Bayesian Generalized Linear Models in **R**

As published in Benchmarks RSS Matters, January 2011

http://web3.unt.edu/benchmarks/issues/2011/1/rss-matters

Jon Starkweather, PhD

Jon Starkweather, PhD
jonathan.starkweather@unt.edu
Consultant
Research and **S**tatistical **S**upport

http://www.unt.edu

http://www.unt.edu/rss

RSS hosts a number of "Short Courses".
A list of them is available at:
http://www.unt.edu/rss/Instructional.htm

The programming scripts contained in this article can also be found at:
http://www.unt.edu/rss/class/Jon/R_SC

Bayesian Generalized Linear Models in **R**

Bayesian statistical analysis has benefited from the explosion of cheap and powerful desktop computing over the last two decades or so. Bayesian techniques can now be applied to complex modeling problems where they could not have been applied previously. It seems likely that the Bayesian perspective will continue to challenge, and perhaps sub-plant, traditional frequentist statistical methods which have dominated many disciplines of science for so long. The current article will review one function which allows the user to conduct linear regression, general linear modeling, and generalized linear modeling (i.e. non-Gaussian; e.g., Poisson, binomial, etc.). A fairly simple model is specified, then modeled using traditional techniques, and then modeled with a Bayesian approach. Do not implement these methods unless you understand the core principles of the Bayesian perspective (i.e. priors, likelihoods, posteriors, etc., and all they entail).

A complete treatment of the Bayesian perspective is beyond the scope of this article and could fill several books; and has. Interested readers can consult a number of introductory texts focusing on the Bayesian perspective (e.g., Berry, 1996; Bolstad, 2004; Gelman, Carlin, Stern, & Rubin, 2004; Hoff, 2009). Very generally speaking, the Bayesian approach to statistical inference differs from traditional frequentist inference by assuming that the data are fixed and model parameters are random, which sets up problems in the form of; what is the probability of a hypothesis (or parameter), given the data at hand? These types of problems can be stated with symbols as: $p\left(H|D\right)$. Traditional frequentist inference assumes that the model parameters are fixed (though unknown) and the data are essentially random; for instance, if the null hypothesis is true, what is the probability of this data? These types of problems can be stated in the general form; what is the probability of the data given a hypothesis? In symbols, this translates to: $p\left(D|H\right)$.

Bayesian methods focus on five essential elements. First, the incorporation of *prior* information (e.g., expert opinion, a thorough literature review of the same or similar variables, and/or prior data). Prior information is generally specified quantitatively in the form of a distribution (e.g., normal/Gaussian, Poisson, binomial, etc.) and represents a probability distribution for a coefficient; meaning, the distribution of probable values for a coefficient we are attempting to model (e.g., a β weight). It may help to think of the prior as an educated *best guess*. Second, the prior is combined with a *likelihood function*. The likelihood function represents the data (i.e. what is the distribution of the estimate produced by the data). Third, the combination of the prior with the likelihood function results in the creation of a *posterior* distribution of coefficient values. Fourth, *simulates* are drawn from the posterior distribution to create an empirical distribution of likely values for the population parameter. Fifth, basic statistics are used to summarize the empirical distribution of simulates from the posterior. The mode (or median or mean) of this empirical distribution represents the maximum likelihood estimate of the true coefficient's population value (i.e. population parameter) and credible intervals can capture the true population value with probability attached.

Keep in mind, priors should be rationally and honestly derived. They can be *weak* or *strong*. These terms refer to the strength of belief we have in the prior(s). Weak priors result when we do not have a great deal of evidence or prior information on which to base the prior(s). When the prior is weak, the prior distribution will be wide, reflecting a great many possible values and the likelihood will be more influential in creating the posterior distribution. Strong priors, conversely, result when we have a great deal of evidence on which to base the prior(s). When the prior is strong, the prior distribution will be narrow, reflecting a smaller range of possible values and the likelihood will be less influential in creating the posterior (strong priors will influence the posterior more than the likelihood). It should be clear the one key feature of the prior is the ability to quantify our uncertainty. The posterior can be thought of as a compromise between the prior and the likelihood. If the prior is weak, then it will be less influential in

creating the posterior; if the prior is strong, then it will be more influential in creating the posterior.

The example used here is a simple linear regression model with one interval/ratio outcome (extro) and three interval/ratio predictors (open, agree, social). The *simulated* data set contains 500 cases, each with complete data (i.e. no missing values).

