Explanation of Functions

Reza Niazi

11/26/2020

Introduction

This document explains how the functions in the package are used. The first function gives plots and statistics to determine whether a given linear model satisfies the regression assumptions. The second, performs a bootstrap in order to calculate regression coefficients and confidence intervals and compares this to the classical version. The third function performs regression using Bayesian methods. Along with diagnostic plots to make sure that the MCMC process converges, it also compares the Bayesian approach to the classical one. The fourth, and final, function really just calls a Shiny web app. Since the RMD cannot fully knit while it also opens the app, another RMD file called Driver_Shiny was made in order to run this on your RMD. You can also just go to the app.R file in the inst/shiny directory and run it there as well, or simply type shinyddt() in the commandline.

Let's begin by looking at the dataset that we will use to demonstrate the functions. This is the QUASAR.xls file. In alL the functions, we use the model that

$$E[RFEWIDTH] = \beta_0 + \beta_1 LINEFLUX + \beta_2 AB1450$$

```
library(readxl)
source("ModelCheck.r")
quasar = read_excel("QUASAR.XLS")
quasar
```

```
## # A tibble: 25 x 7
##    QUASAR REDSHIFT LINEFLUX LUMINOSITY AB1450 ABSMAG RFEWIDTH
##     <dbl>    <dbl>    <dbl>      <dbl>  <dbl>  <dbl>    <dbl>
##  1      1     2.81    -13.5       45.3   19.5  -26.3      117
##  2      2     3.07    -13.7       45.1   19.6  -26.3       82
##  3      3     3.45    -13.9       45.1   18.9  -27.2       33
##  4      4     3.19    -13.3       45.6   18.6  -27.4       92
##  5      5     3.07    -13.6       45.3   19.6  -26.3      114
##  6      6     4.15    -14.0       45.2   19.4  -27.0       50
##  7      7     3.26    -13.8       45.1   19.2  -26.8       43
##  8      8     2.81    -13.5       45.3   20.4  -25.4      259
##  9      9     3.83    -13.7       45.4   18.9  -27.3       58
## 10     10     3.32    -13.7       45.2   20    -26.0      126
## # ... with 15 more rows
```

Function 1 (ModelCheck)

This function produces residual vs fitted and QQ plots given a certain linear model (lm) object. This is done to make sure that the linear model

