Hawkins Exam 1

Stat 451

2/10/2021

For the first set of problems use the data file 'exam1-1.dat'. There are 24 data points in 8 treatments. We will assume the likelihood for the data is normal. You should assume that the variance is homoskedastic across treatments (that is, the variance is the same in all the treatments).

Use the following priors: Normal(mean=5,sd=100) for the cell means, and a gamma(shape=1.5,rate=.5) for the variance. Besides examining the posteriors of the eight cell means and the variance, you will also be examining three other functions of the parameters: (1) the average of the first four cell means, (2) the average of the last four cell means, and (3) the average of the last four cell means minus the average of the first four cell means.

1. Write the JAGS code necessary to produce posterior chains for the eight cell means and the variance. Put a set.seed(1234) command in the file prior to running the JAGS code so that we will all get the same answers. Run 4 chains with 11000 iterations per chain, a burnin of 1000 and thin by 4. This will result in 10000 samples. Print out the JAGS output file.

Read in Data

```
set1 <- read.table("exam1-1.dat", header = TRUE)
```

Run Anova

```
library(R2jags)

set.seed(1234)

mdl <- "
model {

  for(i in 1:24){
    y[i] ~ dnorm(mu[tmt[i]],1/s2)
  }

  for(i in 1:8){
    mu[i] ~ dnorm(5, 0.0001)
  }

  first4 = sum(mu[1] + mu[2] + mu[3] + mu[4])/4
  last4 = sum(mu[5] + mu[6] + mu[7] + mu[8])/4
  diff = last4-first4
  s2 ~ dgamma(1.5, .5)



}
"
# curve(dnorm(x, 5 , sqrt(10000)), from = -150, to = 200)
# curve(dgamma(x, 1.5, .5))

tmt <- set1$tmt
y <- set1$y

writeLines(mdl, 'exam1.txt')
```

```
data.jags <- c('y','tmt')
parms <- c('mu', 's2', 'first4', 'last4', 'diff')

exam1.sim <- jags(data = data.jags, inits = NULL,
                  parameters.to.save = parms,
                  model.file = 'exam1.txt',
                  n.iter = 11000,
                  n.burnin = 1000,
                  n.chains = 4,
                  n.thin = 4)
```

```
 module glm loaded

 Compiling model graph
    Resolving undeclared variables
    Allocating nodes
 Graph information:
    Observed stochastic nodes: 24
    Unobserved stochastic nodes: 9
    Total graph size: 71

 Initializing model
```

```
# Here is the jags output file, it has 10000 samples
exam1.sim
```

```
 Inference for Bugs model at "exam1.txt", fit using jags,
  4 chains, each with 11000 iterations (first 1000 discarded), n.thin = 4
  n.sims = 10000 iterations saved
          mu.vect sd.vect   2.5%    25%    50%    75%  97.5%  Rhat n.eff
 diff       2.483   0.594  1.317  2.091  2.487  2.869  3.663 1.001 10000
 first4     3.242   0.422  2.414  2.967  3.240  3.520  4.083 1.001  6700
 last4      5.724   0.423  4.903  5.448  5.721  5.998  6.567 1.001 10000
 mu[1]      2.111   0.850  0.436  1.563  2.100  2.670  3.796 1.001 10000
 mu[2]      3.099   0.845  1.425  2.524  3.108  3.651  4.806 1.001  7000
 mu[3]      3.045   0.830  1.410  2.507  3.048  3.590  4.678 1.001 10000
 mu[4]      4.712   0.835  3.064  4.152  4.705  5.256  6.356 1.001 10000
 mu[5]      5.197   0.842  3.523  4.647  5.202  5.740  6.888 1.001 10000
 mu[6]      6.097   0.842  4.435  5.564  6.097  6.633  7.785 1.001 10000
 mu[7]      5.881   0.845  4.175  5.338  5.880  6.428  7.552 1.001  8900
 mu[8]      5.721   0.838  4.046  5.175  5.724  6.277  7.346 1.001  8700
 s2         2.129   0.799  1.042  1.572  1.967  2.504  4.067 1.001 10000
 deviance  83.918   5.258 75.948 80.022 83.152 87.072 96.095 1.001  5300

 For each parameter, n.eff is a crude measure of effective sample size,
 and Rhat is the potential scale reduction factor (at convergence, Rhat=1).

 DIC info (using the rule, pD = var(deviance)/2)
 pD = 13.8 and DIC = 97.7
 DIC is an estimate of expected predictive error (lower deviance is better).
```

2. Using coda verify that the chains produced for the eight cell means, the variance, and the three functions of the parameters described above are appropriate for further analysis by showing that the effective sample size for each chain exceeds 5000.

The effective sizes for the difference, first4, last4, mus, and variance are all above 5000 so we are good.

```
library(coda)

sims <- as.mcmc(exam1.sim)
chains <- as.matrix(sims)
sims <- as.mcmc(chains)

effectiveSize(sims)
```

```
  deviance      diff    first4     last4     mu[1]     mu[2]     mu[3]     mu[4]
  8432.137 10000.000 10000.000 10000.000 10000.000  9927.770 10000.000 10000.000
     mu[5]     mu[6]     mu[7]     mu[8]        s2
 11107.053 10000.000 10000.000 10000.000  7218.926
```

3. Using coda verify that the chains produced for the eight cell means, the variance, and the three functions of the parameters described above are appropriate for further analysis by showing that the Raftery-Lewis diagnostic for each chain is smaller than 3.

Using Ratery-Lewis, the dependence factor of the 3 functions, the mus, and the variance are all below 3.

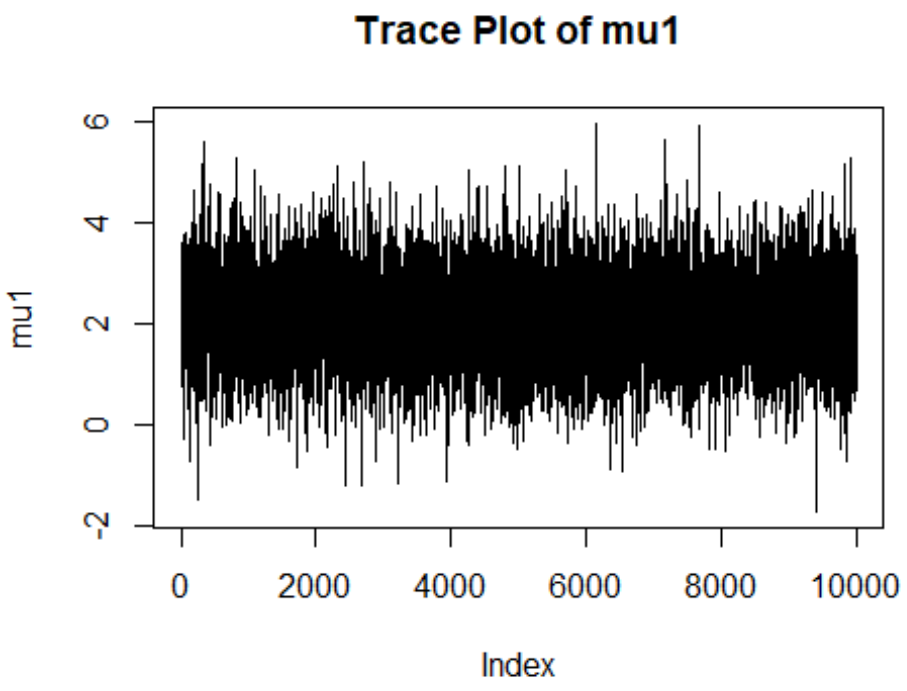
```
library(coda)
raftery.diag(sims)
```

```
 Quantile (q) = 0.025
 Accuracy (r) = +/- 0.005
 Probability (s) = 0.95

           Burn-in  Total Lower bound  Dependence
           (M)      (N)   (Nmin)       factor (I)
  deviance 2        3680  3746         0.982
  diff     2        3680  3746         0.982
  first4   2        3680  3746         0.982
  last4    2        3650  3746         0.974
  mu[1]    2        3865  3746         1.030
  mu[2]    2        3802  3746         1.010
  mu[3]    2        3962  3746         1.060
  mu[4]    2        3834  3746         1.020
  mu[5]    2        3710  3746         0.990
  mu[6]    2        3771  3746         1.010
  mu[7]    2        3771  3746         1.010
  mu[8]    2        3802  3746         1.010
  s2       2        3834  3746         1.020
```