Import the data from the web, get a summary of the data, and take a look at the correlations. We see very little multicollinearity here.

```
R Console
File  Edit  Misc  Packages  Windows  Help

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> data1 <- read.table("http://www.unt.edu/rss/class/Jon/R_SC/Module10/BayesGLM_example.txt",
+    header=TRUE, sep=",", na.strings="NA", dec=".", strip.white=TRUE)
> summary(data1)
     extro            open           agree            social
 Min.   :108.1   Min.   :26.91   Min.   : 47.81   Min.   :13.01
 1st Qu.:144.7   1st Qu.:45.77   1st Qu.: 90.70   1st Qu.:22.58
 Median :155.2   Median :50.32   Median : 99.20   Median :24.93
 Mean   :155.7   Mean   :50.41   Mean   :100.13   Mean   :25.08
 3rd Qu.:167.2   3rd Qu.:55.10   3rd Qu.:110.01   3rd Qu.:27.45
 Max.   :205.3   Max.   :69.44   Max.   :149.29   Max.   :38.84
> cor(data1)
           extro        open       agree      social
extro  1.0000000  0.53601980  0.84510471  0.08392999
open   0.5360198  1.00000000  0.01570736 -0.01462117
agree  0.8451047  0.01570736  1.00000000 -0.00175529
social 0.0839300 -0.01462117 -0.00175529  1.00000000
> |
```

Confirm / take a look at the core Linear Model (`lm`) – traditional Ordinary Least Squares (OLS) regression.

```
> model.1 <- lm(extro ~ open + agree + social, data = data1)
> summary(model.1)

Call:
lm(formula = extro ~ open + agree + social, data = data1)

Residuals:
     Min       1Q   Median       3Q      Max
-2.30609 -0.68503 -0.01365  0.73440  3.04784

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -4.944214   0.530766  -9.315   <2e-16 ***
open         1.199690   0.006343 189.136   <2e-16 ***
agree        0.900636   0.002982 302.022   <2e-16 ***
social       0.397546   0.011839  33.580   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.016 on 496 degrees of freedom
Multiple R-squared: 0.9962,     Adjusted R-squared: 0.9962
F-statistic: 4.324e+04 on 3 and 496 DF,  p-value: < 2.2e-16

> |
```

Notice in the output, the intercept is approximately -5.0. The unstandardized coefficients for each predic-

tor are listed. The open coefficient is approximately 1.2, the agree coefficient is .90, the social coefficient is approximately 0.40. We could calculate a 95% confidence interval (CI_{95}) for each predictor's coefficient; for instance, the CI_{95} for open is 1.187 to 1.212. But what does this *really* tell us? Well, it is interpreted as: if an infinite number of samples were taken from this population, 95% of the open coefficient values would be between 1.187 and 1.212. But it does not tell us the range which contains the true population value.

The same results are below; but, the results below were generated with the Generalized Linear Model (`glm`) function, specifying the default Gaussian (normal) family distribution. The primary benefit of the `glm` function is the ability to specify error distributions other than normal.

```
> model.2 <- glm(extro ~ open + agree + social, data = data1, family = gaussian)
> summary(model.2)

Call:
glm(formula = extro ~ open + agree + social, family = gaussian,
    data = data1)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.30609  -0.68503  -0.01365   0.73440   3.04784

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -4.944214   0.530766  -9.315   <2e-16 ***
open         1.199690   0.006343 189.136   <2e-16 ***
agree        0.900636   0.002982 302.022   <2e-16 ***
social       0.397546   0.011839  33.580   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for gaussian family taken to be 1.032558)

    Null deviance: 134466.65  on 499  degrees of freedom
Residual deviance:    512.15  on 496  degrees of freedom
AIC: 1440.9

Number of Fisher Scoring iterations: 2

>
```

To conduct the Bayesian GLM, load the package 'arm' which contains the `bayesglm` function (Gelman, et al., 2010). You will notice there are several dependencies.

```
> library(arm)
Loading required package: MASS
Loading required package: Matrix
Loading required package: lattice

Attaching package: 'Matrix'

The following object(s) are masked from 'package:base':

    det

Loading required package: lme4

Attaching package: 'lme4'

The following object(s) are masked from 'package:stats':

    AIC

Loading required package: R2WinBUGS
Loading required package: coda

Attaching package: 'coda'

The following object(s) are masked from 'package:lme4':

    HPDinterval

Loading required package: abind
Loading required package: car
Loading required package: nnet
Loading required package: survival
Loading required package: splines

arm (Version 1.3-08, built: 2010-11-20)
Working directory is c:/Documents and Settings/jds0282/Desktop/Work_Stuff/Jon_R
Loading required package: foreign

Attaching package: 'arm'

The following object(s) are masked from 'package:coda':

    traceplot

> |
```

Conduct the Bayesian Generalized linear model (here family = Gaussian) and get the summary of the output. Notice the specification of the prior mean, scale, and degrees of freedom. Each 'family' of distributions requires specific prior specifications (e.g. a binomial distribution would have slightly different prior specification; see the package documentation for details[1] or simply type `help(bayesglm)` in the **R** console).

