A Bayesian Approach to Choosing the Best Physiochemical Properties in a Strain of Red Wine

by

Kirillov Konstantinos

Introduction: Description of the problem.

The wine industry has flourished over the years, giving the opportunity to many renowned vineyards to become rich and famous. One such brand of wine is the "*Vinho Verde*", that is aptly named after the demarcated region in northwestern Portugal in which it is grown, bottled and sold. Under this name that carries a long and successful history are lots of varieties of the wine and as it is natural, when there are a lot of varieties, there are also a lot of opinions. Wine connoisseurs say that some varieties of "*Vinho Verde*" are more successful than others. And even though, taste is a very subjective sense, the aim of this report is to try and identify the factors that may have contributed to the success of the red "*Vinho Verde*" and then utilize this model in order to create the best strain of red "*Vinho Verde*" as it is statistically possible.

The Red Wine Data

In order to do our analysis, we were provided with a set of data in an Excel file. However, as we are informed[1], this selection of data was quite limited, due to privacy and logistic issues. Therefore, in total, our data is comprised of 12 variables (*columns*) and 1599 samples (*lines*). The first 11 columns include the physiochemical data, or in other words, our inputs. These data include objective tests performed on the wines, in order to determine their most common physio-chemical characteristics. The last column presents the sensory output. Here the wine experts were asked to taste the wine and afterwards, grade their samples on a scale from 0 to 10. Then, a median of at least 3 evaluations was taken and put in as a value in the last column of our Excel file. This last column consists of some subjective values, as the human taste is not easily quantifiable. Nevertheless, the results were a median of (at least three) experts, and therefore we shall trust their score and judgement.

Fortunately (as we can easily conclude in *R*)[2] there were no missing data, something that will make this task a lot easier.

Therefore, for the sake of summarizing our data, we will create the following Table (*Table 1*), thus providing a quick statistical overview of the data given.

1. Information taken from: P. Cortez, A. Cerdeira, F. Almeida, T. Matos and J. Reis. Modeling wine preferences by data mining from physicochemical properties. In Decision Support Systems, Elsevier, 47(4): 547-553, 2009, pp. 6 - 7.

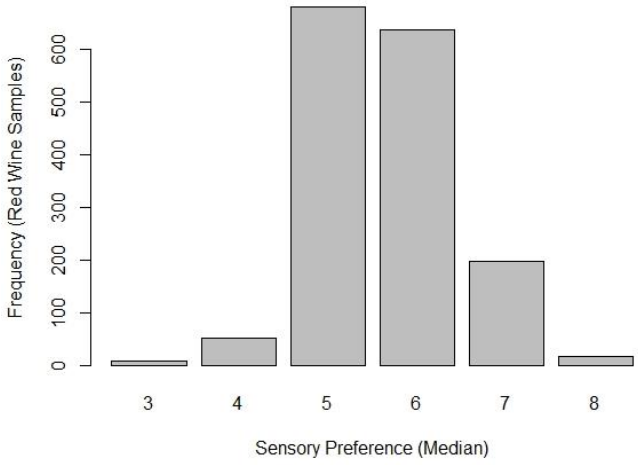
2. Read Appendix 2. (8. *The R Code*)

Table 1. The Physiochemical Data Statistics

Attribute (units)[3]	**Min**	**Max**	**Mean**
fixed acidity $(\boldsymbol{g(tartaric\ acid)/dm^3})$	4.6	15.9	8.319637
volatile acidity $(\boldsymbol{g(acetic\ acid)/dm^3})$	0.12	1.58	0.5278205
citric acid $(\boldsymbol{g/dm^3})$	0	1	0.2709756
residual sugar $(\boldsymbol{g/dm^3})$	0.9	15.5	2.538806
chlorides $(\boldsymbol{g(sodium\ chloride)/dm^3})$	0.012	0.611	0.08746654
free sulfur dioxide $(\boldsymbol{g/dm^3})$	0.001	0.072	0.01587492
total sulfur dioxide $(\boldsymbol{g/dm^3})$	0.006	0.289	0.04646779
density $(\boldsymbol{g/dm^3})$	0.99007	1.00369	0.9967467
pH	2.74	4.01	3.311113
sulphates $(\boldsymbol{g(potassium\ sulphate)/dm^3})$	0.33	2	0.6581488
alcohol $(\%\ \boldsymbol{volume})$	8.4	14.9	10.42298

Furthermore, we shall create a histogram of the sensory data (*Figure 1*). Here we can see that most of the median grades fall in the middle of the plot, while extreme values are not as common. It should be noted that there are only whole numbers, without any decimals.

Figure 1. Sensory Data Plot



3. Unit values were taken from: P. Cortez, A. Cerdeira, F. Almeida, T. Matos and J. Reis. Modeling wine preferences by data mining from physicochemical properties. In Decision Support Systems, Elsevier, 47(4): 547-553, 2009, pp. 23, Table 1.
More on Physiochemical Data in Appendix 1, pp. 9-11.

Model specification

For our analysis, we shall use the *multinomial logistic regression*[4]. Since we are not given any specific information about our coefficients and how they might affect the scores of our judges, we will use low information priors on the betas (β). Therefore, our model will have the following general form:

$$y_i \sim Multinomial(p_{i,K})$$

for $i = 1599$, our number of observations and $K = 1{:}\,6$, our different categories scores (3-8). In this model, our reference category will be 6, which was altered in R to be the most frequent category (the one that had the most observations). Thus, the link function for each level will be:

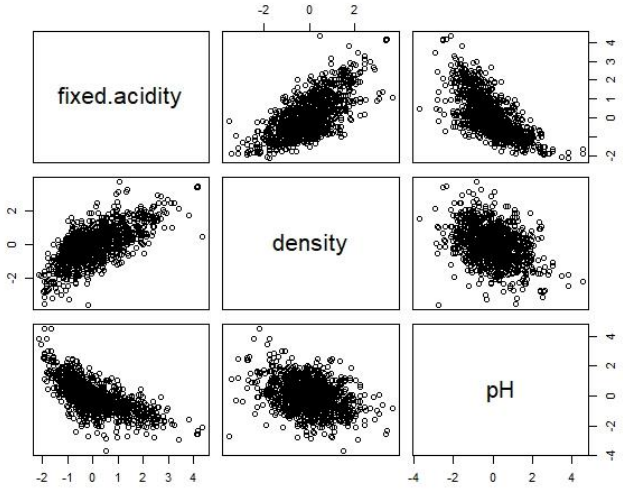
$$Level1{:}\, ln\left(\frac{p_{i1}}{p_{i6}}\right) = \beta_{10} + \beta_{11} \cdot X_{i1} + \beta_{12} \cdot X_{i2} + \cdots + \beta_{1\,11} \cdot X_{i11}$$

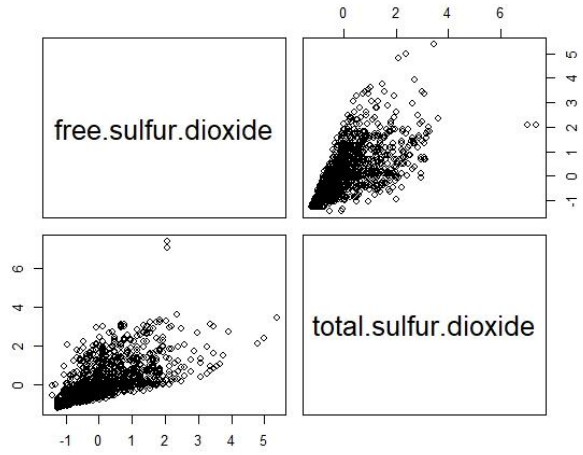
$$Level2{:}\, ln\left(\frac{p_{i2}}{p_{i6}}\right) = \beta_{20} + \beta_{21} \cdot X_{i1} + \beta_{22} \cdot X_{i2} + \cdots + \beta_{2\,11} \cdot X_{i11}$$

$$\vdots$$

$$Level5{:}\, ln\left(\frac{p_{i5}}{p_{i6}}\right) = \beta_{50} + \beta_{51} \cdot X_{i1} + \beta_{52} \cdot X_{i2} + \cdots + \beta_{5\,11} \cdot X_{i11}$$

Initial analysis shows that there might be some form of collinearity between *fixed acidity*, *density* and *pH*, as well as *free* and *total sulfur dioxide*, just as the graphs below indicate:





We are not certain which levels they might affect, but we expect the majority of those trouble-making variables to not coexist, or to do so up to some degree, by the time we select our variables. It should also be noted that to help us with our convergence, we have centered our

4. Appendix 1, pp.12-15.

explanatory variables, by subtracting the mean and dividing the standard deviation of the observations. Therefore, our final results will be bereft of units of measurement.

Variable Selection[5]

Variable selection was done using the *BIC prior*, the *Empirical Bayes* method, the *g-prior* and the *hyper-g* prior on the coefficients of our model, by using both the uniform and beta-binomial priors. Most of these methods were run in both R (by using BAS package) and OpenBUGS software and they did, or at least they should produce similar results that differ only because of the MCMC component in OpenBUGS. Since beta-binomial is more flexible than the uniform on our model priors, because in a sense it lets us choose how parsimonious we want our final model to be, this analysis is based on beta-binomial priors for our model.

After running the models in R, we got the following DIC values:

MODEL	DIC VALUE
Original Model	**3036.0**
Reduced Model (Variable Selection through BIC prior on betas with beta-binomial on the model)	**3063.0**
Reduced Model (Variable Selection through g-prior (of Liang et al.) prior on betas with beta-binomial on the model)	**3058.0**
Reduced Model (Variable Selection through hyper-g prior on betas with beta-binomial on the model)	**3017.0**

The problem with the reduced models which used *BIC prior* and *g-prior* on the betas is that they were too strict with the selection of the statistically significant coefficients up to the point where their final models did not represent the data as well as they should and thus, had much higher DIC values than the original model.

Therefore, since it is the best option of the three, we ended up choosing *hyper-g* priors (as they were modified by Liang et al.) on our model's parameters, with *beta-binomial* priors on the model, in order to make it a bit more parsimonious, if possible. In other words, for the variable selection we had:

$$\beta_{jk} \sim N(0, G \cdot (X^T \cdot X)^{-1} \cdot \sigma^2)$$

for $j = 1, 2, 3, \dots, 11$, our number of estimates and $k = 1, 2, \dots, 5$, our number of levels.

which had the following hyper-prior specification:

$a \longleftarrow 3$
$bw \longleftarrow a/2 - 1$
$w \sim dbeta(1, bw)$
$G \longleftarrow w/(1 - w)$

5. Appendix 1, pp.32-108.

After running our model in BAS package in R, so as to not do any unnecessary mistakes, according to our results, this was the final form of our model's link function, after disposing the superfluous variables:

$$ln\left(\frac{p_{i1}}{p_{i6}}\right) = -8.8 + 1.7 \cdot Z_{i1} + 1.5 \cdot Z_{i2} + 0.6 \cdot Z_{i5} + 1.6 \cdot Z_{i6} - 3.7 \cdot Z_{i7} + 1.6 \cdot Z_{i9} + \\ -1.7 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i2}}{p_{i6}}\right) = -3 + 0.5 \cdot Z_{i2} + 0.2 \cdot Z_{i4} - 0.7 \cdot Z_{i7} + 0.4 \cdot Z_{i9}$$

$$ln\left(\frac{p_{i3}}{p_{i6}}\right) = 0.1 - 0.4 \cdot Z_{i2} - 0.2 \cdot Z_{i5} + 0.3 \cdot Z_{i6} - 0.6 \cdot Z_{i7} + 0.4 \cdot Z_{i10} + 0.8 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i4}}{p_{i6}}\right) = -2.1 + 0.5 \cdot Z_{i1} - 0.8 \cdot Z_{i2} + 0.3 \cdot Z_{i4} - 0.5 \cdot Z_{i5} + 0.3 \cdot Z_{i6} - \\ - 0.9 \cdot Z_{i7} - 0.4 \cdot Z_{i8} + 0.9 \cdot Z_{i10} + 1.4 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i5}}{p_{i6}}\right) = -6.5 - 0.7 \cdot Z_{i1} + 0.7 \cdot Z_{i3} - 0.1 \cdot Z_{i4} - 2 \cdot Z_{i5} + 0.4 \cdot Z_{i6} - \\ - 1.7 \cdot Z_{i7} - 1.1 \cdot Z_{i9} + 1.2 \cdot Z_{i10} + 2.4 \cdot Z_{i11}$$

And now, we are ready to produce some analysis and inference on the exponentiated values of these parameters.

Posterior analysis and interpretation of the results

To summarize our exponentiated parameters, we are giving the following table[6]:

B01	B02	B03	B04	B05	B11	B14	B15	B21	B22
0.0	0.0	1.2	0.1	0.0	6.9	1.6	0.5	4.8	1.7
B23	**B24**	**B35**	**B42**	**B44**	**B45**	**B51**	**B53**	**B54**	**B55**
0.7	0.5	2.2	1.3	1.4	1.0	2.0	0.8	0.6	0.2
B61	**B63**	**B64**	**B65**	**B71**	**B72**	**B73**	**B74**	**B75**	**B84**
7.5	1.3	1.4	1.7	0.1	0.5	0.5	0.4	0.2	0.7
B91	**B92**	**B95**	**B103**	**B104**	**B105**	**B111**	**B113**	**B114**	**B115**
6.3	1.5	0.4	1.5	2.4	3.4	0.2	2.2	4.1	11.1

So let us see what they mean (well, at least for level 1):

6. The result of this table came from the final model, which can also be found in Appendix 1, pp.122-124.

For level 1:

$B01 = exp\{\beta_{01}\} \cong 0$: If we do not include any of our physiochemical characteristics in this level at all, then there is almost (**it is not exactly zero**) no chance that our wine will get a score of 3, but rather that of 5. It is kind of nonsensical as a result, but maybe the other factors that are not included in this level are enough to make this distinction between these two scores.

$B11 = exp\{\beta_{11}\} = 6.9$: Is saying to us that should we keep all the other variables of this level fixed and simultaneously increase the value of fixed acidity by 1 (no units of measurement, remember?), then our odds of receiving a score of 3, rather than 5 will increase by a huge 590%.

$B21 = exp\{\beta_{21}\} = 4.8$: Tells us that should we increase the value of volatile acidity by 1 unit (no units of measurement), all while keeping the other variables constant, then the odds of being in category 1 that gets a median score of 3, will increase drastically by 380%, instead of being in category 6, which gets a median score of 5.

$B51 = exp\{\beta_{51}\} = 2$: Tells us that by keeping all other variables of this level fixed, but also increasing the value of chlorides by 1 (no units of measurement), our odds of receiving a score of 3 instead of 5 will increase by 100%.

$B61 = exp\{\beta_{61}\} = 7.5$: Is an indication that should we keep all other variables of this level constant in their values, while increasing the value of free sulfur dioxide by 1 (no units of measurement), our odds of receiving a score of 3 instead of 5 will increase by an enormous 650%.

$B71 = exp\{\beta_{71}\} = 0.1$: This value points out that if we keep all the other variables of this level fixed and simultaneously increase the value of total sulfur dioxide by 1 (no units of measurement), then the odds of getting a score of 3 rather than 5 will decrease by 90%.

$B91 = exp\{\beta_{91}\} = 6.3$: Tells us that if we keep all the other variables of this level fixed, all the while increasing the value of pH by 1 (no units of measurement), then the odds of getting a score of 3 rather than 5 will increase by an astonishing 530%.

$B111 = exp\{\beta_{111}\} = 0.2$: Is saying to us that should we keep all the other variables of this level fixed, but also increase the value alcohol by 1 (no units of measurement), then the odds of getting a score of 3 rather than 5 will diminish by 80%.

And this analysis follows in the same manner for all other levels[7]. So, what can we surmise from these results?

1. It appears that alcohol as well as sulphates appealed to the taste of our judges and in fact those were the two factors that could, in moderation, make or break our wine's score.
2. There was no good reason to keep a high level of volatile acid in our wine, as it usually made our wine's grade drop to a lower level. In fact, it is not even included in level 5.
3. The same thing can be said about chlorides, as higher quantities of this physiochemical characteristic only make our wine's score drop to a lower value.

7. For more analytical results, please check: Appendix 1, pp.124-129.

4. This can be also extended to pH. It is preferable to keep low values of this physiochemical characteristic in our wines.
5. Free sulfur dioxide adds to the taste, but should be kept in moderation in our wines as it can potentially change our final score to the lowest.
6. Density appears only in level 4 and ruins the chances for our wine to get a higher score, so it should be kept at low values in our wine.
7. Total sulfur dioxide does not contribute to any grade whatsoever, apart from maybe that of 5. Thus, it should be kept at low levels in our wines.

And apart from those variables, the others of our model either contributed or took away from the scores of each level.

Another good question would be, what if we wanted to produce only wines that scored 6 and above? What good physiochemical characteristics should we pay special attention to?

In order to answer that question, we ran a simple binomial model[8], which divided our data into two categories (5 and below, versus 6 and above). Here we used a stricter approach and considered a BIC prior on the betas, with a beta-binomial on the model. The results showed us that free sulfur dioxide, sulphates and alcohol worked in favour of our score and thus should be kept in high values in our model, while volatile acidity, chlorides and total sulfur dioxide should receive values that are as low as possible.

This result along with the one above should work together, when we "fine tune" the physiochemical characteristics in order to produce the best tasting red variant of "*Vinho Verde*", so as to know what variables should be taken with greater care.

Final comments and conclusion

All in all, this was our suggestion based on the analysis done. Making a fine bottle of wine is an art in itself, put too much alcohol in and you will have poison, put too little and you will have grape juice. And even though those physiochemical characteristics are very important to have in small or large quantities in our wines, they are but a small fraction of what makes a wine unique and successful. The climate, the type of grapes, the soil, the way it is bottled, all these and more play an important role in the final product and its score.

Therefore, this recipe is not going to guarantee us a good quality of wine, because it is so much more than that. It does however provide us with a good indication that we are on the right (or the wrong) track to making a tasteful red "*Vinho Verde*" that people can enjoy. And at the end of the day, that is what really matters.

8. More details in: Appendix 1, pp.129-137.

APPENDIX 1

(A more Analytical Solution with Tables & Diagrams)

TABLE OF CONTENTS:

The Data:

We are given a set of data that are related to the red variant of the Portuguese "Vinho Verde" wine. In total, the researchers took 1599 samples and collected their findings in an Excel file.

The file itself consists of 12 columns and 1599 rows. The first 11 columns are the most common physiochemical characteristics of each sample and we are informed that their collection was done in a laboratory that specializes in that procedure and thus can be trusted. These shall be our explanatory variables, and so by definition, they will try to give an explanation, on why the score (our response variable) was given that particular value.

The last column (column 12) contains the response variable of the data set, for each of the 1599 samples that were taken. This value is a median of at least three experts that have tasted a variety of that wine and given a certain score.

The Explanatory Variables:

For our convenience, the table below gives us an overview of all the 11 explanatory variables. It should be noted that the units of the variable *free sulfur dioxide*, as well as the units of *total sulfur dioxide* were transformed form mg/dm^3 to g/dm^3.

Table 1. Physiochemical Characteristics

Χαρακτηριστικό (μονάδες μέτρησης)	**Min**	**Max**	**Μέσος**
fixed acidity $(\boldsymbol{g(tartaric\ acid)/dm^3})$	4.6	15.9	8.319637
volatile acidity $(\boldsymbol{g(acetic\ acid)/dm^3})$	0.12	1.58	0.5278205
citric acid $(\boldsymbol{g/dm^3})$	0	1	0.2709756
residual sugar $(\boldsymbol{g/dm^3})$	0.9	15.5	2.538806
chlorides $(\boldsymbol{g(sodium\ chloride)/dm^3})$	0.012	0.611	0.08746654
free sulfur dioxide $(\boldsymbol{g/dm^3})$	0.001	0.072	0.01587492
total sulfur dioxide $(\boldsymbol{g/dm^3})$	0.006	0.289	0.04646779
density $(\boldsymbol{g/dm^3})$	0.99007	1.00369	0.9967467
pH	2.74	4.01	3.311113
sulphates $(\boldsymbol{g(potassium\ sulphate)/dm^3})$	0.33	2	0.6581488
alcohol $(\%\ \boldsymbol{volume})$	8.4	14.9	10.42298

A bit more on those parameters:

- ***Fixed Acidity*** $\boldsymbol{(g(tartaric\ acid)/dm^3)}$**:** Fixed acids are fruit acids (nonvolatile) that are organic to grapes. This value represents the most acids, which are contained in a bottle of wine that are constant, meaning that when a person opens the bottle, these acids do

not evaporate as easily as others. Here, it is a continuous variable, which is measured in grams of tartaric acid, over cubic decimeter. It takes values from 4.6, to 15.9 and has a mean of 8.319637.

- ***Volatile Acidity*** $(\boldsymbol{g(acetic\ acid)/dm^3})$**:** Volatile acidity is an unstable acid formed by dissolving carbon dioxide in water. It is the basis of carbonated beverages and is related to the carbonate group of compounds. This value represents the amount of acetic acid in a bottle of wine. Should this value be high enough, it gives the wine the unpleasant taste of vinegar. In our data, it is a continuous variable, which is measured in grams of acetic acid, per cubic decimeter. It takes values from 0.12, to 1.58 and has a mean value of 0.5278205.

- ***Citric Acid*** $(\boldsymbol{g/dm^3})$**:** In small quantities, it adds a sense of "freshness" to the wines. It is used in many foods, confections, and soft drinks to improve their stability in natural metals. Here, it is a continuous variable, which is measured in grams of citric acid, over cubic decimeter. It takes values from 0, to 1 and has a mean value of 0.2709756.

- ***Residual Sugar*** $(\boldsymbol{g/dm^3})$**:** It is the amount of residual sugar that remains after the procedure of fermentation has stopped. Usually, it takes values between 1 g/dm^3 and 45 g/dm^3, with the latter value representing the sweet wines. In our data, it is a continuous variable, which is measured in grams of residual sugar, per cubic decimeter. It takes values from 0.9, to 15.5 and has a mean value of 2.538806. Thus, we can safely say that these samples did not contain any sweet wines.

- ***Chlorides*** $(\boldsymbol{g(sodium\ chloride)/dm^3})$**:** Is the amount of salt, which is contained in a wine. Here, it is a continuous variable, which is measured in grams of sodium chloride, per cubic decimeter. It takes values from 0.012, to 0.611 and has a mean value of 0.08746654.

- ***Free Sulfur Dioxide*** $(\boldsymbol{g/dm^3})$**:** Is a chemical component that is used to shield the wine from undesirable microbes and unwanted oxidation, which spoil the good taste of the wine. Here we have altered its unit of measurement from mg/dm^3 to g/dm^3. Thus, in our data, it is a continuous variable, which is measured in grams of free sulfur dioxide, per cubic decimeter. It takes values from 0.001, to 0.072 and has a mean value of 0.01587492.

- ***Total Sulfur Dioxide*** $(\boldsymbol{g/dm^3})$**:** This physiochemical parameter contains the previous variable (meaning, free sulfur dioxide), as well as the bound from of sulfur dioxide in its value. Sulfur dioxide is mostly difficult to detect through the common senses, however in large quantities (greater than 0.05 g/dm^3), it becomes perceptible through its odor and its taste. Once again, we have changed the unit of measurement from mg/dm^3 to g/dm^3. Thus, in our data, it is a continuous variable, which is measured in grams of total sulfur dioxide, per cubic decimeter. It takes values from 0.006, to 0.289 and has a mean

value of 0.04646779. Therefore, we can see that there were samples where it was perceptible and perhaps distasteful.

- ***Density*** (g/dm^3)**:** It measures the density of the wine as a liquid. Dry wines have less density than sweet ones. Density is defined in a qualitative manner as a measure of the relative `heaviness' of an object with constant volume. Here, it is a continuous variable, which is measured in grams over cubic decimeter. It takes values from 0.99007, to 1.00369 and has a mean value of 0.9967467. Hence, here there are only dry wines.

- ***pH*:** This parameter refers to the degree of acidity in our wine. At first it took values on a scale from 0 to 14, but today this scale has been expanded. For the red wine a $pH < 3.4$ is recommended. In our data, it is a continuous variable, which is measured in pH. It takes values from 2.74, to 4.01 and has a mean value of 3.311113.

- ***Sulphates*** $(g(potassium\ sulphate)/dm^3)$**:** It is an additive mainly used in foods. It helps the wine to preserve its "freshness", while also improving the taste. Just like the sulfur dioxide, this also helps shielding the wine from harmful microbes and oxidization. They are also known to cause the annoying headache, that people commonly know as hangover. In these data, it is a continuous variable, which is measured in grams of potassium sulphate per cubic decimeter. It takes values from 0.33, to 2 and has a mean value of 0.6581488. Therefore, if we consider that the maximum limit in the USA is 0.35 g/dm^3, then we can certainly say that although the wines had a good taste, they must have left the wine experts with one of the greatest hangovers in the history of hangovers.

- ***Alcohol*** $(\%\ volume)$**:** This parameter measures the percentage of alcohol in % of volume. This is what separates the wines from grape juice. In our data, it takes values from 8.4% to 14.9%, while having a mean value of 10.42298%.

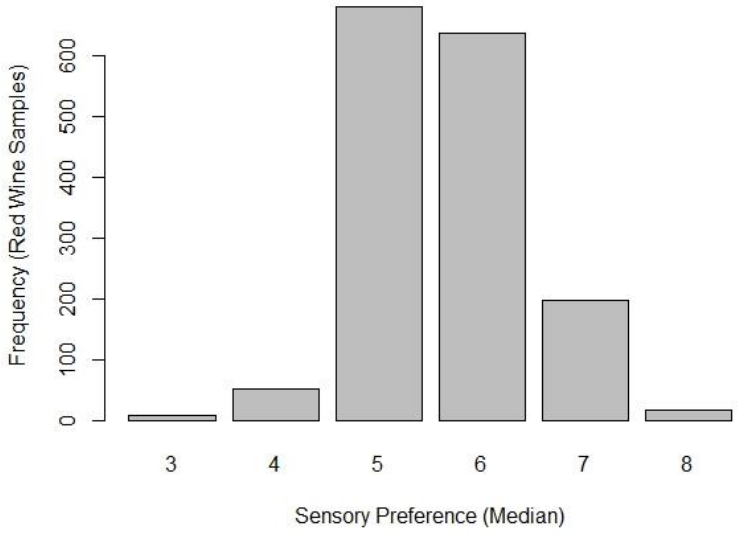
Response Variable:

The wine connoisseurs did not know which wine brand of "Vinho Verde" they were sampling each time. When they tried one wine and graded it according to their taste, this did not give them any information about the previous or the next sample. Thus, we can safely assume that those trials were independent to each other.

All in all, the sampling procedure would unfold in the following manner:

At least three experts would taste a certain type of wine, which had the same physiochemical characteristics and then evaluated it, based on their preference and knowledge in the field of wines. Then the researchers would collect those scores, for those specific physiochemical parameters and take a median value of at least three of them. That is how each line of column 12 was created. A histogram of our response variable would be:

Diagram 1. Sensory preference



As we can see in this histogram, the categories are positive integers. There are no intermediate values like 4.5, or 6.856 for instance. Most of the median values are either scored as 5 or 6, while extreme values of 3 and 8 do not have a lot of observations. Thus, the majority of those medians refer to medium tasting wine. Furthermore, even though the scores should range from 1 to 10, we only have observations for values 3 to 8 and so in total we have 6 observed categories, instead of 11.

The Model (Multinomial Logistic Regression):

After almost two months of suffering and agony, due to the striking the inexperience of the author, the final model to which we concluded was the Multinomial Logistic Regression. There were of course other models that were considered, but then rejected before we ended up with this one.

The most obvious of all the models was at first the Ordinal Logistic Regression (Proportional-Odds Logit Model), because our response variable is ordered, meaning that a grade of 8 denotes a better tasting wine, than if it received a grade of 7, or 6, or God forbid 2. However, this model was rejected, because the proportional-odds (parallel lines) assumption was not in effect. So, for example if we ran that in SPSS, we would get:

Test of Parallel Lines[a]

Model	-2 Log Likelihood	Chi-Square	df	Sig.
Null Hypothesis	3074.767			
General	2919.061[b]	155.707[c]	44	.000

The null hypothesis states that the location parameters (slope coefficients) are the same across response categories.

a. Link function: Logit.

b. The log-likelihood value cannot be further increased after maximum number of step-halving.

c. The Chi-Square statistic is computed based on the log-likelihood value of the last iteration of the general model. Validity of the test is uncertain.

Obviously this is a violation, which means that the coefficients of the model are not the same for each level, and so instead of having something more humane, like:

$$ln\left(\frac{\pi_1}{\pi_2+\pi_3+\cdots+\pi_K}\right)=a_1+\beta_1\cdot X_1+\beta_2\cdot X_2+\cdots+\beta_p\cdot X_p$$

$$ln\left(\frac{\pi_1+\pi_2}{\pi_3+\pi_4+\pi_5}\right)=a_2+\beta_1\cdot X_1+\beta_2\cdot X_2+\cdots+\beta_p\cdot X_p$$

$$\vdots$$

$$ln\left(\frac{\pi_1+\pi_2+\cdots+\pi_4}{\pi_5}\right)=a_5+\beta_1\cdot X_1+\beta_2\cdot X_2+\cdots+\beta_p\cdot X_p$$

Which has only 16 total variables ($p=11$ here), instead we have:

$$Level\ 1=a_1+\beta_{11}\cdot X_1+\beta_{12}\cdot X_2+\cdots+\beta_{1p}\cdot X_p$$

$$Level\ 2=a_2+\beta_{21}\cdot X_1+\beta_{22}\cdot X_2+\cdots+\beta_{2p}\cdot X_p$$

$$\vdots$$

$$Level\ 5=a_5+\beta_{51}\cdot X_1+\beta_{52}\cdot X_2+\cdots+\beta_{5p}\cdot X_p$$

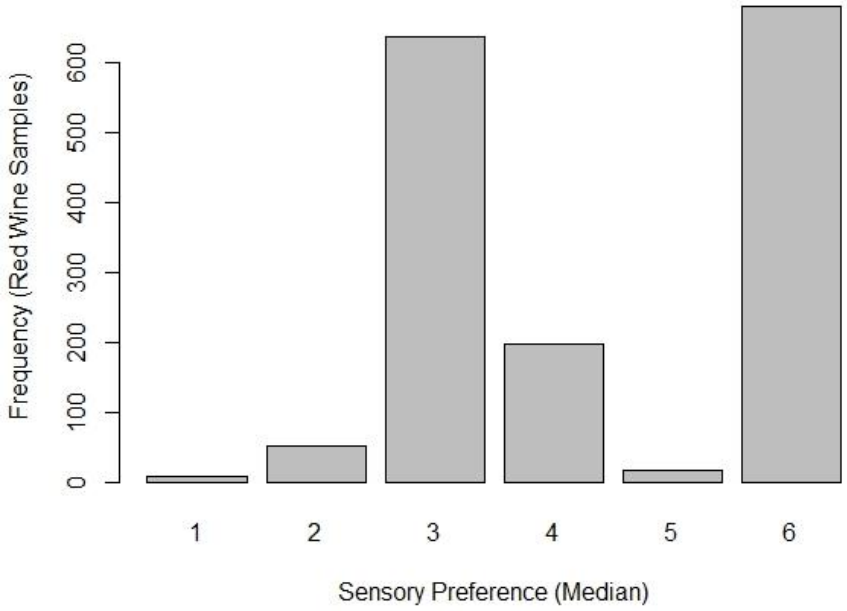
That are a total of 60 variables to estimate. But thankfully there are ways to do that as well. And just like good wine, these ways take time.

So, the proportional-odds model does not work, what can one do then? Well, the first thing that should come to mind, would be a Multinomial Logistic Regression, which is a more general case of the model that did not fit our data.

This model does not suppose an order between the levels, nor does it impose a common value for our covariates. Therefore, in our wine data set, we will ignore that the value 2 is greater than 1, or that 5 is better than 4, 3 and 2, for instance, and instead treat them as independent categories.

The idea of how this model works is it separates the multinomial model of K categories, into a series of (binary) logistic $K-1$ models, for each pair of response categories. In order to compare the log-odds model, we need a common baseline category which will be our point of reference.

Usually, the most "meaningful" category is chosen to be the reference category, or the most frequent category, or either the first or the last category of our response variable. Almost any category, as long as it is not an outlier or something rare, that has very few observations. In our data, the most frequent category is the third one and so it will be chosen to be the reference category. What we shall do now, is switch their order and place the reference category in the last position, which will help us later in the programming part. Therefore, if we see a graph after we organize our data, we will have:



Notice that our categories are individual observations and are not grouped, so the response is a variable, which denotes the level (so 1, 2, 3, etc.) at which we currently find ourselves for each of the 1599 observations. Therefore, we have:

$$y_i \sim Categorical(p_{i,K})$$
$$or\ equally$$

$$y_i \sim Multinomial(p_{i,K}, N_i = 1)$$

In total there are 11 explanatory variables and six categories (3-8), so if we make the sixth category our reference group (because it has the largest frequency in the group), then the baseline-category probability will be:

$$p_{i6} = \frac{1}{1 + \sum_{k=1}^{5} exp\{\beta_{0k} + \beta_{1k} \cdot X_{i1} + \cdots + \beta_{11k} \cdot X_{i11}\}}$$

While the other probabilities will be:

$$p_{ij} = \frac{exp\{\beta_{j0} + \beta_{j1} \cdot X_{i1} + \cdots + \beta_{j11} \cdot X_{i11}\}}{1 + \sum_{k=1}^{5} exp\{\beta_{k0} + \beta_{k1} \cdot X_{i1} + \cdots + \beta_{k11} \cdot X_{i11}\}}, \qquad for\ j = 1, 2, \dots, 5$$

Or alternatively, for each level we will have:

$$ln\left(\frac{p_{i1}}{p_{i6}}\right) = \beta_{10} + \beta_{11} \cdot X_{i1} + \beta_{12} \cdot X_{i2} + \cdots + \beta_{1\,11} \cdot X_{i11}$$

$$ln\left(\frac{p_{i2}}{p_{i6}}\right) = \beta_{20} + \beta_{21} \cdot X_{i1} + \beta_{22} \cdot X_{i2} + \cdots + \beta_{2\,11} \cdot X_{i11}$$

$$\vdots$$

$$ln\left(\frac{p_{i5}}{p_{i6}}\right) = \beta_{50} + \beta_{51} \cdot X_{i1} + \beta_{52} \cdot X_{i2} + \cdots + \beta_{5\,11} \cdot X_{i11}$$

As for the last probability:

$$p_{i6} = 1 - \sum_{j=1}^{5} p_{i5}$$

Since all these probabilities will eventually sum to 1.