4. Produce the trace plot for the mean parameter of treatment 1.

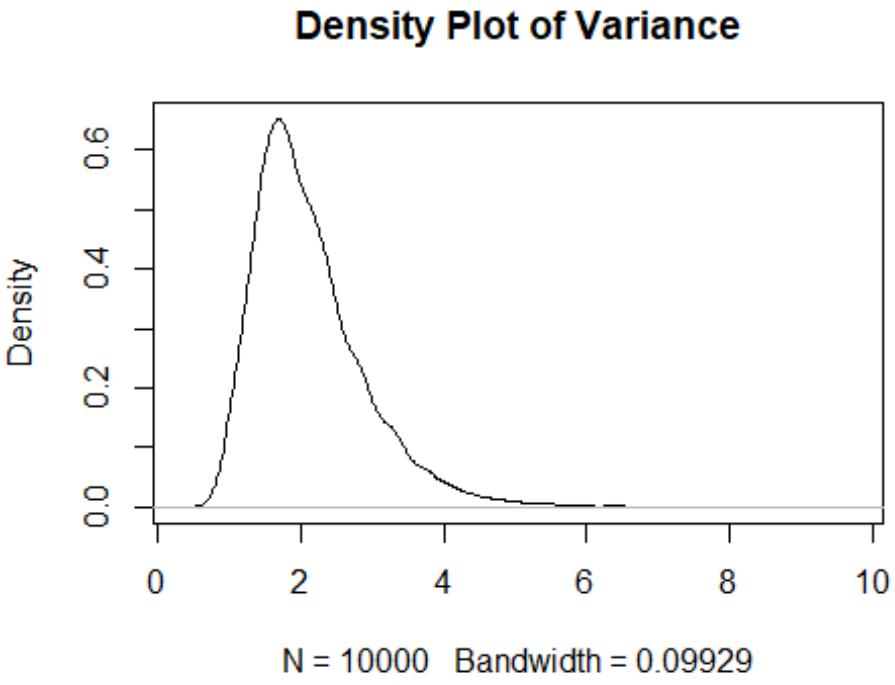
Trace plot of mu1 shows no trends.

```
mu1 <- chains[,5]
plot(mu1, type = 'l', main = "Trace Plot of mu1")
```



5. Produce the density plot for the variance parameter.

```
plot(density(chains[,13]), main = "Density Plot of Variance")
```



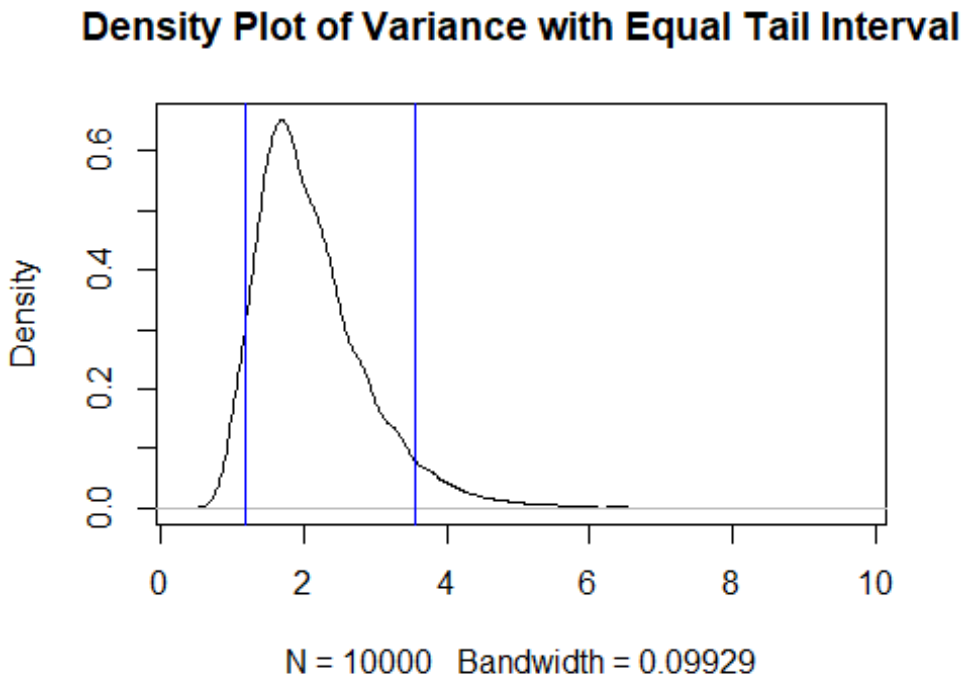
6. What is the equal tail 95% posterior probability interval of the variance.

95% Equal tail posterior probability interval is (1.17, 3.55)

```
library(bayestestR)

ci(sims, method = "ETI")
```

```
# Equal-Tailed Interval

Parameter |        89% ETI
--------------------------
deviance  | [76.85, 93.39]
diff      | [ 1.55,  3.42]
first4    | [ 2.57,  3.91]
last4     | [ 5.05,  6.41]
mu[1]     | [ 0.76,  3.45]
mu[2]     | [ 1.77,  4.46]
mu[3]     | [ 1.73,  4.36]
mu[4]     | [ 3.40,  6.05]
mu[5]     | [ 3.87,  6.54]
mu[6]     | [ 4.76,  7.45]
mu[7]     | [ 4.53,  7.24]
mu[8]     | [ 4.38,  7.06]
s2        | [ 1.17,  3.55]
```

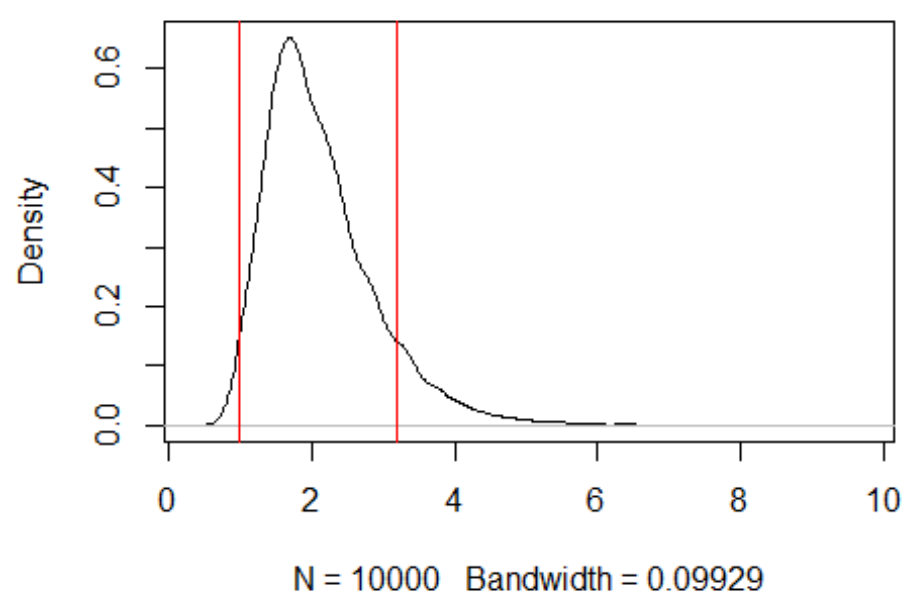
```
plot(density(chains[,13]), main = "Density Plot of Variance with Equal Tail Interval")
abline(v =1.17, col = 'blue')
abline(v= 3.55, col = 'blue')
```



7. What is the highest posterior density 95% interval of the variance.

highest posterior density of the 95% interval is (0.97, 3.19)