[1] `http://cran.r-project.org/web/packages/arm/arm.pdf`

```
> model.3 <- bayesglm (extro ~ open + agree + social, family = gaussian, data = data1,
+                      prior.mean=0, prior.scale=Inf, prior.df=Inf)
> summary(model.3)

Call:
bayesglm(formula = extro ~ open + agree + social, family = gaussian,
    data = data1, prior.mean = 0, prior.scale = Inf, prior.df = Inf)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.30609  -0.68503  -0.01365   0.73440   3.04784

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -4.944214   0.528638  -9.353   <2e-16 ***
open         1.199690   0.006318 189.897   <2e-16 ***
agree        0.900636   0.002970 303.238   <2e-16 ***
social       0.397546   0.011791  33.715   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for gaussian family taken to be 1.024298)

    Null deviance: 134466.65  on 499  degrees of freedom
Residual deviance:    512.15  on 500  degrees of freedom
AIC: 1440.9

Number of Fisher Scoring iterations: 5

> |
```

We can see the output matches up with the traditional linear model (OLS regression) as well as the traditional GLM. As sample sizes increase the results should converge to the same values.

One of the benefits of the Bayesian perspective (for any analysis) is that it allows us to make *credible interval* statements. Credible intervals are similar to confidence intervals, but in the Bayesian framework, the interval REALLY IS believed to contain the true population parameter. For instance: a 95% credible interval for a parameter being estimated is interpreted as; there is a 95% probability that the actual parameter is included in that interval. This is because the interval is based on information from the posterior distribution; of for instance, one of the predictor's coefficient posterior distribution (e.g. the open variable's coefficient posterior distribution).

The `bayesglm` function represents a kind of short cut of the Bayesian approach to inference. Typically, the posterior is not used directly for making inferences. Instead, an empirical distribution is constructed based on draws from the posterior and that empirical distribution is what informs the inference(s). Here, we are using the `bayesglm` as a proxy for doing the added empirical distribution. With the `bayesglm` we get a distribution of 'simulates' which are used in place of an actual empirical distribution (which will be covered further below).

Retrieve the posterior distributions of the coefficients for the intercept and all three predictors. The `head` function simply lists the first 10 rows of the object on which it is run (the default `head` is the first 6).