Running the original model:

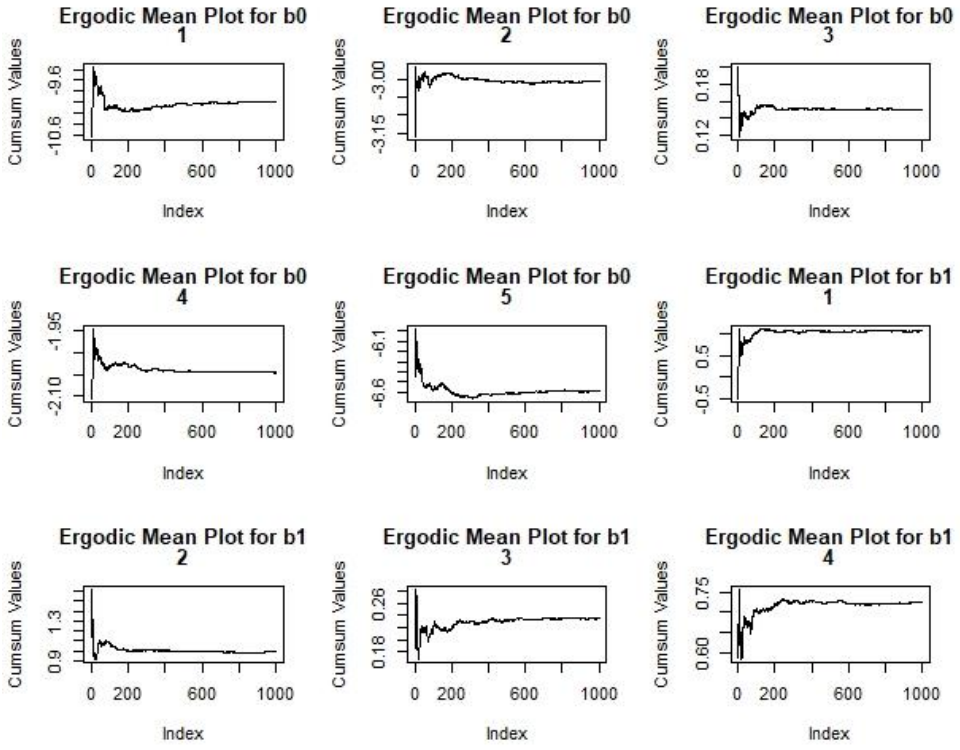
Before we start doing variable selection, we should do a trial run and see whether the model we chose converges for the parameters that we wish to estimate. Here of course we want to

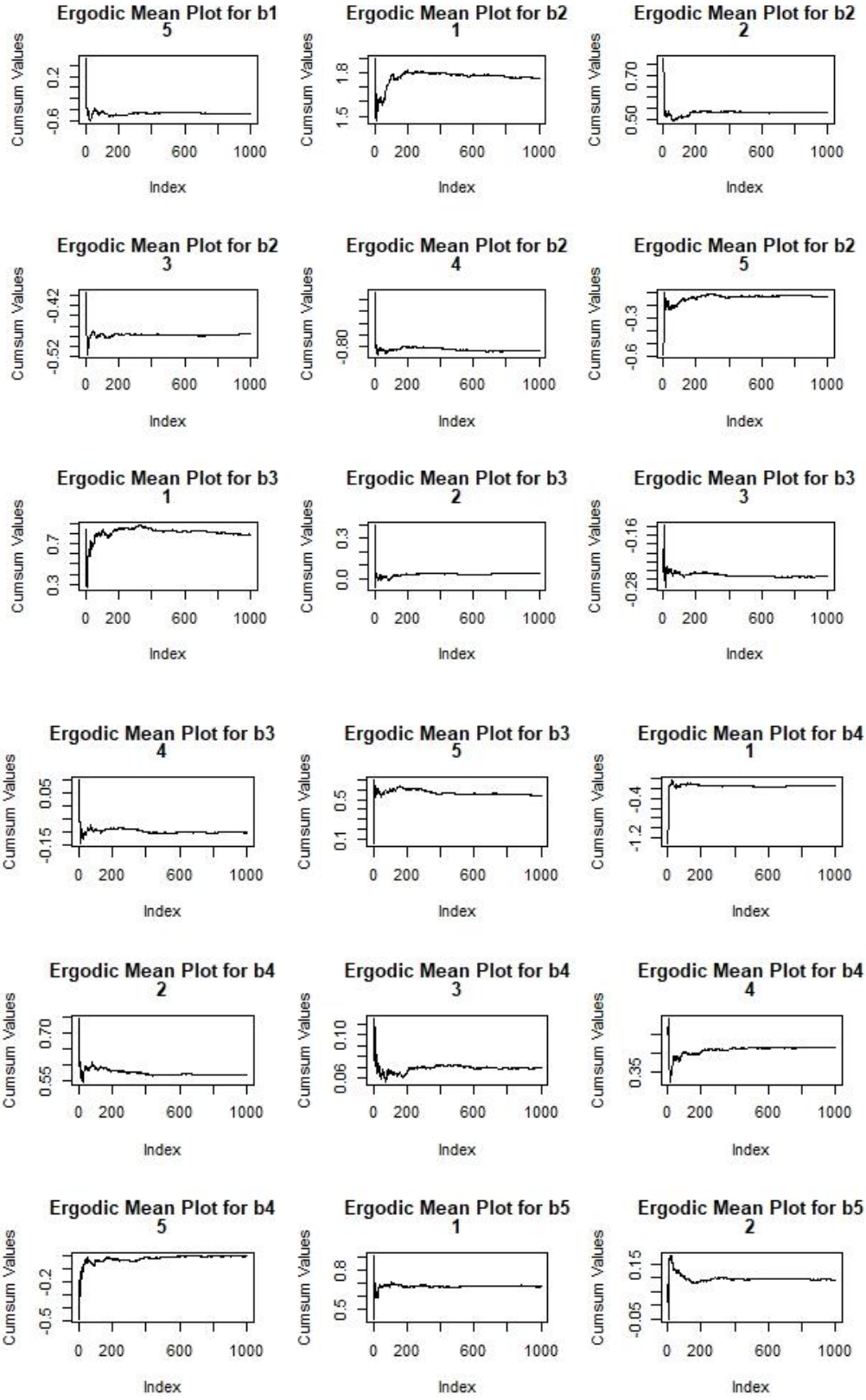
estimate the coefficients of the physiochemical data, but we are given no information as to a set of values that they should or should not take. Therefore, we are going to be vague in our priors chosen. By setting the mean of all βs equal to zero (because it is the center of the real numbers set $\mathbb{R}$ that goes from $-\infty$ to ∞) and the precision to a low value (here 0.0001 or 1.0E-4), we state that we have low information priors on those coefficients and so we shall let the data determine their real values. Also, before running our program, we shall center the values of the physiochemical data, to assist us in both the speed of the convergence and autocorrelation problems.

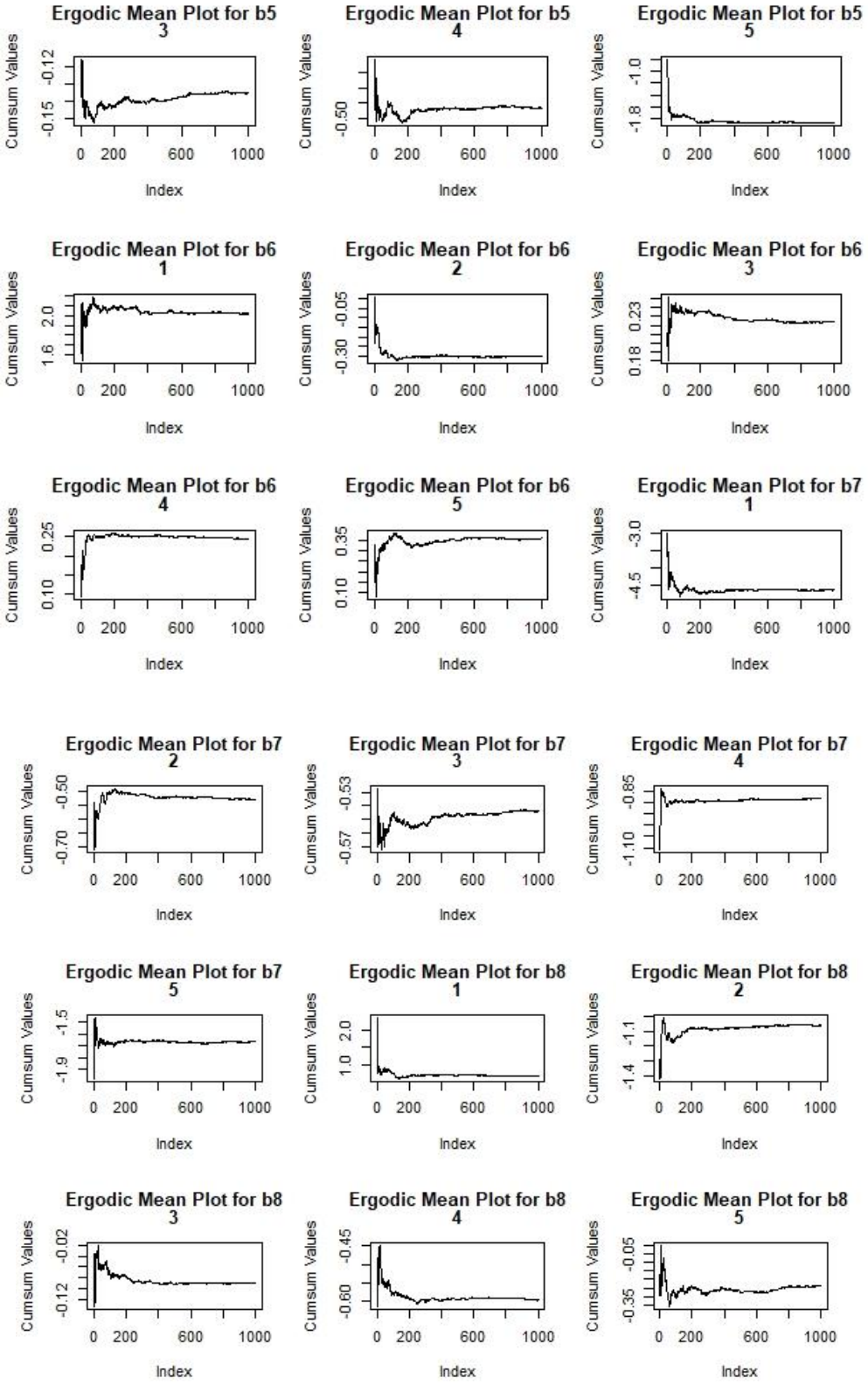
Having done some initial runs, the plots showed some significant autocorrelation problems up until 30 lags. Therefore, we will also apply a thin of 30, to rectify this problem. This procedure, even though we have only 1 chain is quite slow, and therefore we shall do the following: We will discard the first 1000 iterations as a burn-in period and then run our program for a total of 30000 iterations, which according to the thinning value that we have set, should return 1000 samples to us.

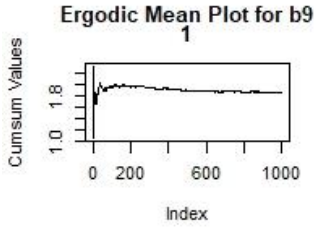
Now having done all that, we run our program based on the *multinomial.txt* model and get the following results.

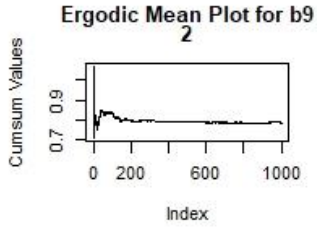
Convergence plots on the multinomial logistic regression (original model):

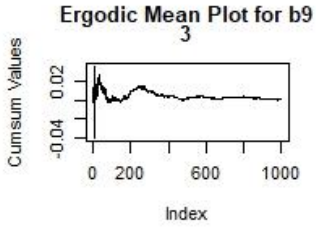


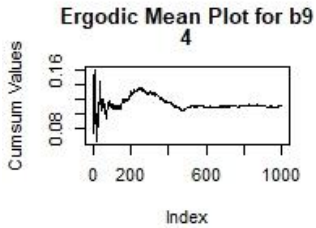

Ergodic Mean Plot for b1 5
Ergodic Mean Plot for b2 1
Ergodic Mean Plot for b2 2
Ergodic Mean Plot for b2 3
Ergodic Mean Plot for b2 4
Ergodic Mean Plot for b2 5
Ergodic Mean Plot for b3 1
Ergodic Mean Plot for b3 2
Ergodic Mean Plot for b3 3
Ergodic Mean Plot for b3 4
Ergodic Mean Plot for b3 5
Ergodic Mean Plot for b4 1
Ergodic Mean Plot for b4 2
Ergodic Mean Plot for b4 3
Ergodic Mean Plot for b4 4
Ergodic Mean Plot for b4 5
Ergodic Mean Plot for b5 1
Ergodic Mean Plot for b5 2
Cumsum Values
Index
0
200
600
1000

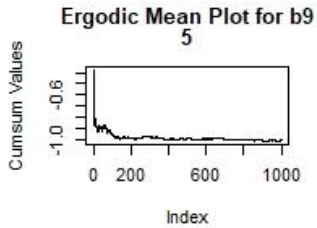

Ergodic Mean Plot for b5 3
Ergodic Mean Plot for b5 4
Ergodic Mean Plot for b5 5
Ergodic Mean Plot for b6 1
Ergodic Mean Plot for b6 2
Ergodic Mean Plot for b6 3
Ergodic Mean Plot for b6 4
Ergodic Mean Plot for b6 5
Ergodic Mean Plot for b7 1
Ergodic Mean Plot for b7 2
Ergodic Mean Plot for b7 3
Ergodic Mean Plot for b7 4
Ergodic Mean Plot for b7 5
Ergodic Mean Plot for b8 1
Ergodic Mean Plot for b8 2
Ergodic Mean Plot for b8 3
Ergodic Mean Plot for b8 4
Ergodic Mean Plot for b8 5
Cumsum Values
Index
0 200 600 1000

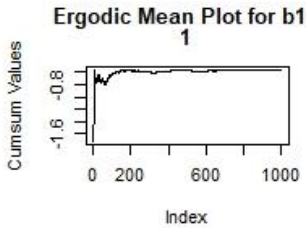

Ergodic Mean Plot for b9
1
Cumsum Values
1.0 1.8
0 200 600 1000
Index

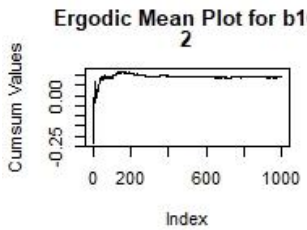

Ergodic Mean Plot for b9
2
Cumsum Values
0.7 0.9
0 200 600 1000
Index

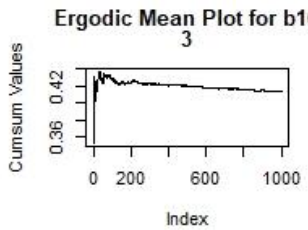

Ergodic Mean Plot for b9
3
Cumsum Values
-0.04 0.02
0 200 600 1000
Index

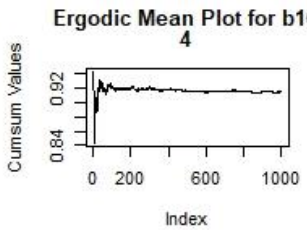

Ergodic Mean Plot for b9
4
Cumsum Values
0.08 0.16
0 200 600 1000
Index

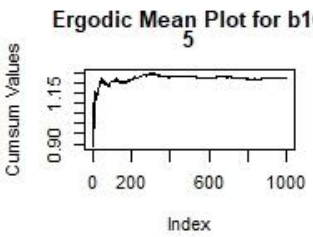

Ergodic Mean Plot for b9
5
Cumsum Values
-1.0 -0.6
0 200 600 1000
Index

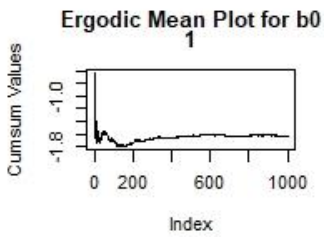

Ergodic Mean Plot for b10
1
Cumsum Values
-1.6 -0.8
0 200 600 1000
Index

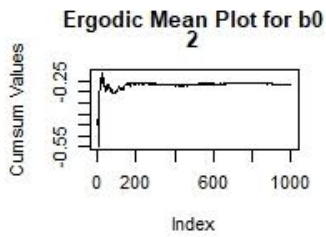

Ergodic Mean Plot for b10
2
Cumsum Values
-0.25 0.00
0 200 600 1000
Index

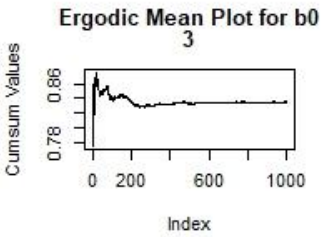

Ergodic Mean Plot for b10
3
Cumsum Values
0.36 0.42
0 200 600 1000
Index

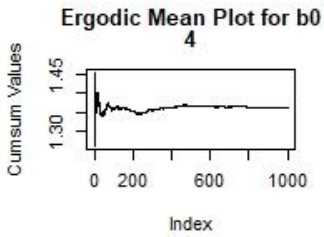

Ergodic Mean Plot for b10
4
Cumsum Values
0.84 0.92
0 200 600 1000
Index

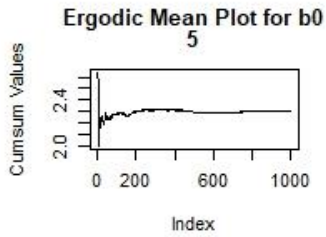

Ergodic Mean Plot for b10
5
Cumsum Values
0.90 1.15
0 200 600 1000
Index

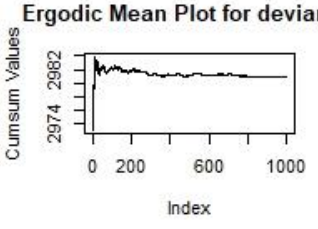

Ergodic Mean Plot for b0
1
Cumsum Values
-1.8 -1.0
0 200 600 1000
Index

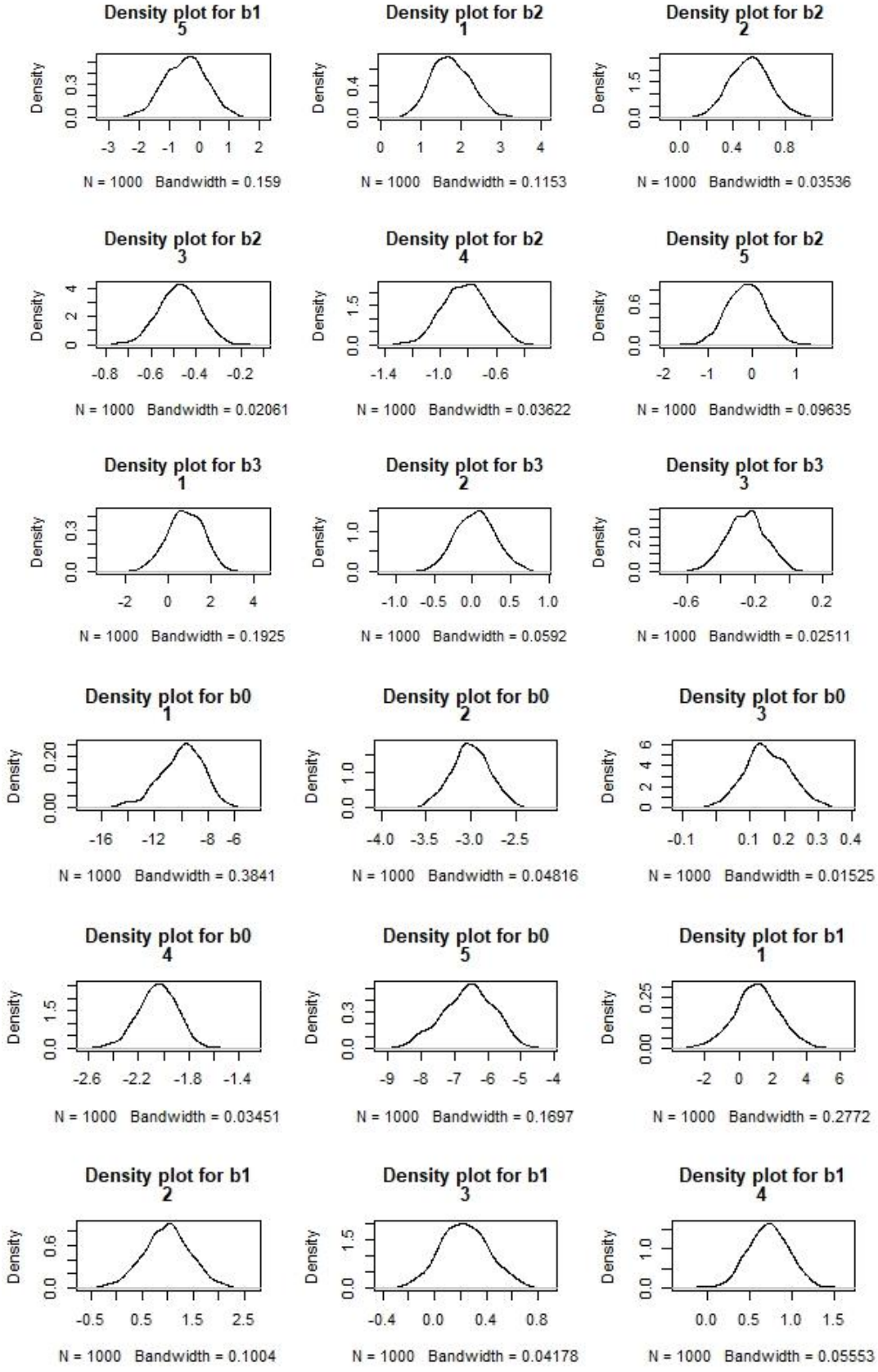

Ergodic Mean Plot for b0
2
Cumsum Values
-0.55 -0.25
0 200 600 1000
Index

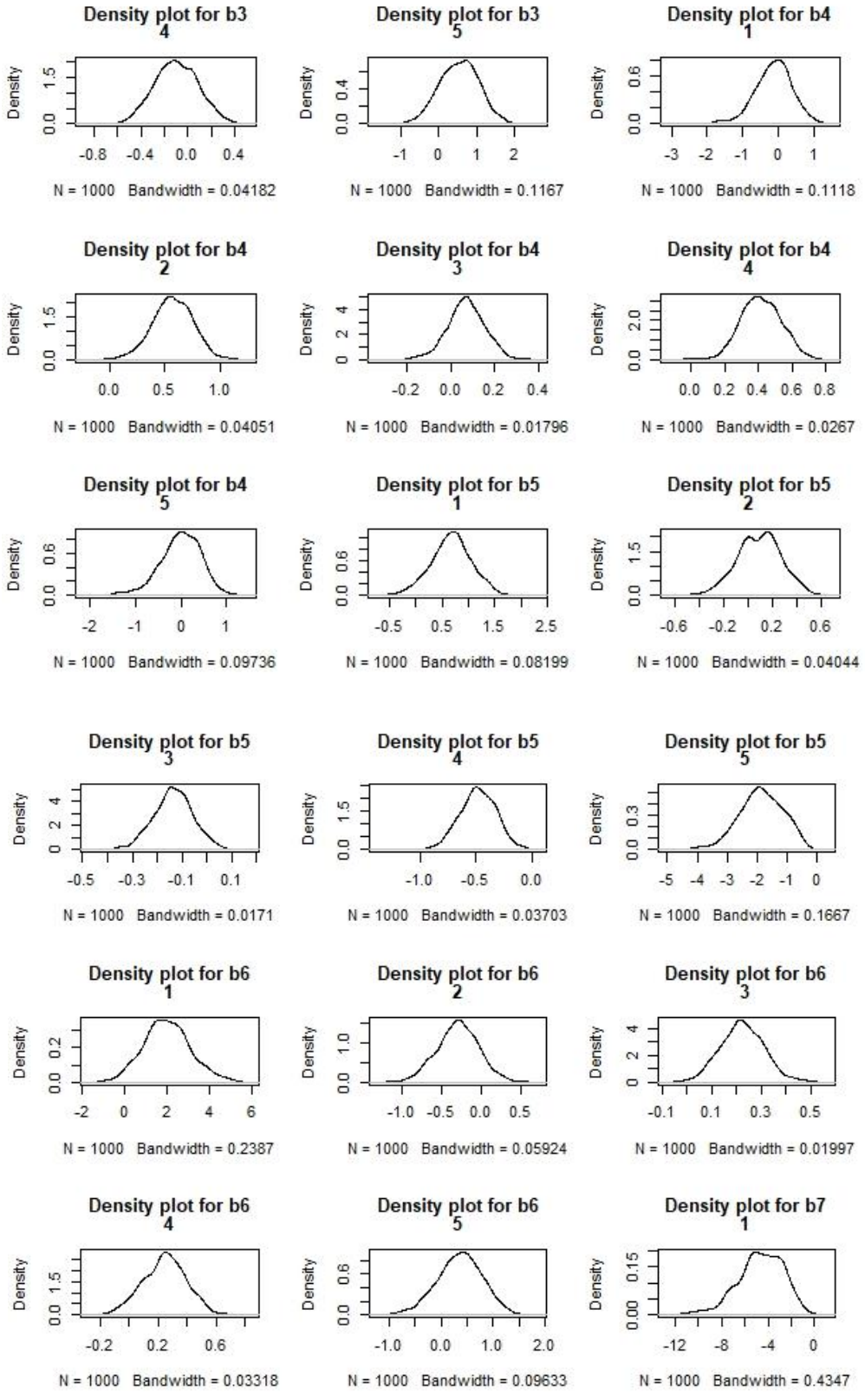

Ergodic Mean Plot for b0
3
Cumsum Values
0.78 0.86
0 200 600 1000
Index

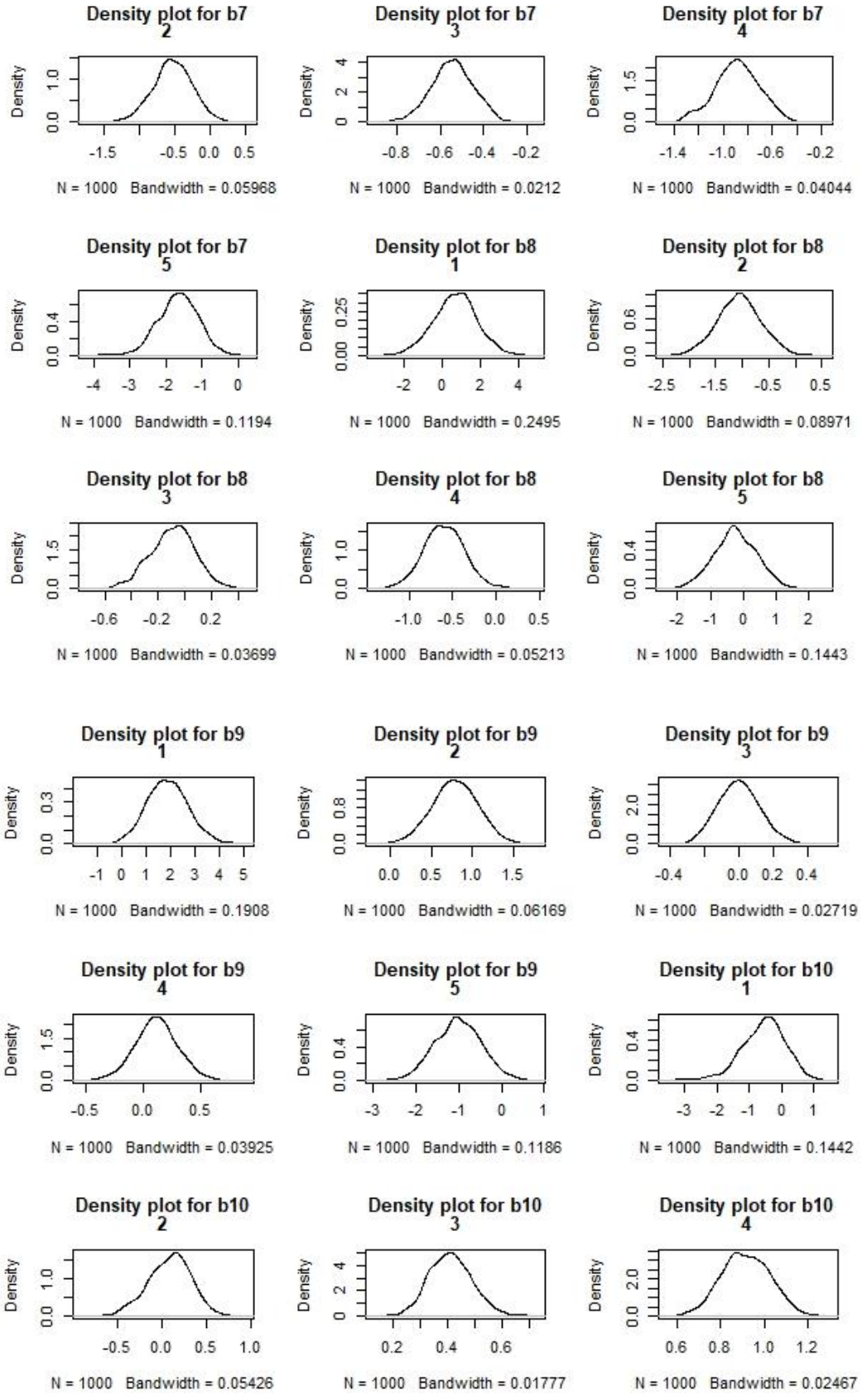

Ergodic Mean Plot for b0
4
Cumsum Values
1.30 1.45
0 200 600 1000
Index

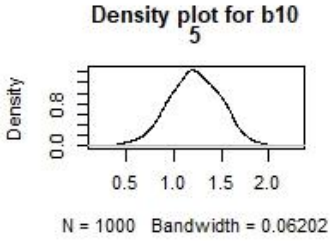

Ergodic Mean Plot for b0
5
Cumsum Values
2.0 2.4
0 200 600 1000
Index

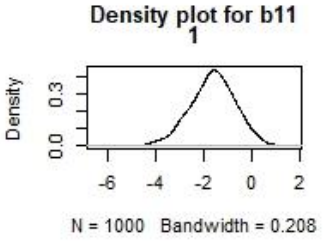

Ergodic Mean Plot for devianc
Cumsum Values
2974 2982
0 200 600 1000
Index

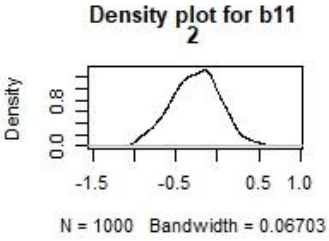

Density plot for b1 5
Density
N = 1000 Bandwidth = 0.159
Density plot for b2 1
Density
N = 1000 Bandwidth = 0.1153
Density plot for b2 2
Density
N = 1000 Bandwidth = 0.03536
Density plot for b2 3
Density
N = 1000 Bandwidth = 0.02061
Density plot for b2 4
Density
N = 1000 Bandwidth = 0.03622
Density plot for b2 5
Density
N = 1000 Bandwidth = 0.09635
Density plot for b3 1
Density
N = 1000 Bandwidth = 0.1925
Density plot for b3 2
Density
N = 1000 Bandwidth = 0.0592
Density plot for b3 3
Density
N = 1000 Bandwidth = 0.02511
Density plot for b0 1
Density
N = 1000 Bandwidth = 0.3841
Density plot for b0 2
Density
N = 1000 Bandwidth = 0.04816
Density plot for b0 3
Density
N = 1000 Bandwidth = 0.01525
Density plot for b0 4
Density
N = 1000 Bandwidth = 0.03451
Density plot for b0 5
Density
N = 1000 Bandwidth = 0.1697
Density plot for b1 1
Density
N = 1000 Bandwidth = 0.2772
Density plot for b1 2
Density
N = 1000 Bandwidth = 0.1004
Density plot for b1 3
Density
N = 1000 Bandwidth = 0.04178
Density plot for b1 4
Density
N = 1000 Bandwidth = 0.05553

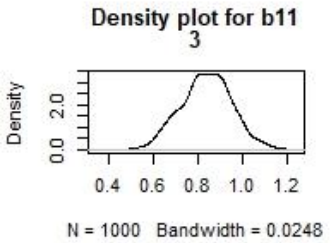

Density plot for b3 4
Density plot for b3 5
Density plot for b4 1
Density plot for b4 2
Density plot for b4 3
Density plot for b4 4
Density plot for b4 5
Density plot for b5 1
Density plot for b5 2
Density plot for b5 3
Density plot for b5 4
Density plot for b5 5
Density plot for b6 1
Density plot for b6 2
Density plot for b6 3
Density plot for b6 4
Density plot for b6 5
Density plot for b7 1
Density
N = 1000 Bandwidth = 0.04182
N = 1000 Bandwidth = 0.1167
N = 1000 Bandwidth = 0.1118
N = 1000 Bandwidth = 0.04051
N = 1000 Bandwidth = 0.01796
N = 1000 Bandwidth = 0.0267
N = 1000 Bandwidth = 0.09736
N = 1000 Bandwidth = 0.08199
N = 1000 Bandwidth = 0.04044
N = 1000 Bandwidth = 0.0171
N = 1000 Bandwidth = 0.03703
N = 1000 Bandwidth = 0.1667
N = 1000 Bandwidth = 0.2387
N = 1000 Bandwidth = 0.05924
N = 1000 Bandwidth = 0.01997
N = 1000 Bandwidth = 0.03318
N = 1000 Bandwidth = 0.09633
N = 1000 Bandwidth = 0.4347

