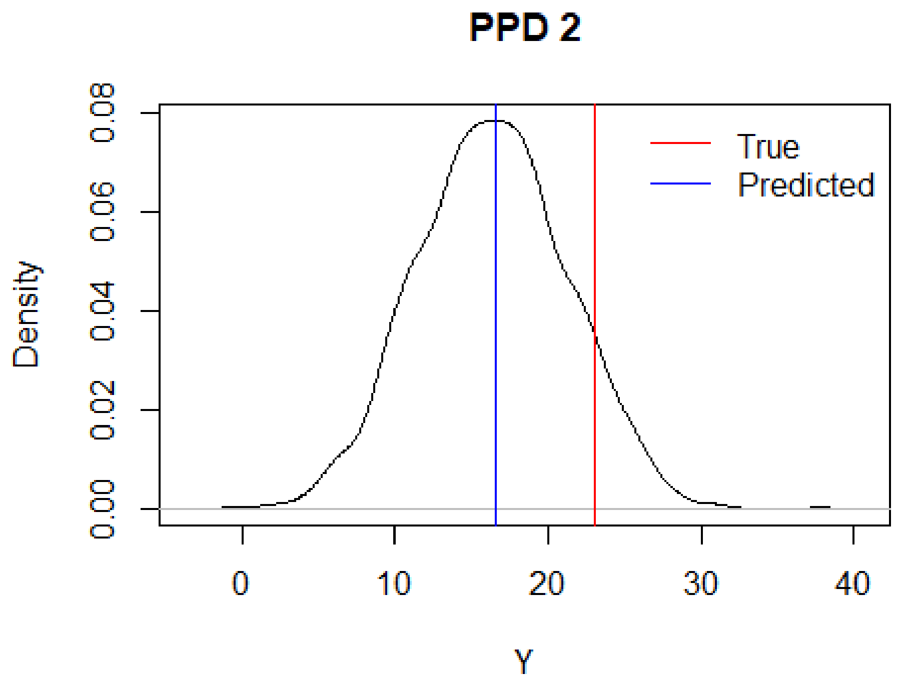
Comparing the Bayesian linear regression model in part a (uninformative prior) to part c, we see the all the summary values are different. It appears the prior has a greater effect on the data than in part a. Also, the ci in part c appears much more narrow , the means and sd for each parameter appears much smaller when compared to part a.

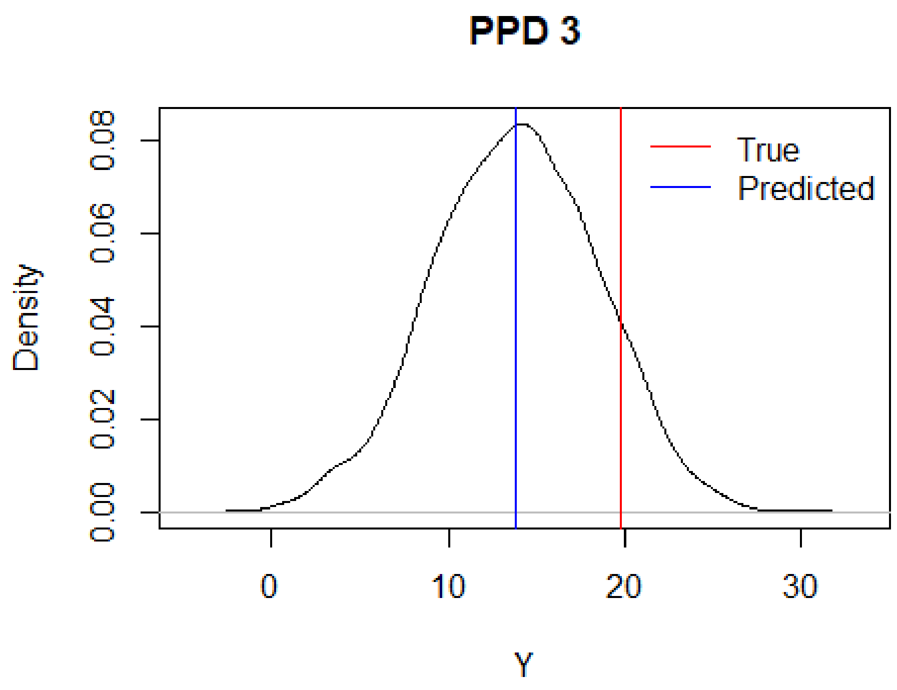
(D)