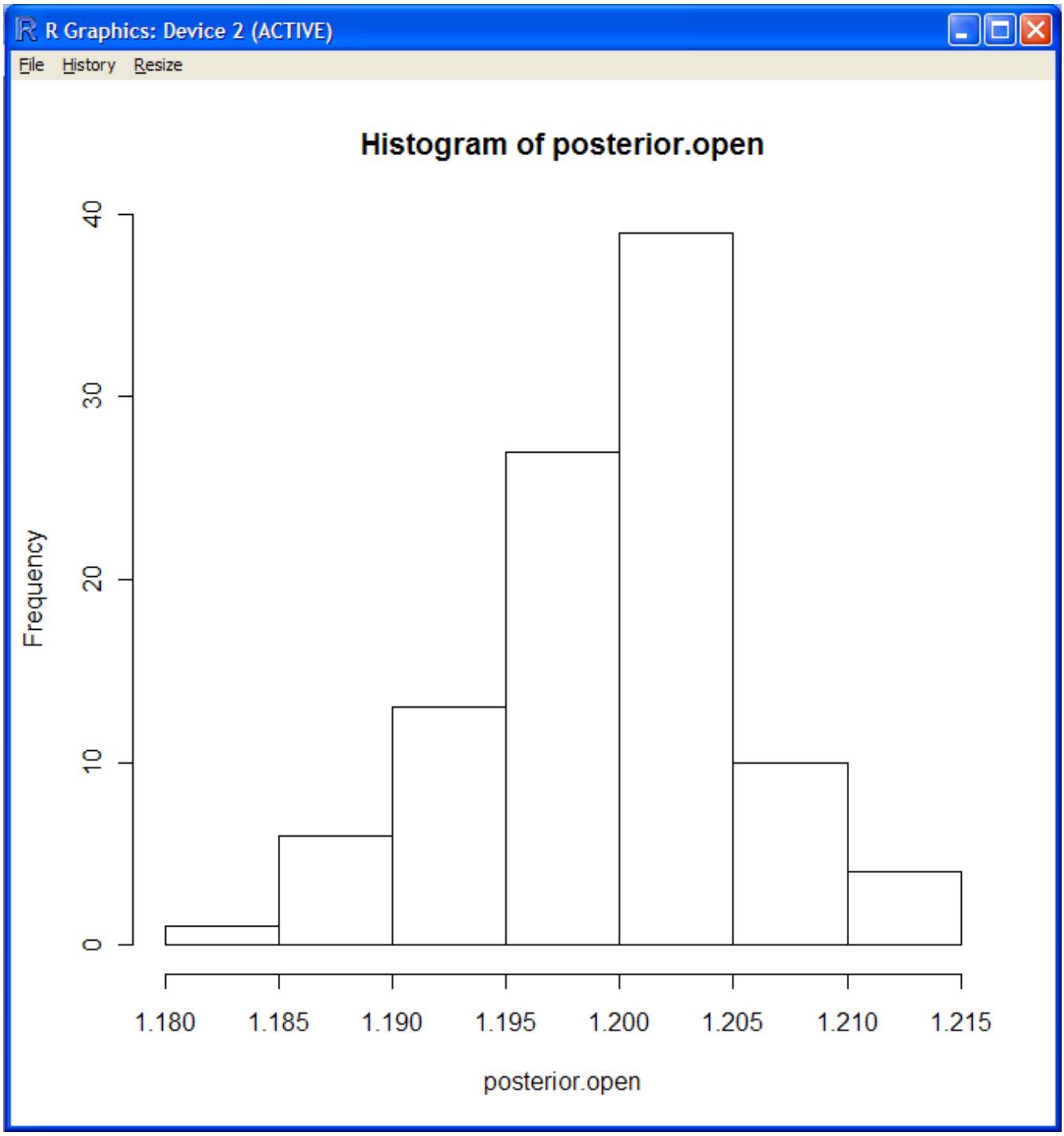
```
> simulates <- coef(sim(model.3))
> head(simulates, 10)
      (Intercept)     open     agree    social
 [1,]   -5.256479 1.204338 0.8973143 0.4136634
 [2,]   -5.188813 1.198698 0.9020349 0.4057331
 [3,]   -4.609328 1.198168 0.8988569 0.3961592
 [4,]   -4.279686 1.183936 0.9005968 0.4027266
 [5,]   -5.439462 1.203092 0.8992952 0.4155758
 [6,]   -4.962830 1.202151 0.8987105 0.4024284
 [7,]   -5.190100 1.202345 0.9019534 0.3953540
 [8,]   -4.397217 1.191770 0.9018699 0.3882703
 [9,]   -4.432026 1.189979 0.8994867 0.4012326
[10,]   -4.213517 1.198061 0.8965457 0.3869824
> |
```

Extract just the posterior distribution of the 'open' variable's coefficient. Again, the `head` function simply lists the first 10 items of the object.

```
> posterior.open <- simulates[,2]
> head(posterior.open, 10)
 [1] 1.204338 1.198698 1.198168 1.183936 1.203092 1.202151 1.202345 1.191770
 [9] 1.189979 1.198061
> |
```