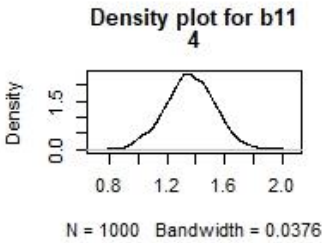

Density plot for b7 2
Density
N = 1000 Bandwidth = 0.05968
Density plot for b7 3
Density
N = 1000 Bandwidth = 0.0212
Density plot for b7 4
Density
N = 1000 Bandwidth = 0.04044
Density plot for b7 5
Density
N = 1000 Bandwidth = 0.1194
Density plot for b8 1
Density
N = 1000 Bandwidth = 0.2495
Density plot for b8 2
Density
N = 1000 Bandwidth = 0.08971
Density plot for b8 3
Density
N = 1000 Bandwidth = 0.03699
Density plot for b8 4
Density
N = 1000 Bandwidth = 0.05213
Density plot for b8 5
Density
N = 1000 Bandwidth = 0.1443
Density plot for b9 1
Density
N = 1000 Bandwidth = 0.1908
Density plot for b9 2
Density
N = 1000 Bandwidth = 0.06169
Density plot for b9 3
Density
N = 1000 Bandwidth = 0.02719
Density plot for b9 4
Density
N = 1000 Bandwidth = 0.03925
Density plot for b9 5
Density
N = 1000 Bandwidth = 0.1186
Density plot for b10 1
Density
N = 1000 Bandwidth = 0.1442
Density plot for b10 2
Density
N = 1000 Bandwidth = 0.05426
Density plot for b10 3
Density
N = 1000 Bandwidth = 0.01777
Density plot for b10 4
Density
N = 1000 Bandwidth = 0.02467

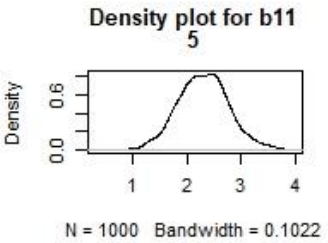

Density plot for b10
5
Density
N = 1000 Bandwidth = 0.06202

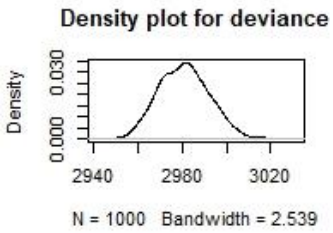

Density plot for b11
1
Density
N = 1000 Bandwidth = 0.208

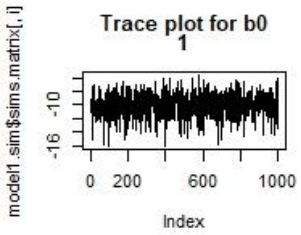

Density plot for b11
2
Density
N = 1000 Bandwidth = 0.06703

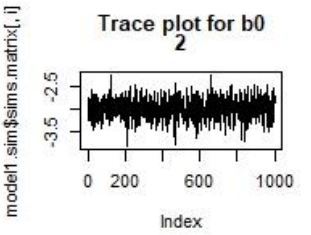

Density plot for b11
3
Density
N = 1000 Bandwidth = 0.0248

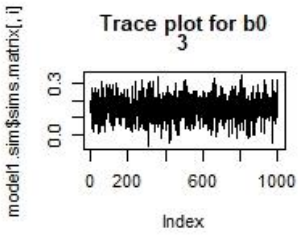

Density plot for b11
4
Density
N = 1000 Bandwidth = 0.0376

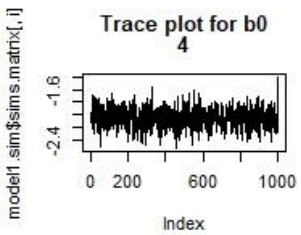

Density plot for b11
5
Density
N = 1000 Bandwidth = 0.1022

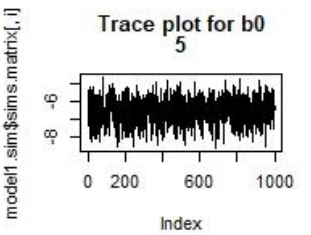

Density plot for deviance
Density
N = 1000 Bandwidth = 2.539

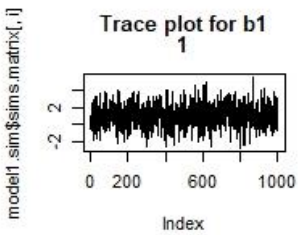

Trace plot for b0
1
model1.sim$sims.matrix[, i]
Index

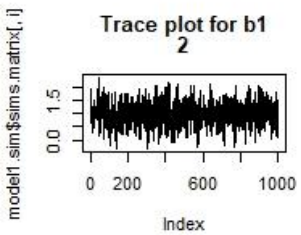

Trace plot for b0
2
model1.sim$sims.matrix[, i]
Index

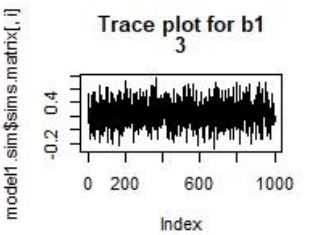

Trace plot for b0
3
model1.sim$sims.matrix[, i]
Index

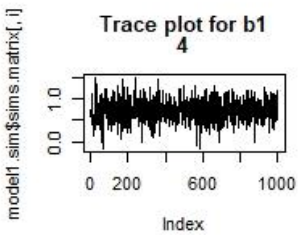

Trace plot for b0
4
model1.sim$sims.matrix[, i]
Index

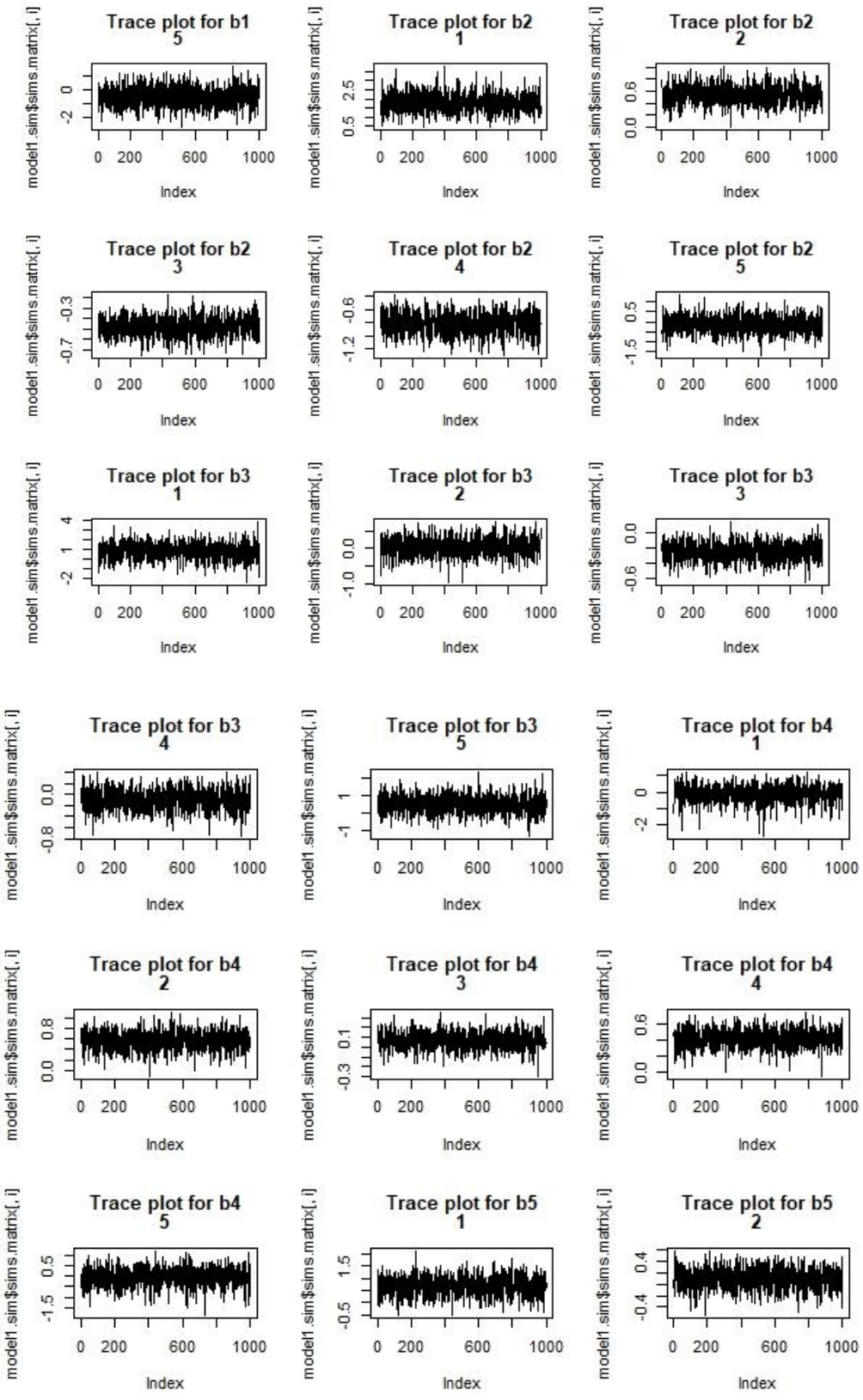

Trace plot for b0
5
model1.sim$sims.matrix[, i]
Index

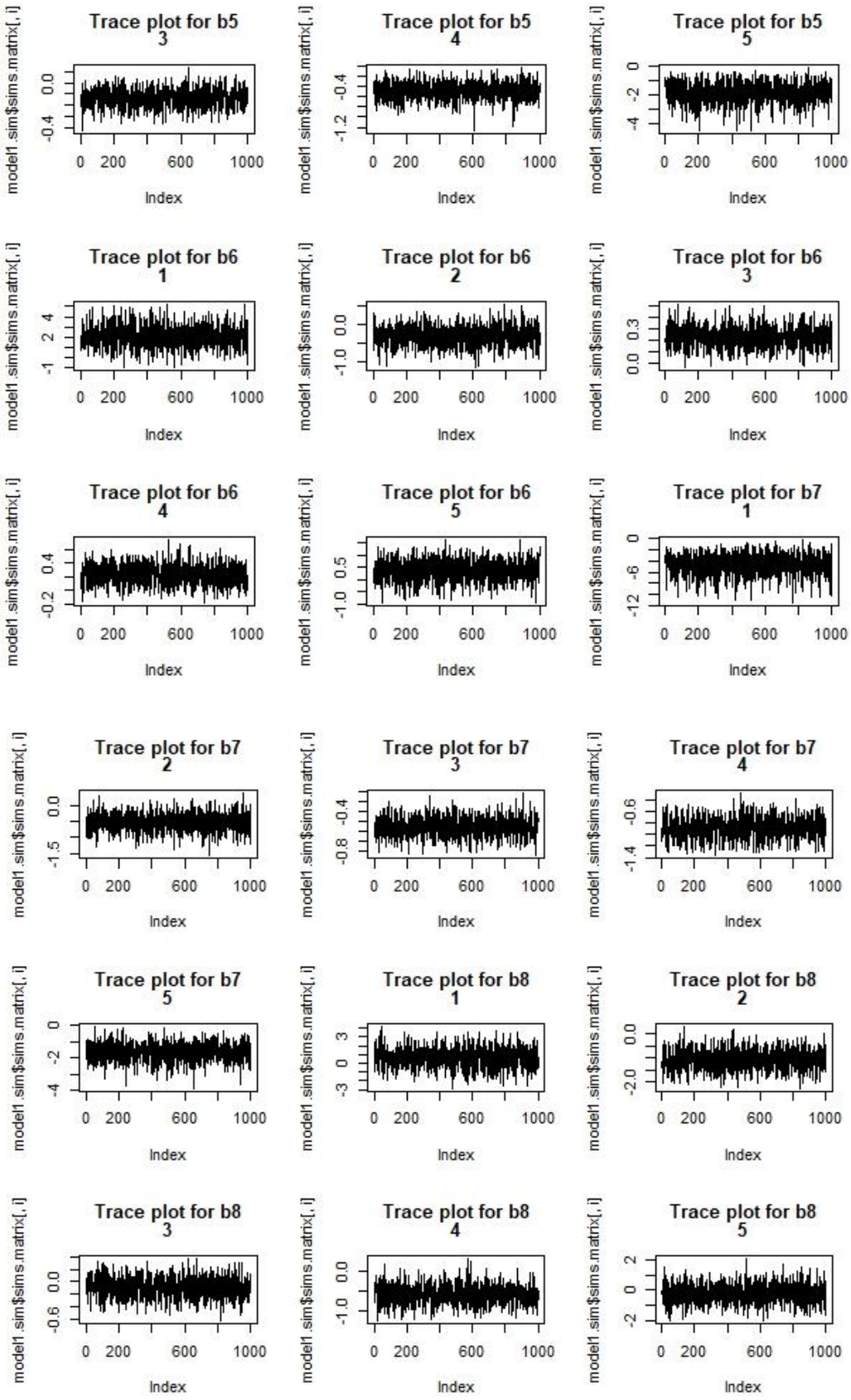

Trace plot for b1
1
model1.sim$sims.matrix[, i]
Index

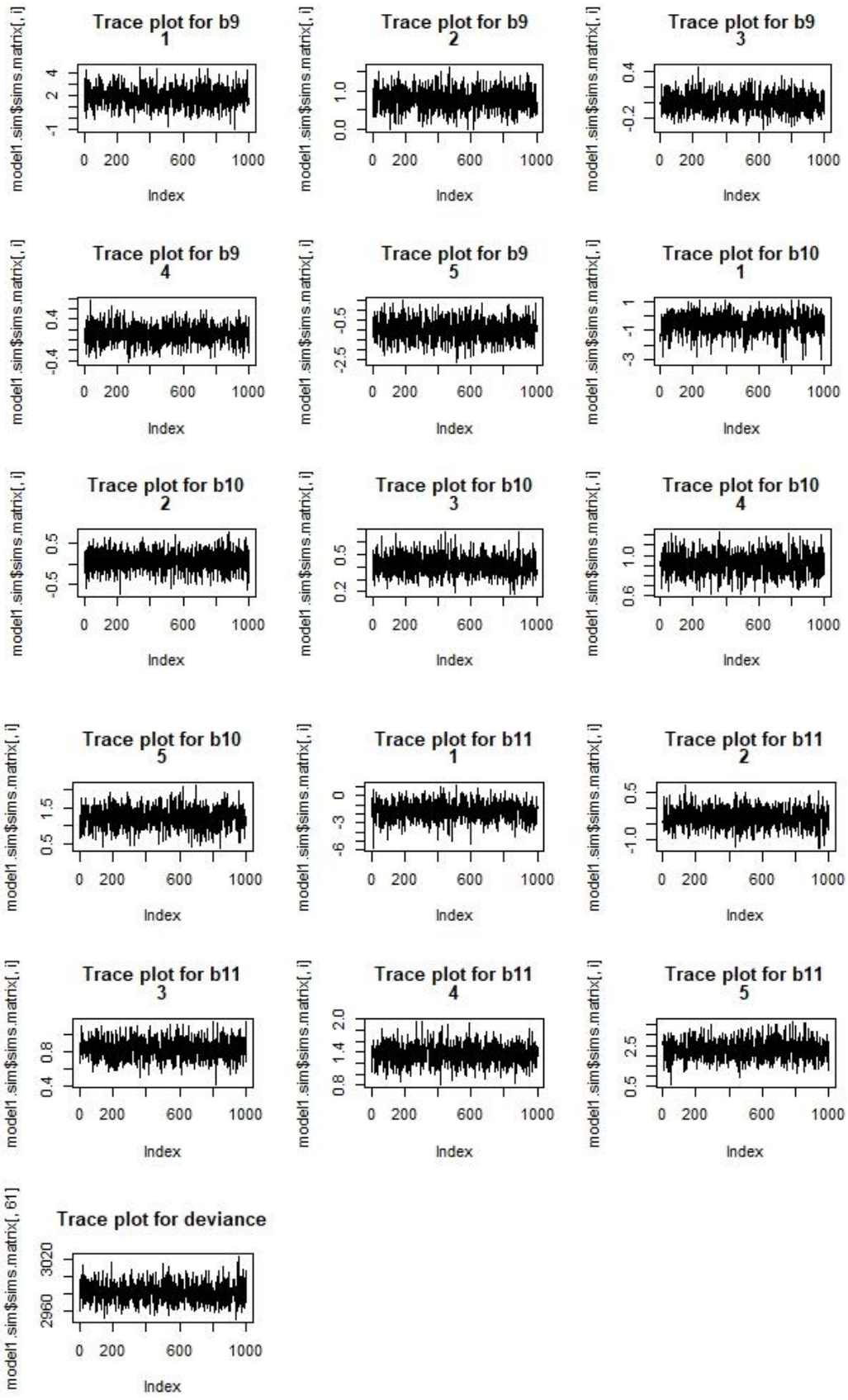

Trace plot for b1
2
model1.sim$sims.matrix[, i]
Index

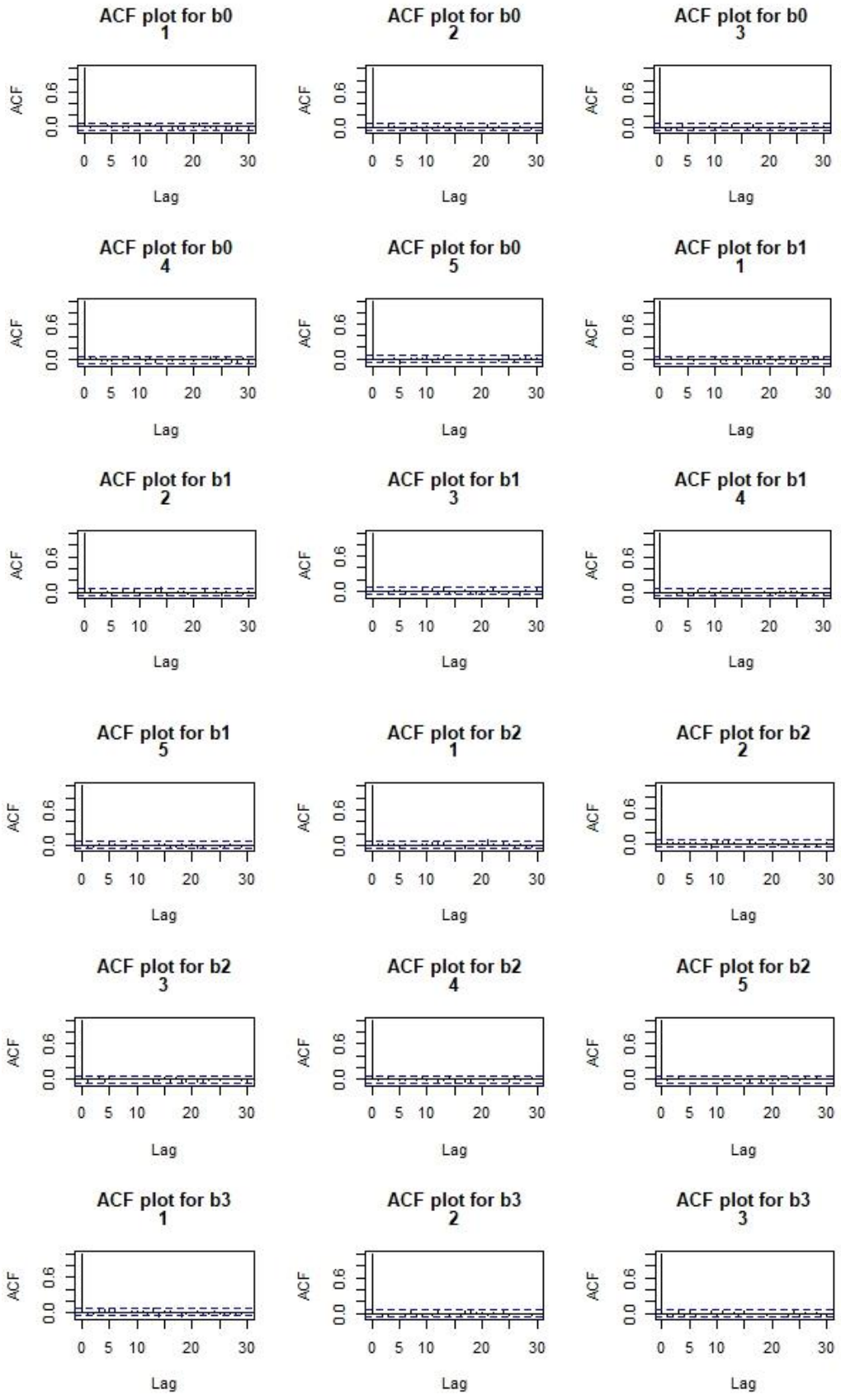

Trace plot for b1
3
model1.sim$sims.matrix[, i]
Index

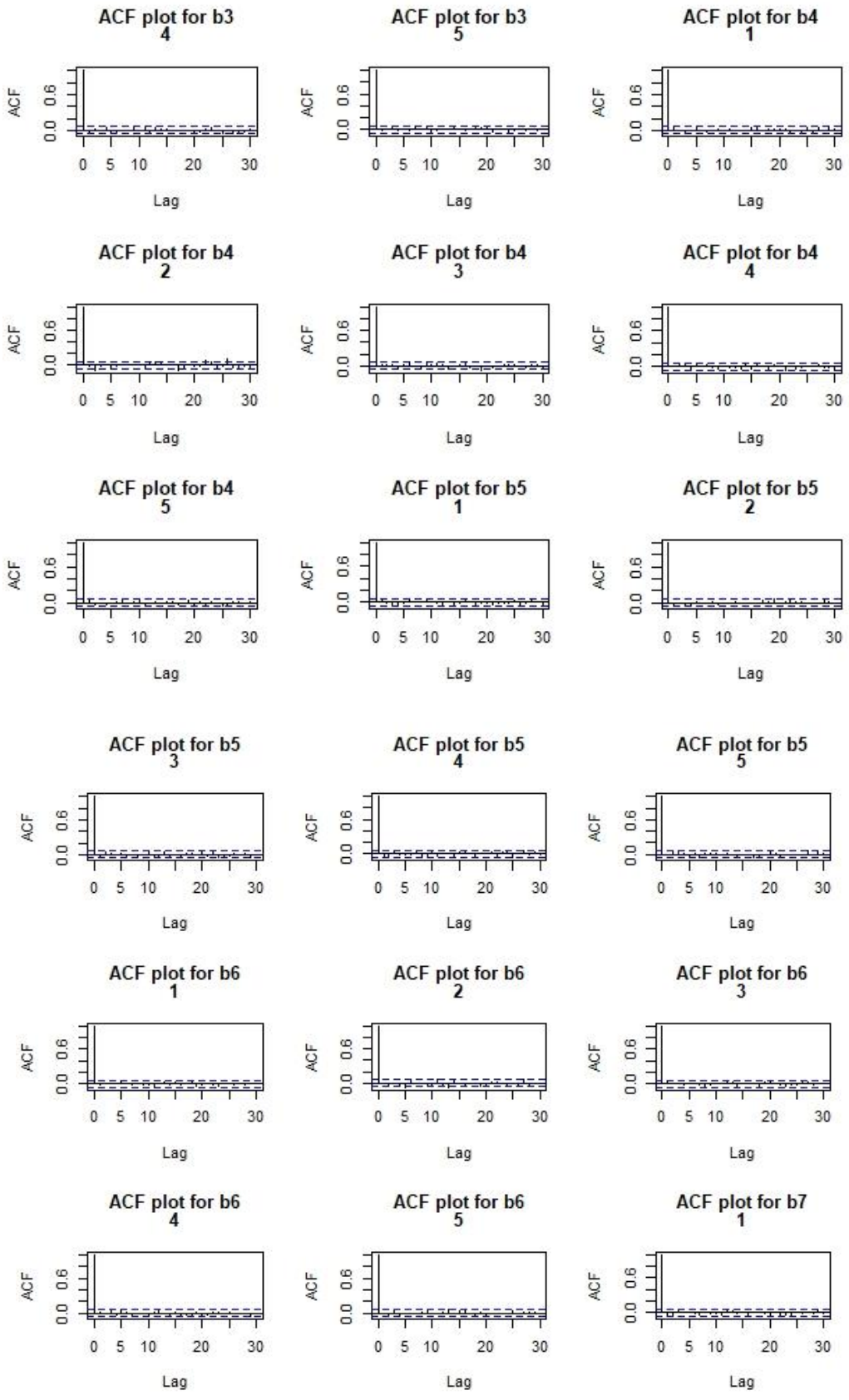

Trace plot for b1
4
model1.sim$sims.matrix[, i]
Index

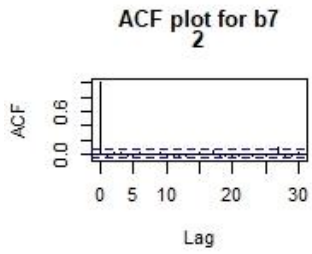

Trace plot for b1 5
Trace plot for b2 1
Trace plot for b2 2
Trace plot for b2 3
Trace plot for b2 4
Trace plot for b2 5
Trace plot for b3 1
Trace plot for b3 2
Trace plot for b3 3
Trace plot for b3 4
Trace plot for b3 5
Trace plot for b4 1
Trace plot for b4 2
Trace plot for b4 3
Trace plot for b4 4
Trace plot for b4 5
Trace plot for b5 1
Trace plot for b5 2
model1.sim$sims.matrix[, i]
Index

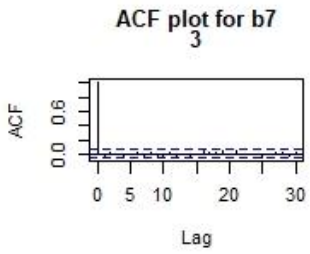

Trace plot for b5
3
Trace plot for b5
4
Trace plot for b5
5
Trace plot for b6
1
Trace plot for b6
2
Trace plot for b6
3
Trace plot for b6
4
Trace plot for b6
5
Trace plot for b7
1
Trace plot for b7
2
Trace plot for b7
3
Trace plot for b7
4
Trace plot for b7
5
Trace plot for b8
1
Trace plot for b8
2
Trace plot for b8
3
Trace plot for b8
4
Trace plot for b8
5
model1.sim$sims.matrix[, i]
Index
0
200
600
1000

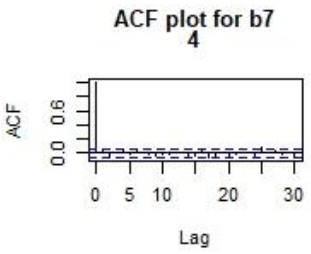

Trace plot for b9 1
Trace plot for b9 2
Trace plot for b9 3
Trace plot for b9 4
Trace plot for b9 5
Trace plot for b10 1
Trace plot for b10 2
Trace plot for b10 3
Trace plot for b10 4
Trace plot for b10 5
Trace plot for b11 1
Trace plot for b11 2
Trace plot for b11 3
Trace plot for b11 4
Trace plot for b11 5
Trace plot for deviance
model1.sim$sims.matrix[, i]
model1.sim$sims.matrix[, 61]
Index
0 200 600 1000

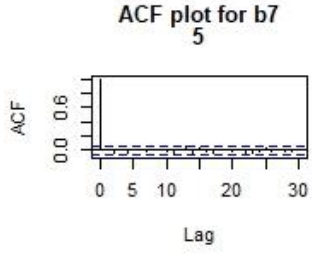

ACF plot for b0
1
ACF plot for b0
2
ACF plot for b0
3
ACF plot for b0
4
ACF plot for b0
5
ACF plot for b1
1
ACF plot for b1
2
ACF plot for b1
3
ACF plot for b1
4
ACF plot for b1
5
ACF plot for b2
1
ACF plot for b2
2
ACF plot for b2
3
ACF plot for b2
4
ACF plot for b2
5
ACF plot for b3
1
ACF plot for b3
2
ACF plot for b3
3
ACF
Lag
0.0
0.6
0
5
10
20
30

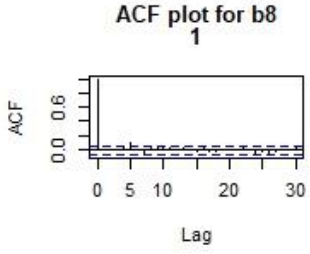

ACF plot for b3 4
ACF plot for b3 5
ACF plot for b4 1
ACF plot for b4 2
ACF plot for b4 3
ACF plot for b4 4
ACF plot for b4 5
ACF plot for b5 1
ACF plot for b5 2
ACF plot for b5 3
ACF plot for b5 4
ACF plot for b5 5
ACF plot for b6 1
ACF plot for b6 2
ACF plot for b6 3
ACF plot for b6 4
ACF plot for b6 5
ACF plot for b7 1
ACF
Lag
0.0
0.6
0
5
10
20
30

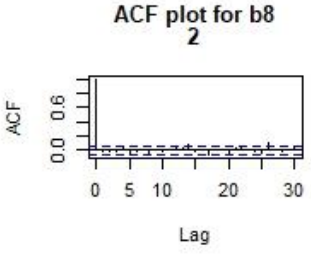

ACF plot for b7
2
ACF
0.0 0.6
0 5 10 20 30
Lag

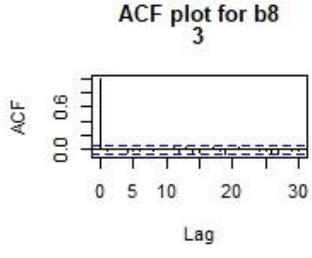

ACF plot for b7
3
ACF
0.0 0.6
0 5 10 20 30
Lag

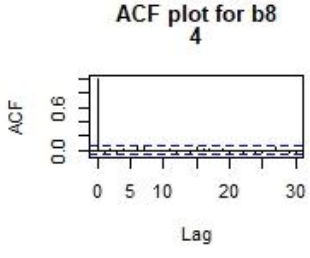

ACF plot for b7
4
ACF
0.0 0.6
0 5 10 20 30
Lag

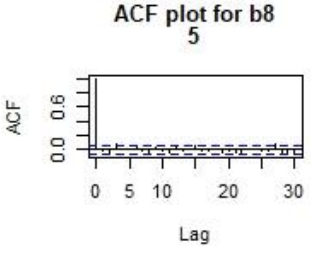

ACF plot for b7
5
ACF
0.0 0.6
0 5 10 20 30
Lag

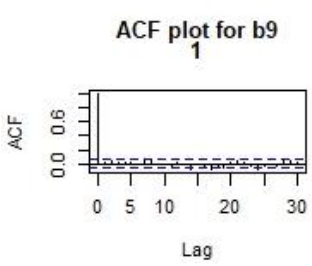

ACF plot for b8
1
ACF
0.0 0.6
0 5 10 20 30
Lag

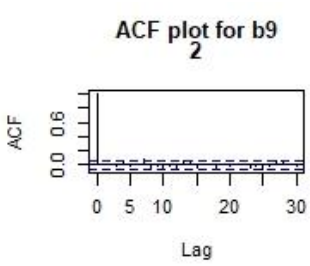

ACF plot for b8
2
ACF
0.0 0.6
0 5 10 20 30
Lag

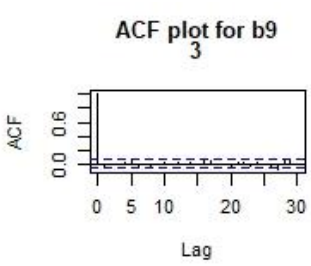

ACF plot for b8
3
ACF
0.0 0.6
0 5 10 20 30
Lag

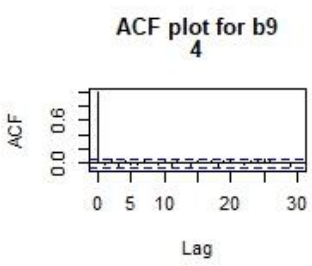

ACF plot for b8
4
ACF
0.0 0.6
0 5 10 20 30
Lag

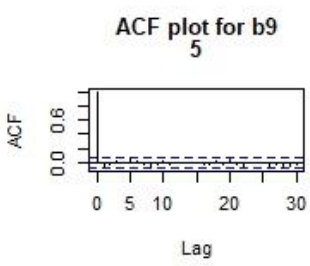

ACF plot for b8
5
ACF
0.0 0.6
0 5 10 20 30
Lag

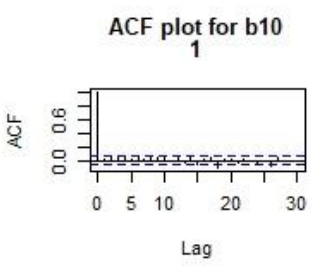

ACF plot for b9
1
ACF
0.0 0.6
0 5 10 20 30
Lag

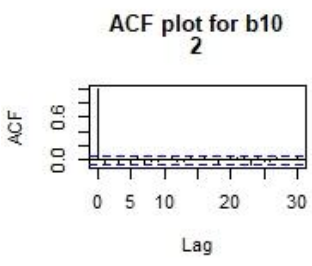

ACF plot for b9
2
ACF
0.0 0.6
0 5 10 20 30
Lag

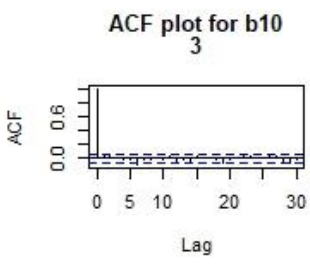

ACF plot for b9
3
ACF
0.0 0.6
0 5 10 20 30
Lag

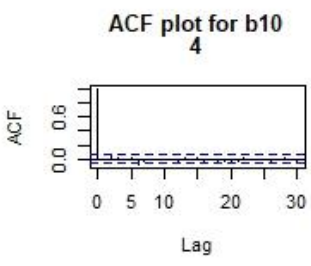

ACF plot for b9
4
ACF
0.0 0.6
0 5 10 20 30
Lag

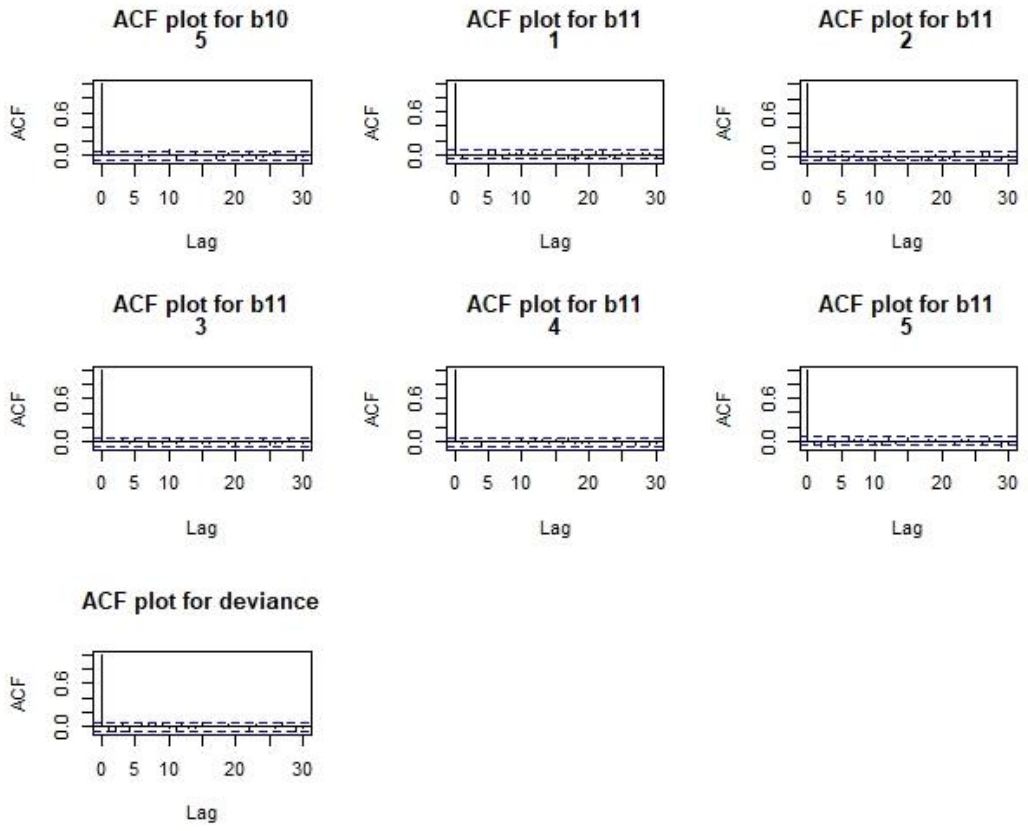

ACF plot for b9
5
ACF
0.0 0.6
0 5 10 20 30
Lag


ACF plot for b10
1
ACF
0.0 0.6
0 5 10 20 30
Lag


ACF plot for b10
2
ACF
0.0 0.6
0 5 10 20 30
Lag


ACF plot for b10
3
ACF
0.0 0.6
0 5 10 20 30
Lag


ACF plot for b10
4
ACF
0.0 0.6
0 5 10 20 30
Lag



All those graphs indicate convergence for our model. While some parameters behave better than others, their erratic behaviour can be fixed by adding more iterations, which should smooth out our final results.