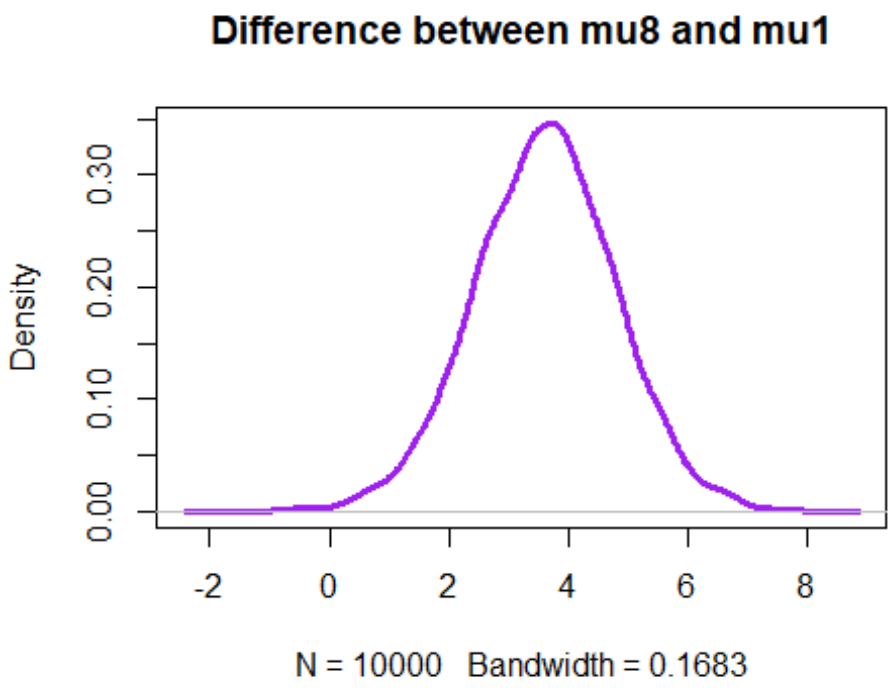
```
plot(density(chains[,13]), main = "Density Plot of Variance with Highest Posterior Density
Interval")
abline(v = 0.97, col = 'red')
abline(v = 3.19, col = 'red')
```




```
ci(sims, method = "HDI")

 # Highest Density Interval

 Parameter |        89% HDI
 --------------------------
 deviance  | [75.78, 91.60]
 diff      | [ 1.55,  3.41]
 first4    | [ 2.57,  3.90]
 last4     | [ 5.00,  6.35]
 mu[1]     | [ 0.80,  3.48]
 mu[2]     | [ 1.79,  4.47]
 mu[3]     | [ 1.65,  4.27]
 mu[4]     | [ 3.39,  6.03]
 mu[5]     | [ 3.86,  6.53]
 mu[6]     | [ 4.74,  7.42]
 mu[7]     | [ 4.61,  7.30]
 mu[8]     | [ 4.38,  7.05]
 s2        | [ 0.97,  3.19]
```

8. Say we want to know if the mean for treatment 1 is different than the mean of treatment 8. Compute the chain that represents the difference of the mean of treatment 8 minus the mean of treatment 1. Plot the density of this chain.

```
mu1 <- chains[,5]
mu8 <- chains[,12]

diffmus <- mu8-mu1
plot(density(diffmus), main = "Difference between mu8 and mu1", col = 'purple', lwd = 3)
```



9. Would you conclude the mean of treatment 8 exceeds the mean of treatment 1? Why?

Yes I would conclude tha mu8 > mu1 because the density plot of the differences shows it's positive and doesn't span 0. A 95% quantile confirms that conclusion.

```
quantile(diffmus, probs = c(.025, .975))
```

```
     2.5%    97.5%
 1.234603 5.939878
```

10. Compute pD using the JAGS formula using one of the chains you have already produced. That is, you are computing pD yourself, not just reading it from the output.

to calculate pD, I take the variance of the deviance and divide it by 2. This gives me 13.82. Which matches the output.

```
var(chains[,1])/2
```

```
 [1] 13.82269
```

11. How is the DIC for the model computed.

DIC is pD plus the mean deviance.

```
mean(chains[,1]) + var(chains[,1])/2
```

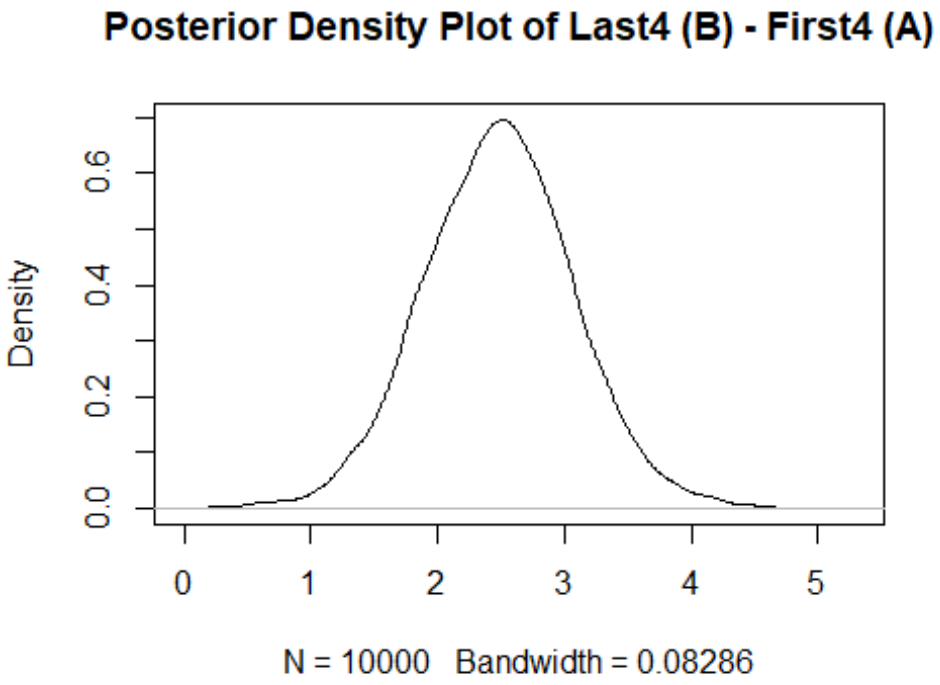
```
 [1] 97.74086
```

12. Say the first four treatment means represent 4 levels of a treatment. We'll call this treatment A. Say treatment means five through eight represent 4 levels of another treatment that we will call treatment B. Plot the posterior density for the combination of the parameters that you would use to test the assertion that treatment B yields higher responses that treatment A.

```
colnames(chains)
```

```
  [1] "deviance" "diff"     "first4"   "last4"    "mu[1]"    "mu[2]"
  [7] "mu[3]"    "mu[4]"    "mu[5]"    "mu[6]"    "mu[7]"    "mu[8]"
 [13] "s2"
```

```
last4 <- chains[,4]
first4 <- chains[,3]
plot(density(last4-first4), main = "Posterior Density Plot of Last4 (B) - First4 (A)")
```



13. Would you conclude treatment B yields higher responses than treatment A? Why?

Yes I conclude that treatment B yields a higher response than treatment A because the density plot of the differences is well above 0, centered around 2.5. So the difference is positive for sure.

B

For the next set of problems use the data file 'exam1-3.dat'. Use a normal likelihood as you would with a standard frequentist multiple regression. For these data we are attempting to predict Defective using Temperature, Density, and Rate. Use the square root of Defective as the dependent variable, and please standardize all variables (including the square root of Defective) prior to running any model.

14. For the model include only main effects for Temperature, Density, and Rate. Write code to solve the problem in Proc MCMC. Please include the Proc MCMC code in your output file so that it is readable. Produce posterior chains for the parameters you are estimating. Use normal priors for the β parameters, N(0,var=100), and use a gamma with shape of 1.1 and a scale of 1 for the variance. Also, set the seed value as 1234 so all output will be identical. Use 350000 for the number of iterations, 50000 for the burn in iterations, and thin by 50. Compute the DIC, Raftery-Louis diagnostics, and the effective sample size.