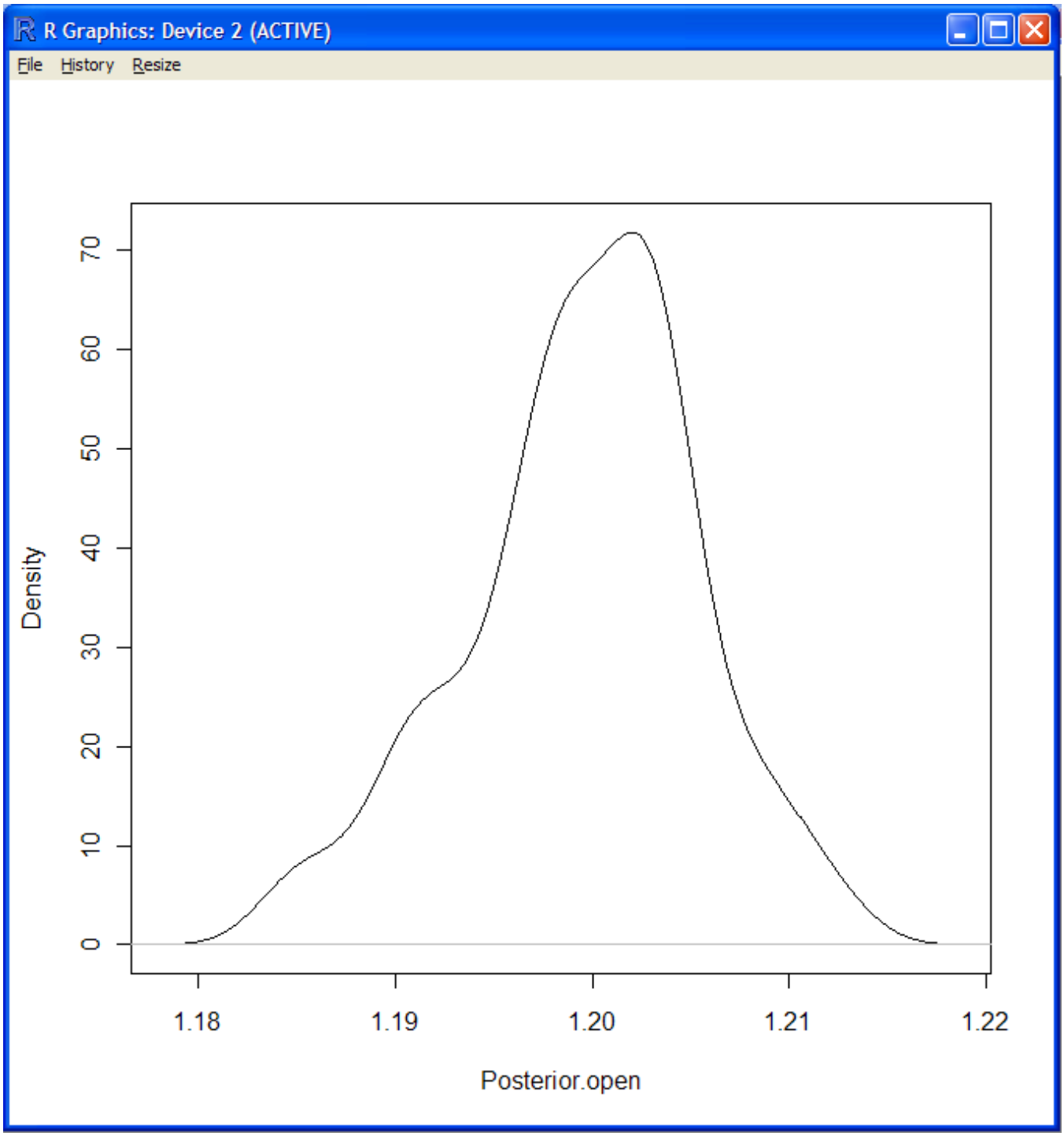
Take a look at the posterior distribution of the open variable's coefficient (normally a histogram would not be used, it is used here simply as a graphical reference).

```
> hist(posterior.open)
> |
```



The 'density' plot is normally used to display a posterior. The density plot can be thought of as a highly modified histogram. Imagine a histogram with 100 bins (instead of the 7 as displayed in the histogram above), then imagine plotting a line from the x-axis through each bin's midpoint at the top of each bin, and then back down to the x-axis; the result would be the density plot.

```
> plot(density(posterior.open), main = "", xlab = "Posterior.open", ylab = "Density")
>
```



Now we can retrieve the 95% credible interval for the open variable's coefficient.

```
> quantile(posterior.open, c(.025, .975))
    2.5%    97.5%
1.186170 1.210246
>
```

Recall, this credible interval is interpreted as; there is a 95% probability that the true population value of the open coefficient is between 1.186 and 2.210. Keep in mind, these numbers will fluctuate slightly based on the iterative nature of the function.

To make truly Bayesian inferences about our coefficients, we need to do the extra step of creating the em-

pirical distribution(s) mentioned above. Going further entails actually creating an empirical distribution based on iterative draws from the posterior. The `MCMCregress` function in the package 'MCMCpack' (Martin, Quinn, & Park, 2010)[2] provides us with the Markov Chain Monte Carlo simulation method of creating the empirical distribution; which itself allows us to then compute the descriptive statistics used for inference. Meaning, the mode, median, or mean of the empirical MCMC simulates' distribution is the 'maximum likelihood' estimate (i.e. top of a density function) of the population parameter. The `MCMCregress` function also gives us the credible interval which includes the actual population parameter value.