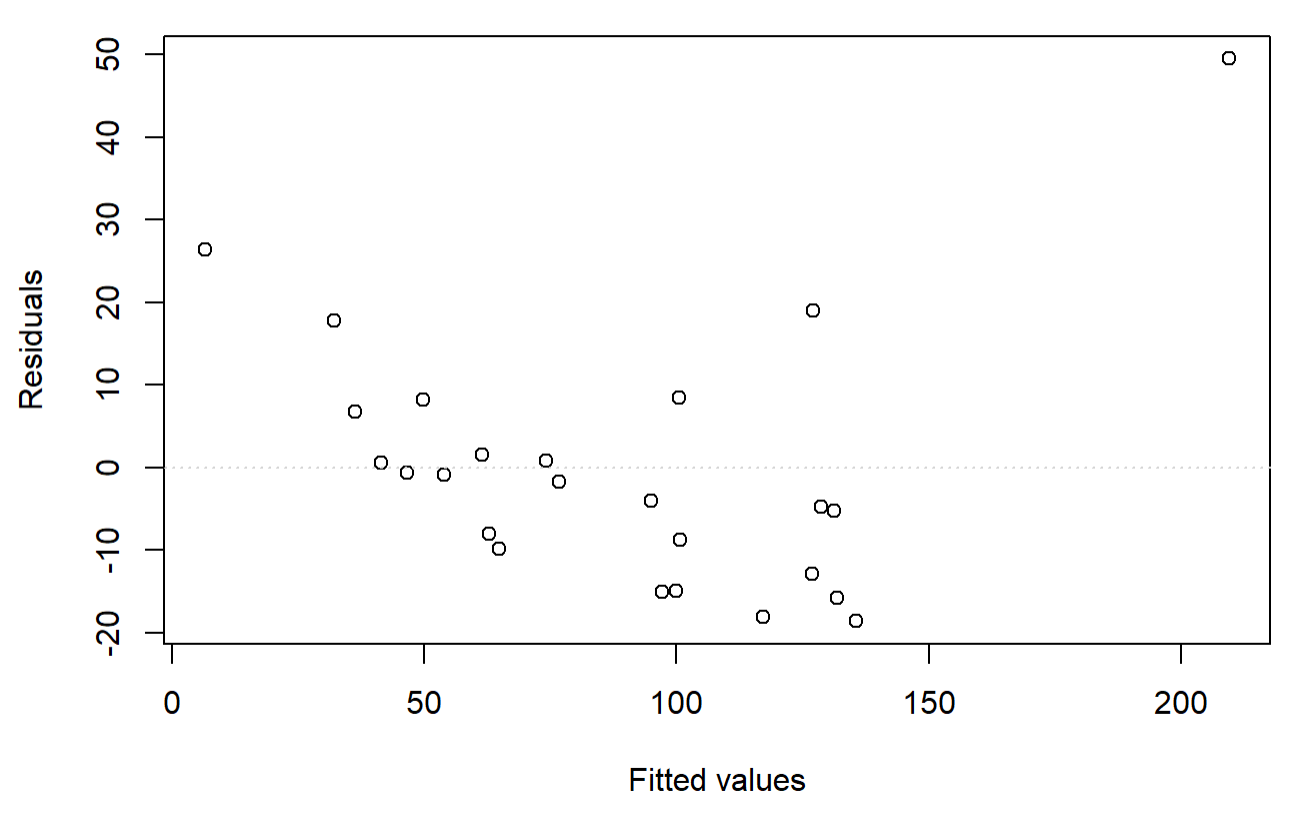
$$Y = X\beta + \epsilon$$

satisfies the assumption that $\epsilon \overset{iid}{\sim} N(0, \sigma^2 I)$. The residuals vs. fitted plot is used to determine whether the mean is 0 and whether the variance is a constant. The first is done by seeing whether there are as many points above and below the horizontal line at 0 units. The second is checked by seeing whether the datapoints in the scatterplot have a cloud-like shape as opposed to a bowed or megaphone shape. This shows that the variance is not influenced by the lower and upper values of the independent variables.

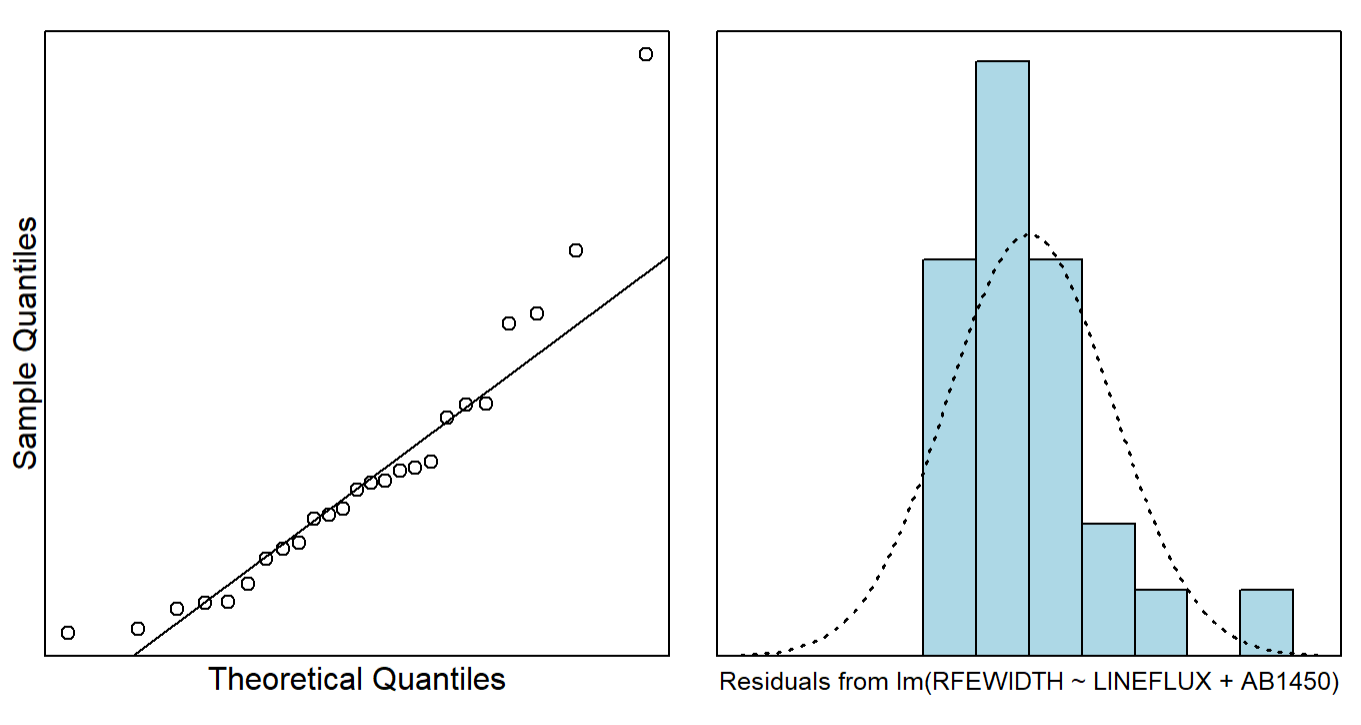
The QQ plot is used to check the normality condition by plotting the theoretical quantiles vs the fitted quantiles from data. The closer the data points lie to the image of the identity function, the stronger evidence that the errors are distrusted normally. Finally, the Shapiro-Wilk Statistic along with its p-value is presented as another check of normality. The NULL hypothesis of this test is that the errors are distributed normally.

```
model = lm(RFEWIDTH~LINEFLUX+AB1450,data=quasar)
ModelCheck(model)
```

```
## [1] "Plot for checking equality of Variance, mean of 0, and independence of errors"
```




```
## [1] "Plot and statistic for checking normality of errors"
```