15.

```
#Standardize and write out

q2 <- read.table('exam1-3.dat', header = TRUE)
q2

q2$sqdef <- sqrt(q2$Defective)

q2$sqdef <- (q2$sqdef - mean(q2$sqdef))/sd(q2$sqdef)
q2$Temperature <- (q2$Temperature - mean(q2$Temperature))/sd(q2$Temperature)
q2$Density <- (q2$Density - mean(q2$Density))/sd(q2$Density)
q2$Rate <- (q2$Rate - mean(q2$Rate))/sd(q2$Rate)
q2$Case <- (q2$Case - mean(q2$Case))/sd(q2$Case)
```

```
q2 <- cbind(seq(1,30), q2)


write.table(q2[,c(-6)], 'standardq2.dat')
```

Sas Code. It runs, just doesn't knit.

library(SASmarkdown)

saspath <- "C:/Program Files/SASHome/SASFoundation/9.4/sas.exe" sasopts <- "-nosplash -ls 75" knitr::opts_chunk$set(engine="sas", engine.path=saspath, engine.opts=sasopts, comment=NA)

knitr::opts_chunk*get*()engine knitr::opts_chunk*get*()engine.path knitr::opts_chunk*get*()engine.opts

data q2; infile 'C:/Users/nateh/Documents/Stat 451/standardq2.dat'; input yes rep Case Temperature Density Rate sqdef; run;

proc mcmc data = q2 nbi = 50000 nmc = 350000 thin = 50 seed = 1234 outpost = 'C:/Users/nateh/Documents/Stat 451/exam1_q2.sas7bdat' dic propcov=quanew monitor=(*parms*) stats = all diagnostics = all; parms b0 0; parms btemp 0; parms bdensity 0; parms brate 0; parms vv 1.1;

prior b0 ~ normal(0, var = 100); prior btemp: ~ normal(0, var = 100); prior bdensity: ~ normal(0, var = 100); prior brate: ~ normal(0, var = 100);

prior vv ~ gamma(shape = 1.1, scale = 1.0);

mu = b0 + btemp*Temperature + bdensity*Density + brate*Rate; model sqdef ~ normal(mu, var = vv); run;

SAS OUTPUT

Parameters				
Block	**Parameter**	**Sampling Method**	**Initial Value**	**Prior Distribution**
1	**b0**	N-Metropolis	0	normal(0, var = 100)
2	**btemp**	N-Metropolis	0	normal(0, var = 100)
3	**bdensity**	N-Metropolis	0	normal(0, var = 100)
4	**brate**	N-Metropolis	0	normal(0, var = 100)
5	**vv**	N-Metropolis	1.1000	gamma(shape = 1.1, scale = 1.0)

Posterior Summaries						
Parameter	N	Mean	Standard Deviation	Percentiles		
				25	50	75
b0	7000	-0.00043	0.0503	-0.0331	-0.00079	0.0326
btemp	7000	0.4105	0.1912	0.2875	0.4113	0.5366
bdensity	7000	-0.4311	0.1987	-0.5628	-0.4283	-0.3009
brate	7000	0.1473	0.1337	0.0597	0.1473	0.2352
vv	7000	0.0757	0.0238	0.0592	0.0717	0.0871

Posterior Intervals					
Parameter	Alpha	Equal-Tail Interval		HPD Interval	
b0	0.050	-0.0997	0.0999	-0.0982	0.1006
btemp	0.050	0.0316	0.7859	0.0197	0.7711
bdensity	0.050	-0.8295	-0.0401	-0.8350	-0.0494
brate	0.050	-0.1157	0.4090	-0.1226	0.3968
vv	0.050	0.0427	0.1359	0.0382	0.1251

Posterior Correlation Matrix					
Parameter	**b0**	**btemp**	**bdensity**	**brate**	**vv**
b0	1.0000	0.0081	0.0136	0.0122	0.0091
btemp	0.0081	1.0000	0.7623	-0.2581	-0.0234
bdensity	0.0136	0.7623	1.0000	0.3779	-0.0186
brate	0.0122	-0.2581	0.3779	1.0000	0.0001
vv	0.0091	-0.0234	-0.0186	0.0001	1.0000

Posterior Covariance Matrix					
Parameter	**b0**	**btemp**	**bdensity**	**brate**	**vv**
b0	0.00253	0.000078	0.000135	0.000082	0.000011
btemp	0.000078	0.0366	0.0290	-0.00660	-0.00011
bdensity	0.000135	0.0290	0.0395	0.0100	-0.00009
brate	0.000082	-0.00660	0.0100	0.0179	2.776E-7
vv	0.000011	-0.00011	-0.00009	2.776E-7	0.000565

Monte Carlo Standard Errors			
Parameter	**MCSE**	**Standard Deviation**	**MCSE/SD**
b0	0.000601	0.0503	0.0120
btemp	0.00386	0.1912	0.0202
bdensity	0.00399	0.1987	0.0201
brate	0.00200	0.1337	0.0150
vv	0.000295	0.0238	0.0124

Posterior Autocorrelations				
Parameter	**Lag 1**	**Lag 5**	**Lag 10**	**Lag 50**
b0	-0.0034	-0.0111	0.0121	0.0155
btemp	0.4415	0.0261	0.0150	-0.0235
bdensity	0.4541	0.0388	0.0001	-0.0233
brate	0.1763	0.0217	0.0012	-0.0139
vv	0.0379	-0.0124	-0.0021	0.0059

Geweke Diagnostics		
Parameter	**z**	**Pr > |z|**
b0	-0.2045	0.8379
btemp	0.0266	0.9788
bdensity	-0.2384	0.8116
brate	-0.1566	0.8755
vv	0.2050	0.8375

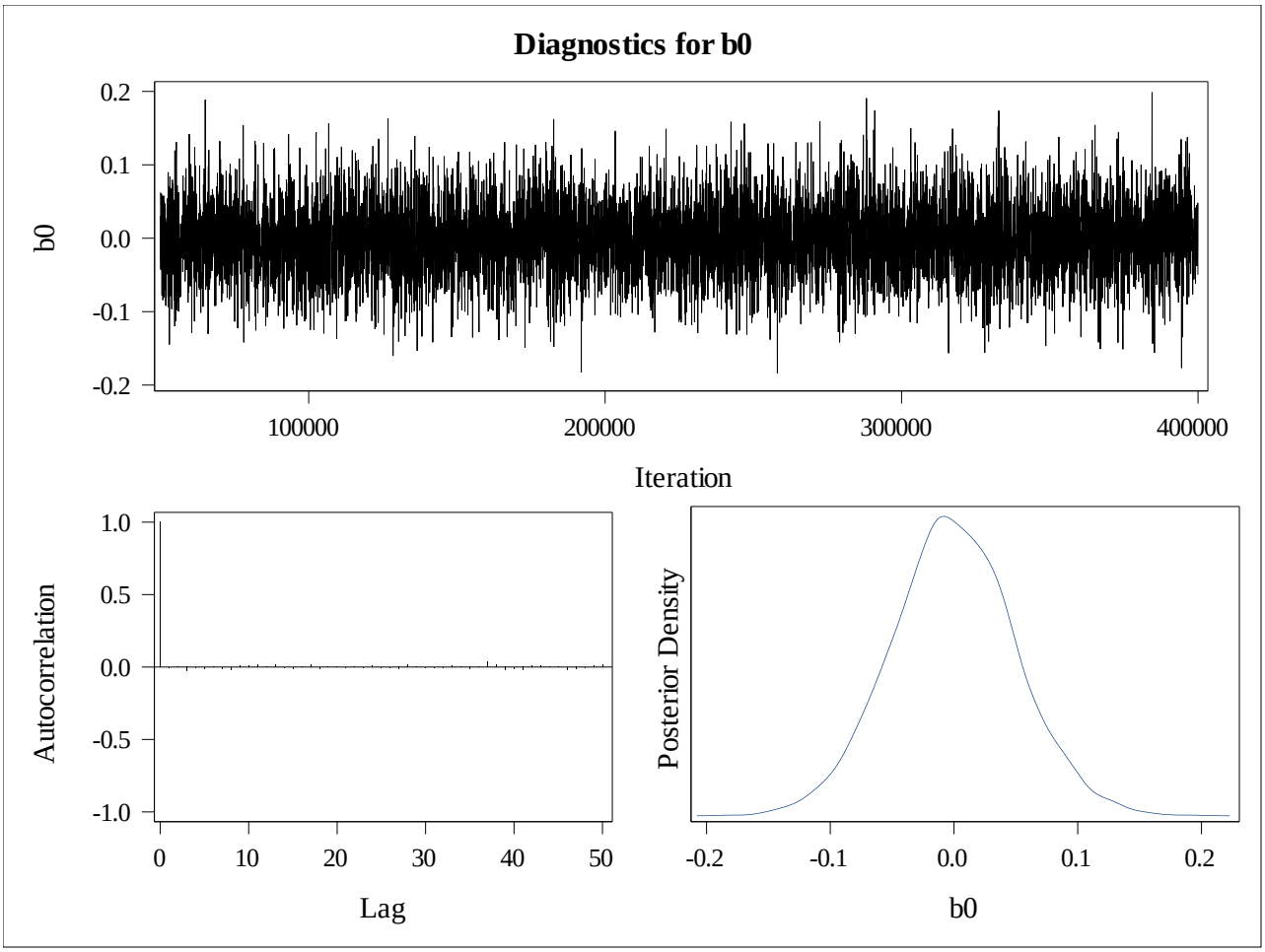
Raftery-Lewis Diagnostics				
Quantile=0.025 Accuracy=+/-0.005 Probability=0.95 Epsilon=0.001				
Parameter	Number of Samples			Dependence Factor
	Burn-In	Total	Minimum	
b0	2	3732	3746	0.9963
btemp	4	4781	3746	1.2763
bdensity	4	4781	3746	1.2763
brate	2	3912	3746	1.0443
vv	2	3689	3746	0.9848