Results of the initial run:

After the pilot run, which is described above, we get the following results:

```
Inference for Bugs model at "C:\Users\30697\Desktop\multinomial.txt",
fit using OpenBUGS,
 1 chains, each with 31000 iterations (first 1000 discarded), n.thin =
30
 n.sims = 1000 iterations saved
           mean   sd   2.5%    25%    50%    75%  97.5%
b0[1]     -10.0  1.7  -14.0  -11.1   -9.8   -8.8   -7.1
b0[2]      -3.0  0.2   -3.4   -3.1   -3.0   -2.9   -2.6
b0[3]       0.2  0.1    0.0    0.1    0.1    0.2    0.3
b0[4]      -2.0  0.2   -2.4   -2.1   -2.0   -1.9   -1.8
b0[5]      -6.6  0.8   -8.1   -7.1   -6.5   -6.0   -5.2
b1[1]       1.1  1.3   -1.5    0.3    1.1    1.9    3.6
b1[2]       1.0  0.5    0.1    0.7    1.0    1.3    1.9
b1[3]       0.2  0.2   -0.1    0.1    0.2    0.4    0.6
```

b1[4]	0.7	0.2	0.2	0.6	0.7	0.9	1.2
b1[5]	-0.5	0.7	-1.9	-1.0	-0.4	0.0	0.9
b2[1]	1.8	0.5	0.8	1.4	1.7	2.1	2.8
b2[2]	0.5	0.2	0.2	0.4	0.5	0.6	0.8
b2[3]	-0.5	0.1	-0.7	-0.5	-0.5	-0.4	-0.3
b2[4]	-0.8	0.2	-1.1	-0.9	-0.8	-0.7	-0.5
b2[5]	-0.1	0.4	-1.0	-0.4	-0.1	0.2	0.7
b3[1]	0.8	0.9	-1.0	0.2	0.8	1.4	2.3
b3[2]	0.0	0.3	-0.5	-0.1	0.0	0.2	0.6
b3[3]	-0.3	0.1	-0.5	-0.3	-0.3	-0.2	0.0
b3[4]	-0.1	0.2	-0.4	-0.2	-0.1	0.0	0.3
b3[5]	0.5	0.5	-0.5	0.2	0.6	0.9	1.5
b4[1]	-0.1	0.5	-1.3	-0.5	-0.1	0.2	0.8
b4[2]	0.6	0.2	0.2	0.5	0.6	0.7	0.9
b4[3]	0.1	0.1	-0.1	0.0	0.1	0.1	0.2
b4[4]	0.4	0.1	0.2	0.3	0.4	0.5	0.6
b4[5]	0.0	0.4	-1.0	-0.3	0.0	0.3	0.8
b5[1]	0.7	0.4	-0.1	0.4	0.7	0.9	1.4
b5[2]	0.1	0.2	-0.3	0.0	0.1	0.2	0.4
b5[3]	-0.1	0.1	-0.3	-0.2	-0.1	-0.1	0.0
b5[4]	-0.5	0.2	-0.8	-0.6	-0.5	-0.4	-0.2
b5[5]	-1.9	0.7	-3.4	-2.3	-1.9	-1.3	-0.6
b6[1]	2.0	1.1	0.0	1.3	2.0	2.7	4.3
b6[2]	-0.3	0.3	-0.8	-0.5	-0.3	-0.1	0.2
b6[3]	0.2	0.1	0.1	0.2	0.2	0.3	0.4
b6[4]	0.2	0.1	0.0	0.1	0.2	0.3	0.5
b6[5]	0.4	0.4	-0.6	0.1	0.4	0.7	1.2
b7[1]	-4.6	1.9	-8.9	-5.8	-4.5	-3.1	-1.4
b7[2]	-0.5	0.3	-1.1	-0.7	-0.5	-0.3	0.0
b7[3]	-0.5	0.1	-0.7	-0.6	-0.5	-0.5	-0.4
b7[4]	-0.9	0.2	-1.3	-1.0	-0.9	-0.8	-0.5
b7[5]	-1.7	0.5	-2.7	-2.0	-1.6	-1.3	-0.6
b8[1]	0.7	1.1	-1.6	-0.1	0.7	1.4	2.8
b8[2]	-1.1	0.4	-1.9	-1.3	-1.1	-0.8	-0.3
b8[3]	-0.1	0.2	-0.4	-0.2	-0.1	0.0	0.2
b8[4]	-0.6	0.2	-1.0	-0.8	-0.6	-0.4	-0.1
b8[5]	-0.2	0.6	-1.5	-0.7	-0.3	0.2	1.0
b9[1]	1.9	0.8	0.2	1.3	1.8	2.4	3.6
b9[2]	0.8	0.3	0.2	0.6	0.8	1.0	1.3
b9[3]	0.0	0.1	-0.2	-0.1	0.0	0.1	0.3
b9[4]	0.1	0.2	-0.2	0.0	0.1	0.2	0.5
b9[5]	-1.0	0.5	-2.0	-1.4	-1.0	-0.7	0.0
b10[1]	-0.6	0.7	-2.1	-1.0	-0.5	-0.1	0.6
b10[2]	0.1	0.2	-0.4	-0.1	0.1	0.3	0.5
b10[3]	0.4	0.1	0.3	0.4	0.4	0.5	0.6
b10[4]	0.9	0.1	0.7	0.8	0.9	1.0	1.1
b10[5]	1.2	0.3	0.7	1.0	1.2	1.4	1.7
b11[1]	-1.6	1.0	-3.8	-2.2	-1.6	-1.0	0.2

```
b11[2]     -0.3  0.3   -0.9   -0.5   -0.3   -0.1    0.3
b11[3]      0.8  0.1    0.6    0.8    0.8    0.9    1.1
b11[4]      1.4  0.2    1.0    1.3    1.4    1.5    1.7
b11[5]      2.3  0.5    1.4    2.0    2.3    2.6    3.2
deviance 2981.0 11.2 2960.5 2972.6 2980.7 2988.6 3003.6

DIC info (using the rule, pD = Dbar-Dhat)
pD = 55.2 and DIC = 3036.0
DIC is an estimate of expected predictive error (lower deviance is
better).
```

We can discern that with the exception of the 7th estimate for the β of total sulfur dioxide, which affected all the levels negatively, the other estimates either had a positive or a negative effect on each level. It could mean that if the levels that had a positive estimated value, then they had the correct dosage of a physiochemical characteristic, which worked well when it was combined with all the other components, while if they had a negative value, then that particular physiochemical deviant was not playing well with the others.

According to our results, our full model is:

$$ln\left(\frac{p_{i1}}{p_{i6}}\right) = -10 + 1.1 \cdot Z_{i1} + 1.8 \cdot Z_{i2} + \cdots - 1.6 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i2}}{p_{i6}}\right) = -3 + 1 \cdot Z_{i1} + 0.5 \cdot Z_{i2} + \cdots - 0.3 \cdot Z_{i11}$$

$$\vdots$$

$$ln\left(\frac{p_{i5}}{p_{i6}}\right) = -6.6 - 0.5 \cdot Z_{i1} - 0.1 \cdot Z_{i2} + \cdots + 2.3 \cdot Z_{i11}$$

where Z_{ip} are our centered values of X_{ip}, meaning that:

$$Z_{ip} = \frac{X_{ip} - \bar{X}_p}{S_p}$$

where $\bar{X}_p$ is the mean value of the physiochemical characteristic p (the whole column of 1599 observations), while S_p is the standard deviation of the physiochemical characteristic p.

Variable Selection:

Now we are getting to the good part. The aim of this study is after all, to take the model above and select only the most significant physiochemical characteristics of those that we have data on. We will do it, by using the following methods:

Variable Selection through BIC on prior coefficients (using BAS package in R):

Just as it says, this method uses the Bayes Information Criterion (BIC) to pick and choose the most important variables in our model. This procedure will be done for centered values. We can run our code by using a bunch of prior distributions on the model. In our problem we will do it by running the following two:

1. Using a Uniform Distribution on the priors:

Which is as if we are specifying that we want to choose different models with the same probability which will be imposed on their dimensions.

Since there is no way to implement the BAS package for the multinomial logistic regression as it is, we separate our problem into 5 binomial problems (the number of levels that we have) in our code. And if we run the code for it, then we shall get the following results:

For level 1 (category 1 with reference level 6):

```
 Marginal Posterior Summaries of Coefficients:

 Using  BMA

 Based on the top  234 models
                      post mean  post SD    post p(B != 0)
Intercept             -5.528401   0.764418   1.000000
fixed.acidity          0.022590   0.156528   0.055957
volatile.acidity       1.068056   0.289777   0.995752
citric.acid            0.042818   0.213934   0.069580
residual.sugar         0.006080   0.078205   0.039355
chlorides              0.055310   0.156334   0.140918
free.sulfur.dioxide    0.057447   0.363146   0.092334
total.sulfur.dioxide  -0.821427   0.943397   0.595020
density                0.132811   0.341706   0.168555
pH                     0.025142   0.156214   0.059082
sulphates              0.002636   0.102711   0.041406
alcohol               -0.102082   0.400745   0.096729
> summary(res1.1)
                     P(B != 0 | Y)     model 1   model 2     model 3
Intercept               1.00000000   1.0000000   1.00000   1.0000000
fixed.acidity           0.05595703   0.0000000   0.00000   0.0000000
volatile.acidity        0.99575195   1.0000000   1.00000   1.0000000
citric.acid             0.06958008   0.0000000   0.00000   0.0000000
residual.sugar          0.03935547   0.0000000   0.00000   0.0000000
chlorides               0.14091797   0.0000000   0.00000   0.0000000
free.sulfur.dioxide     0.09233398   0.0000000   0.00000   0.0000000
total.sulfur.dioxide    0.59501953   1.0000000   0.00000   1.0000000
density                 0.16855469   0.0000000   0.00000   1.0000000
pH                      0.05908203   0.0000000   0.00000   0.0000000
```

```
sulphates               0.04140625   0.0000000   0.00000   0.0000000
alcohol                 0.09672852   0.0000000   0.00000   0.0000000
BF                              NA   0.9975965   1.00000   0.2595098
PostProbs                       NA   0.2343000   0.23200   0.0568000
R2                              NA   0.2611000   0.19870   0.2979000
dim                             NA   3.0000000   2.00000   4.0000000
logmarg                         NA -45.1674217 -45.16502 -46.5139760
                         model 4     model 5
Intercept              1.0000000   1.0000000
fixed.acidity          0.0000000   0.0000000
volatile.acidity       1.0000000   1.0000000
citric.acid            0.0000000   0.0000000
residual.sugar         0.0000000   0.0000000
chlorides              1.0000000   0.0000000
free.sulfur.dioxide    0.0000000   0.0000000
total.sulfur.dioxide   1.0000000   1.0000000
density                0.0000000   0.0000000
pH                     0.0000000   0.0000000
sulphates              0.0000000   0.0000000
alcohol                0.0000000   1.0000000
BF                     0.1765086   0.1445988
PostProbs              0.0442000   0.0291000
R2                     0.2905000   0.2867000
dim                    4.0000000   4.0000000
logmarg              -46.8994009 -47.0988074
```

For level 2 (category 2 with reference level 6):

```
 Marginal Posterior Summaries of Coefficients:

 Using  BMA

 Based on the top  263 models
                      post mean  post SD    post p(B != 0)
Intercept             -2.889649   0.203378   1.000000
fixed.acidity         -0.006037   0.064136   0.051025
volatile.acidity       0.575179   0.150088   0.992920
citric.acid            0.005414   0.058890   0.048926
residual.sugar         0.044970   0.116062   0.164014
chlorides              0.002428   0.030505   0.039404
free.sulfur.dioxide   -0.030289   0.132021   0.089160
total.sulfur.dioxide  -0.560638   0.243236   0.920166
density               -0.025939   0.106533   0.087939
pH                     0.062857   0.145760   0.199561
sulphates              0.003964   0.039327   0.044189
alcohol                0.077925   0.177308   0.200439
> summary(res1.2)
                     P(B != 0 | Y)   model 1      model 2      model 3
Intercept               1.00000000    1.0000    1.0000000    1.0000000
```

```
fixed.acidity           0.05102539    0.0000    0.0000000    0.0000000
volatile.acidity        0.99291992    1.0000    1.0000000    1.0000000
citric.acid             0.04892578    0.0000    0.0000000    0.0000000
residual.sugar          0.16401367    0.0000    0.0000000    0.0000000
chlorides               0.03940430    0.0000    0.0000000    0.0000000
free.sulfur.dioxide     0.08916016    0.0000    0.0000000    0.0000000
total.sulfur.dioxide    0.92016602    1.0000    1.0000000    1.0000000
density                 0.08793945    0.0000    0.0000000    0.0000000
pH                      0.19956055    0.0000    0.0000000    1.0000000
sulphates               0.04418945    0.0000    0.0000000    0.0000000
alcohol                 0.20043945    0.0000    1.0000000    0.0000000
BF                              NA    1.0000    0.3167589    0.2574623
PostProbs                       NA    0.3694    0.1068000    0.0936000
R2                              NA    0.0961    0.1074000    0.1063000
dim                             NA    3.0000    4.0000000    4.0000000
logmarg                         NA -178.6481 -179.7977127 -180.0049804
                          model 4      model 5
Intercept               1.0000000    1.0000000
fixed.acidity           0.0000000    0.0000000
volatile.acidity        1.0000000    1.0000000
citric.acid             0.0000000    0.0000000
residual.sugar          1.0000000    0.0000000
chlorides               0.0000000    0.0000000
free.sulfur.dioxide     0.0000000    0.0000000
total.sulfur.dioxide    1.0000000    1.0000000
density                 0.0000000    1.0000000
pH                      0.0000000    0.0000000
sulphates               0.0000000    0.0000000
alcohol                 0.0000000    0.0000000
BF                      0.1789942    0.0885787
PostProbs               0.0591000    0.0282000
R2                      0.1044000    0.1007000
dim                     4.0000000    4.0000000
logmarg              -180.3685001 -181.0719622
```

For level 3 (category 3 with reference level 6):

```
 Marginal Posterior Summaries of Coefficients:

 Using  BMA

 Based on the top  75 models
                     post mean   post SD     post p(B != 0)
Intercept             0.1442064   0.0659399   1.0000000
fixed.acidity         0.0061386   0.0374363   0.0517578
volatile.acidity     -0.4072339   0.0867200   1.0000000
citric.acid          -0.0337026   0.0871079   0.1713867
residual.sugar        0.0016025   0.0149511   0.0333496
```

```
chlorides             -0.0413485   0.0774618   0.2664551
free.sulfur.dioxide    0.2243361   0.1368383   0.8024414
total.sulfur.dioxide  -0.5984030   0.1281603   0.9992188
density                0.0014770   0.0186418   0.0305176
pH                    -0.0002957   0.0143161   0.0306152
sulphates              0.3505989   0.0760334   0.9984375
alcohol                0.8283591   0.0850498   0.9998047
> summary(res1.3)
                    P(B != 0 | Y)   model 1      model 2     model 3
Intercept              1.00000000    1.0000    1.0000000    1.000000
fixed.acidity          0.05175781    0.0000    0.0000000    0.000000
volatile.acidity       1.00000000    1.0000    1.0000000    1.000000
citric.acid            0.17138672    0.0000    0.0000000    0.000000
residual.sugar         0.03334961    0.0000    0.0000000    0.000000
chlorides              0.26645508    0.0000    1.0000000    0.000000
free.sulfur.dioxide    0.80244141    1.0000    1.0000000    0.000000
total.sulfur.dioxide   0.99921875    1.0000    1.0000000    1.000000
density                0.03051758    0.0000    0.0000000    0.000000
pH                     0.03061523    0.0000    0.0000000    0.000000
sulphates              0.99843750    1.0000    1.0000000    1.000000
alcohol                0.99980469    1.0000    1.0000000    1.000000
BF                             NA    1.0000    0.4045382    0.176709
PostProbs                      NA    0.4576    0.1772000    0.068900
R2                             NA    0.1784    0.1814000    0.172600
dim                            NA    6.0000    7.0000000    5.000000
logmarg                        NA -768.5041 -769.4090773 -770.237319
                       model 4      model 5
Intercept             1.0000000    1.0000000
fixed.acidity         0.0000000    0.0000000
volatile.acidity      1.0000000    1.0000000
citric.acid           1.0000000    1.0000000
residual.sugar        0.0000000    0.0000000
chlorides             0.0000000    0.0000000
free.sulfur.dioxide   1.0000000    0.0000000
total.sulfur.dioxide  1.0000000    1.0000000
density               0.0000000    0.0000000
pH                    0.0000000    0.0000000
sulphates             1.0000000    1.0000000
alcohol               1.0000000    1.0000000
BF                    0.1416244    0.1080856
PostProbs             0.0528000    0.0435000
R2                    0.1802000    0.1760000
dim                   7.0000000    6.0000000
logmarg            -770.4586449 -770.7289000
```

For level 4 (category 4 with reference level 6):

```
 Marginal Posterior Summaries of Coefficients:
```

```
 Using  BMA

 Based on the top  110 models
                     post mean  post SD    post p(B != 0)
Intercept             -1.937507   0.156390   1.000000
fixed.acidity          0.061361   0.136611   0.228418
volatile.acidity      -0.981452   0.164900   0.998682
citric.acid           -0.005383   0.079777   0.056348
residual.sugar         0.016653   0.066310   0.085840
chlorides             -0.122066   0.191919   0.362207
free.sulfur.dioxide    0.116267   0.209230   0.286084
total.sulfur.dioxide  -0.589230   0.222370   0.991211
density               -0.008337   0.085674   0.058789
pH                    -0.028556   0.092393   0.126807
sulphates              0.644060   0.132453   0.997705
alcohol                1.750716   0.170207   0.999121
> summary(res1.4)
                    P(B != 0 | Y)   model 1      model 2      model 3
Intercept              1.00000000    1.0000    1.0000000    1.0000000
fixed.acidity          0.22841797    0.0000    0.0000000    0.0000000
volatile.acidity       0.99868164    1.0000    1.0000000    1.0000000
citric.acid            0.05634766    0.0000    0.0000000    0.0000000
residual.sugar         0.08583984    0.0000    0.0000000    0.0000000
chlorides              0.36220703    0.0000    1.0000000    0.0000000
free.sulfur.dioxide    0.28608398    0.0000    0.0000000    1.0000000
total.sulfur.dioxide   0.99121094    1.0000    1.0000000    1.0000000
density                0.05878906    0.0000    0.0000000    0.0000000
pH                     0.12680664    0.0000    0.0000000    0.0000000
sulphates              0.99770508    1.0000    1.0000000    1.0000000
alcohol                0.99912109    1.0000    1.0000000    1.0000000
BF                             NA    1.0000    0.5284735    0.3574823
PostProbs                      NA    0.2460    0.1499000    0.0959000
R2                             NA    0.5192    0.5250000    0.5242000
dim                            NA    5.0000    6.0000000    6.0000000
logmarg                        NA -239.7406 -240.3783653 -240.7692722
                      model 4      model 5
Intercept           1.0000000    1.0000000
fixed.acidity       1.0000000    1.0000000
volatile.acidity    1.0000000    1.0000000
citric.acid         0.0000000    0.0000000
residual.sugar      0.0000000    0.0000000
chlorides           0.0000000    1.0000000
free.sulfur.dioxide 0.0000000    0.0000000
total.sulfur.dioxide 1.0000000   1.0000000
density             0.0000000    0.0000000
pH                  0.0000000    0.0000000
sulphates           1.0000000    1.0000000
```

```
alcohol                  1.0000000    1.0000000
BF                       0.2508113    0.1633278
PostProbs                0.0718000    0.0455000
R2                       0.5235000    0.5298000
dim                      6.0000000    7.0000000
logmarg               -241.1236571 -241.5525986
```

For level 5 (category 5 with reference level 6):

```
 Marginal Posterior Summaries of Coefficients:

 Using  BMA

 Based on the top  401 models
                      post mean  post SD   post p(B != 0)
Intercept              -6.90153    1.48388   1.00000
fixed.acidity          -0.07906    0.37641   0.10449
volatile.acidity       -0.41466    0.65142   0.35020
citric.acid             1.00256    0.99446   0.60117
residual.sugar         -1.17122    1.11341   0.61919
chlorides              -1.19416    1.49193   0.52090
free.sulfur.dioxide     0.02615    0.22948   0.07310
total.sulfur.dioxide   -1.58583    0.86360   0.88843
density                -0.32705    0.74951   0.22988
pH                     -0.23728    0.51044   0.23784
sulphates               1.26924    0.54658   0.92886
alcohol                 2.48258    0.87481   0.99849
> summary(res1.5)
                     P(B != 0 | Y)   model 1    model 2     model 3
Intercept                1.0000000   1.00000   1.000000   1.0000000
fixed.acidity            0.1044922   0.00000   0.000000   0.0000000
volatile.acidity         0.3501953   0.00000   0.000000   1.0000000
citric.acid              0.6011719   1.00000   1.000000   0.0000000
residual.sugar           0.6191895   1.00000   0.000000   1.0000000
chlorides                0.5208984   1.00000   1.000000   0.0000000
free.sulfur.dioxide      0.0730957   0.00000   0.000000   0.0000000
total.sulfur.dioxide     0.8884277   1.00000   1.000000   1.0000000
density                  0.2298828   0.00000   1.000000   0.0000000
pH                       0.2378418   0.00000   0.000000   0.0000000
sulphates                0.9288574   1.00000   1.000000   1.0000000
alcohol                  0.9984863   1.00000   1.000000   1.0000000
BF                             NA    1.00000   0.755376   0.8293052
PostProbs                      NA    0.12020   0.098900   0.0944000
R2                             NA    0.66940   0.666000   0.6280000
dim                            NA    7.00000   7.000000   6.0000000
logmarg                        NA -47.30155 -47.582095 -47.4887219
                         model 4     model 5
Intercept              1.0000000   1.0000000
```

```
fixed.acidity          0.0000000   0.0000000
volatile.acidity       0.0000000   1.0000000
citric.acid            1.0000000   0.0000000
residual.sugar         1.0000000   1.0000000
chlorides              0.0000000   0.0000000
free.sulfur.dioxide    0.0000000   0.0000000
total.sulfur.dioxide   1.0000000   1.0000000
density                0.0000000   0.0000000
pH                     0.0000000   1.0000000
sulphates              1.0000000   1.0000000
alcohol                1.0000000   1.0000000
BF                     0.8464082   0.3194614
PostProbs              0.0886000   0.0386000
R2                     0.6282000   0.6557000
dim                    6.0000000   7.0000000
logmarg              -47.4683084 -48.4426737
```

The summaries above show us the top 5 models for each of the categories, while the coefficients are for the MAP, (or in other words HMP) models. We will choose the best model based on the Bayes Factor (BF). Therefore, if it is 1, then this is the best model. Thus, for instance, for the first table, we saw that model 2 was the best, with a very close second alternative from model 1. According to the results above, we can see that our model is of the following form:

$$ln\left(\frac{p_{i1}}{p_{i6}}\right) = -5.53 + 1.07 \cdot Z_{i2}$$

$$ln\left(\frac{p_{i2}}{p_{i6}}\right) = -2.89 + 0.58 \cdot Z_{i2} - 0.56 \cdot Z_{i7}$$

$$ln\left(\frac{p_{i3}}{p_{i6}}\right) = 0.14 - 0.41 \cdot Z_{i2} + 0.22 \cdot Z_{i6} - 0.6 \cdot Z_{i7} + 0.35 \cdot Z_{i10} + 0.83 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i4}}{p_{i6}}\right) = -1.94 - 0.98 \cdot Z_{i2} - 0.6 \cdot Z_{i7} + 0.64 \cdot Z_{i10} + 1.75 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i5}}{p_{i6}}\right) = -6.9 + 1 \cdot Z_{i3} - 1.17 \cdot Z_{i4} - 1.19 \cdot Z_{i5} - 1.6 \cdot Z_{i7} + 1.3 \cdot Z_{i10} + 2.48 \cdot Z_{i11}$$

We can see that the physiochemical characteristics of *Fixed Acidity*, *pH* and *Density* were not considered useful at all, while others were used to a degree, depending on the level on which they were.

2. Using a Beta-Binomial Distribution on the priors:

Same as before, only this time instead of imposing a *Uniform* distribution on the priors, we shall try the *Beta-Binomial*. Here we will choose the hyperparameter *a* to be smaller than the hyperparameter *b*, thus stating that we would prefer a more parsimonious model in the end. Based on our code, we get the following results:

For level 1 (category 1 with reference level 6):

```
 Marginal Posterior Summaries of Coefficients:

 Using  BMA

 Based on the top  67 models
                       post mean   post SD     post p(B != 0)
Intercept              -5.259e+00   5.402e-01   1.000e+00
fixed.acidity           5.495e-03   7.699e-02   1.099e-02
volatile.acidity        1.060e+00   2.660e-01   9.886e-01
citric.acid             1.628e-03   5.875e-02   7.715e-03
residual.sugar          9.681e-04   2.973e-02   7.373e-03
chlorides               5.733e-03   5.071e-02   1.675e-02
free.sulfur.dioxide    -3.446e-03   9.055e-02   1.255e-02
total.sulfur.dioxide   -1.524e-01   4.575e-01   1.291e-01
density                 1.011e-02   9.218e-02   1.636e-02
pH                      4.531e-03   6.775e-02   9.375e-03
sulphates               7.002e-05   2.980e-02   4.297e-03
alcohol                -7.579e-03   1.075e-01   1.113e-02
> summary(res1.1)
                     P(B != 0 | Y)   model 1     model 2     model 3
Intercept              1.000000000   1.00000   1.0000000   1.0000000
fixed.acidity          0.010986328   0.00000   0.0000000   0.0000000
volatile.acidity       0.988574219   1.00000   1.0000000   1.0000000
citric.acid            0.007714844   0.00000   0.0000000   0.0000000
residual.sugar         0.007373047   0.00000   0.0000000   0.0000000
chlorides              0.016748047   0.00000   0.0000000   0.0000000
free.sulfur.dioxide    0.012548828   0.00000   0.0000000   0.0000000
total.sulfur.dioxide   0.129101562   0.00000   1.0000000   0.0000000
density                0.016357422   0.00000   0.0000000   1.0000000
pH                     0.009375000   0.00000   0.0000000   0.0000000
sulphates              0.004296875   0.00000   0.0000000   0.0000000
alcohol                0.011132813   0.00000   0.0000000   0.0000000
BF                              NA   1.00000   0.9975965   0.1151104
PostProbs                       NA   0.79260   0.1122000   0.0114000
R2                              NA   0.19870   0.2611000   0.2198000
dim                             NA   2.00000   3.0000000   3.0000000
logmarg                         NA -45.16502 -45.1674217 -47.3268794
                       model 4      model 5
```

```
Intercept              1.0000000   1.00000000
fixed.acidity          0.0000000   0.00000000
volatile.acidity       1.0000000   1.00000000
citric.acid            0.0000000   0.00000000
residual.sugar         0.0000000   0.00000000
chlorides              1.0000000   0.00000000
free.sulfur.dioxide    0.0000000   1.00000000
total.sulfur.dioxide   0.0000000   0.00000000
density                0.0000000   0.00000000
pH                     0.0000000   0.00000000
sulphates              0.0000000   0.00000000
alcohol                0.0000000   0.00000000
BF                     0.1116504   0.08472875
PostProbs              0.0103000   0.00840000
R2                     0.2193000   0.21400000
dim                    3.0000000   3.00000000
logmarg              -47.3573984 -47.63331571
```

For level 2 (category 2 with reference level 6):

```
 Marginal Posterior Summaries of Coefficients:

 Using  BMA

 Based on the top  92 models
                     post mean   post SD     post p(B != 0)
Intercept             -2.8982978   0.1996788   1.0000000
fixed.acidity         -0.0024105   0.0314922   0.0149414
volatile.acidity       0.5786542   0.1617787   0.9791992
citric.acid            0.0002903   0.0243806   0.0083984
residual.sugar         0.0109902   0.0586060   0.0430176
chlorides              0.0004225   0.0148975   0.0106445
free.sulfur.dioxide   -0.0228008   0.1140097   0.0520508
total.sulfur.dioxide  -0.5490599   0.2592015   0.8873535
density               -0.0060057   0.0498721   0.0220703
pH                     0.0224283   0.0935222   0.0670410
sulphates              0.0007647   0.0180731   0.0102051
alcohol                0.0350480   0.1284128   0.0828613
> summary(res1.2)
                    P(B != 0 | Y)   model 1      model 2      model 3
Intercept             1.000000000    1.0000    1.0000000    1.0000000
fixed.acidity         0.014941406    0.0000    0.0000000    0.0000000
volatile.acidity      0.979199219    1.0000    1.0000000    1.0000000
citric.acid           0.008398437    0.0000    0.0000000    0.0000000
residual.sugar        0.043017578    0.0000    0.0000000    0.0000000
chlorides             0.010644531    0.0000    0.0000000    0.0000000
free.sulfur.dioxide   0.052050781    0.0000    0.0000000    0.0000000
total.sulfur.dioxide  0.887353516    1.0000    1.0000000    1.0000000
```

```
density                0.022070313    0.0000    0.0000000    0.0000000
pH                     0.067041016    0.0000    0.0000000    1.0000000
sulphates              0.010205078    0.0000    0.0000000    0.0000000
alcohol                0.082861328    0.0000    1.0000000    0.0000000
BF                              NA    1.0000    0.3167589    0.2574623
PostProbs                       NA    0.6815    0.0532000    0.0363000
R2                              NA    0.0961    0.1074000    0.1063000
dim                             NA    3.0000    4.0000000    4.0000000
logmarg                         NA -178.6481 -179.7977127 -180.0049804
                           model 4      model 5
Intercept             1.000000e+00    1.0000000
fixed.acidity         0.000000e+00    0.0000000
volatile.acidity      1.000000e+00    1.0000000
citric.acid           0.000000e+00    0.0000000
residual.sugar        0.000000e+00    1.0000000
chlorides             0.000000e+00    0.0000000
free.sulfur.dioxide   0.000000e+00    0.0000000
total.sulfur.dioxide  0.000000e+00    1.0000000
density               0.000000e+00    0.0000000
pH                    0.000000e+00    0.0000000
sulphates             0.000000e+00    0.0000000
alcohol               0.000000e+00    0.0000000
BF                    9.554042e-03    0.1789942
PostProbs             3.430000e-02    0.0297000
R2                    5.430000e-02    0.1044000
dim                   2.000000e+00    4.0000000
logmarg              -1.832989e+02 -180.3685001
```