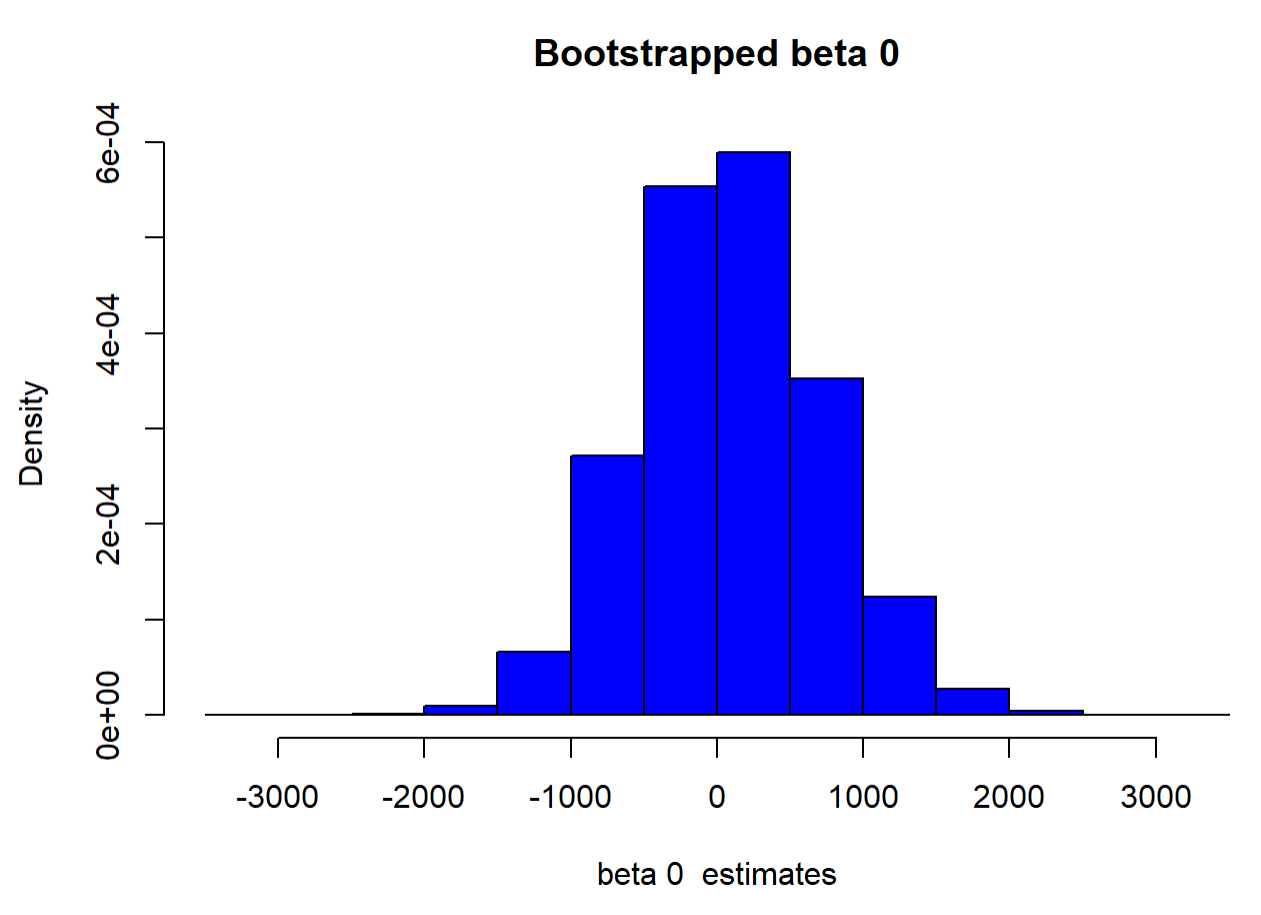


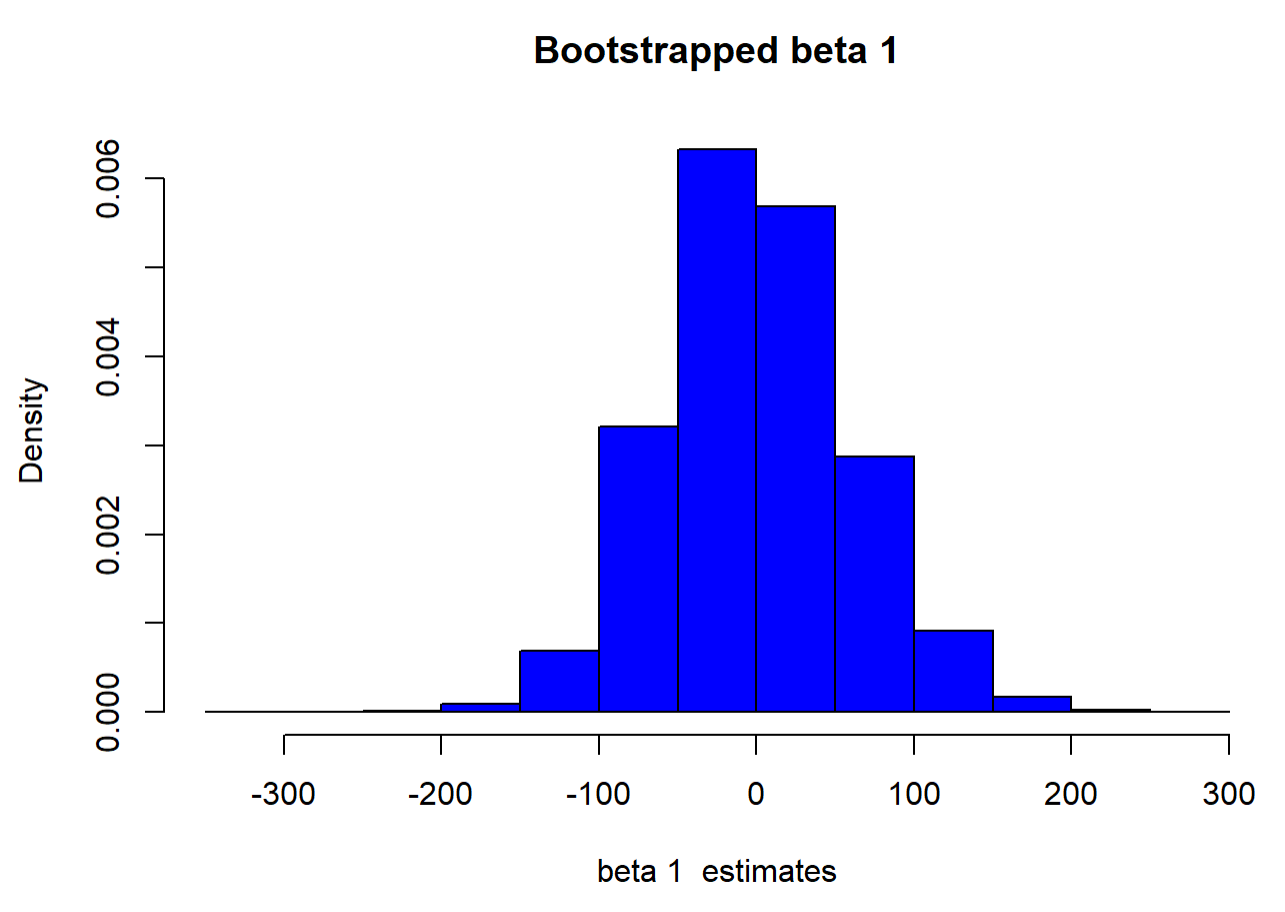

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals
## W = 0.88202, p-value = 0.007607
```

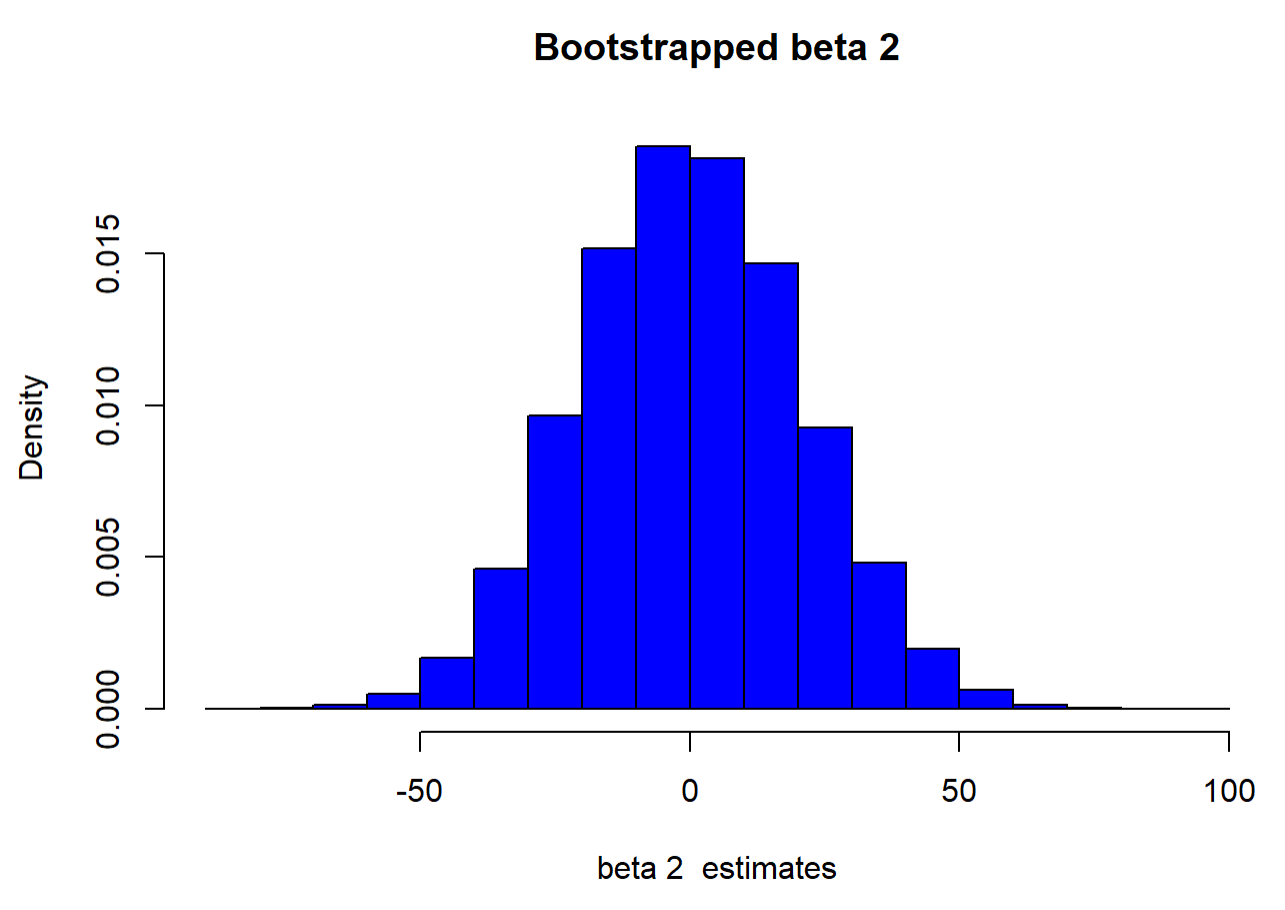
Function 2 (Bootstrap)

This function performs a bootstrap to determine regression coefficients and their confidence intervals to compare to the standard projection matrix approach. It also produces histograms to show the distribution of the estimated coefficients over all the iterations. The function takes in 4 parameters: iter (number of iterations to perform), alpha (significance level for confidence intervals), X (the design matrix, including a column of 1 in the first column), and Y (the response vector).