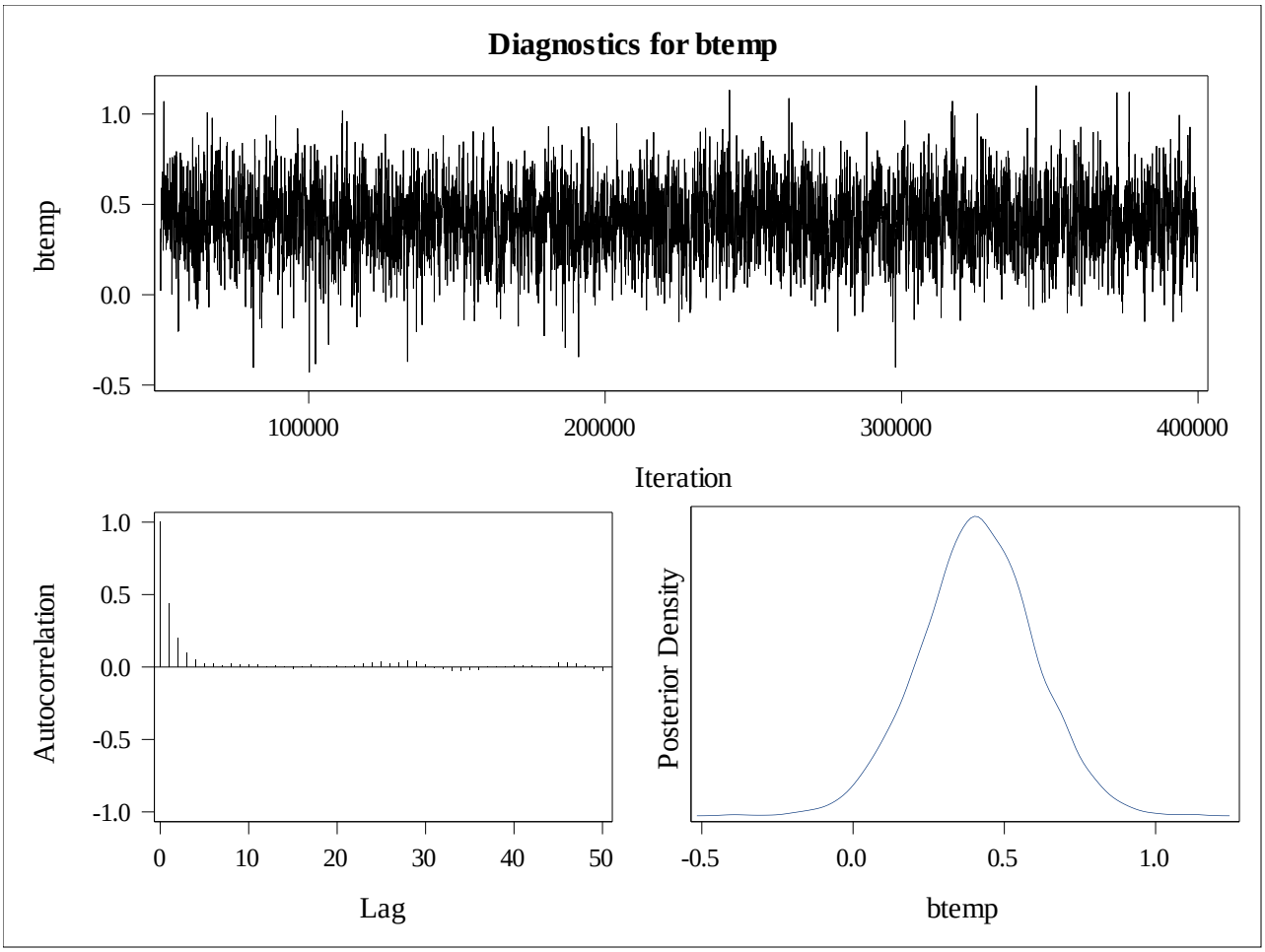
Heidelberger-Welch Diagnostics								
Parameter	Stationarity Test				Half-Width Test			
	Cramer-von Mises Stat	p-Value	Test Outcome	Iterations Discarded	Half-Width	Mean	Relative Half-Width	Test Outcome
b0	0.2536	0.1838	Passed	0	0.00123	-0.00043	-2.8317	Failed
btemp	0.0859	0.6582	Passed	0	0.00889	0.4105	0.0217	Passed
bdensity	0.0672	0.7688	Passed	0	0.00828	-0.4311	-0.0192	Passed
brate	0.0623	0.7998	Passed	0	0.00320	0.1473	0.0217	Passed
vv	0.0941	0.6146	Passed	0	0.000538	0.0757	0.00710	Passed

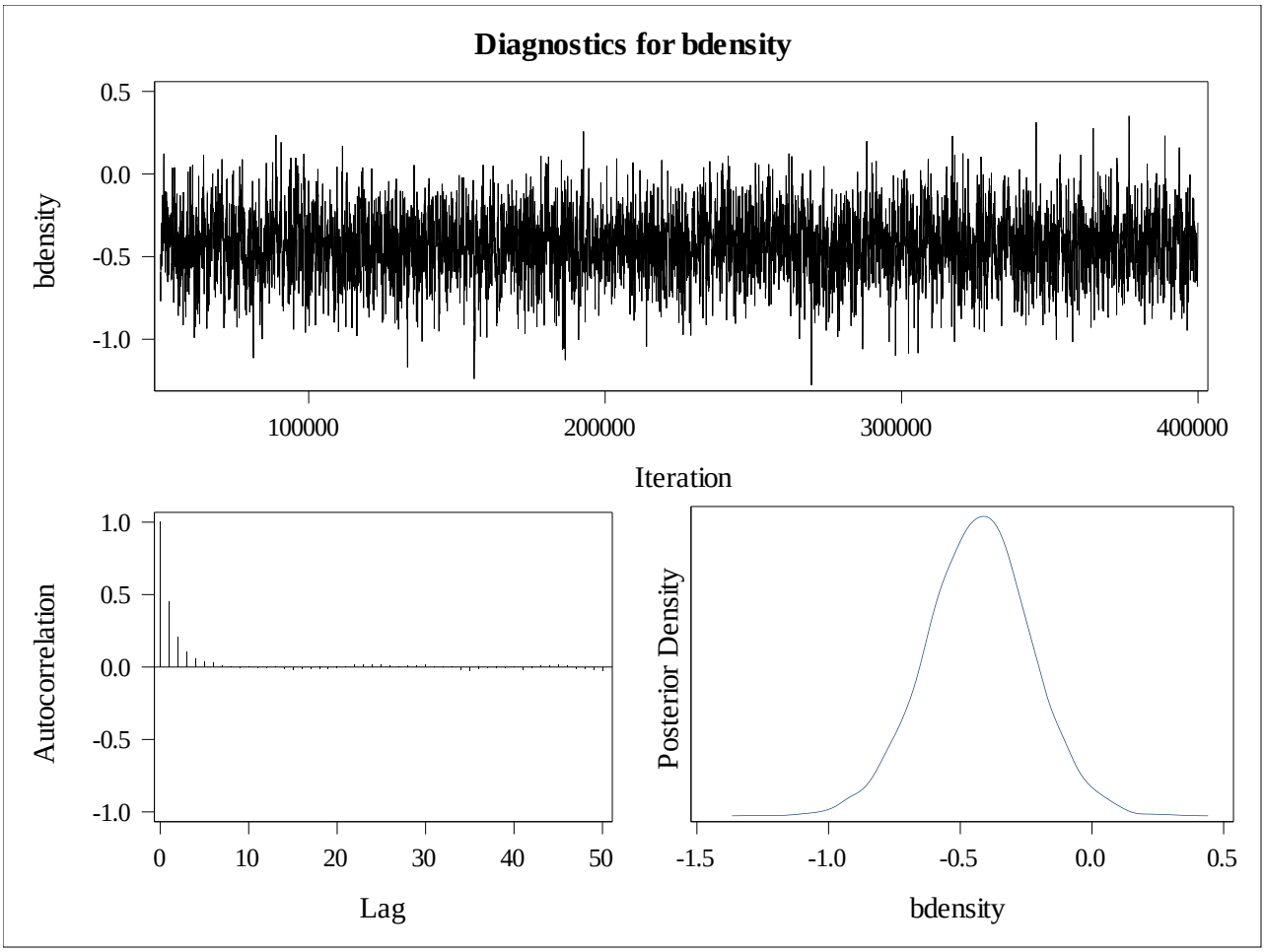
Effective Sample Sizes			
Parameter	ESS	Autocorrelation Time	Efficiency
b0	7000.0	1.0000	1.0000
btemp	2458.4	2.8474	0.3512
bdensity	2483.8	2.8183	0.3548