For level 3 (category 3 with reference level 6):

```
 Marginal Posterior Summaries of Coefficients:

 Using  BMA

 Based on the top  56 models
                      post mean   post SD     post p(B != 0)
Intercept              1.463e-01   6.624e-02   1.000e+00
fixed.acidity          1.712e-03   2.020e-02   1.626e-02
volatile.acidity      -4.043e-01   8.166e-02   9.999e-01
citric.acid           -1.981e-02   6.667e-02   1.019e-01
residual.sugar         8.029e-04   1.073e-02   1.812e-02
chlorides             -2.484e-02   6.359e-02   1.583e-01
free.sulfur.dioxide    1.822e-01   1.523e-01   6.465e-01
total.sulfur.dioxide  -5.678e-01   1.360e-01   9.998e-01
density                7.787e-04   1.646e-02   1.660e-02
pH                    -3.233e-05   1.012e-02   1.792e-02
sulphates              3.393e-01   7.310e-02   9.986e-01
alcohol                8.348e-01   8.997e-02   9.984e-01
```

```
> summary(res1.3)
                      P(B != 0 | Y)    model 1     model 2      model 3
Intercept               1.00000000    1.0000    1.000000    1.0000000
fixed.acidity           0.01625977    0.0000    0.000000    0.0000000
volatile.acidity        0.99990234    1.0000    1.000000    1.0000000
citric.acid             0.10190430    0.0000    0.000000    0.0000000
residual.sugar          0.01811523    0.0000    0.000000    0.0000000
chlorides               0.15825195    0.0000    0.000000    1.0000000
free.sulfur.dioxide     0.64648438    1.0000    0.000000    1.0000000
total.sulfur.dioxide    0.99975586    1.0000    1.000000    1.0000000
density                 0.01660156    0.0000    0.000000    0.0000000
pH                      0.01791992    0.0000    0.000000    0.0000000
sulphates               0.99858398    1.0000    1.000000    1.0000000
alcohol                 0.99838867    1.0000    1.000000    1.0000000
BF                             NA    1.0000    0.176709    0.4045382
PostProbs                      NA    0.4693    0.239100    0.1029000
R2                             NA    0.1784    0.172600    0.1814000
dim                            NA    6.0000    5.000000    7.0000000
logmarg                        NA -768.5041 -770.237319 -769.4090773
                        model 4       model 5
Intercept             1.0000000    1.00000000
fixed.acidity         0.0000000    0.00000000
volatile.acidity      1.0000000    1.00000000
citric.acid           1.0000000    0.00000000
residual.sugar        0.0000000    0.00000000
chlorides             0.0000000    1.00000000
free.sulfur.dioxide   0.0000000    0.00000000
total.sulfur.dioxide  1.0000000    1.00000000
density               0.0000000    0.00000000
pH                    0.0000000    0.00000000
sulphates             1.0000000    1.00000000
alcohol               1.0000000    1.00000000
BF                    0.1080856    0.08753025
PostProbs             0.0477000    0.03740000
R2                    0.1760000    0.17580000
dim                   6.0000000    6.00000000
logmarg            -770.7289000 -770.93983893
```

For level 4 (category 4 with reference level 6):

```
 Marginal Posterior Summaries of Coefficients:

 Using  BMA

 Based on the top  80 models
                     post mean  post SD    post p(B != 0)
Intercept            -1.928005   0.154828   1.000000
fixed.acidity         0.027436   0.090846   0.110107
```

```
volatile.acidity      -1.002758   0.155906   0.999756
citric.acid           -0.001380   0.039418   0.020313
residual.sugar         0.007085   0.043262   0.038379
chlorides             -0.063223   0.149847   0.189697
free.sulfur.dioxide    0.052857   0.151766   0.131348
total.sulfur.dioxide  -0.538386   0.188332   0.988525
density               -0.001178   0.040672   0.020801
pH                    -0.011699   0.059048   0.055273
sulphates              0.626892   0.123760   0.998682
alcohol                1.760819   0.156445   0.999951
> summary(res1.4)
                    P(B != 0 | Y)   model 1      model 2      model 3
Intercept              1.00000000    1.0000    1.0000000    1.0000000
fixed.acidity          0.11010742    0.0000    0.0000000    0.0000000
volatile.acidity       0.99975586    1.0000    1.0000000    1.0000000
citric.acid            0.02031250    0.0000    0.0000000    0.0000000
residual.sugar         0.03837891    0.0000    0.0000000    0.0000000
chlorides              0.18969727    0.0000    1.0000000    0.0000000
free.sulfur.dioxide    0.13134766    0.0000    0.0000000    1.0000000
total.sulfur.dioxide   0.98852539    1.0000    1.0000000    1.0000000
density                0.02080078    0.0000    0.0000000    0.0000000
pH                     0.05527344    0.0000    0.0000000    0.0000000
sulphates              0.99868164    1.0000    1.0000000    1.0000000
alcohol                0.99995117    1.0000    1.0000000    1.0000000
BF                             NA    1.0000    0.5284735    0.3574823
PostProbs                      NA    0.5808    0.1132000    0.0698000
R2                             NA    0.5192    0.5250000    0.5242000
dim                            NA    5.0000    6.0000000    6.0000000
logmarg                        NA -239.7406 -240.3783653 -240.7692722
                       model 4      model 5
Intercept             1.0000000    1.0000000
fixed.acidity         1.0000000    0.0000000
volatile.acidity      1.0000000    1.0000000
citric.acid           0.0000000    0.0000000
residual.sugar        0.0000000    0.0000000
chlorides             0.0000000    0.0000000
free.sulfur.dioxide   0.0000000    0.0000000
total.sulfur.dioxide  1.0000000    1.0000000
density               0.0000000    0.0000000
pH                    0.0000000    1.0000000
sulphates             1.0000000    1.0000000
alcohol               1.0000000    1.0000000
BF                    0.2508113    0.1176858
PostProbs             0.0495000    0.0237000
R2                    0.5235000    0.5218000
dim                   6.0000000    6.0000000
logmarg            -241.1236571 -241.8803398
```

For level 5 (category 5 with reference level 6):

```
 Marginal Posterior Summaries of Coefficients:

 Using  BMA

 Based on the top  288 models
                     post mean  post SD   post p(B != 0)
Intercept             -5.69923    1.40947   1.00000
fixed.acidity         -0.02627    0.21167   0.03550
volatile.acidity      -0.60438    0.71479   0.46704
citric.acid            0.49775    0.78559   0.33643
residual.sugar        -0.66771    0.99523   0.36714
chlorides             -0.54429    1.20641   0.21753
free.sulfur.dioxide   -0.00152    0.12386   0.02661
total.sulfur.dioxide  -0.88077    0.96336   0.52803
density               -0.12867    0.45461   0.09658
pH                    -0.15009    0.40770   0.15039
sulphates              0.70311    0.68441   0.57793
alcohol                2.18824    0.71022   0.99844
> summary(res1.5)
                    P(B != 0 | Y)      model 1   model 2     model 3
Intercept              1.00000000   1.00000000   1.00000   1.0000000
fixed.acidity          0.03549805   0.00000000   0.00000   0.0000000
volatile.acidity       0.46704102   1.00000000   0.00000   1.0000000
citric.acid            0.33642578   0.00000000   1.00000   0.0000000
residual.sugar         0.36713867   0.00000000   1.00000   1.0000000
chlorides              0.21752930   0.00000000   0.00000   0.0000000
free.sulfur.dioxide    0.02661133   0.00000000   0.00000   0.0000000
total.sulfur.dioxide   0.52802734   0.00000000   1.00000   1.0000000
density                0.09658203   0.00000000   0.00000   0.0000000
pH                     0.15039062   0.00000000   0.00000   0.0000000
sulphates              0.57792969   0.00000000   1.00000   1.0000000
alcohol                0.99843750   1.00000000   1.00000   1.0000000
BF                             NA   0.04817807   1.00000   0.9797934
PostProbs                      NA   0.17370000   0.07660   0.0747000
R2                             NA   0.47450000   0.62820   0.6280000
dim                            NA   3.00000000   6.00000   6.0000000
logmarg                        NA -50.50115974 -47.46831 -47.4887219
                       model 4     model 5
Intercept            1.00000000   1.0000000
fixed.acidity        0.00000000   0.0000000
volatile.acidity     0.00000000   1.0000000
citric.acid          0.00000000   0.0000000
residual.sugar       0.00000000   0.0000000
chlorides            0.00000000   0.0000000
free.sulfur.dioxide  0.00000000   0.0000000
total.sulfur.dioxide 1.00000000   1.0000000
density              0.00000000   0.0000000
```

```
pH                    0.00000000   0.0000000
sulphates             1.00000000   1.0000000
alcohol               1.00000000   1.0000000
BF                    0.07164376   0.2830567
PostProbs             0.05470000   0.0478000
R2                    0.51840000   0.5740000
dim                   4.00000000   5.0000000
logmarg             -50.10435766 -48.7304164
```

Just like before, we see that in the end our model will be looking like this:

$$ln\left(\frac{p_{i1}}{p_{i6}}\right) = -5.3 + 1.06 \cdot Z_{i2}$$

$$ln\left(\frac{p_{i2}}{p_{i6}}\right) = -2.89 + 0.57 \cdot Z_{i2} - 0.55 \cdot Z_{i7}$$

$$ln\left(\frac{p_{i3}}{p_{i6}}\right) = 0.15 - 0.4 \cdot Z_{i2} + 0.18 \cdot Z_{i6} - 0.57 \cdot Z_{i7} + 0.34 \cdot Z_{i10} + 0.83 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i4}}{p_{i6}}\right) = -1.93 - 1 \cdot Z_{i2} - 0.54 \cdot Z_{i7} + 0.63 \cdot Z_{i10} + 1.76 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i5}}{p_{i6}}\right) = -5.7 + 0.5 \cdot Z_{i3} - 0.67 \cdot Z_{i4} - 0.88 \cdot Z_{i7} + 0.7 \cdot Z_{i10} + 2.19 \cdot Z_{i11}$$

Here we did not use the physiochemical characteristics of *Fixed Acidity*, *chlorides*, *pH* and *Density* at all. This selection was a bit more parsimonious than the last one.

Variable Selection through Empirical Bayes (using OpenBUGS):

The outlawed approach that will get us in trouble if we talk to any old school Bayesians. This method is assuming that:

$$\beta_{jk} \sim N\left(\tilde{\beta}_{jk}, n \cdot \tilde{S}^2_{\beta_{jk}}\right)$$

for $j = 1, 2, 3, \dots, 11$, our number of estimates and $k = 1, 2, \dots, 5$, our number of levels.

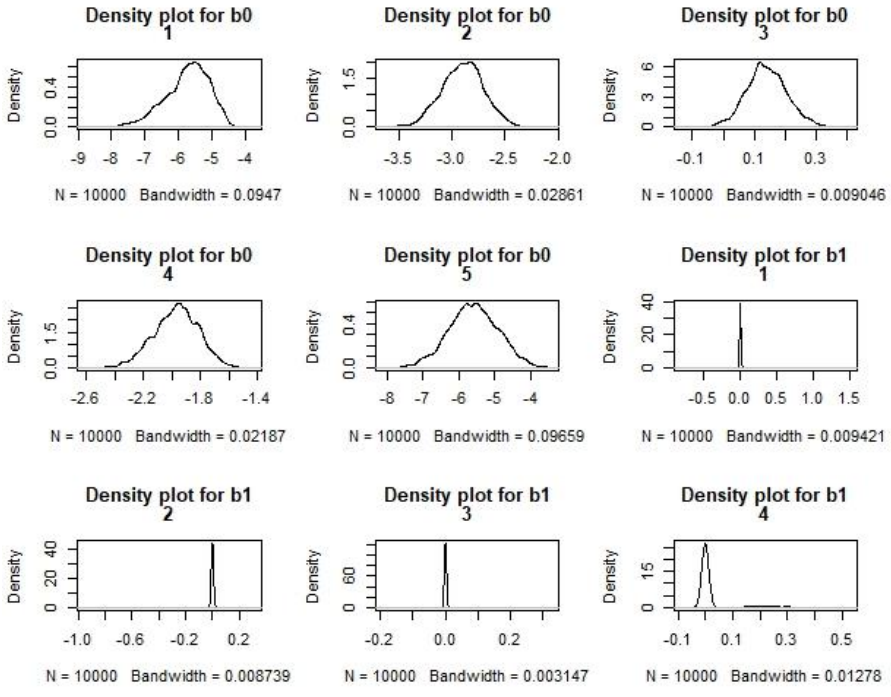
where, $\tilde{\beta}_{jk}$ is the posterior mean and $\tilde{S}^2_{\beta_{jk}}$ is the posterior variance of the full model. In our code, we have set our constants to always be present in the final result. Due to time constraints, we ran each model for 11000 iterations with a thinning step of 30.

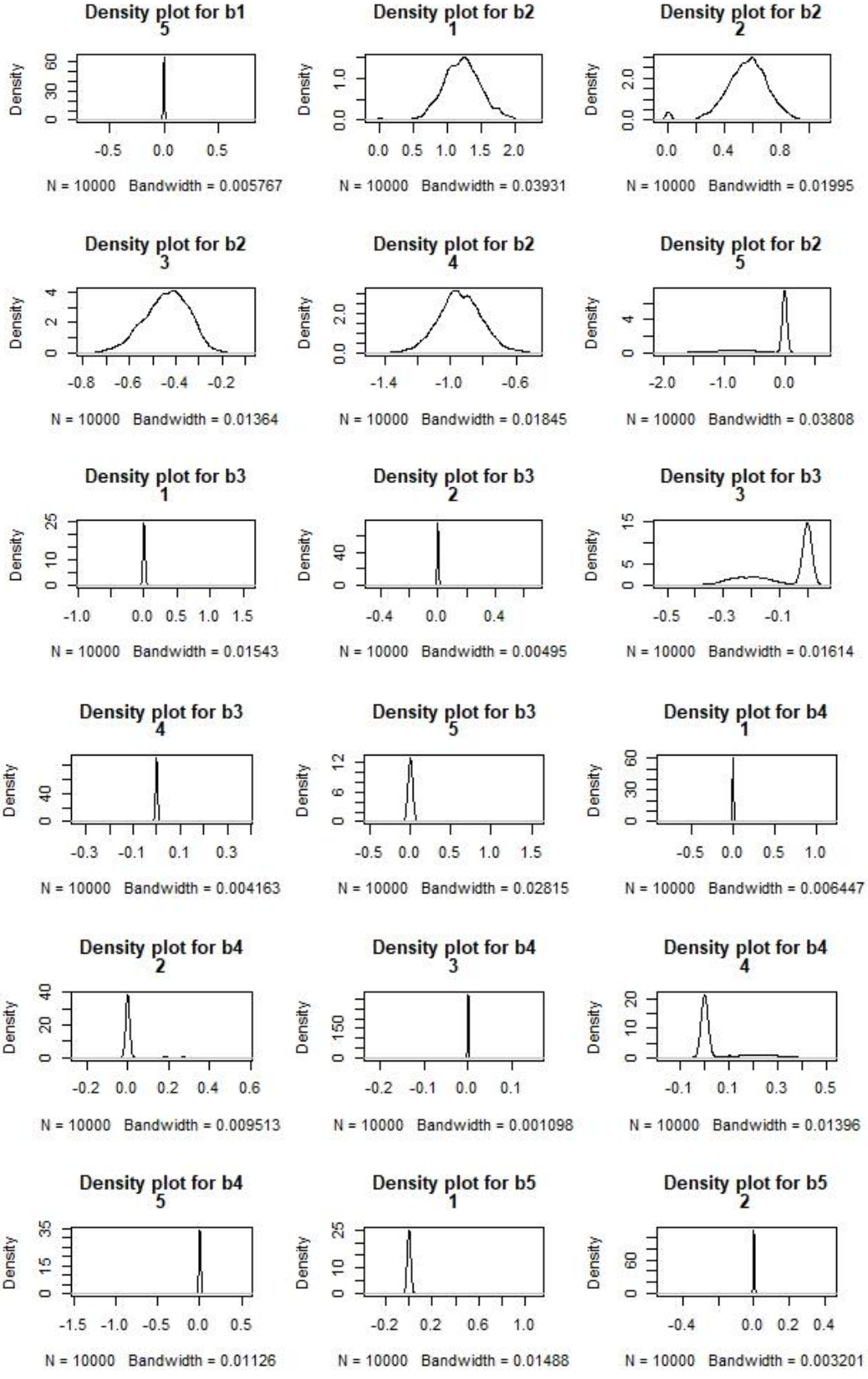
1. Using a Uniform Distribution on the priors:

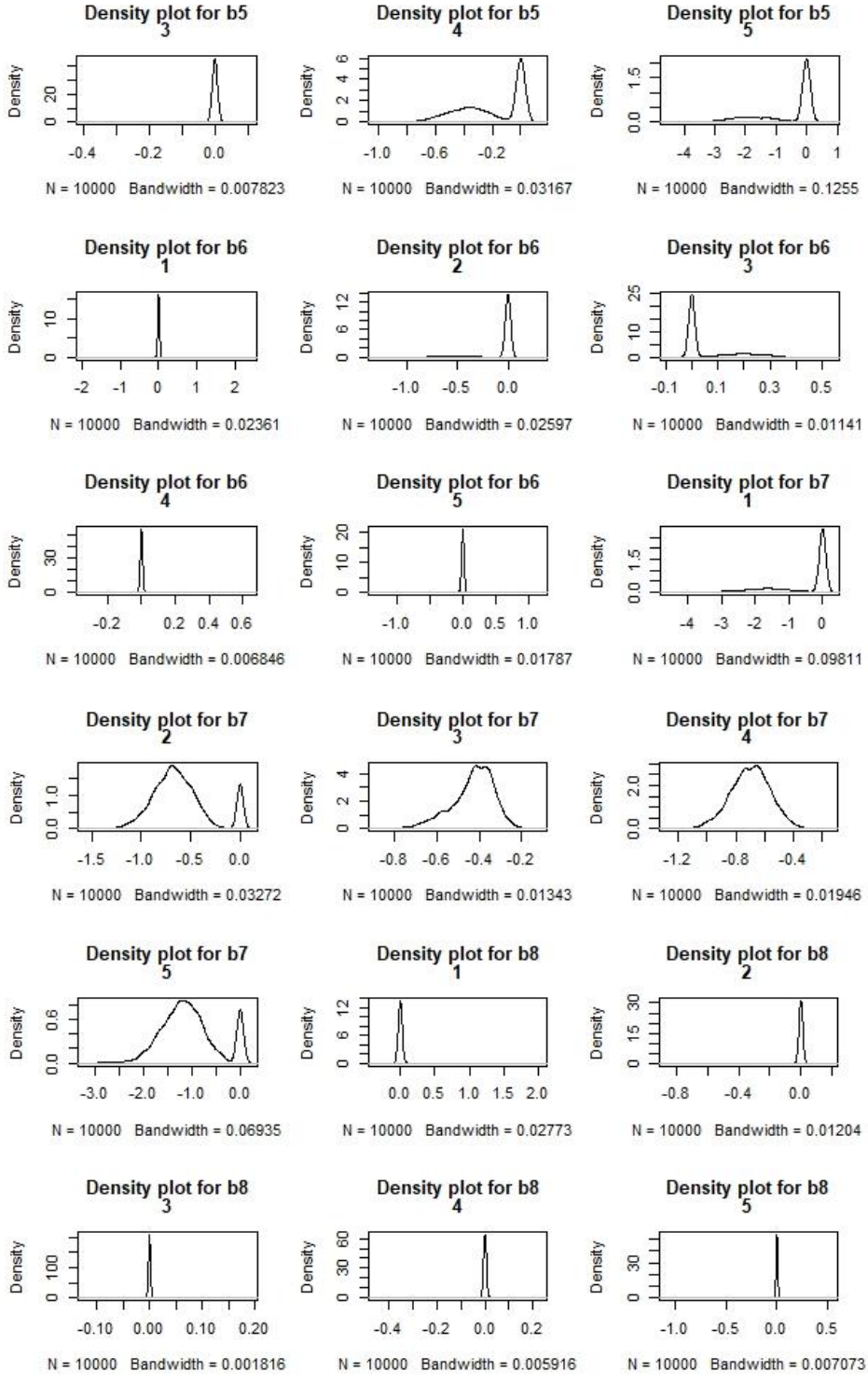
Here the results are:

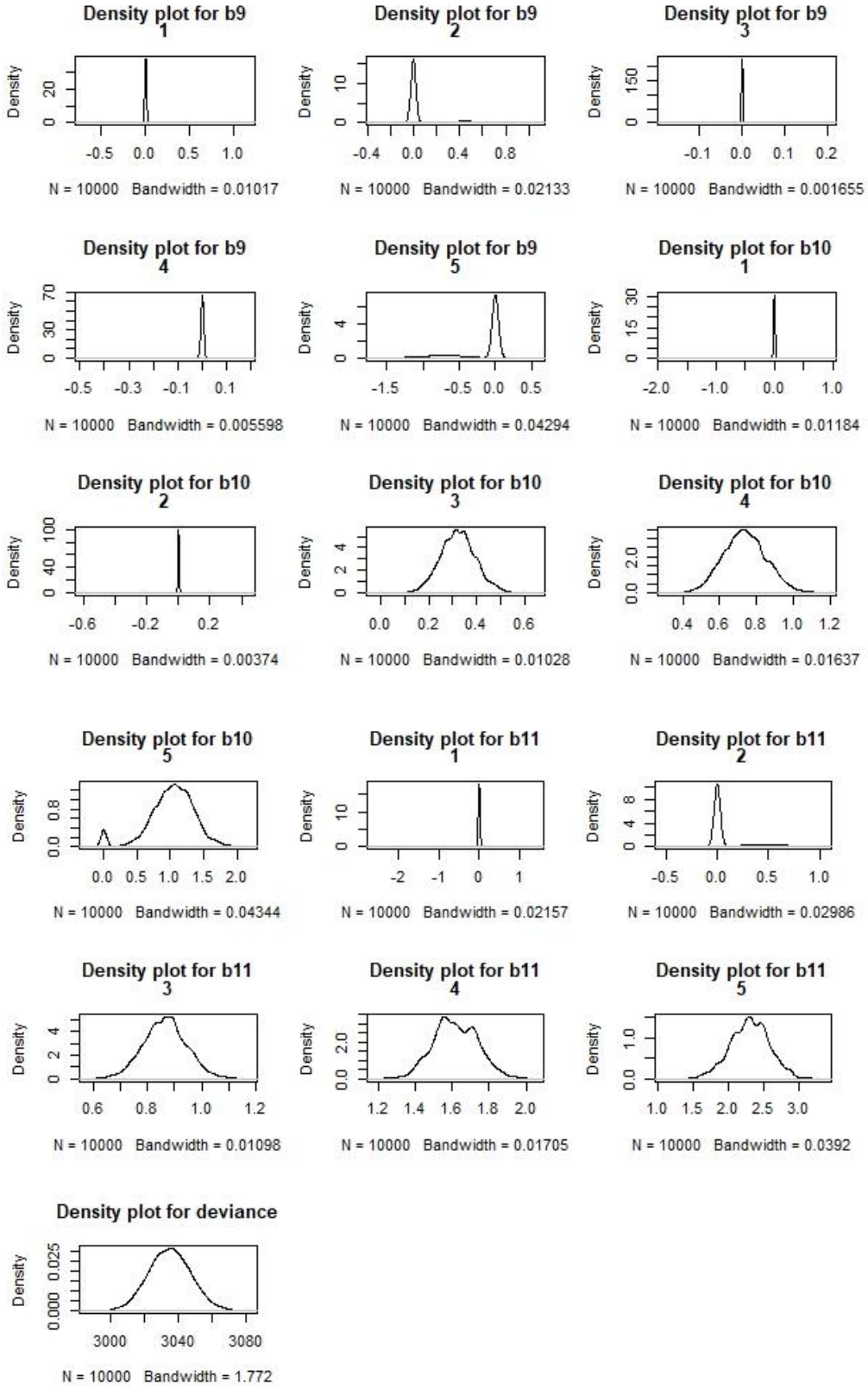
```
Inference for Bugs model at "C:\Users\30697\Desktop\empirical.txt",
fit using OpenBUGS,
 1 chains, each with 11000 iterations (first 1000 discarded), n.thin =
30
 n.sims = 333 iterations saved
           mean   sd   2.5%    25%    50%    75%  97.5%
gb0[1]     -5.7  0.6   -6.9   -6.1   -5.6   -5.3   -4.7
gb0[2]     -2.9  0.2   -3.3   -3.0   -2.9   -2.8   -2.5
gb0[3]      0.1  0.1    0.0    0.1    0.1    0.2    0.3
gb0[4]     -2.0  0.2   -2.3   -2.1   -2.0   -1.9   -1.7
gb0[5]     -5.6  0.7   -6.9   -6.0   -5.6   -5.2   -4.2
gb1[1]      0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb1[2]      0.0  0.1    0.0    0.0    0.0    0.0    0.0
gb1[3]      0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb1[4]      0.0  0.1    0.0    0.0    0.0    0.0    0.3
gb1[5]      0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb2[1]      1.2  0.3    0.7    1.0    1.2    1.4    1.8
gb2[2]      0.5  0.2    0.0    0.5    0.6    0.6    0.8
gb2[3]     -0.4  0.1   -0.7   -0.5   -0.4   -0.4   -0.3
gb2[4]     -0.9  0.1   -1.2   -1.0   -1.0   -0.9   -0.7
gb2[5]     -0.2  0.4   -1.4    0.0    0.0    0.0    0.0
gb3[1]      0.0  0.2    0.0    0.0    0.0    0.0    0.1
gb3[2]      0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb3[3]     -0.1  0.1   -0.3   -0.2    0.0    0.0    0.0
gb3[4]      0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb3[5]      0.1  0.2    0.0    0.0    0.0    0.0    0.8
gb4[1]      0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb4[2]      0.0  0.1    0.0    0.0    0.0    0.0    0.3
gb4[3]      0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb4[4]      0.0  0.1    0.0    0.0    0.0    0.0    0.3
gb4[5]      0.0  0.1    0.0    0.0    0.0    0.0    0.0
gb5[1]      0.0  0.1    0.0    0.0    0.0    0.0    0.3
gb5[2]      0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb5[3]      0.0  0.0   -0.2    0.0    0.0    0.0    0.0
gb5[4]     -0.2  0.2   -0.6   -0.4   -0.1    0.0    0.0
gb5[5]     -0.6  0.9   -2.8   -1.2    0.0    0.0    0.0
gb6[1]      0.0  0.2    0.0    0.0    0.0    0.0    0.0
gb6[2]     -0.1  0.2   -0.7    0.0    0.0    0.0    0.0
gb6[3]      0.1  0.1    0.0    0.0    0.0    0.1    0.3
gb6[4]      0.0  0.1    0.0    0.0    0.0    0.0    0.2
gb6[5]      0.0  0.1   -0.2    0.0    0.0    0.0    0.0
gb7[1]     -0.5  0.8   -2.6   -0.8    0.0    0.0    0.0
gb7[2]     -0.6  0.3   -1.1   -0.8   -0.7   -0.5    0.0
gb7[3]     -0.4  0.1   -0.7   -0.5   -0.4   -0.4   -0.3
```

```
gb7[4]      -0.7  0.1   -1.0   -0.8   -0.7   -0.6   -0.4
gb7[5]      -1.1  0.6   -2.1   -1.5   -1.1   -0.7    0.0
gb8[1]       0.0  0.2    0.0    0.0    0.0    0.0    0.8
gb8[2]       0.0  0.1   -0.3    0.0    0.0    0.0    0.0
gb8[3]       0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb8[4]       0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb8[5]       0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb9[1]       0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb9[2]       0.1  0.2    0.0    0.0    0.0    0.0    0.5
gb9[3]       0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb9[4]       0.0  0.0   -0.1    0.0    0.0    0.0    0.0
gb9[5]      -0.1  0.3   -1.0    0.0    0.0    0.0    0.0
gb10[1]      0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb10[2]      0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb10[3]      0.3  0.1    0.2    0.3    0.3    0.4    0.5
gb10[4]      0.7  0.1    0.5    0.7    0.7    0.8    1.0
gb10[5]      1.0  0.3    0.0    0.8    1.0    1.2    1.6
gb11[1]      0.0  0.1    0.0    0.0    0.0    0.0    0.0
gb11[2]      0.1  0.2    0.0    0.0    0.0    0.0    0.7
gb11[3]      0.9  0.1    0.7    0.8    0.9    0.9    1.0
gb11[4]      1.6  0.1    1.4    1.5    1.6    1.7    1.8
gb11[5]      2.3  0.3    1.8    2.1    2.3    2.5    2.8
deviance  3036.4 13.0 3011.6 3028.3 3036.6 3044.3 3060.2

DIC info (using the rule, pD = var(deviance)/2)
pD = 84.3 and DIC = 3120.7
DIC is an estimate of expected predictive error (lower deviance is
better).
```




Density plot for b1 5
Density plot for b2 1
Density plot for b2 2
Density plot for b2 3
Density plot for b2 4
Density plot for b2 5
Density plot for b3 1
Density plot for b3 2
Density plot for b3 3
Density plot for b3 4
Density plot for b3 5
Density plot for b4 1
Density plot for b4 2
Density plot for b4 3
Density plot for b4 4
Density plot for b4 5
Density plot for b5 1
Density plot for b5 2
Density
N = 10000 Bandwidth = 0.005767
N = 10000 Bandwidth = 0.03931
N = 10000 Bandwidth = 0.01995
N = 10000 Bandwidth = 0.01364
N = 10000 Bandwidth = 0.01845
N = 10000 Bandwidth = 0.03808
N = 10000 Bandwidth = 0.01543
N = 10000 Bandwidth = 0.00495
N = 10000 Bandwidth = 0.01614
N = 10000 Bandwidth = 0.004163
N = 10000 Bandwidth = 0.02815
N = 10000 Bandwidth = 0.006447
N = 10000 Bandwidth = 0.009513
N = 10000 Bandwidth = 0.001098
N = 10000 Bandwidth = 0.01396
N = 10000 Bandwidth = 0.01126
N = 10000 Bandwidth = 0.01488
N = 10000 Bandwidth = 0.003201


Density plot for b5 3
Density
N = 10000 Bandwidth = 0.007823
Density plot for b5 4
Density
N = 10000 Bandwidth = 0.03167
Density plot for b5 5
Density
N = 10000 Bandwidth = 0.1255
Density plot for b6 1
Density
N = 10000 Bandwidth = 0.02361
Density plot for b6 2
Density
N = 10000 Bandwidth = 0.02597
Density plot for b6 3
Density
N = 10000 Bandwidth = 0.01141
Density plot for b6 4
Density
N = 10000 Bandwidth = 0.006846
Density plot for b6 5
Density
N = 10000 Bandwidth = 0.01787
Density plot for b7 1
Density
N = 10000 Bandwidth = 0.09811
Density plot for b7 2
Density
N = 10000 Bandwidth = 0.03272
Density plot for b7 3
Density
N = 10000 Bandwidth = 0.01343
Density plot for b7 4
Density
N = 10000 Bandwidth = 0.01946
Density plot for b7 5
Density
N = 10000 Bandwidth = 0.06935
Density plot for b8 1
Density
N = 10000 Bandwidth = 0.02773
Density plot for b8 2
Density
N = 10000 Bandwidth = 0.01204
Density plot for b8 3
Density
N = 10000 Bandwidth = 0.001816
Density plot for b8 4
Density
N = 10000 Bandwidth = 0.005916
Density plot for b8 5
Density
N = 10000 Bandwidth = 0.007073


Density plot for b9
1
Density
N = 10000 Bandwidth = 0.01017
Density plot for b9
2
Density
N = 10000 Bandwidth = 0.02133
Density plot for b9
3
Density
N = 10000 Bandwidth = 0.001655
Density plot for b9
4
Density
N = 10000 Bandwidth = 0.005598
Density plot for b9
5
Density
N = 10000 Bandwidth = 0.04294
Density plot for b10
1
Density
N = 10000 Bandwidth = 0.01184
Density plot for b10
2
Density
N = 10000 Bandwidth = 0.00374
Density plot for b10
3
Density
N = 10000 Bandwidth = 0.01028
Density plot for b10
4
Density
N = 10000 Bandwidth = 0.01637
Density plot for b10
5
Density
N = 10000 Bandwidth = 0.04344
Density plot for b11
1
Density
N = 10000 Bandwidth = 0.02157
Density plot for b11
2
Density
N = 10000 Bandwidth = 0.02986
Density plot for b11
3
Density
N = 10000 Bandwidth = 0.01098
Density plot for b11
4
Density
N = 10000 Bandwidth = 0.01705
Density plot for b11
5
Density
N = 10000 Bandwidth = 0.0392
Density plot for deviance
Density
N = 10000 Bandwidth = 1.772

Then in this case, the model will be looking like this:

$$ln\left(\frac{p_{i1}}{p_{i6}}\right) = -5.7 + 1.2 \cdot Z_{i2} - 0.5 \cdot Z_{i7}$$

$$ln\left(\frac{p_{i2}}{p_{i6}}\right) = -2.9 + 0.5 \cdot Z_{i2} - 0.1 \cdot Z_{i6} - 0.6 \cdot Z_{i7} + 0.1 \cdot Z_{i9} + 0.1 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i3}}{p_{i6}}\right) = 0.1 - 0.4 \cdot Z_{i2} - 0.1 \cdot Z_{i3} + 0.1 \cdot Z_{i6} - 0.4 \cdot Z_{i7} + 0.3 \cdot Z_{i10} + 0.9 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i4}}{p_{i6}}\right) = -2 - 0.9 \cdot Z_{i2} - 0.2 \cdot Z_{i5} - 0.7 \cdot Z_{i7} + 0.7 \cdot Z_{i10} + 1.6 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i5}}{p_{i6}}\right) = -5.60 - 0.2 \cdot Z_{i2} + 0.1 \cdot Z_{i3} - 0.6 \cdot Z_{i5} - 1.1 \cdot Z_{i7} + 1 \cdot Z_{i10} + 2.3 \cdot Z_{i11}$$

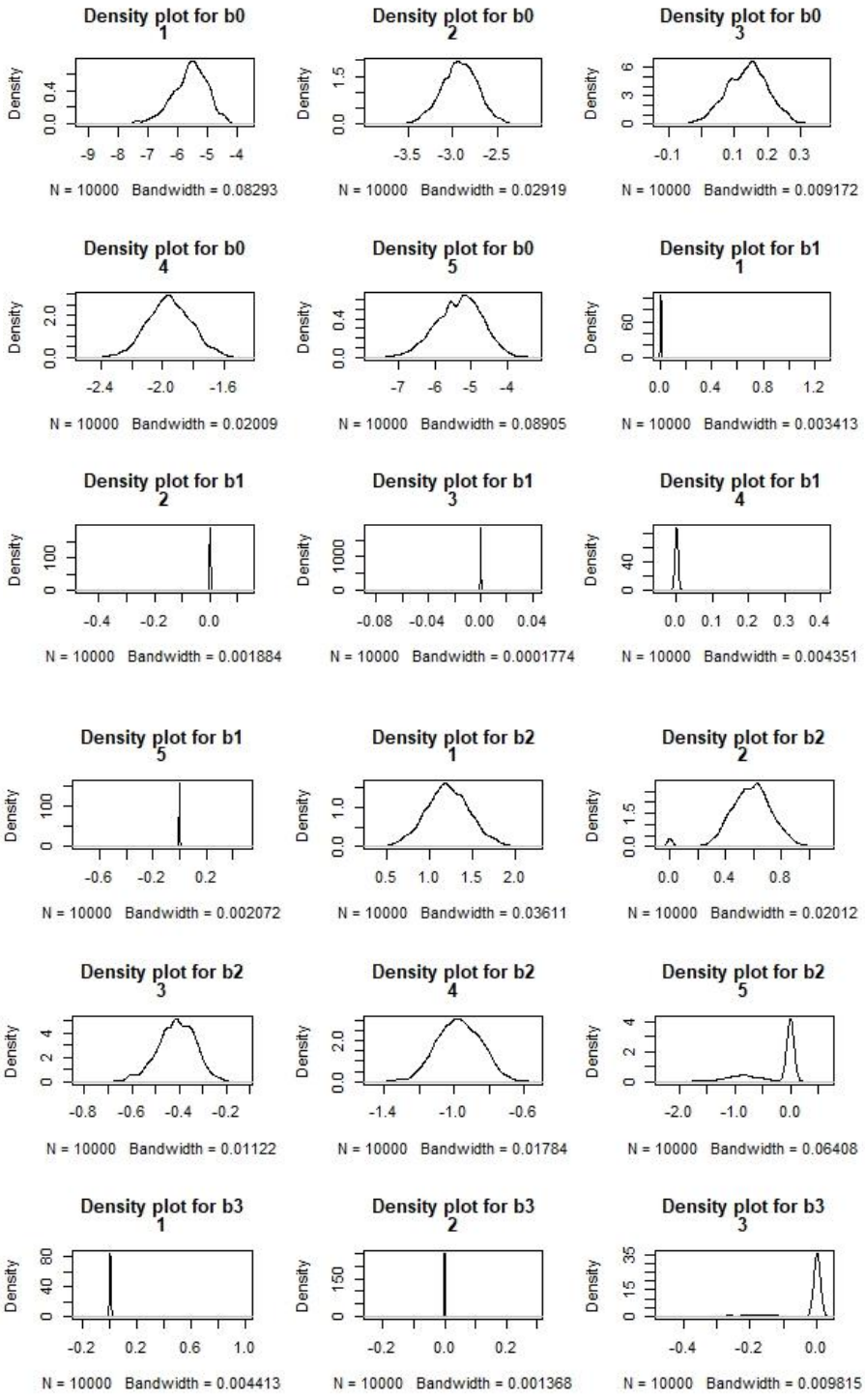
Surely, had we taken more iterations, the results would look better. Here we did not use the physiochemical characteristics of *Fixed Acidity*, *Residual Sugar* and *Density* at all, while others were used up to a certain degree at each level.