```
source("Bootstrap.r")
Y= as.matrix(quasar[7])
cols = c(3,5)
X= cbind(rep(1,nrow(quasar)),as.matrix(quasar[,cols]))
Bootstrap(100000,.05,X,Y)
```








```
## [1] "Point Estimate from bootstrap for beta  0"
## [1] 89.95716
## [1] "CI from bootstrap for beta  0"
## [1] -1177.673  1357.587
## [1] "Point Estimate from bootstrap for beta  1"
## [1] 0.2142385
## [1] "CI from bootstrap for beta  1"
## [1] -120.2151  120.6436
## [1] "Point Estimate from bootstrap for beta  2"
## [1] 0.06680556
## [1] "CI from bootstrap for beta  2"
## [1] -41.18577  41.31938
## [1] "Regression Coefficeints (starting at beta0) from classical ML formulas:"
##        RFEWIDTH
## [1,] 1232.77811
## [2,]  205.42383
## [3,]   85.73936
## [1] "Confidence interval for regression coefficients from classical ML formulas:"
## [1]  819.8791 1645.6772
## [1] "Confidence interval for regression coefficients from classical ML formulas:"
## [1] 166.3961 244.4516
## [1] "Confidence interval for regression coefficients from classical ML formulas:"
## [1] 72.16257 99.31616
```