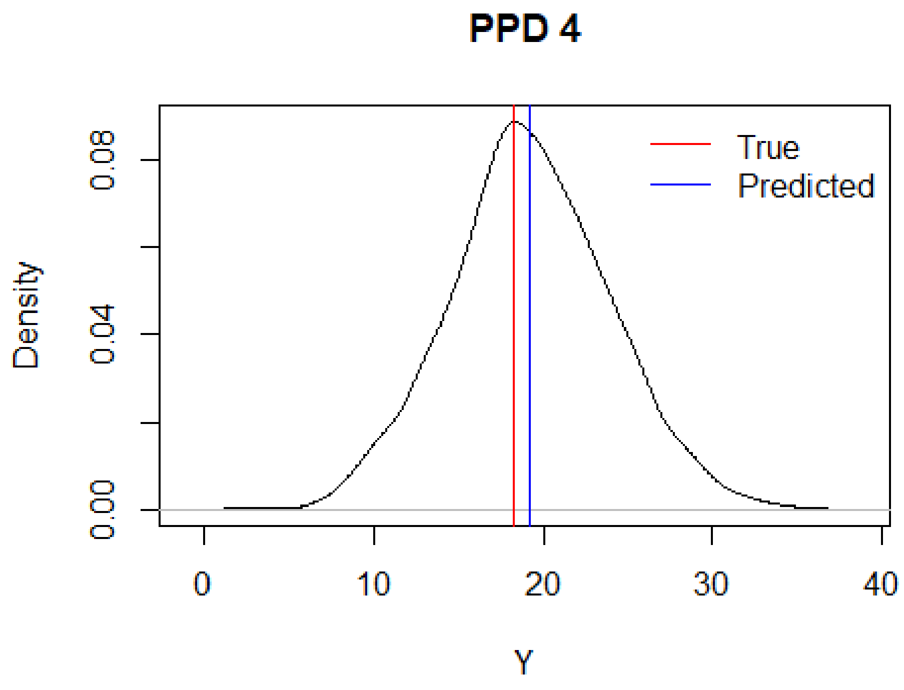
```
##data
#taking the frist 500 rows
y<- Boston$medv[1:500]

#taking the covar, 500 rows
x<- (Boston[1:13])[1:500,]

##ppd data to be passed to JAGS
yPp<- y[495:500]
xPp<- x[495:500, ]
#scaling matrix
X_<- as.matrix(xPp)


#obs Data to be passed to JAGS
Y<- y[1:494]
xob<- x[1:494, ]
#scaling
X<- as.matrix(xob)


# of obs in obs matrix
n <- length(Y)
p <- ncol(xob)
```