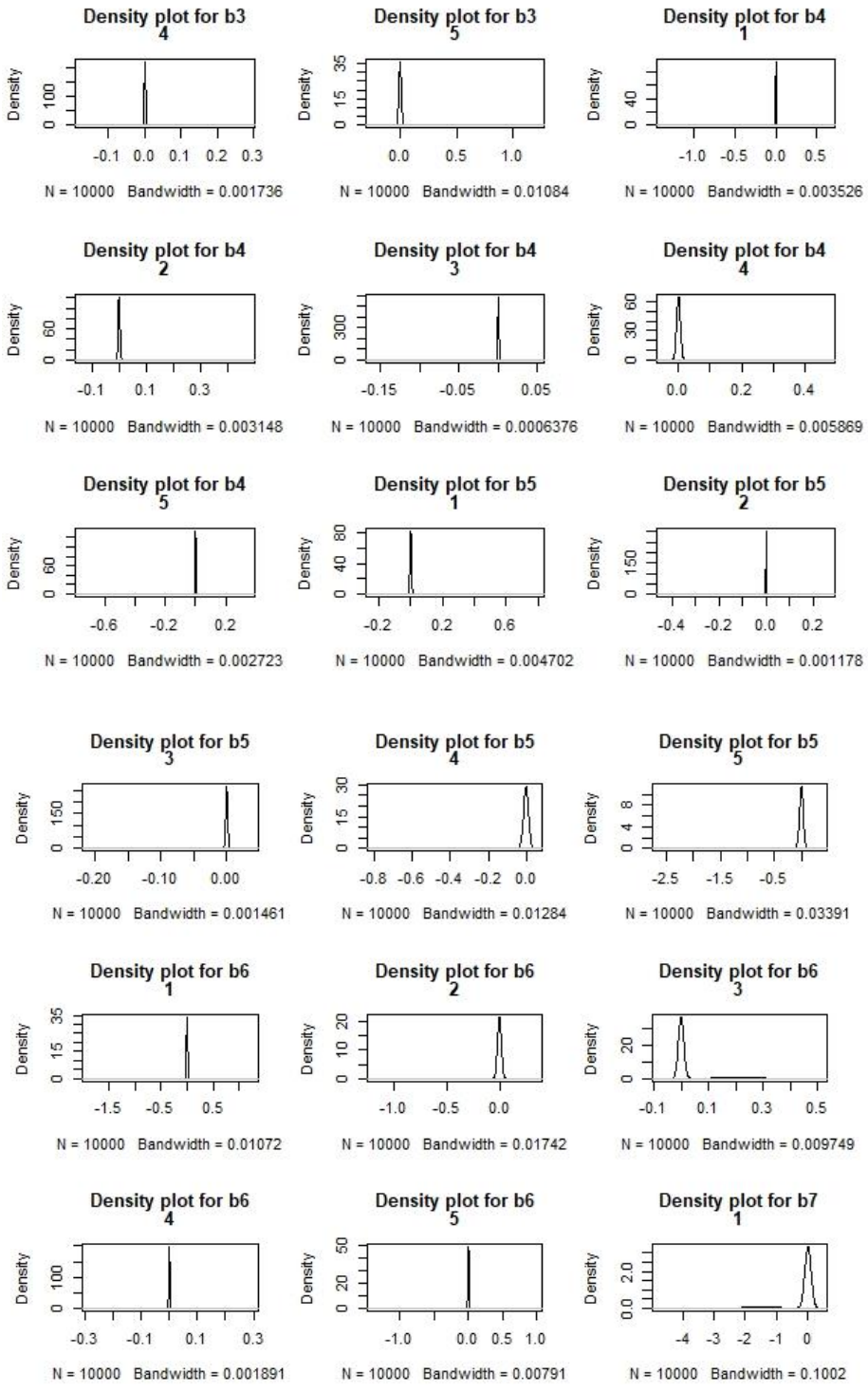
2. Using a Beta-Binomial Distribution on the priors:

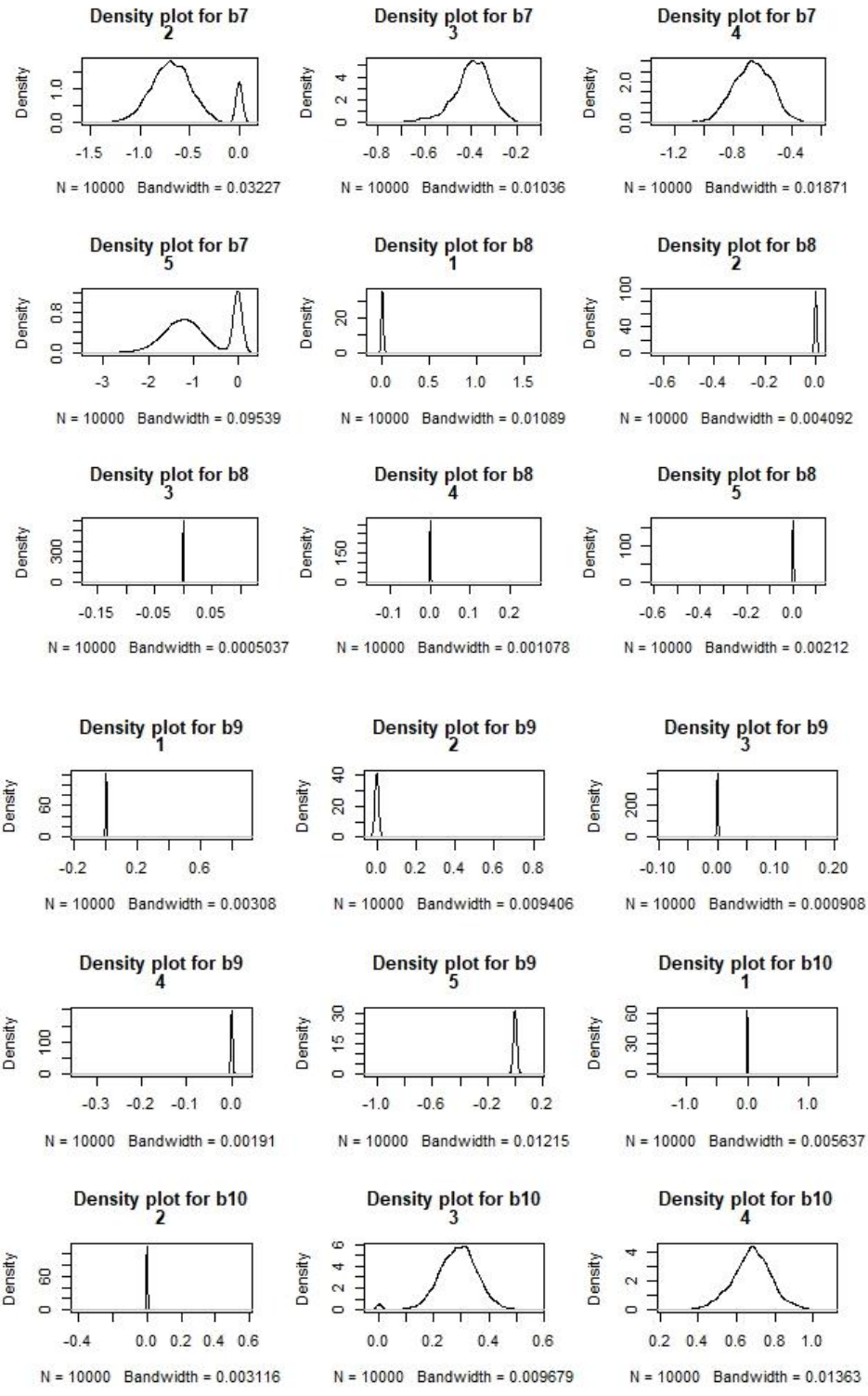
```
Inference for Bugs model at "C:\Users\30697\Desktop\empirical.txt",
fit using OpenBUGS,
 1 chains, each with 11000 iterations (first 1000 discarded), n.thin =
30
 n.sims = 333 iterations saved
          mean   sd   2.5%    25%    50%    75%  97.5%
gb0[1]    -5.6  0.6   -7.0   -5.9   -5.5   -5.2   -4.5
gb0[2]    -2.9  0.2   -3.3   -3.1   -2.9   -2.8   -2.5
gb0[3]     0.1  0.1    0.0    0.1    0.1    0.2    0.3
gb0[4]    -2.0  0.1   -2.2   -2.0   -2.0   -1.9   -1.7
gb0[5]    -5.3  0.6   -6.6   -5.8   -5.3   -4.9   -4.2
gb1[1]     0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb1[2]     0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb1[3]     0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb1[4]     0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb1[5]     0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb2[1]     1.2  0.3    0.7    1.0    1.2    1.4    1.7
gb2[2]     0.6  0.2    0.3    0.5    0.6    0.7    0.9
gb2[3]    -0.4  0.1   -0.6   -0.5   -0.4   -0.4   -0.3
gb2[4]    -1.0  0.1   -1.2   -1.1   -1.0   -0.9   -0.7
gb2[5]    -0.3  0.4   -1.3   -0.6    0.0    0.0    0.0
gb3[1]     0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb3[2]     0.0  0.0    0.0    0.0    0.0    0.0    0.0
```

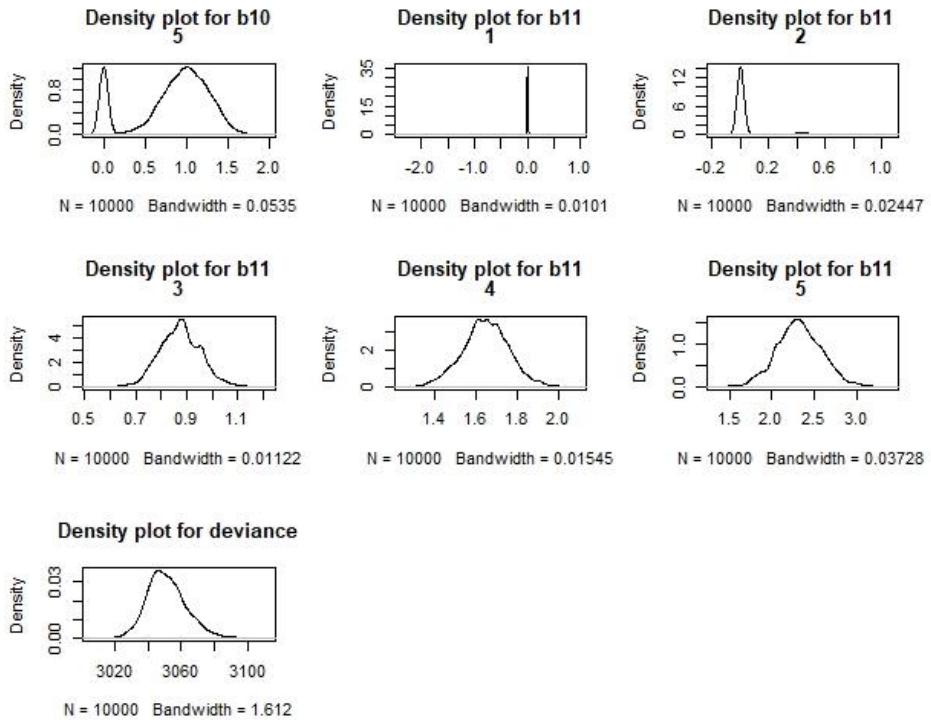
```
gb3[3]      0.0  0.1   -0.3    0.0    0.0    0.0    0.0
gb3[4]      0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb3[5]      0.0  0.1    0.0    0.0    0.0    0.0    0.0
gb4[1]      0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb4[2]      0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb4[3]      0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb4[4]      0.0  0.0    0.0    0.0    0.0    0.0    0.2
gb4[5]      0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb5[1]      0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb5[2]      0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb5[3]      0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb5[4]      0.0  0.1   -0.4    0.0    0.0    0.0    0.0
gb5[5]      0.0  0.2   -0.5    0.0    0.0    0.0    0.0
gb6[1]      0.0  0.1    0.0    0.0    0.0    0.0    0.0
gb6[2]      0.0  0.1   -0.5    0.0    0.0    0.0    0.0
gb6[3]      0.0  0.1    0.0    0.0    0.0    0.0    0.3
gb6[4]      0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb6[5]      0.0  0.1    0.0    0.0    0.0    0.0    0.0
gb7[1]     -0.3  0.7   -2.6    0.0    0.0    0.0    0.0
gb7[2]     -0.6  0.3   -1.1   -0.8   -0.7   -0.5    0.0
gb7[3]     -0.4  0.1   -0.6   -0.4   -0.4   -0.3   -0.3
gb7[4]     -0.7  0.1   -0.9   -0.8   -0.7   -0.6   -0.4
gb7[5]     -0.9  0.7   -2.0   -1.4   -1.0    0.0    0.0
gb8[1]      0.0  0.1    0.0    0.0    0.0    0.0    0.0
gb8[2]      0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb8[3]      0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb8[4]      0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb8[5]      0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb9[1]      0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb9[2]      0.0  0.1    0.0    0.0    0.0    0.0    0.1
gb9[3]      0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb9[4]      0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb9[5]      0.0  0.1    0.0    0.0    0.0    0.0    0.0
gb10[1]     0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb10[2]     0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb10[3]     0.3  0.1    0.1    0.2    0.3    0.3    0.4
gb10[4]     0.7  0.1    0.5    0.6    0.7    0.7    0.9
gb10[5]     0.8  0.4    0.0    0.7    0.9    1.2    1.5
gb11[1]     0.0  0.1    0.0    0.0    0.0    0.0    0.0
gb11[2]     0.1  0.2    0.0    0.0    0.0    0.0    0.6
gb11[3]     0.9  0.1    0.7    0.8    0.9    0.9    1.0
gb11[4]     1.6  0.1    1.4    1.6    1.6    1.7    1.9
gb11[5]     2.3  0.3    1.8    2.2    2.3    2.5    2.8
deviance 3051.3 11.7 3030.4 3043.2 3050.2 3058.4 3076.7

DIC info (using the rule, pD = var(deviance)/2)
pD = 68.2 and DIC = 3119.5
```

DIC is an estimate of expected predictive error (lower deviance is better).




Density plot for b3 4
Density
N = 10000 Bandwidth = 0.001736
Density plot for b3 5
Density
N = 10000 Bandwidth = 0.01084
Density plot for b4 1
Density
N = 10000 Bandwidth = 0.003526
Density plot for b4 2
Density
N = 10000 Bandwidth = 0.003148
Density plot for b4 3
Density
N = 10000 Bandwidth = 0.0006376
Density plot for b4 4
Density
N = 10000 Bandwidth = 0.005869
Density plot for b4 5
Density
N = 10000 Bandwidth = 0.002723
Density plot for b5 1
Density
N = 10000 Bandwidth = 0.004702
Density plot for b5 2
Density
N = 10000 Bandwidth = 0.001178
Density plot for b5 3
Density
N = 10000 Bandwidth = 0.001461
Density plot for b5 4
Density
N = 10000 Bandwidth = 0.01284
Density plot for b5 5
Density
N = 10000 Bandwidth = 0.03391
Density plot for b6 1
Density
N = 10000 Bandwidth = 0.01072
Density plot for b6 2
Density
N = 10000 Bandwidth = 0.01742
Density plot for b6 3
Density
N = 10000 Bandwidth = 0.009749
Density plot for b6 4
Density
N = 10000 Bandwidth = 0.001891
Density plot for b6 5
Density
N = 10000 Bandwidth = 0.00791
Density plot for b7 1
Density
N = 10000 Bandwidth = 0.1002


Density plot for b7 2
Density
N = 10000 Bandwidth = 0.03227
Density plot for b7 3
Density
N = 10000 Bandwidth = 0.01036
Density plot for b7 4
Density
N = 10000 Bandwidth = 0.01871
Density plot for b7 5
Density
N = 10000 Bandwidth = 0.09539
Density plot for b8 1
Density
N = 10000 Bandwidth = 0.01089
Density plot for b8 2
Density
N = 10000 Bandwidth = 0.004092
Density plot for b8 3
Density
N = 10000 Bandwidth = 0.0005037
Density plot for b8 4
Density
N = 10000 Bandwidth = 0.001078
Density plot for b8 5
Density
N = 10000 Bandwidth = 0.00212
Density plot for b9 1
Density
N = 10000 Bandwidth = 0.00308
Density plot for b9 2
Density
N = 10000 Bandwidth = 0.009406
Density plot for b9 3
Density
N = 10000 Bandwidth = 0.000908
Density plot for b9 4
Density
N = 10000 Bandwidth = 0.00191
Density plot for b9 5
Density
N = 10000 Bandwidth = 0.01215
Density plot for b10 1
Density
N = 10000 Bandwidth = 0.005637
Density plot for b10 2
Density
N = 10000 Bandwidth = 0.003116
Density plot for b10 3
Density
N = 10000 Bandwidth = 0.009679
Density plot for b10 4
Density
N = 10000 Bandwidth = 0.01363



And so, this model will be:

$$ln\left(\frac{p_{i1}}{p_{i6}}\right) = -5.6 + 1.2 \cdot Z_{i2} - 0.3 \cdot Z_{i7}$$

$$ln\left(\frac{p_{i2}}{p_{i6}}\right) = -2.9 + 0.6 \cdot Z_{i2} - 0.6 \cdot Z_{i7} + 0.1 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i3}}{p_{i6}}\right) = 0.1 - 0.4 \cdot Z_{i2} - 0.4 \cdot Z_{i7} + 0.3 \cdot Z_{i10} + 0.9 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i4}}{p_{i6}}\right) = -2 - 1 \cdot Z_{i2} - 0.7 \cdot Z_{i7} + 0.7 \cdot Z_{i10} + 1.6 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i5}}{p_{i6}}\right) = -5.3 - 0.3 \cdot Z_{i2} - 0.9 \cdot Z_{i7} + 0.8 \cdot Z_{i10} + 2.3 \cdot Z_{i11}$$

Which is a more parsimonious model than the last one, as it did not contain the physiochemical characteristics of *Fixed Acidity*, *Citric Acid*, *Residual Sugar*, *Chlorides*, *Free Sulphur Dioxide*, *Density* and *pH*.

Variable Selection through g-prior on prior coefficients (using BAS package in R):

Another alternative for the choice of a prior distribution of βs when no prior information is available, is the Zellner's g-prior. Here, we will have:

$$\beta_{jk} \sim N(0, n \cdot (X^T \cdot X)^{-1} \cdot \sigma^2)$$

The algorithm uses the modified version of Liang et al. of this prior, with improper priors for the intercept of the model, as well as the precision (*tau*). Just like before, the process of variable selection will be done in the following two ways:

1. Using a Uniform Distribution on the priors:

The results here for each group, will be:

For level 1 (category 1 with reference level 6):

```
 Marginal Posterior Summaries of Coefficients:

 Using  BMA

 Based on the top  237 models
                      post mean  post SD    post p(B != 0)
Intercept             -5.523561   0.753000   1.000000
fixed.acidity          0.021996   0.157864   0.058594
volatile.acidity       1.066413   0.289899   0.994531
citric.acid            0.042204   0.208483   0.073779
residual.sugar         0.006579   0.083114   0.043604
chlorides              0.051107   0.149910   0.131201
free.sulfur.dioxide    0.056498   0.366369   0.096045
total.sulfur.dioxide  -0.813376   0.943329   0.589404
density                0.137584   0.343356   0.179639
pH                     0.019555   0.139275   0.051758
sulphates              0.002821   0.087830   0.034473
alcohol               -0.098715   0.394548   0.095264
> summary(res2.1)
                     P(B != 0 | Y)     model 1   model 2     model 3
Intercept               1.00000000   1.0000000   1.00000   1.0000000
fixed.acidity           0.05859375   0.0000000   0.00000   0.0000000
volatile.acidity        0.99453125   1.0000000   1.00000   1.0000000
citric.acid             0.07377930   0.0000000   0.00000   0.0000000
residual.sugar          0.04360352   0.0000000   0.00000   0.0000000
chlorides               0.13120117   0.0000000   0.00000   0.0000000
free.sulfur.dioxide     0.09604492   0.0000000   0.00000   0.0000000
total.sulfur.dioxide    0.58940430   0.0000000   1.00000   1.0000000
density                 0.17963867   0.0000000   0.00000   1.0000000
pH                      0.05175781   0.0000000   0.00000   0.0000000
sulphates               0.03447266   0.0000000   0.00000   0.0000000
```

```
alcohol                0.09526367   0.0000000   0.00000   0.0000000
BF                             NA   0.9791067   1.00000   0.2620167
PostProbs                      NA   0.2254000   0.22420   0.0613000
R2                             NA   0.1987000   0.26110   0.2979000
dim                            NA   2.0000000   3.00000   4.0000000
logmarg                        NA -45.3435427 -45.32243 -46.6617752
                        model 4     model 5
Intercept             1.0000000   1.0000000
fixed.acidity         0.0000000   0.0000000
volatile.acidity      1.0000000   1.0000000
citric.acid           0.0000000   0.0000000
residual.sugar        0.0000000   0.0000000
chlorides             1.0000000   0.0000000
free.sulfur.dioxide   0.0000000   0.0000000
total.sulfur.dioxide  1.0000000   0.0000000
density               0.0000000   1.0000000
pH                    0.0000000   0.0000000
sulphates             0.0000000   0.0000000
alcohol               0.0000000   0.0000000
BF                    0.1783827   0.1125876
PostProbs             0.0410000   0.0308000
R2                    0.2905000   0.2198000
dim                   4.0000000   3.0000000
logmarg             -47.0462518 -47.5064518
```

For level 2 (category 2 with reference level 6):

```
 Marginal Posterior Summaries of Coefficients:

 Using  BMA

 Based on the top  275 models
                      post mean  post SD    post p(B != 0)
Intercept             -2.887121   0.204377   1.000000
fixed.acidity         -0.005761   0.068728   0.057129
volatile.acidity       0.571058   0.154100   0.988916
citric.acid            0.004545   0.057720   0.050342
residual.sugar         0.050481   0.121995   0.183984
chlorides              0.002467   0.030742   0.040381
free.sulfur.dioxide   -0.035058   0.140407   0.100586
total.sulfur.dioxide  -0.552940   0.249282   0.909424
density               -0.032040   0.117474   0.107178
pH                     0.064811   0.147766   0.206152
sulphates              0.004552   0.041501   0.048584
alcohol                0.082297   0.180961   0.212842
> summary(res2.2)
                     P(B != 0 | Y)   model 1     model 2      model 3
Intercept               1.00000000    1.0000    1.000000    1.0000000
fixed.acidity           0.05712891    0.0000    0.000000    0.0000000
```

```
volatile.acidity        0.98891602    1.0000    1.000000    1.0000000
citric.acid             0.05034180    0.0000    0.000000    0.0000000
residual.sugar          0.18398438    0.0000    0.000000    0.0000000
chlorides               0.04038086    0.0000    0.000000    0.0000000
free.sulfur.dioxide     0.10058594    0.0000    0.000000    0.0000000
total.sulfur.dioxide    0.90942383    1.0000    1.000000    1.0000000
density                 0.10717773    0.0000    0.000000    0.0000000
pH                      0.20615234    0.0000    0.000000    1.0000000
sulphates               0.04858398    0.0000    0.000000    0.0000000
alcohol                 0.21284180    0.0000    1.000000    0.0000000
BF                             NA    1.0000    0.317415    0.2577731
PostProbs                      NA    0.3190    0.101500    0.0961000
R2                             NA    0.0961    0.107400    0.1063000
dim                            NA    3.0000    4.000000    4.0000000
logmarg                        NA -179.6678 -180.815324 -181.0234544
                         model 4       model 5
Intercept              1.0000000    1.00000000
fixed.acidity          0.0000000    0.00000000
volatile.acidity       1.0000000    1.00000000
citric.acid            0.0000000    0.00000000
residual.sugar         1.0000000    0.00000000
chlorides              0.0000000    0.00000000
free.sulfur.dioxide    0.0000000    0.00000000
total.sulfur.dioxide   1.0000000    1.00000000
density                0.0000000    1.00000000
pH                     0.0000000    0.00000000
sulphates              0.0000000    0.00000000
alcohol                0.0000000    0.00000000
BF                     0.1791273    0.08857708
PostProbs              0.0679000    0.03490000
R2                     0.1044000    0.10070000
dim                    4.0000000    4.00000000
logmarg             -181.3874374 -182.09166084
```

For level 3 (category 3 with reference level 6):

```
 Marginal Posterior Summaries of Coefficients:

 Using  BMA

 Based on the top  66 models
                     post mean   post SD     post p(B != 0)
Intercept             0.1443183   0.0659369   1.0000000
fixed.acidity         0.0056524   0.0359002   0.0476562
volatile.acidity     -0.4062907   0.0867942   0.9993652
citric.acid          -0.0330804   0.0850380   0.1722168
residual.sugar        0.0011866   0.0130161   0.0263672
chlorides            -0.0427213   0.0784243   0.2743652
free.sulfur.dioxide   0.2228540   0.1371573   0.7988770
```

```
total.sulfur.dioxide  -0.5970115   0.1282187   0.9991699
density                0.0010487   0.0177876   0.0345215
pH                    -0.0003257   0.0125768   0.0264648
sulphates              0.3512195   0.0755211   0.9992676
alcohol                0.8273962   0.0845784   0.9999023
> summary(res2.3)
                    P(B != 0 | Y)   model 1      model 2      model 3
Intercept              1.00000000    1.0000    1.0000000    1.0000000
fixed.acidity          0.04765625    0.0000    0.0000000    0.0000000
volatile.acidity       0.99936523    1.0000    1.0000000    1.0000000
citric.acid            0.17221680    0.0000    0.0000000    0.0000000
residual.sugar         0.02636719    0.0000    0.0000000    0.0000000
chlorides              0.27436523    0.0000    1.0000000    0.0000000
free.sulfur.dioxide    0.79887695    1.0000    1.0000000    0.0000000
total.sulfur.dioxide   0.99916992    1.0000    1.0000000    1.0000000
density                0.03452148    0.0000    0.0000000    0.0000000
pH                     0.02646484    0.0000    0.0000000    0.0000000
sulphates              0.99926758    1.0000    1.0000000    1.0000000
alcohol                0.99990234    1.0000    1.0000000    1.0000000
BF                             NA    1.0000    0.4048037    0.1763485
PostProbs                      NA    0.4276    0.1916000    0.0831000
R2                             NA    0.1784    0.1814000    0.1726000
dim                            NA    6.0000    7.0000000    5.0000000
logmarg                        NA -770.4470 -771.3513829 -772.1823228
                       model 4      model 5
Intercept              1.0000000    1.0000000
fixed.acidity          0.0000000    0.0000000
volatile.acidity       1.0000000    1.0000000
citric.acid            1.0000000    1.0000000
residual.sugar         0.0000000    0.0000000
chlorides              0.0000000    0.0000000
free.sulfur.dioxide    1.0000000    0.0000000
total.sulfur.dioxide   1.0000000    1.0000000
density                0.0000000    0.0000000
pH                     0.0000000    0.0000000
sulphates              1.0000000    1.0000000
alcohol                1.0000000    1.0000000
BF                     0.1416914    0.1079878
PostProbs              0.0723000    0.0433000
R2                     0.1802000    0.1760000
dim                    7.0000000    6.0000000
logmarg             -772.4011336 -772.6727669
```

For level 4 (category 4 with reference level 6):

```
 Marginal Posterior Summaries of Coefficients:

 Using  BMA
```

```
 Based on the top  106 models
                     post mean  post SD    post p(B != 0)
Intercept            -1.938019   0.156460   1.000000
fixed.acidity         0.056255   0.132291   0.206592
volatile.acidity     -0.980949   0.163226   0.999170
citric.acid          -0.006592   0.067517   0.050146
residual.sugar        0.018000   0.068111   0.094287
chlorides            -0.126321   0.194024   0.373975
free.sulfur.dioxide   0.117227   0.209602   0.288721
total.sulfur.dioxide -0.593707   0.219810   0.995605
density              -0.007231   0.076584   0.063037
pH                   -0.031068   0.094747   0.134766
sulphates             0.646540   0.129095   0.999805
alcohol               1.748001   0.164713   0.999658
> summary(res2.4)
                    P(B != 0 | Y)   model 1      model 2      model 3
Intercept              1.00000000    1.0000    1.0000000    1.0000000
fixed.acidity          0.20659180    0.0000    0.0000000    0.0000000
volatile.acidity       0.99916992    1.0000    1.0000000    1.0000000
citric.acid            0.05014648    0.0000    0.0000000    0.0000000
residual.sugar         0.09428711    0.0000    0.0000000    0.0000000
chlorides              0.37397461    0.0000    1.0000000    0.0000000
free.sulfur.dioxide    0.28872070    0.0000    0.0000000    1.0000000
total.sulfur.dioxide   0.99560547    1.0000    1.0000000    1.0000000
density                0.06303711    0.0000    0.0000000    0.0000000
pH                     0.13476562    0.0000    0.0000000    0.0000000
sulphates              0.99980469    1.0000    1.0000000    1.0000000
alcohol                0.99965820    1.0000    1.0000000    1.0000000
BF                             NA    1.0000    0.5316337    0.3590766
PostProbs                      NA    0.2611    0.1467000    0.0971000
R2                             NA    0.5192    0.5250000    0.5242000
dim                            NA    5.0000    6.0000000    6.0000000
logmarg                        NA -241.0493 -241.6810716 -242.0734905
                       model 4      model 5
Intercept            1.0000000    1.0000000
fixed.acidity        1.0000000    0.0000000
volatile.acidity     1.0000000    1.0000000
citric.acid          0.0000000    0.0000000
residual.sugar       0.0000000    0.0000000
chlorides            0.0000000    1.0000000
free.sulfur.dioxide  0.0000000    1.0000000
total.sulfur.dioxide 1.0000000    1.0000000
density              0.0000000    0.0000000
pH                   0.0000000    0.0000000
sulphates            1.0000000    1.0000000
alcohol              1.0000000    1.0000000
BF                   0.2516325    0.1408933
PostProbs            0.0623000    0.0441000
```

```
R2                      0.5235000    0.5294000
dim                     6.0000000    7.0000000
logmarg              -242.4290568 -243.0090236
```

For level 5 (category 5 with reference level 6):

```
 Marginal Posterior Summaries of Coefficients:

 Using  BMA

 Based on the top  405 models
                      post mean  post SD   post p(B != 0)
Intercept             -6.89717    1.48251   1.00000
fixed.acidity         -0.05734    0.32288   0.07422
volatile.acidity      -0.36479    0.62581   0.30776
citric.acid            1.04250    0.97959   0.63057
residual.sugar        -1.17978    1.11055   0.62251
chlorides             -1.22246    1.51072   0.52515
free.sulfur.dioxide    0.02373    0.21598   0.06611
total.sulfur.dioxide  -1.56124    0.87070   0.87734
density               -0.34645    0.75974   0.23804
pH                    -0.22178    0.49156   0.22397
sulphates              1.25975    0.55290   0.92310
alcohol                2.46388    0.87950   0.99824
> summary(res2.5)
                     P(B != 0 | Y)   model 1     model 2     model 3
Intercept               1.00000000   1.00000   1.0000000   1.0000000
fixed.acidity           0.07421875   0.00000   0.0000000   0.0000000
volatile.acidity        0.30776367   0.00000   0.0000000   0.0000000
citric.acid             0.63056641   1.00000   1.0000000   1.0000000
residual.sugar          0.62250977   1.00000   0.0000000   1.0000000
chlorides               0.52514648   1.00000   1.0000000   0.0000000
free.sulfur.dioxide     0.06611328   0.00000   0.0000000   0.0000000
total.sulfur.dioxide    0.87734375   1.00000   1.0000000   1.0000000
density                 0.23803711   0.00000   1.0000000   0.0000000
pH                      0.22397461   0.00000   0.0000000   0.0000000
sulphates               0.92309570   1.00000   1.0000000   1.0000000
alcohol                 0.99824219   1.00000   1.0000000   1.0000000
BF                             NA   1.00000   0.7499369   0.8118072
PostProbs                      NA   0.13470   0.1025000   0.0986000
R2                             NA   0.66940   0.6660000   0.6282000
dim                            NA   7.00000   7.0000000   6.0000000
logmarg                        NA -47.45245 -47.7402123 -47.6609386
                        model 4     model 5
Intercept              1.0000000   1.0000000
fixed.acidity          0.0000000   0.0000000
volatile.acidity       1.0000000   1.0000000
citric.acid            0.0000000   0.0000000
residual.sugar         1.0000000   1.0000000
```

```
chlorides              0.0000000   0.0000000
free.sulfur.dioxide    0.0000000   0.0000000
total.sulfur.dioxide   1.0000000   1.0000000
density                0.0000000   0.0000000
pH                     0.0000000   1.0000000
sulphates              1.0000000   1.0000000
alcohol                1.0000000   1.0000000
BF                     0.7928453   0.3153408
PostProbs              0.0940000   0.0375000
R2                     0.6280000   0.6557000
dim                    6.0000000   7.0000000
logmarg              -47.6845734 -48.6065476
```

Thus, if we select the best model of them all, which has BF equal to 1, then we would get the following model:

$$ln\left(\frac{p_{i1}}{p_{i6}}\right) = -5.5 + 1.07 \cdot Z_{i2} - 0.81 \cdot Z_{i7}$$

$$ln\left(\frac{p_{i2}}{p_{i6}}\right) = -2.89 + 0.57 \cdot Z_{i2} - 0.55 \cdot Z_{i7}$$

$$ln\left(\frac{p_{i3}}{p_{i6}}\right) = 0.14 - 0.41 \cdot Z_{i2} + 0.22 \cdot Z_{i6} - 0.59 \cdot Z_{i7} + 0.35 \cdot Z_{i10} + 0.83 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i4}}{p_{i6}}\right) = -1.94 - 0.98 \cdot Z_{i2} - 0.59 \cdot Z_{i7} + 0.65 \cdot Z_{i10} + 1.75 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i5}}{p_{i6}}\right) = -6.9 + 1.04 \cdot Z_{i3} - 1.18 \cdot Z_{i4} - 1.22 \cdot Z_{i5} - 0.88 \cdot Z_{i7} + 1.26 \cdot Z_{i10} + 2.46 \cdot Z_{i11}$$

This method did not use the physiochemical characteristics of *Fixed Acidity*, *Density* and *pH*.