Function 3 (BayesRegress)

This function performs a Bayesian Regression in order to calculate the regression coefficients of our model. Along with these point estimates, it produces posterior quantiles that emulate the confidence intervals of classical/frequentist regression. The function also produces the classical estiamtes and confidence intervals for comparison. Lastly, diagnostic plots like posterior densities and trace plots are shown to assess the efficacy of the regression. The parameters of the function are: formula(a formula such as y~x1+x2 that gives the desired model to be analyzed) and data(the data that our model will come from).

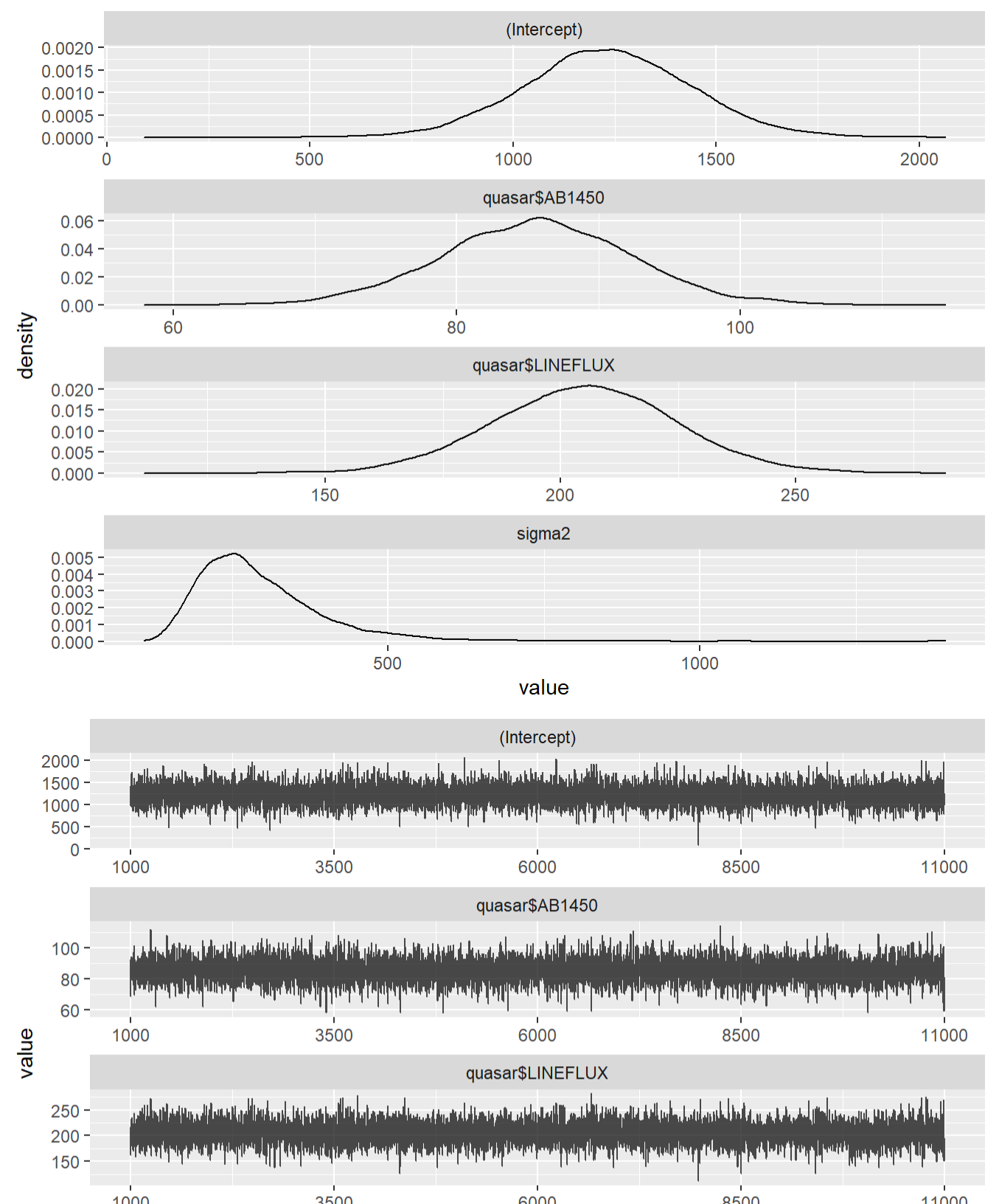
```
source("BayesRegress.r")
formula = quasar$RFEWIDTH~quasar$LINEFLUX+quasar$AB1450
BayesRegress(formula,quasar)
```

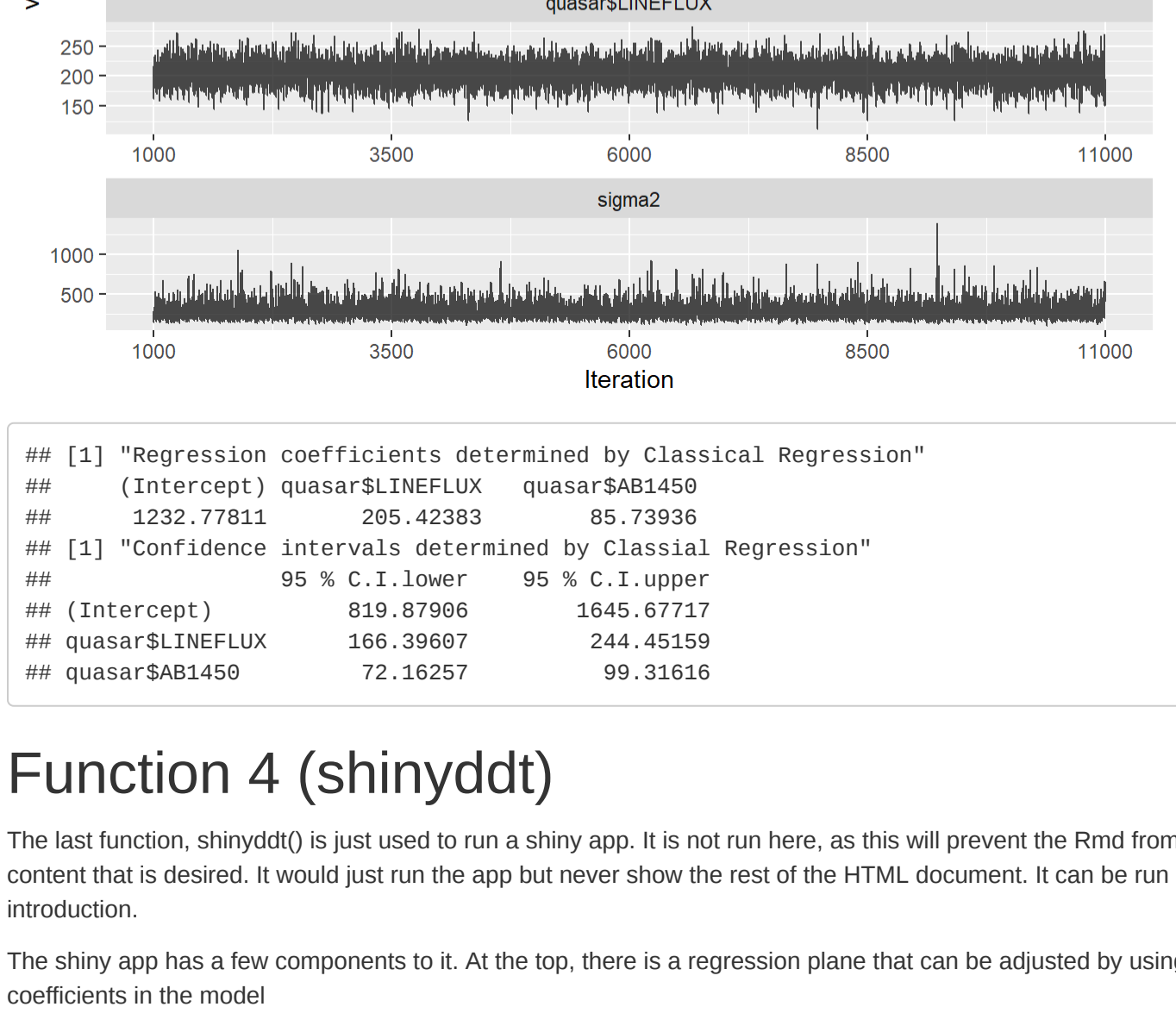
```
## Warning: package 'MCMCpack' was built under R version 4.0.3
```