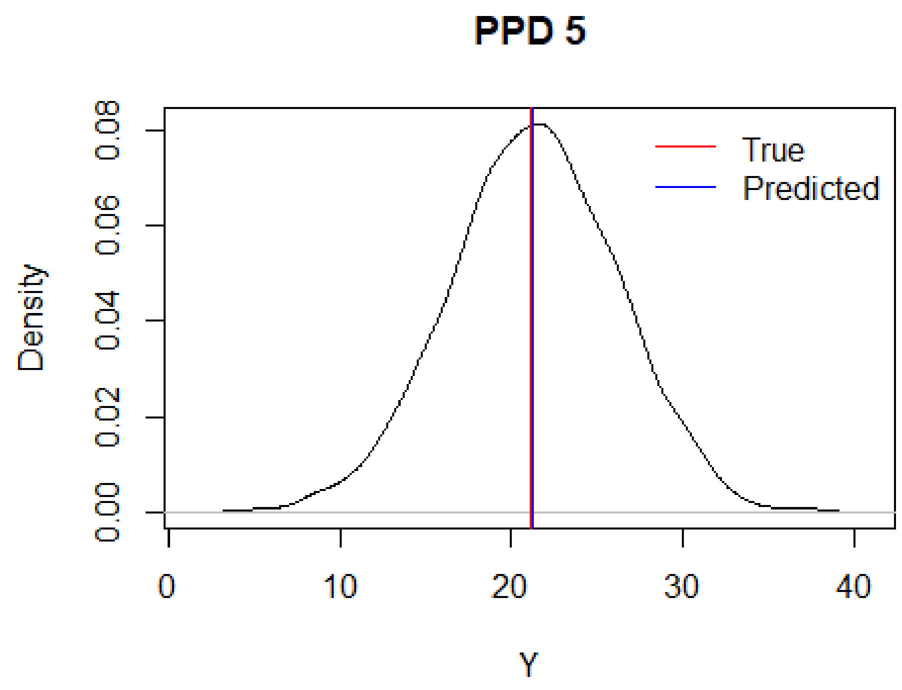
```
# of obs in predi matrix
n_<- length(yPp)

data   <- list(Y=Y,X=X,n=n,p=p, n_= n_, X_= X_)

#jags model
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] ~ dnorm(0,0.001)

   }

   alpha ~ dnorm(0,0.001)
   taue  ~ dgamma(0.1, 0.1)

  #prediction
   for(i in 1:n_){
    Y_[i] ~ dnorm(alpha+inprod(X_[i,],beta[]),taue)
   }

}")


model <- jags.model(model_string,data = data, n.chains=2,quiet=TRUE)
update(model, 10000, progress.bar="none")
params <- c("Y_")

samples <- coda.samples(model,variable.names=params,n.iter=20000,thin=10,  pr
ogress.bar="none")

preM<- summary(samples)

pMean<- preM$statistics[,1]

sub<- samples[[1]]

#plot the ppd
for(i in 1:length(yPp)){
  plot(density(sub[,i]),xlab="Y",main=paste("PPD", i))
  #true means
  abline(v = yPp[i], col= "red")
  #pred means
  abline(v= pMean[i], col = "blue")
  legend("topright", legend = c("True", "Predicted"),col=c("red", "blue"), lt
y=c(1,1), bty = "n")
}
```

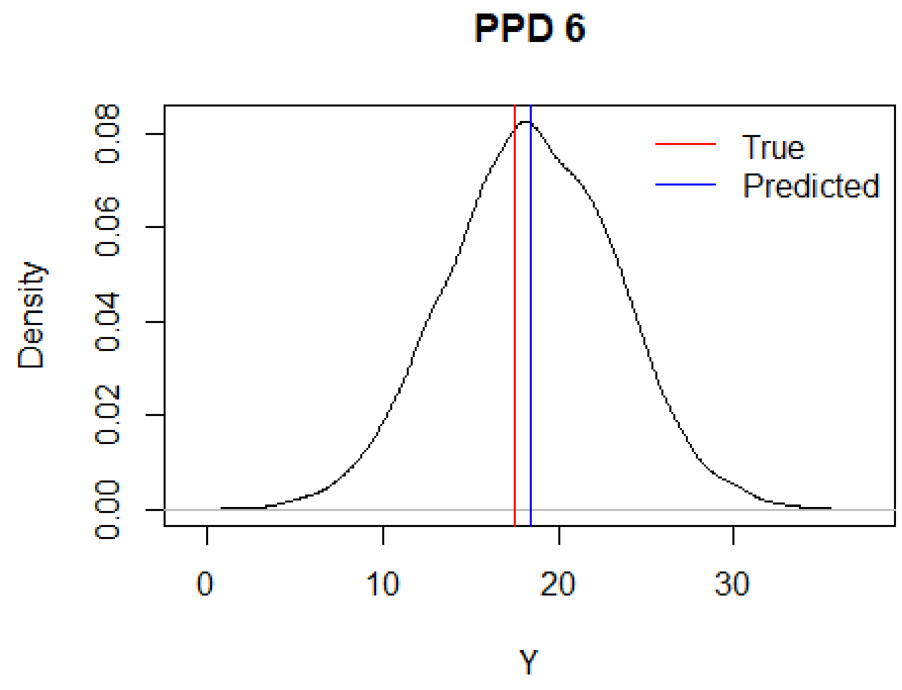





PPD 3
Density
Y
True
Predicted
0.00
0.02
0.04
0.06
0.08
0
10
20
30


PPD 4
Density
Y
True
Predicted
0.00
0.04
0.08
0
10
20
30
40





I think the predictions are reasonable, The above plots indicate this for each of the 6 plots showing the predicted means(blue) with the true means(red). As we can see these values are not too far apart.