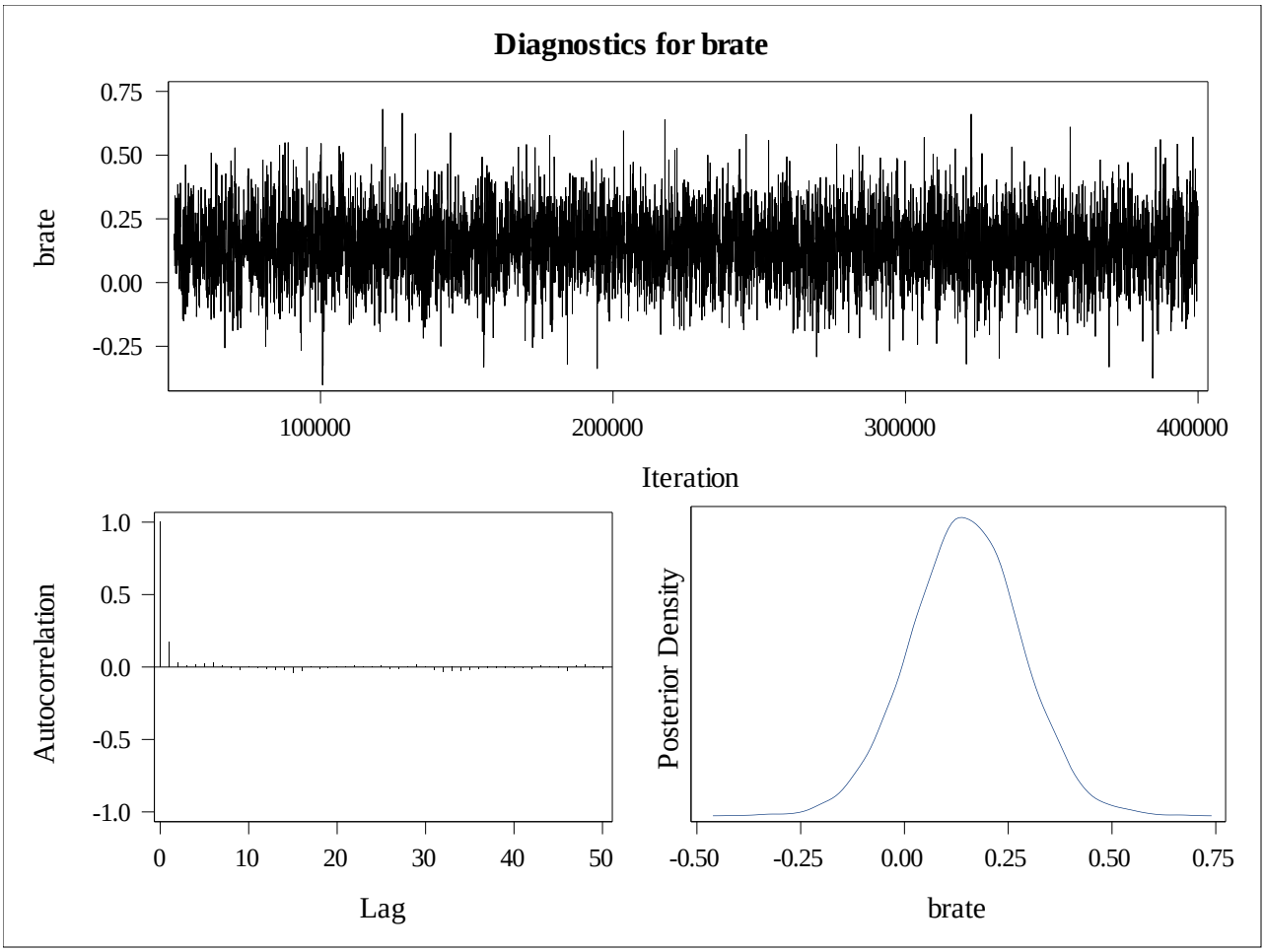
Effective Sample Sizes			
Parameter	**ESS**	**Autocorrelation Time**	**Efficiency**
brate	4470.9	1.5657	0.6387
vv	6506.4	1.0759	0.9295

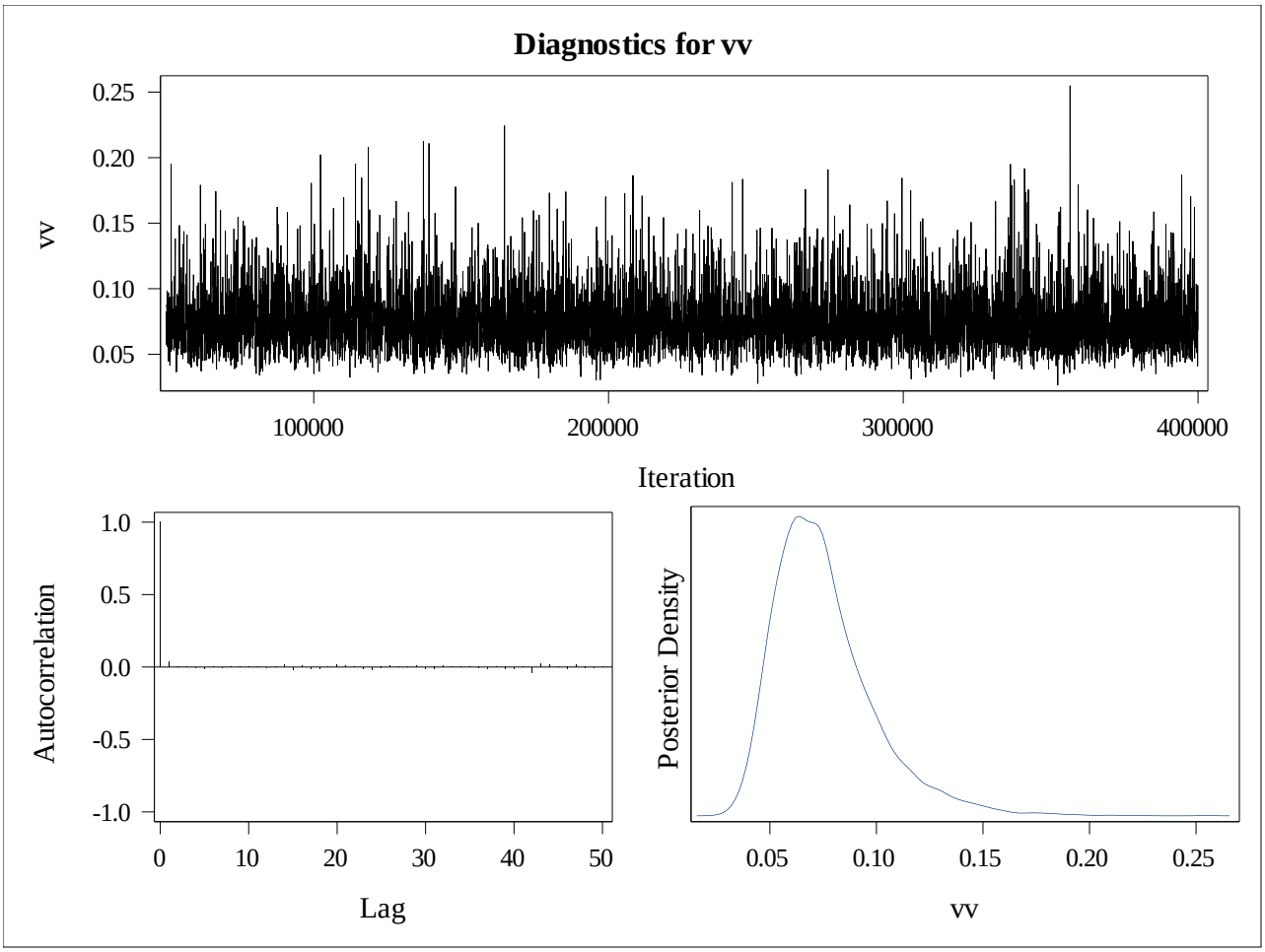
Deviance Information Criterion	
Dbar (posterior mean of deviance)	4.233
Dmean (deviance evaluated at posterior mean)	-0.464
pD (effective number of parameters)	4.697
DIC (smaller is better)	8.931