First, load the 'MCMCpack' library.

```
> library(MCMCpack)
##
## Markov Chain Monte Carlo Package (MCMCpack)
## Copyright (C) 2003-2010 Andrew D. Martin, Kevin M. Quinn, and Jong Hee Park
##
## Support provided by the U.S. National Science Foundation
## (Grants SES-0350646 and SES-0350613)
##
>
```

Next, apply the `MCMCregress` function. Notice, the model formula is the same, but here we have some new options. The 'burnin' argument is used because MCMC iterates are sensitive to their initial start values, so the first few (i.e. 3000) iterations are discarded. The 'mcmc' simply issues how many (post-burnin) iterations will be used to build the empirical distribution. The 'thin' defaults to 1 and represents a control on convergence, such that once approximate convergence has been reached it can be beneficial to keep only a few simulates and discard the rest to conserve computer resources (Gelman, Carlin, Stern, & Rubin, 2004). The verbose option (by default is off) simply does or does not print the iteration history as the function runs. The seed argument simply allows the user to set the random number generator seed. The 'beta.start' argument allows the user to set a start value for the beta vector.

```
> model.4 <- MCMCregress(extro ~ open + agree + social, data = data1, burnin = 3000, mcmc = 10000,
+                        thin = 1, verbose = 0, seed = NA, beta.start = NA)
> summary(model.4)

Iterations = 3001:13000
Thinning interval = 1
Number of chains = 1
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) -4.9460 0.529359 5.294e-03      5.485e-03
open         1.1996 0.006430 6.430e-05      5.125e-05
agree        0.9007 0.002983 2.983e-05      2.870e-05
social       0.3976 0.011705 1.170e-04      1.188e-04
sigma2       1.0367 0.066217 6.622e-04      6.404e-04

2. Quantiles for each variable:

               2.5%     25%     50%     75%   97.5%
(Intercept) -5.9681 -5.3023 -4.9422 -4.5963 -3.8876
open         1.1869  1.1953  1.1997  1.2039  1.2120
agree        0.8947  0.8987  0.9007  0.9027  0.9064
social       0.3743  0.3896  0.3977  0.4054  0.4205
sigma2       0.9149  0.9910  1.0338  1.0795  1.1752

>
```

Notice in the summary, we get the coefficient estimates ("1. Empirical...") and credible intervals ("2. Quantiles..."). So, we can say there is a 95% probability that the true population value of the open coefficient is between 1.1869 and 2.2120.

[2]The package 'MCMCpack' contains functions for many other types of models and contains other ancillary functions for working with MCMC objects.

References & Resources

Albert, J. (2007). *Bayesian Computation with R.* New York: Springer Science+Business Media, LLC.

Berry, D. A. (1996). *Statistics: A Bayesian perspective.* Belmont, CA: Wadsworth Publishing Company.

Bolker, B. M. (2008). *Ecological Models and Data in R.* Princeton, NJ: Princeton University Press.

Bolstad, W. M. (2004). *Introduction to Bayesian statistics.* Hoboken, NJ: John Wiley & Sons, Inc.

Gelman, A., Carlin, J. B., Stern, H. S., & Rubin, D. B. (2004). *Bayesian Data Analysis* (2nd ed.). Boca Raton, FL: Chapman & Hall/CRC.

Gelman, A., Jakulin, A., Pittau, M. G., & Su, Y. (2009). A Weakly Informative Default Prior Distribution For Logistic And Other Regression Models. *The Annals of Applied Statistics, 2*(4), 1360-1383.

Gelman, A., Su, Y., Yajima, M., Hill, J., Pittau, M. G., Kerman, J., & Zheng, T. (2010). *Package 'arm'.* Available at: `http://cran.r-project.org/web/packages/arm`

Hoff, P. D. (2009). *A First Course in Bayesian Statistical Methods.* New York: Springer Science+Business Media, LLC.

Martin, A. D., Quinn, K. M., & Park, J. H. (2010). Package 'MCMCpack'. Available at: `http://cran.r-project.org/web/packages/MCMCpack/MCMCpack.pdf`

Until next time, "Stop; hey, what's that sound..."

This article was last updated on January 14, 2011.

This document was created using LaTeX