```
## Warning: package 'coda' was built under R version 4.0.3
```

```
## Warning: package 'ggmcmc' was built under R version 4.0.3
```

```
## [1] "Regression coefficients (as posteriors) and quantiles determined by Bayesian Regression"
##
## Iterations = 1001:11000
## 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)      1231.6 208.392  2.08392        2.18773
## quasar$LINEFLUX   205.1  19.725  0.19725        0.19725
## quasar$AB1450      85.6   6.879  0.06879        0.06879
## sigma2            292.1  98.485  0.98485        1.12191
##
## 2. Quantiles for each variable:
##
##                    2.5%     25%     50%     75%   97.5%
## (Intercept)      829.23 1098.48 1230.49 1368.46 1644.22
## quasar$LINEFLUX 166.55  192.07  205.32  218.12  243.67
## quasar$AB1450    72.02   81.09   85.64   90.14   99.33
## sigma2          159.25  223.31  271.85  338.74  535.39
```






```
## [1] "Regression coefficients determined by Classical Regression"
##     (Intercept) quasar$LINEFLUX   quasar$AB1450
##      1232.77811       205.42383        85.73936
## [1] "Confidence intervals determined by Classial Regression"
##                 95 % C.I.lower    95 % C.I.upper
## (Intercept)          819.87906        1645.67717
## quasar$LINEFLUX      166.39607         244.45159
## quasar$AB1450         72.16257          99.31616
```

Function 4 (shinyddt)

The last function, shinyddt() is just used to run a shiny app. It is not run here, as this will prevent the Rmd from knitting to an HTML with all the content that is desired. It would just run the app but never show the rest of the HTML document. It can be run in the ways mentioned in the introduction.

The shiny app has a few components to it. At the top, there is a regression plane that can be adjusted by using the sliders. The sliders allow the coefficients in the model

$$E[RFEWIDTH] = \beta_0 + \beta_1 LINEFLUX + \beta_2 AB1450$$

to be tuned in order to see how changing the parameters affects the fit of the regression plane. The second plot is still interactive, but the regression plane is the one where the parameters were found using classical regression. You can still do interesting things, as it is a plotly graph.

The last functional part of the app is that allows the user to select data,as a .xls file and with a specific format, from their local disk. If the file is formatted correctly, upon selection the app will produce a residual vs fitted and QQ plot to show whether the chosen model satisfies the regression assumptions. Likewise, the actual data frame is shown as a table, and then under that the regression coefficients of the model using classical regression are given as well. Therefore, the user can with a click determine the appropriate β parameters of their model!

FORMATTING: The format of the .xls file must be the following for the above mentioned functionality to work properly. The column names do not matter, but their order does. This first column must be the response vector. The subsequent columns must be the terms in the model that are desired to be included, where a column of all 1's is needed first.

For example, if we want the model $E[y] = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2$, the first column should be the response Y, the second should be the column of all 1's, followed by the column of x_1, x_2 and $x_1 \cdot x_2$ values, respectively. Effectively, the first column is the response, and the rest of the dataframe should be the design matrix.

SEE QUASAR_FORM.xls file to see an example of the proper formatting.