Diagnostics for b0
b0
0.2
0.1
0.0
-0.1
-0.2
100000
200000
300000
400000
Iteration
Autocorrelation
1.0
0.5
0.0
-0.5
-1.0
0
10
20
30
40
50
Lag
Posterior Density
-0.2
-0.1
0.0
0.1
0.2
b0


Diagnostics for btemp
btemp
1.0
0.5
0.0
-0.5
100000
200000
300000
400000
Iteration
Autocorrelation
1.0
0.5
0.0
-0.5
-1.0
0
10
20
30
40
50
Lag
Posterior Density
-0.5
0.0
0.5
1.0
btemp


Diagnostics for bdensity
bdensity
0.5
0.0
-0.5
-1.0
100000
200000
300000
400000
Iteration
Autocorrelation
1.0
0.5
0.0
-0.5
-1.0
0
10
20
30
40
50
Lag
Posterior Density
-1.5
-1.0
-0.5
0.0
0.5
bdensity


Diagnostics for brate
brate
0.75
0.50
0.25
0.00
-0.25
100000
200000
300000
400000
Iteration
Autocorrelation
1.0
0.5
0.0
-0.5
-1.0
0
10
20
30
40
50
Lag
Posterior Density
-0.50
-0.25
0.00
0.25
0.50
0.75
brate



15. Verify that the chains you have produced have converged appropriately by examining (and reproducing) the effective sample size, and the Raftery-Lewis diagnostics. Produce these diagnostics in the SAS output file. Since SAS produces output that is 132 characters in width by default, you may need to change the output width to fit on the pages you will be handing in.

Thinning by 50 in SAS got rid of a lot. I assume you didn't want us to thin that much. But I ran it anyways with thin = 50 to not lose points. Because of that thin, the sample sizes are a little bit small. But the raftery-ewis diagnostics are good. None of the trace plots show trends or patterns so I conclude that they converge.

I'm not sure what you mean by reproducing. Here I read in my saved output and used the effectiveSize function.

Raftery-Lewis Diagnostics

b0 0.9963 btemp 1.2763 bdensity 1.2763 brate 1.0443 vv 0.9848

```
library(haven)
sasoutput <- read_sas('exam1_q2.sas7bdat')
```

```
sims2 <- as.mcmc(sasoutput)
effectiveSize(sasoutput)

 Iteration        b0     btemp  bdensity     brate        vv  LogPrior
LogLike
     0.000  7000.000  2712.024  2627.253  4901.489  6487.453  7000.000
5745.136
   LogPost
  5745.102

dim(sasoutput)

 [1] 7000    9
```

16. In the SAS output file find 95% highest posterior density intervals for the parameters you have estimated.

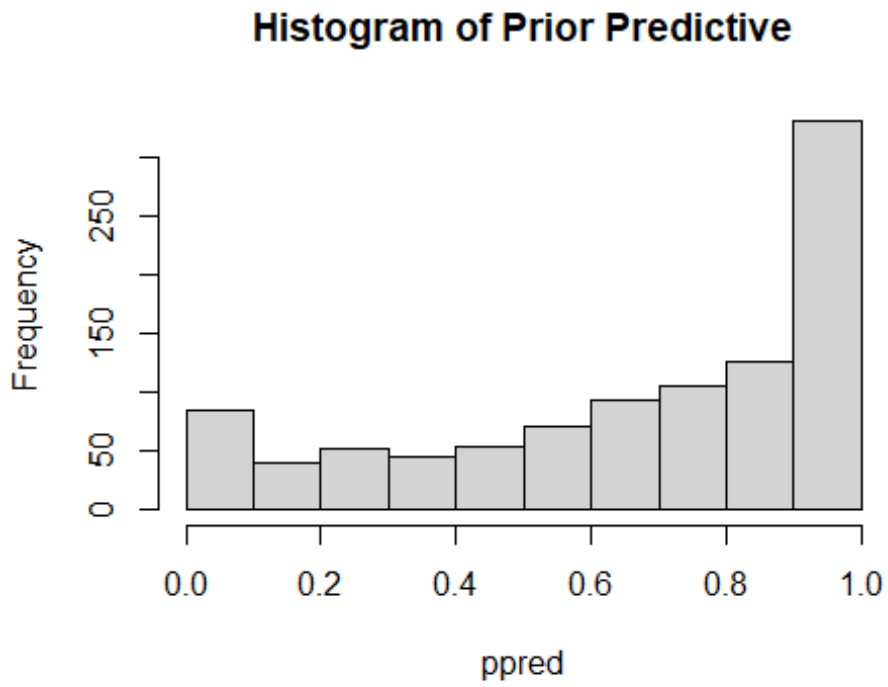
Posterior Intervals					
Parameter	**Alpha**	**Equal-Tail Interval**		**HPD Interval**	
b0	0.050	-0.0997	0.0999	-0.0982	0.1006
btemp	0.050	0.0316	0.7859	0.0197	0.7711
bdensity	0.050	-0.8295	-0.0401	-0.8350	-0.0494
brate	0.050	-0.1157	0.4090	-0.1226	0.3968
vv	0.050	0.0427	0.1359	0.0382	0.1251

For the last set of problems, you have been brought data by an anthropologist. She has found five adult skeletons of ancient humanoids. She has been studying the ratio of the length of humerus (upper arm bone) to the length of the femur (thigh bone) in primates. For monkeys she knows that ratio is about 0.95. For modern man, the ratio is about 0.72. The ratios for the five adult skeletons she has found are as follows: 0.857, 0.824, 0.820, 0.875, 0.844. She is interested in the probability that the population mean ratio for the group of people whose skeletons she has found is !between 0.80 and 0.90.

For your likelihood, you should be aware that the appropriate support for these data is between 0 and 1. So we will use a beta likelihood. You may remember the beta distribution from 251 when you worked on binomial (bernoulli) data. You should also know that these ratios range from about 0.70 to 0.98. The parameters of a beta must be positive, so we will use gamma priors for the parameters. If the beta is parameterized with (a,b), then use a gamma(shape=1,rate=.2) for a, and a gamma(shape=1.5,rate=1) for b.

17. Since you know the likelihood and the prior distributions for the parameters, you can draw values from the prior predictive. Draw 1000 values from the prior predictive and plot a histogram of the values drawn from the prior predictive. Remember that the values must be between 0 and 1.

```
ppred <- NULL
for(i in 1:1000){
  a <- rgamma(1, 1, 0.2)
  b <- rgamma(1, 1.5, 1)
  ppred[i] <- rbeta(1, a, b)
}

hist(ppred, main = "Histogram of Prior Predictive")
```



18. Write code in JAGS to address the problem. Use 20000 burnin and 100000 iterations, thin by 10 and produce 5 chains. This will give you 40000 MCMC draws of the posterior. Print the summary of the simulation.

```
set.seed(1234)
```

```
library(R2jags)

mdl4 <- "
  model{

  for(i in 1:5){
    y1[i] ~ dnorm(mu, 1/vv)
  }


  # Priors
  a ~ dgamma(1, 0.2)
  b ~ dgamma(1.5, 1)

  mu = a/(a+b)
  vv = (a*b)/((a+b)^2 * (a+b+1))

}"

writeLines(mdl4, 'mod4output')
data.jags <- list(
    y1 =c(0.857, 0.824, 0.820, 0.875, 0.844)
  )

parms.jags <- c("mu", "vv")


beta.sim <- jags(data= data.jags, parameters.to.save = parms.jags,
                 model.file = 'mod4output', inits = NULL,
                 n.iter = 100000, n.thin = 10, n.chains = 5,
                 n.burnin = 20000)

 Compiling model graph
    Resolving undeclared variables
    Allocating nodes
 Graph information:
    Observed stochastic nodes: 5
    Unobserved stochastic nodes: 2
    Total graph size: 19

 Initializing model

beta.sim

 Inference for Bugs model at "mod4output", fit using jags,
  5 chains, each with 1e+05 iterations (first 20000 discarded), n.thin = 10
  n.sims = 40000 iterations saved
          mu.vect sd.vect    2.5%     25%     50%     75%  97.5%  Rhat n.eff
 mu         0.863   0.041   0.769   0.840   0.868   0.890   0.930 1.001 35000
```

```
 vv         0.010   0.007   0.004   0.006   0.009   0.012  0.027 1.001 21000
 deviance -12.931   2.446 -17.044 -14.634 -13.160 -11.493 -7.529 1.001 18000

 For each parameter, n.eff is a crude measure of effective sample size,
 and Rhat is the potential scale reduction factor (at convergence, Rhat=1).

 DIC info (using the rule, pD = var(deviance)/2)
 pD = 3.0 and DIC = -9.9
 DIC is an estimate of expected predictive error (lower deviance is better).
```

```
sims3 <- as.mcmc(beta.sim)
chains3 <- as.matrix(sims3)
sims3 <- as.mcmc(chains3)


colnames(chains3)
```

```
 [1] "deviance" "mu"       "vv"
```

```
post.draws <- chains3[,2]
post.vv <- chains3[,3]
```

19. Verify that the chains you have produced have converged appropriately and have enough information to use to make inference by examining (and reporting) the effective sample size. Effective sample sizes should exceed 5000.

Effective sample size for deviance is 32,018 Effective Sample size for mu is 40000 Effective sample size for vv is 36,194.9

```
effectiveSize(sims3)
```

```
 deviance       mu       vv
 32018.41 40000.00 36194.92
```

20. Verify that the chains you have produced have converged appropriately and have enough information to use to make inference by examining (and reporting) the Raftery-Louis diagnostic. Raftery-Louis diagnostics should be less than 3.

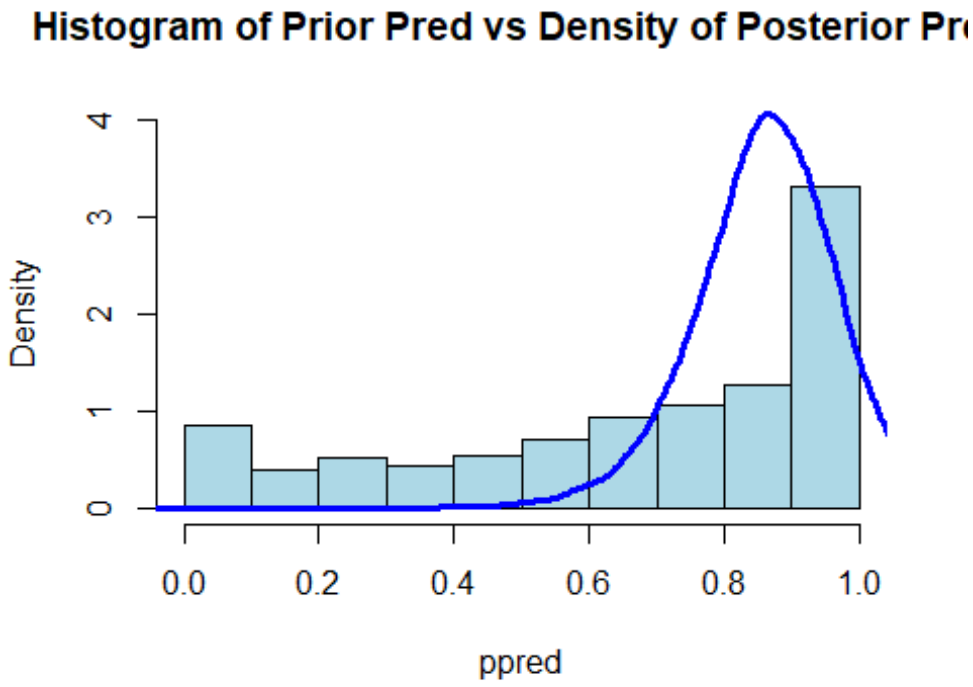
RLD for deviance is 1.13 RLD for mu is 0.998 RLD for vv is 1.070

```
raftery.diag(sims3)


 Quantile (q) = 0.025
 Accuracy (r) = +/- 0.005
 Probability (s) = 0.95

           Burn-in  Total Lower bound  Dependence
           (M)      (N)   (Nmin)       factor (I)
  deviance 3        4249  3746         1.130
```

```
  mu       2        3740  3746          0.998
  vv       3        4020  3746          1.070
```

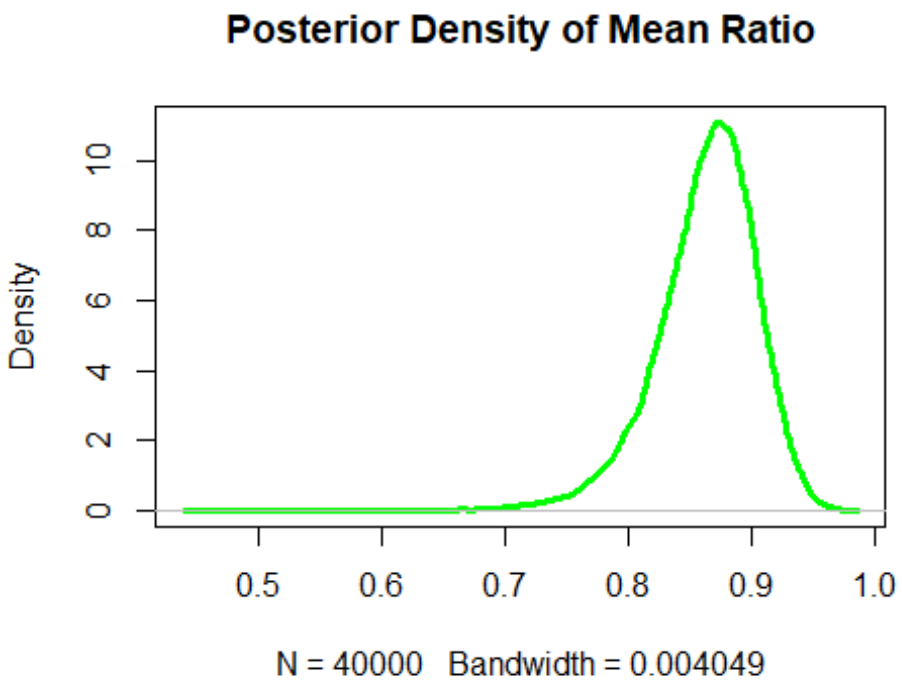
21. Using your output chains, draw 40000 values from the posterior predictive. Plot the posterior predictive density and the histogram of the prior predictive density on the same set of axes.

```
popdat1 <- NULL
for(i in 1:40000){
            #Noise in mean, #Noise in Error
  popdat1[i] <- post.draws[i] + rnorm(1,0,sqrt(post.vv[i]))
}

hist(ppred, freq = FALSE, ylim = c(0,4), xlim = c(0,1), col = "lightblue"
,main = "Histogram of Prior Pred vs Density of Posterior Pred")
lines(density(popdat1), col = "blue", lwd = 3)
```



22. Plot the posterior density of the mean ratio which would be computed as a/(a+b).

```
plot(density(post.draws), col = "green", lwd = 3, main = "Posterior Density
of Mean Ratio")
```



23. What is the probability the ratio for the population mean given the data from these five skeletons is between 0.80 and 0.90?

There is 86% posterior probability that the ratio for the population mean is between 0.8 and 0.9.

```
mean(post.draws[post.draws >= .8 & post.draws <= .9])
```

```
 [1] 0.8599267
```