2. Using a Beta-Binomial Distribution on the priors:

Here we will have the following results for each group:

For level 1 (category 1 with reference level 6):

```
 Marginal Posterior Summaries of Coefficients:

 Using  BMA

 Based on the top  90 models
                     post mean   post SD     post p(B != 0)
Intercept            -5.2720092   0.5508172   1.0000000
fixed.acidity         0.0039901   0.0587205   0.0102051
```

```
volatile.acidity       1.0594639   0.2660126   0.9891602
citric.acid            0.0042257   0.0656519   0.0103027
residual.sugar         0.0013579   0.0346564   0.0093262
chlorides              0.0080483   0.0579613   0.0245605
free.sulfur.dioxide   -0.0023368   0.1130868   0.0171387
total.sulfur.dioxide  -0.1798177   0.5021136   0.1481934
density                0.0161466   0.1173092   0.0254395
pH                     0.0048286   0.0631537   0.0126465
sulphates              0.0006452   0.0339597   0.0064941
alcohol               -0.0123864   0.1397073   0.0154785
> summary(res2.1)
                    P(B != 0 | Y)     model 1   model 2     model 3
Intercept              1.000000000   1.0000000   1.00000   1.0000000
fixed.acidity          0.010205078   0.0000000   0.00000   0.0000000
volatile.acidity       0.989160156   1.0000000   1.00000   1.0000000
citric.acid            0.010302734   0.0000000   0.00000   0.0000000
residual.sugar         0.009326172   0.0000000   0.00000   0.0000000
chlorides              0.024560547   0.0000000   0.00000   1.0000000
free.sulfur.dioxide    0.017138672   0.0000000   0.00000   0.0000000
total.sulfur.dioxide   0.148193359   0.0000000   1.00000   0.0000000
density                0.025439453   0.0000000   0.00000   0.0000000
pH                     0.012646484   0.0000000   0.00000   0.0000000
sulphates              0.006494141   0.0000000   0.00000   0.0000000
alcohol                0.015478516   0.0000000   0.00000   0.0000000
BF                              NA   0.9791067   1.00000   0.1098552
PostProbs                       NA   0.7650000   0.11090   0.0152000
R2                              NA   0.1987000   0.26110   0.2193000
dim                             NA   2.0000000   3.00000   3.0000000
logmarg                         NA -45.3435427 -45.32243 -47.5310203
                      model 4      model 5
Intercept             1.0000000   1.00000000
fixed.acidity         0.0000000   0.00000000
volatile.acidity      1.0000000   1.00000000
citric.acid           0.0000000   0.00000000
residual.sugar        0.0000000   0.00000000
chlorides             0.0000000   0.00000000
free.sulfur.dioxide   0.0000000   1.00000000
total.sulfur.dioxide  0.0000000   0.00000000
density               1.0000000   0.00000000
pH                    0.0000000   0.00000000
sulphates             0.0000000   0.00000000
alcohol               0.0000000   0.00000000
BF                    0.1125876   0.08361466
PostProbs             0.0146000   0.00960000
R2                    0.2198000   0.21400000
dim                   3.0000000   3.00000000
logmarg             -47.5064518 -47.80396441
```

For level 2 (category 2 with reference level 6):

```
 Marginal Posterior Summaries of Coefficients:

 Using  BMA

 Based on the top  91 models
                      post mean   post SD     post p(B != 0)
Intercept             -2.8981454   0.1999721   1.0000000
fixed.acidity         -0.0022860   0.0319280   0.0129883
volatile.acidity       0.5756445   0.1649886   0.9758301
citric.acid            0.0003056   0.0221035   0.0083496
residual.sugar         0.0119482   0.0617194   0.0453125
chlorides              0.0003418   0.0128964   0.0082031
free.sulfur.dioxide   -0.0211946   0.1094167   0.0468750
total.sulfur.dioxide  -0.5478193   0.2597361   0.8856934
density               -0.0070800   0.0555156   0.0235840
pH                     0.0247246   0.0973603   0.0745605
sulphates              0.0007053   0.0170610   0.0093262
alcohol                0.0324694   0.1238535   0.0768066
> summary(res2.2)
                    P(B != 0 | Y)   model 1     model 2       model 3
Intercept              1.000000000    1.0000    1.000000  1.000000e+00
fixed.acidity          0.012988281    0.0000    0.000000  0.000000e+00
volatile.acidity       0.975830078    1.0000    1.000000  1.000000e+00
citric.acid            0.008349609    0.0000    0.000000  0.000000e+00
residual.sugar         0.045312500    0.0000    0.000000  0.000000e+00
chlorides              0.008203125    0.0000    0.000000  0.000000e+00
free.sulfur.dioxide    0.046875000    0.0000    0.000000  0.000000e+00
total.sulfur.dioxide   0.885693359    1.0000    1.000000  0.000000e+00
density                0.023583984    0.0000    0.000000  0.000000e+00
pH                     0.074560547    0.0000    0.000000  0.000000e+00
sulphates              0.009326172    0.0000    0.000000  0.000000e+00
alcohol                0.076806641    0.0000    1.000000  0.000000e+00
BF                              NA    1.0000    0.317415  9.500522e-03
PostProbs                       NA    0.6874    0.046600  4.180000e-02
R2                              NA    0.0961    0.107400  5.430000e-02
dim                             NA    3.0000    4.000000  2.000000e+00
logmarg                         NA -179.6678 -180.815324 -1.843242e+02
                       model 4      model 5
Intercept             1.0000000    1.0000000
fixed.acidity         0.0000000    0.0000000
volatile.acidity      1.0000000    1.0000000
citric.acid           0.0000000    0.0000000
residual.sugar        0.0000000    1.0000000
chlorides             0.0000000    0.0000000
free.sulfur.dioxide   0.0000000    0.0000000
total.sulfur.dioxide  1.0000000    1.0000000
density               0.0000000    0.0000000
```

```
pH                      1.0000000    0.0000000
sulphates               0.0000000    0.0000000
alcohol                 0.0000000    0.0000000
BF                      0.2577731    0.1791273
PostProbs               0.0416000    0.0270000
R2                      0.1063000    0.1044000
dim                     4.0000000    4.0000000
logmarg              -181.0234544 -181.3874374
```

For level 3 (category 3 with reference level 6):

```
 Marginal Posterior Summaries of Coefficients:

 Using  BMA

 Based on the top  58 models
                      post mean   post SD     post p(B != 0)
Intercept              1.466e-01   6.601e-02   1.000e+00
fixed.acidity          1.770e-03   2.070e-02   2.153e-02
volatile.acidity      -4.055e-01   8.240e-02   9.996e-01
citric.acid           -2.212e-02   6.921e-02   1.188e-01
residual.sugar         4.647e-04   8.390e-03   9.521e-03
chlorides             -2.279e-02   6.124e-02   1.459e-01
free.sulfur.dioxide    1.772e-01   1.528e-01   6.308e-01
total.sulfur.dioxide  -5.631e-01   1.368e-01   9.990e-01
density                5.051e-04   1.500e-02   1.621e-02
pH                     6.238e-05   9.232e-03   1.377e-02
sulphates              3.388e-01   7.159e-02   1.000e+00
alcohol                8.361e-01   8.533e-02   9.995e-01
> summary(res2.3)
                     P(B != 0 | Y)   model 1      model 2      model 3
Intercept              1.000000000    1.0000    1.0000000    1.0000000
fixed.acidity          0.021533203    0.0000    0.0000000    0.0000000
volatile.acidity       0.999609375    1.0000    1.0000000    1.0000000
citric.acid            0.118798828    0.0000    0.0000000    0.0000000
residual.sugar         0.009521484    0.0000    0.0000000    0.0000000
chlorides              0.145898438    0.0000    0.0000000    1.0000000
free.sulfur.dioxide    0.630810547    1.0000    0.0000000    1.0000000
total.sulfur.dioxide   0.999023438    1.0000    1.0000000    1.0000000
density                0.016210938    0.0000    0.0000000    0.0000000
pH                     0.013769531    0.0000    0.0000000    0.0000000
sulphates              1.000000000    1.0000    1.0000000    1.0000000
alcohol                0.999462891    1.0000    1.0000000    1.0000000
BF                              NA    1.0000    0.1763485    0.4048037
PostProbs                       NA    0.4766    0.2392000    0.0753000
R2                              NA    0.1784    0.1726000    0.1814000
dim                             NA    6.0000    5.0000000    7.0000000
logmarg                         NA -770.4470 -772.1823228 -771.3513829
                         model 4       model 5
```

```
Intercept                1.0000000    1.00000000
fixed.acidity            0.0000000    0.00000000
volatile.acidity         1.0000000    1.00000000
citric.acid              1.0000000    0.00000000
residual.sugar           0.0000000    0.00000000
chlorides                0.0000000    1.00000000
free.sulfur.dioxide      0.0000000    0.00000000
total.sulfur.dioxide     1.0000000    1.00000000
density                  0.0000000    0.00000000
pH                       0.0000000    0.00000000
sulphates                1.0000000    1.00000000
alcohol                  1.0000000    1.00000000
BF                       0.1079878    0.08741685
PostProbs                0.0520000    0.04740000
R2                       0.1760000    0.17580000
dim                      6.0000000    6.00000000
logmarg               -772.6727669 -772.88409712
```

For level 4 (category 4 with reference level 6):

```
 Marginal Posterior Summaries of Coefficients:

 Using  BMA

 Based on the top  75 models
                      post mean  post SD    post p(B != 0)
Intercept             -1.928137   0.154825   1.000000
fixed.acidity          0.033836   0.103107   0.128760
volatile.acidity      -1.001509   0.156026   1.000000
citric.acid           -0.003752   0.049809   0.026221
residual.sugar         0.007770   0.045452   0.041504
chlorides             -0.054911   0.141177   0.164941
free.sulfur.dioxide    0.059618   0.159883   0.148486
total.sulfur.dioxide  -0.539286   0.194703   0.985303
density               -0.003803   0.051549   0.023779
pH                    -0.012482   0.060765   0.057129
sulphates              0.623905   0.121529   0.999463
alcohol                1.760783   0.158483   0.999756
> summary(res2.4)
                     P(B != 0 | Y)   model 1      model 2      model 3
Intercept               1.00000000    1.0000    1.0000000    1.0000000
fixed.acidity           0.12875977    0.0000    0.0000000    0.0000000
volatile.acidity        1.00000000    1.0000    1.0000000    1.0000000
citric.acid             0.02622070    0.0000    0.0000000    0.0000000
residual.sugar          0.04150391    0.0000    0.0000000    0.0000000
chlorides               0.16494141    0.0000    1.0000000    0.0000000
free.sulfur.dioxide     0.14848633    0.0000    0.0000000    1.0000000
total.sulfur.dioxide    0.98530273    1.0000    1.0000000    1.0000000
density                 0.02377930    0.0000    0.0000000    0.0000000
```

```
pH                      0.05712891    0.0000    0.0000000    0.0000000
sulphates               0.99946289    1.0000    1.0000000    1.0000000
alcohol                 0.99975586    1.0000    1.0000000    1.0000000
BF                             NA    1.0000    0.5316337    0.3590766
PostProbs                      NA    0.5545    0.0997000    0.0838000
R2                             NA    0.5192    0.5250000    0.5242000
dim                            NA    5.0000    6.0000000    6.0000000
logmarg                        NA -241.0493 -241.6810716 -242.0734905
                         model 4      model 5
Intercept              1.0000000    1.0000000
fixed.acidity          1.0000000    0.0000000
volatile.acidity       1.0000000    1.0000000
citric.acid            0.0000000    0.0000000
residual.sugar         0.0000000    0.0000000
chlorides              0.0000000    0.0000000
free.sulfur.dioxide    0.0000000    0.0000000
total.sulfur.dioxide   1.0000000    1.0000000
density                0.0000000    0.0000000
pH                     0.0000000    1.0000000
sulphates              1.0000000    1.0000000
alcohol                1.0000000    1.0000000
BF                     0.2516325    0.1179672
PostProbs              0.0627000    0.0241000
R2                     0.5235000    0.5218000
dim                    6.0000000    6.0000000
logmarg             -242.4290568 -243.1866198
```

For level 5 (category 5 with reference level 6):

```
 Marginal Posterior Summaries of Coefficients:

 Using  BMA

 Based on the top  303 models
                       post mean   post SD     post p(B != 0)
Intercept              -5.7972473   1.4221738   1.0000000
fixed.acidity          -0.0429014   0.2731041   0.0518066
volatile.acidity       -0.5973103   0.7091462   0.4698730
citric.acid             0.5579142   0.8388904   0.3642090
residual.sugar         -0.6606628   0.9853124   0.3676270
chlorides              -0.6449986   1.2808107   0.2627441
free.sulfur.dioxide    -0.0001341   0.1298969   0.0284668
total.sulfur.dioxide   -0.9263280   0.9597479   0.5592773
density                -0.1727900   0.5335677   0.1228027
pH                     -0.1626270   0.4346457   0.1562988
sulphates               0.7545693   0.6846952   0.6170898
alcohol                 2.1752184   0.7135057   0.9988281
> summary(res2.5)
                    P(B != 0 | Y)      model 1     model 2      model 3
```

```
Intercept              1.00000000   1.00000000   1.0000000   1.00000000
fixed.acidity          0.05180664   0.00000000   0.0000000   0.00000000
volatile.acidity       0.46987305   1.00000000   1.0000000   0.00000000
citric.acid            0.36420898   0.00000000   0.0000000   0.00000000
residual.sugar         0.36762695   0.00000000   1.0000000   0.00000000
chlorides              0.26274414   0.00000000   0.0000000   0.00000000
free.sulfur.dioxide    0.02846680   0.00000000   0.0000000   0.00000000
total.sulfur.dioxide   0.55927734   0.00000000   1.0000000   1.00000000
density                0.12280273   0.00000000   0.0000000   0.00000000
pH                     0.15629883   0.00000000   0.0000000   0.00000000
sulphates              0.61708984   0.00000000   1.0000000   1.00000000
alcohol                0.99882812   1.00000000   1.0000000   1.00000000
BF                             NA   0.04219674   0.9766424   0.06508855
PostProbs                      NA   0.15410000   0.0703000   0.06050000
R2                             NA   0.47450000   0.6280000   0.51840000
dim                            NA   3.00000000   6.0000000   4.00000000
logmarg                        NA -50.82635091 -47.6845734 -50.39294531
                        model 4      model 5
Intercept                1.00000   1.0000000
fixed.acidity            0.00000   0.0000000
volatile.acidity         0.00000   1.0000000
citric.acid              1.00000   0.0000000
residual.sugar           1.00000   0.0000000
chlorides                0.00000   0.0000000
free.sulfur.dioxide      0.00000   0.0000000
total.sulfur.dioxide     1.00000   1.0000000
density                  0.00000   0.0000000
pH                       0.00000   0.0000000
sulphates                1.00000   1.0000000
alcohol                  1.00000   1.0000000
BF                       1.00000   0.2707585
PostProbs                0.05350   0.0501000
R2                       0.62820   0.5740000
dim                      6.00000   5.0000000
logmarg                -47.66094 -48.9674665
```

And so, the model this time will be:

$$ln\left(\frac{p_{i1}}{p_{i6}}\right) = -5.27 + 1.06 \cdot Z_{i2} - 0.17 \cdot Z_{i7}$$

$$ln\left(\frac{p_{i2}}{p_{i6}}\right) = -2.9 + 0.58 \cdot Z_{i2} - 0.55 \cdot Z_{i7}$$

$$ln\left(\frac{p_{i3}}{p_{i6}}\right) = 0.15 - 0.41 \cdot Z_{i2} + 1.78 \cdot Z_{i6} - 0.56 \cdot Z_{i7} + 0.34 \cdot Z_{i10} + 0.84 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i4}}{p_{i6}}\right) = -1.93 - 1 \cdot Z_{i2} - 0.54 \cdot Z_{i7} + 0.62 \cdot Z_{i10} + 1.76 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i5}}{p_{i6}}\right) = -5.8 + 0.55 \cdot Z_{i3} - 0.66 \cdot Z_{i4} - 0.93 \cdot Z_{i7} + 0.75 \cdot Z_{i10} + 2.18 \cdot Z_{i11}$$

Looking at the model above, we see that we did not use the physiochemical characteristics of *Fixed Acidity, chlorides, pH* and *Density* at all.

Variable Selection through (Liang et al.) g-prior (using OpenBUGS):

This will be the same method as before, only this time it will be implemented in OpenBUGS. Once more we will use the version of Liang et al. Here:

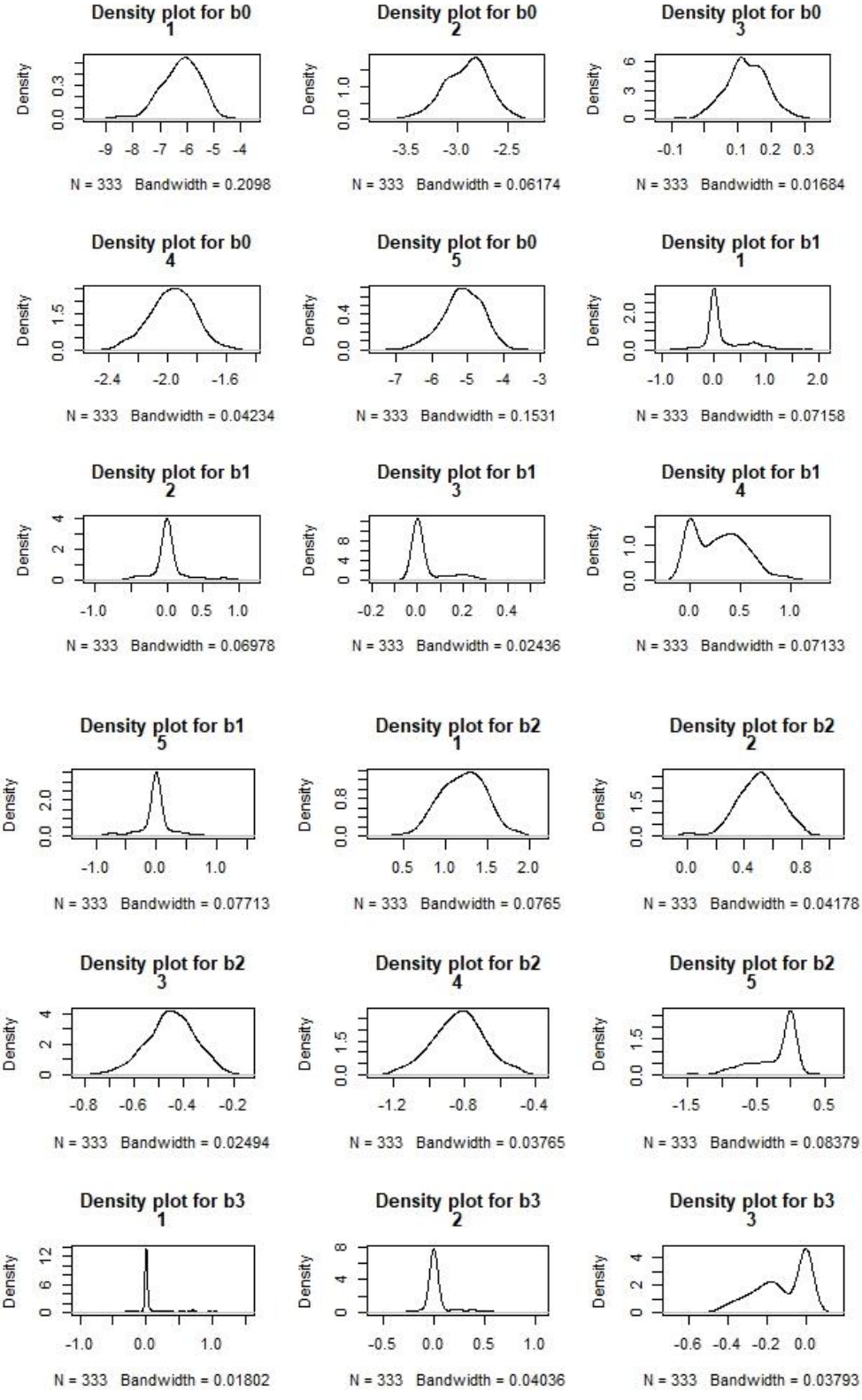
$$G \leftarrow (n = 1599)$$

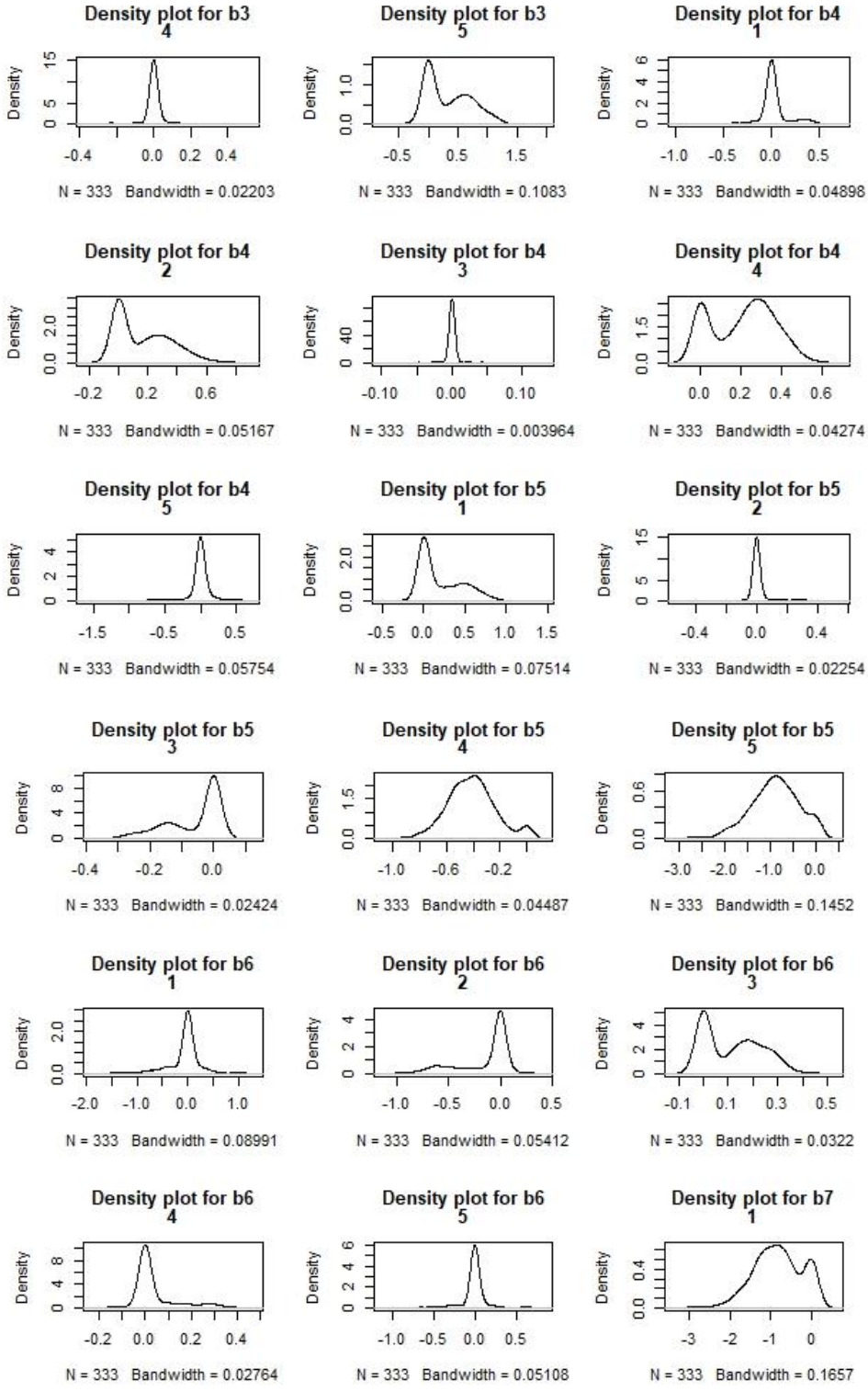
This method should produce similar results to the previous one, the only minor differences, should be because it uses MCMC. We will repeat the procedure by:

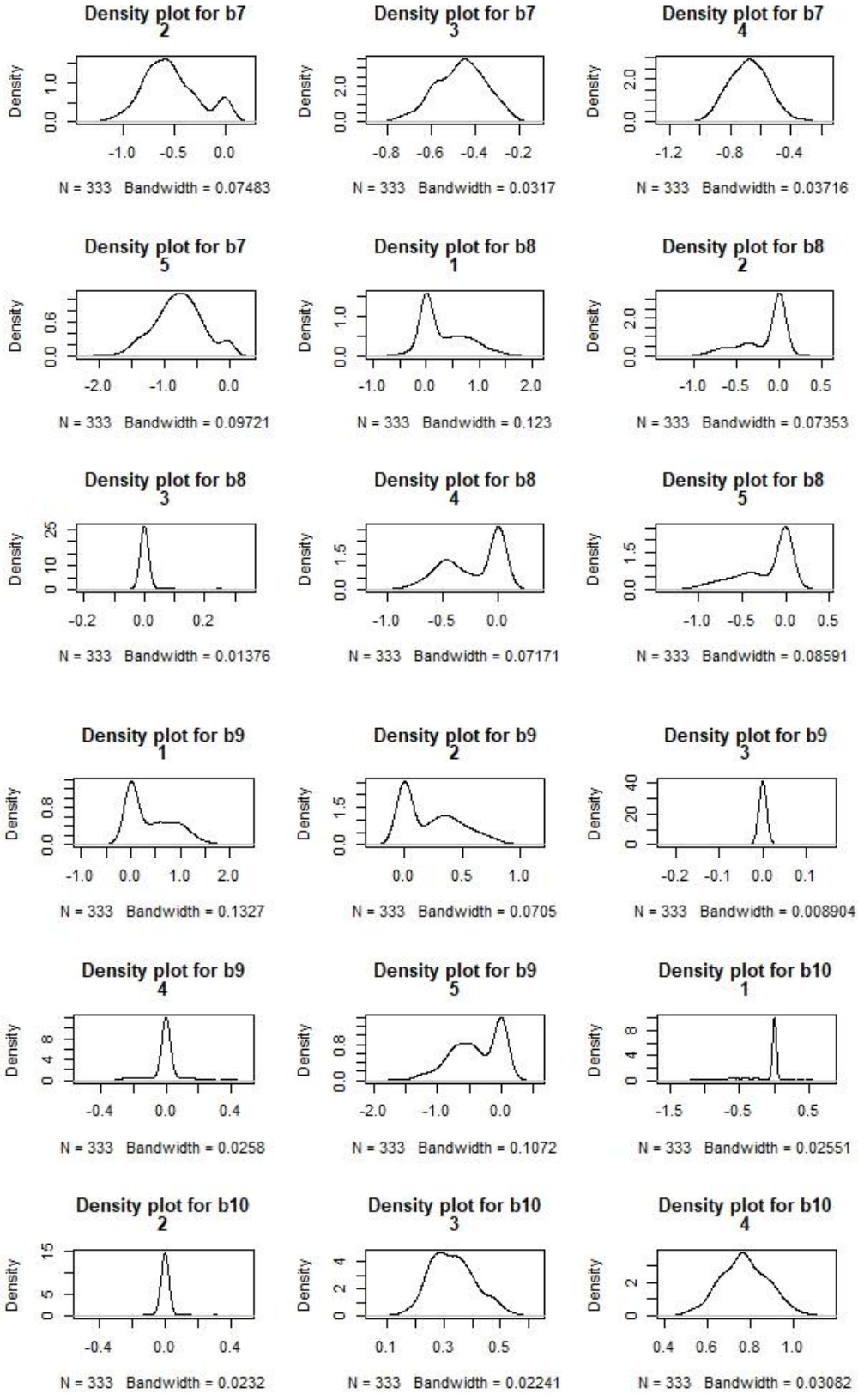
1. Using a Uniform Distribution on the priors:

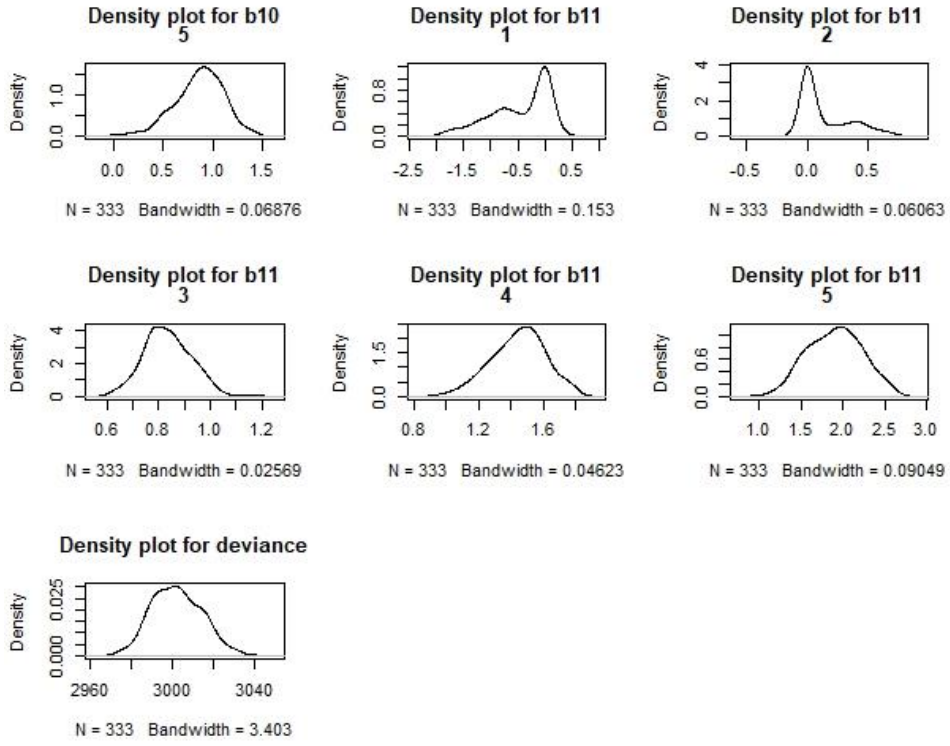
```
Inference for Bugs model at "C:\Users\30697\Desktop\gprior.txt", fit
using OpenBUGS,
 1 chains, each with 11000 iterations (first 1000 discarded), n.thin =
30
 n.sims = 333 iterations saved
          mean   sd   2.5%    25%    50%    75%  97.5%
alpha[1]   -6.2  0.8   -8.1   -6.7   -6.2   -5.7   -5.1
alpha[2]   -2.9  0.2   -3.3   -3.1   -2.9   -2.8   -2.5
alpha[3]    0.1  0.1    0.0    0.1    0.1    0.2    0.3
alpha[4]   -2.0  0.2   -2.3   -2.1   -2.0   -1.9   -1.7
alpha[5]   -5.1  0.6   -6.3   -5.5   -5.1   -4.7   -4.1
gb1[1]      0.2  0.4   -0.4    0.0    0.0    0.3    1.3
gb1[2]      0.0  0.2   -0.5    0.0    0.0    0.0    0.8
gb1[3]      0.0  0.1    0.0    0.0    0.0    0.0    0.3
gb1[4]      0.3  0.3    0.0    0.0    0.3    0.5    0.9
gb1[5]      0.0  0.3   -0.7    0.0    0.0    0.0    0.6
gb2[1]      1.2  0.3    0.7    1.0    1.2    1.4    1.7
gb2[2]      0.5  0.2    0.2    0.4    0.5    0.6    0.8
gb2[3]     -0.5  0.1   -0.6   -0.5   -0.4   -0.4   -0.3
gb2[4]     -0.8  0.1   -1.1   -0.9   -0.8   -0.7   -0.5
gb2[5]     -0.2  0.3   -1.0   -0.4    0.0    0.0    0.0
gb3[1]      0.1  0.3   -0.4    0.0    0.0    0.1    1.0
gb3[2]      0.0  0.1   -0.2    0.0    0.0    0.0    0.5
gb3[3]     -0.1  0.1   -0.4   -0.2   -0.1    0.0    0.0
gb3[4]      0.0  0.1   -0.2    0.0    0.0    0.0    0.2
gb3[5]      0.3  0.4   -0.1    0.0    0.2    0.7    1.1
```

```
gb4[1]      0.0  0.2   -0.4    0.0    0.0    0.0    0.4
gb4[2]      0.2  0.2    0.0    0.0    0.1    0.3    0.6
gb4[3]      0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb4[4]      0.2  0.2    0.0    0.0    0.2    0.3    0.5
gb4[5]      0.0  0.2   -0.6    0.0    0.0    0.0    0.3
gb5[1]      0.2  0.3    0.0    0.0    0.0    0.4    0.8
gb5[2]      0.0  0.1   -0.1    0.0    0.0    0.0    0.3
gb5[3]     -0.1  0.1   -0.3   -0.1    0.0    0.0    0.0
gb5[4]     -0.4  0.2   -0.8   -0.5   -0.4   -0.3    0.0
gb5[5]     -0.9  0.5   -2.0   -1.2   -0.9   -0.5    0.0
gb6[1]     -0.1  0.3   -0.9    0.0    0.0    0.0    0.5
gb6[2]     -0.1  0.3   -0.8   -0.3    0.0    0.0    0.1
gb6[3]      0.1  0.1    0.0    0.0    0.1    0.2    0.3
gb6[4]      0.0  0.1   -0.1    0.0    0.0    0.0    0.3
gb6[5]      0.0  0.2   -0.5    0.0    0.0    0.0    0.3
gb7[1]     -0.8  0.6   -2.0   -1.2   -0.8   -0.4    0.0
gb7[2]     -0.5  0.3   -1.0   -0.7   -0.6   -0.4    0.0
gb7[3]     -0.5  0.1   -0.7   -0.5   -0.5   -0.4   -0.3
gb7[4]     -0.7  0.1   -0.9   -0.8   -0.7   -0.6   -0.4
gb7[5]     -0.8  0.4   -1.5   -1.0   -0.8   -0.5    0.0
gb8[1]      0.3  0.4   -0.3    0.0    0.0    0.6    1.3
gb8[2]     -0.2  0.3   -0.8   -0.3    0.0    0.0    0.1
gb8[3]      0.0  0.0    0.0    0.0    0.0    0.0    0.2
gb8[4]     -0.2  0.3   -0.7   -0.5   -0.2    0.0    0.0
gb8[5]     -0.2  0.3   -0.9   -0.4    0.0    0.0    0.0
gb9[1]      0.4  0.5   -0.1    0.0    0.2    0.8    1.4
gb9[2]      0.2  0.3    0.0    0.0    0.2    0.4    0.8
gb9[3]      0.0  0.0   -0.1    0.0    0.0    0.0    0.0
gb9[4]      0.0  0.1   -0.2    0.0    0.0    0.0    0.2
gb9[5]     -0.4  0.4   -1.2   -0.7   -0.4    0.0    0.0
gb10[1]    -0.1  0.3   -1.1   -0.1    0.0    0.0    0.3
gb10[2]     0.0  0.1   -0.1    0.0    0.0    0.0    0.3
gb10[3]     0.3  0.1    0.2    0.3    0.3    0.4    0.5
gb10[4]     0.8  0.1    0.5    0.7    0.8    0.8    1.0
gb10[5]     0.9  0.3    0.3    0.7    0.9    1.0    1.3
gb11[1]    -0.5  0.5   -1.7   -0.8   -0.3    0.0    0.1
gb11[2]     0.1  0.2   -0.1    0.0    0.0    0.3    0.7
gb11[3]     0.8  0.1    0.7    0.8    0.8    0.9    1.0
gb11[4]     1.4  0.2    1.1    1.3    1.5    1.6    1.8
gb11[5]     1.9  0.3    1.3    1.7    1.9    2.1    2.5
deviance 3002.2 12.1 2980.8 2993.1 3001.7 3011.3 3026.2

DIC info (using the rule, pD = var(deviance)/2)
pD = 73.0 and DIC = 3075.2
DIC is an estimate of expected predictive error (lower deviance is
better).
```


Density plot for b0 1
Density
N = 333 Bandwidth = 0.2098
Density plot for b0 2
Density
N = 333 Bandwidth = 0.06174
Density plot for b0 3
Density
N = 333 Bandwidth = 0.01684
Density plot for b0 4
Density
N = 333 Bandwidth = 0.04234
Density plot for b0 5
Density
N = 333 Bandwidth = 0.1531
Density plot for b1 1
Density
N = 333 Bandwidth = 0.07158
Density plot for b1 2
Density
N = 333 Bandwidth = 0.06978
Density plot for b1 3
Density
N = 333 Bandwidth = 0.02436
Density plot for b1 4
Density
N = 333 Bandwidth = 0.07133
Density plot for b1 5
Density
N = 333 Bandwidth = 0.07713
Density plot for b2 1
Density
N = 333 Bandwidth = 0.0765
Density plot for b2 2
Density
N = 333 Bandwidth = 0.04178
Density plot for b2 3
Density
N = 333 Bandwidth = 0.02494
Density plot for b2 4
Density
N = 333 Bandwidth = 0.03765
Density plot for b2 5
Density
N = 333 Bandwidth = 0.08379
Density plot for b3 1
Density
N = 333 Bandwidth = 0.01802
Density plot for b3 2
Density
N = 333 Bandwidth = 0.04036
Density plot for b3 3
Density
N = 333 Bandwidth = 0.03793


Density plot for b3 4
Density
N = 333 Bandwidth = 0.02203
Density plot for b3 5
Density
N = 333 Bandwidth = 0.1083
Density plot for b4 1
Density
N = 333 Bandwidth = 0.04898
Density plot for b4 2
Density
N = 333 Bandwidth = 0.05167
Density plot for b4 3
Density
N = 333 Bandwidth = 0.003964
Density plot for b4 4
Density
N = 333 Bandwidth = 0.04274
Density plot for b4 5
Density
N = 333 Bandwidth = 0.05754
Density plot for b5 1
Density
N = 333 Bandwidth = 0.07514
Density plot for b5 2
Density
N = 333 Bandwidth = 0.02254
Density plot for b5 3
Density
N = 333 Bandwidth = 0.02424
Density plot for b5 4
Density
N = 333 Bandwidth = 0.04487
Density plot for b5 5
Density
N = 333 Bandwidth = 0.1452
Density plot for b6 1
Density
N = 333 Bandwidth = 0.08991
Density plot for b6 2
Density
N = 333 Bandwidth = 0.05412
Density plot for b6 3
Density
N = 333 Bandwidth = 0.0322
Density plot for b6 4
Density
N = 333 Bandwidth = 0.02764
Density plot for b6 5
Density
N = 333 Bandwidth = 0.05108
Density plot for b7 1
Density
N = 333 Bandwidth = 0.1657


Density plot for b7 2
Density
N = 333 Bandwidth = 0.07483
Density plot for b7 3
Density
N = 333 Bandwidth = 0.0317
Density plot for b7 4
Density
N = 333 Bandwidth = 0.03716
Density plot for b7 5
Density
N = 333 Bandwidth = 0.09721
Density plot for b8 1
Density
N = 333 Bandwidth = 0.123
Density plot for b8 2
Density
N = 333 Bandwidth = 0.07353
Density plot for b8 3
Density
N = 333 Bandwidth = 0.01376
Density plot for b8 4
Density
N = 333 Bandwidth = 0.07171
Density plot for b8 5
Density
N = 333 Bandwidth = 0.08591
Density plot for b9 1
Density
N = 333 Bandwidth = 0.1327
Density plot for b9 2
Density
N = 333 Bandwidth = 0.0705
Density plot for b9 3
Density
N = 333 Bandwidth = 0.008904
Density plot for b9 4
Density
N = 333 Bandwidth = 0.0258
Density plot for b9 5
Density
N = 333 Bandwidth = 0.1072
Density plot for b10 1
Density
N = 333 Bandwidth = 0.02551
Density plot for b10 2
Density
N = 333 Bandwidth = 0.0232
Density plot for b10 3
Density
N = 333 Bandwidth = 0.02241
Density plot for b10 4
Density
N = 333 Bandwidth = 0.03082



Therefore, the model that we get will be:

$$ln\left(\frac{p_{i1}}{p_{i6}}\right) = -6.2 + 1.2 \cdot Z_{i2} + 0.2 \cdot Z_{i5} - 0.8 \cdot Z_{i7} + 0.3 \cdot Z_{i8} + 0.4 \cdot Z_{i9} - 0.5 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i2}}{p_{i6}}\right) = -2.9 + 0.5 \cdot Z_{i2} + 0.4 \cdot Z_{i4} - 0.5 \cdot Z_{i7} - 0.2 \cdot Z_{i8} + 0.2 \cdot Z_{i9} + 0.1 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i3}}{p_{i6}}\right) = 0.1 - 0.5 \cdot Z_{i2} - 0.1 \cdot Z_{i3} - 0.9 \cdot Z_{i5} + 0.1 \cdot Z_{i6} - 0.5 \cdot Z_{i7} + 0.3 \cdot Z_{i10} + 0.8 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i4}}{p_{i6}}\right) = -2 + 0.3 \cdot Z_{i1} - 0.8 \cdot Z_{i2} + 0.2 \cdot Z_{i4} - 0.4 \cdot Z_{i5} - 0.7 \cdot Z_{i7} - 0.2 \cdot Z_{i8} + \\ + 0.8 \cdot Z_{i10} + 1.4 \cdot Z_{i11}$$

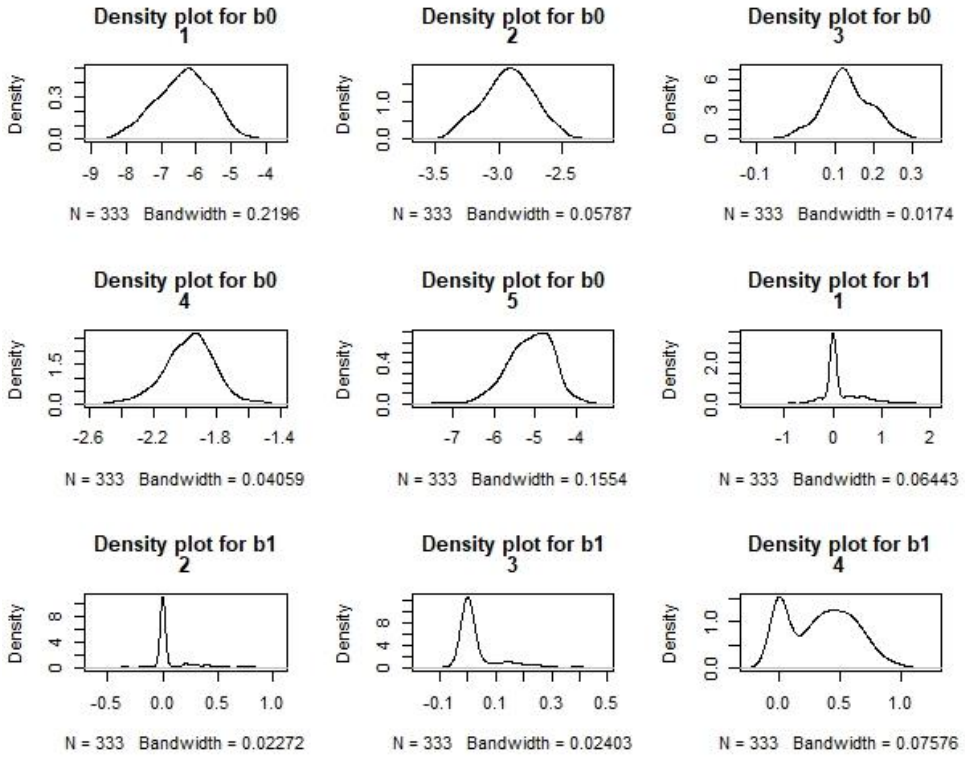
$$ln\left(\frac{p_{i5}}{p_{i6}}\right) = -5.1 - 0.2 \cdot Z_{i2} + 0.3 \cdot Z_{i3} - 0.9 \cdot Z_{i5} - 0.8 \cdot Z_{i7} - 0.2 \cdot Z_{i8} - 0.4 \cdot Z_{i9} + \\ + 0.9 \cdot Z_{i10} + 1.9 \cdot Z_{i11}$$

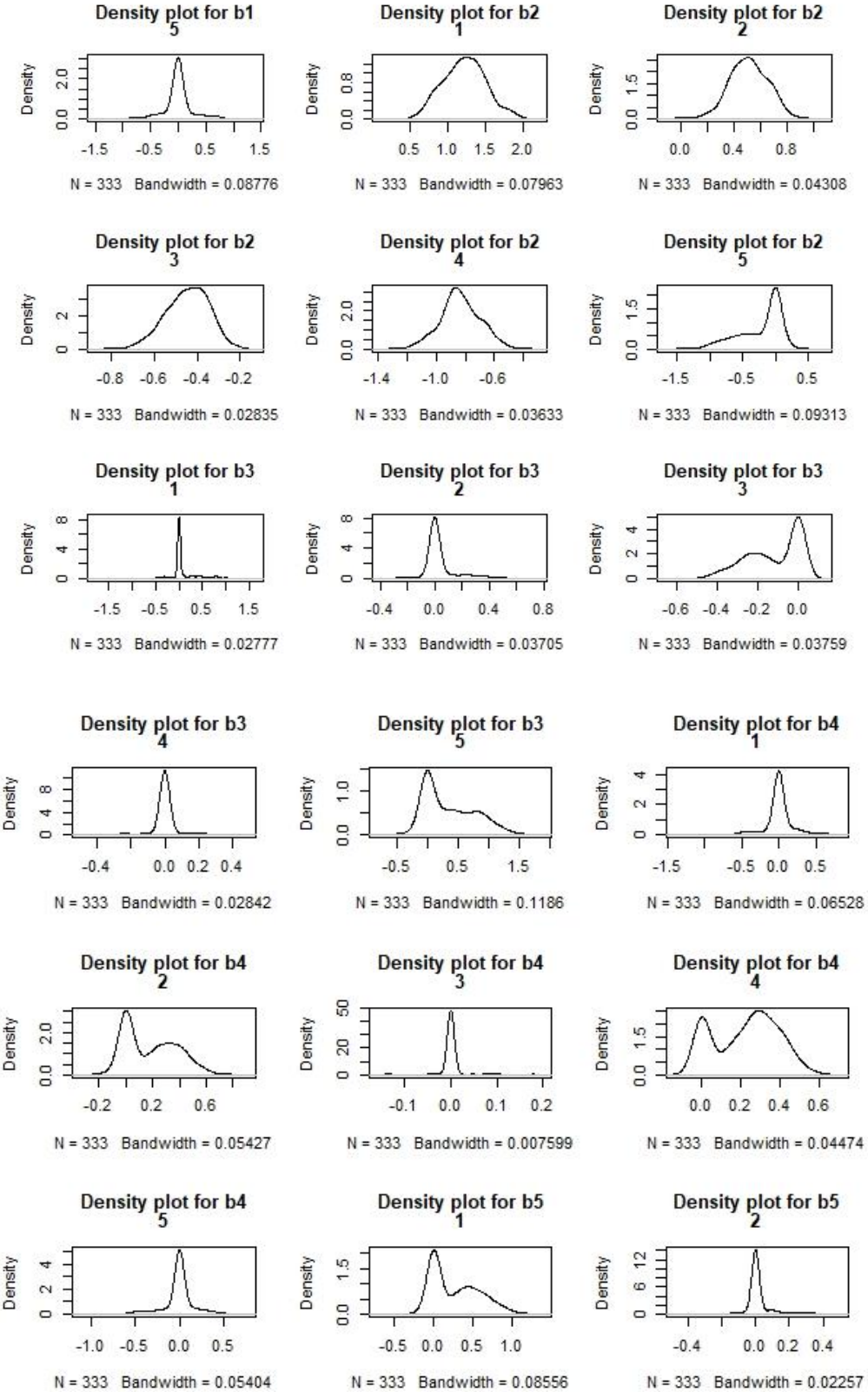
This is not an ideal result, as it does not resemble the result of the BAS package and some variables that were previously excluded were included in this method. This could be due to the MCMC that is used, or an error in the code itself. Perhaps more iterations could smooth out the results and make this model look somewhat more like the one in BAS above.

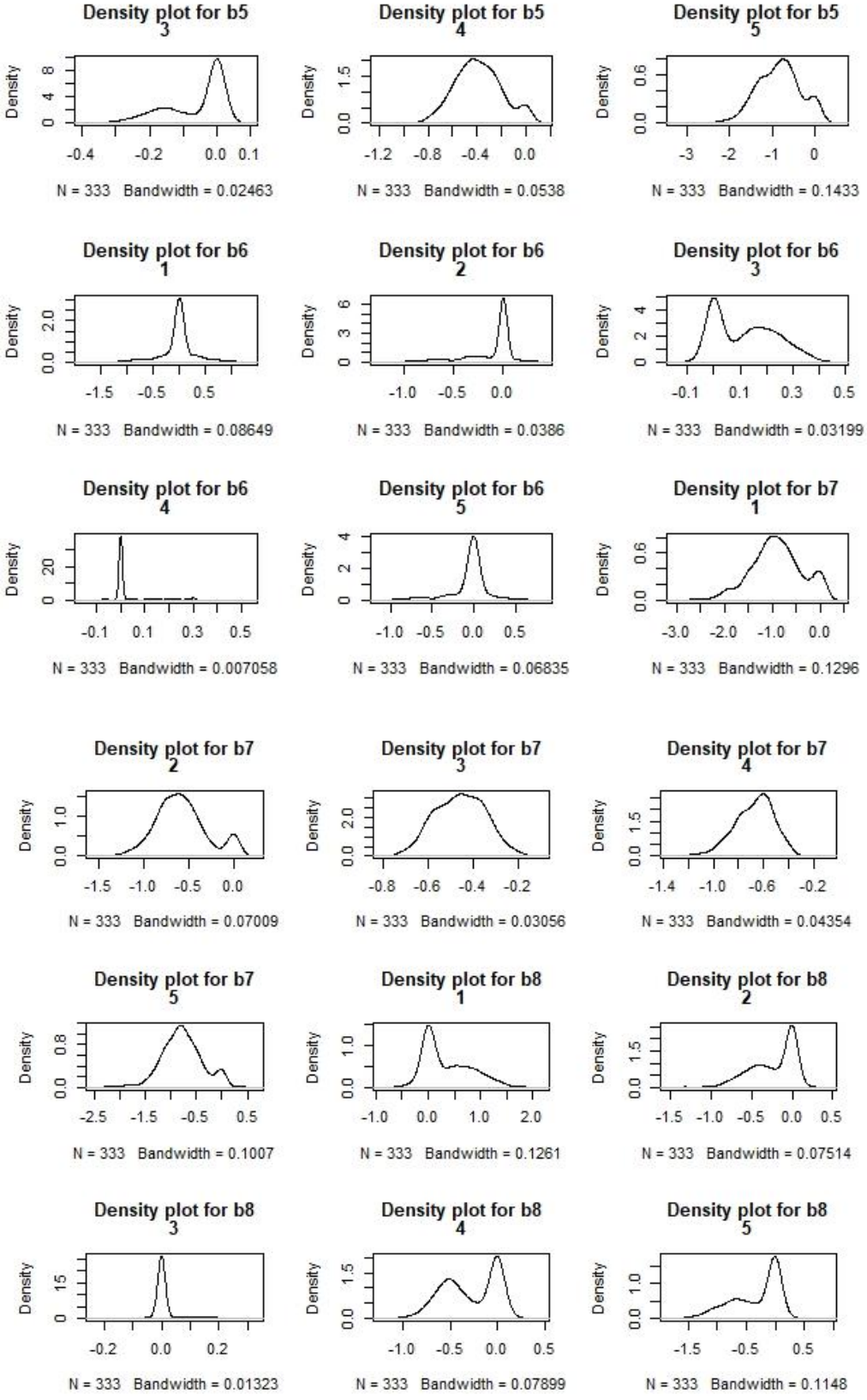
2. Using a Beta-Binomial Distribution on the priors:

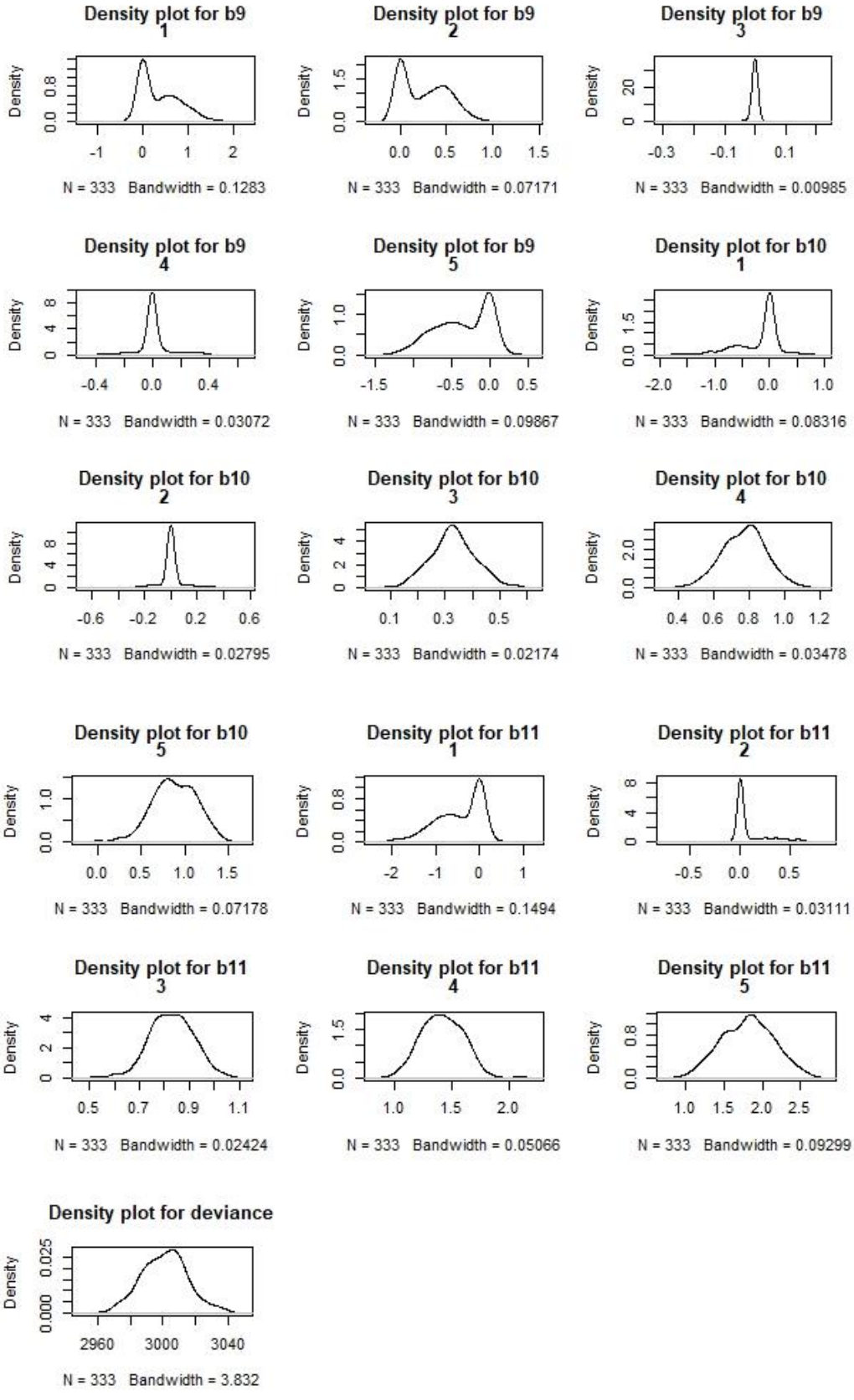
```
Inference for Bugs model at "C:\Users\30697\Desktop\gprior.txt", fit
using OpenBUGS,
 1 chains, each with 11000 iterations (first 1000 discarded), n.thin =
30
 n.sims = 333 iterations saved
           mean   sd   2.5%    25%    50%    75%  97.5%
alpha[1]   -6.4  0.8   -7.9   -6.9   -6.3   -5.8   -5.0
alpha[2]   -2.9  0.2   -3.3   -3.1   -2.9   -2.8   -2.5
alpha[3]    0.1  0.1    0.0    0.1    0.1    0.2    0.3
alpha[4]   -2.0  0.2   -2.3   -2.1   -2.0   -1.9   -1.7
alpha[5]   -5.1  0.6   -6.3   -5.5   -5.1   -4.7   -4.1
gb1[1]      0.2  0.4   -0.5    0.0    0.0    0.3    1.2
gb1[2]      0.1  0.2   -0.3    0.0    0.0    0.1    0.8
gb1[3]      0.0  0.1    0.0    0.0    0.0    0.0    0.3
gb1[4]      0.3  0.3    0.0    0.0    0.4    0.6    0.8
gb1[5]      0.0  0.3   -0.8    0.0    0.0    0.0    0.7
gb2[1]      1.2  0.3    0.7    1.0    1.2    1.4    1.8
gb2[2]      0.5  0.2    0.2    0.4    0.5    0.6    0.8
gb2[3]     -0.4  0.1   -0.7   -0.5   -0.4   -0.4   -0.3
gb2[4]     -0.8  0.1   -1.1   -0.9   -0.8   -0.7   -0.6
gb2[5]     -0.2  0.3   -1.0   -0.5    0.0    0.0    0.1
gb3[1]      0.1  0.3   -0.4    0.0    0.0    0.1    1.0
gb3[2]      0.0  0.1   -0.2    0.0    0.0    0.0    0.4
gb3[3]     -0.1  0.1   -0.4   -0.2   -0.1    0.0    0.0
gb3[4]      0.0  0.1   -0.3    0.0    0.0    0.0    0.3
gb3[5]      0.4  0.4    0.0    0.0    0.3    0.7    1.2
gb4[1]      0.0  0.2   -0.6    0.0    0.0    0.0    0.5
gb4[2]      0.2  0.2    0.0    0.0    0.2    0.4    0.6
gb4[3]      0.0  0.0    0.0    0.0    0.0    0.0    0.1
gb4[4]      0.2  0.2    0.0    0.0    0.3    0.3    0.5
gb4[5]      0.0  0.2   -0.5    0.0    0.0    0.0    0.4
gb5[1]      0.3  0.3    0.0    0.0    0.2    0.5    0.9
gb5[2]      0.0  0.1   -0.1    0.0    0.0    0.0    0.3
gb5[3]     -0.1  0.1   -0.3   -0.1    0.0    0.0    0.0
gb5[4]     -0.4  0.2   -0.7   -0.5   -0.4   -0.3    0.0
gb5[5]     -0.9  0.5   -1.9   -1.2   -0.8   -0.5    0.0
gb6[1]      0.0  0.3   -0.8    0.0    0.0    0.0    0.7
gb6[2]     -0.1  0.2   -0.7   -0.2    0.0    0.0    0.1
gb6[3]      0.1  0.1    0.0    0.0    0.1    0.2    0.3
gb6[4]      0.0  0.1   -0.1    0.0    0.0    0.0    0.3
gb6[5]     -0.1  0.2   -0.7    0.0    0.0    0.0    0.4
gb7[1]     -0.9  0.5   -2.0   -1.2   -0.9   -0.6    0.0
gb7[2]     -0.6  0.3   -1.1   -0.8   -0.6   -0.4    0.0
gb7[3]     -0.5  0.1   -0.7   -0.5   -0.5   -0.4   -0.2
gb7[4]     -0.7  0.2   -1.0   -0.8   -0.7   -0.6   -0.4
gb7[5]     -0.8  0.4   -1.5   -1.0   -0.8   -0.5    0.0
```

```
gb8[1]       0.4  0.4   -0.2    0.0    0.2    0.7    1.3
gb8[2]      -0.2  0.3   -0.8   -0.4   -0.1    0.0    0.0
gb8[3]       0.0  0.0    0.0    0.0    0.0    0.0    0.2
gb8[4]      -0.3  0.3   -0.8   -0.5   -0.3    0.0    0.0
gb8[5]      -0.3  0.4   -1.2   -0.6   -0.1    0.0    0.0
gb9[1]       0.4  0.5   -0.1    0.0    0.3    0.7    1.4
gb9[2]       0.3  0.3    0.0    0.0    0.3    0.5    0.7
gb9[3]       0.0  0.0   -0.1    0.0    0.0    0.0    0.0
gb9[4]       0.0  0.1   -0.2    0.0    0.0    0.0    0.3
gb9[5]      -0.4  0.4   -1.1   -0.6   -0.3    0.0    0.0
gb10[1]     -0.2  0.4   -1.2   -0.4    0.0    0.0    0.4
gb10[2]      0.0  0.1   -0.2    0.0    0.0    0.0    0.3
gb10[3]      0.3  0.1    0.2    0.3    0.3    0.4    0.5
gb10[4]      0.8  0.1    0.5    0.7    0.8    0.9    1.0
gb10[5]      0.9  0.3    0.4    0.7    0.9    1.1    1.3
gb11[1]     -0.5  0.5   -1.7   -0.9   -0.4    0.0    0.0
gb11[2]      0.1  0.2   -0.1    0.0    0.0    0.1    0.6
gb11[3]      0.8  0.1    0.7    0.8    0.8    0.9    1.0
gb11[4]      1.4  0.2    1.1    1.3    1.4    1.5    1.7
gb11[5]      1.8  0.3    1.2    1.6    1.8    2.0    2.4
deviance 3001.1 14.0 2973.4 2991.6 3001.6 3009.8 3031.2

DIC info (using the rule, pD = var(deviance)/2)
pD = 97.7 and DIC = 3098.8
DIC is an estimate of expected predictive error (lower deviance is
better).
```




Density plot for b1 5
Density plot for b2 1
Density plot for b2 2
Density plot for b2 3
Density plot for b2 4
Density plot for b2 5
Density plot for b3 1
Density plot for b3 2
Density plot for b3 3
Density plot for b3 4
Density plot for b3 5
Density plot for b4 1
Density plot for b4 2
Density plot for b4 3
Density plot for b4 4
Density plot for b4 5
Density plot for b5 1
Density plot for b5 2
Density
N = 333 Bandwidth = 0.08776
N = 333 Bandwidth = 0.07963
N = 333 Bandwidth = 0.04308
N = 333 Bandwidth = 0.02835
N = 333 Bandwidth = 0.03633
N = 333 Bandwidth = 0.09313
N = 333 Bandwidth = 0.02777
N = 333 Bandwidth = 0.03705
N = 333 Bandwidth = 0.03759
N = 333 Bandwidth = 0.02842
N = 333 Bandwidth = 0.1186
N = 333 Bandwidth = 0.06528
N = 333 Bandwidth = 0.05427
N = 333 Bandwidth = 0.007599
N = 333 Bandwidth = 0.04474
N = 333 Bandwidth = 0.05404
N = 333 Bandwidth = 0.08556
N = 333 Bandwidth = 0.02257


Density plot for b5 3
Density plot for b5 4
Density plot for b5 5
Density plot for b6 1
Density plot for b6 2
Density plot for b6 3
Density plot for b6 4
Density plot for b6 5
Density plot for b7 1
Density plot for b7 2
Density plot for b7 3
Density plot for b7 4
Density plot for b7 5
Density plot for b8 1
Density plot for b8 2
Density plot for b8 3
Density plot for b8 4
Density plot for b8 5
Density
N = 333 Bandwidth = 0.02463
N = 333 Bandwidth = 0.0538
N = 333 Bandwidth = 0.1433
N = 333 Bandwidth = 0.08649
N = 333 Bandwidth = 0.0386
N = 333 Bandwidth = 0.03199
N = 333 Bandwidth = 0.007058
N = 333 Bandwidth = 0.06835
N = 333 Bandwidth = 0.1296
N = 333 Bandwidth = 0.07009
N = 333 Bandwidth = 0.03056
N = 333 Bandwidth = 0.04354
N = 333 Bandwidth = 0.1007
N = 333 Bandwidth = 0.1261
N = 333 Bandwidth = 0.07514
N = 333 Bandwidth = 0.01323
N = 333 Bandwidth = 0.07899
N = 333 Bandwidth = 0.1148


Density plot for b9 1
Density
N = 333 Bandwidth = 0.1283
Density plot for b9 2
Density
N = 333 Bandwidth = 0.07171
Density plot for b9 3
Density
N = 333 Bandwidth = 0.00985
Density plot for b9 4
Density
N = 333 Bandwidth = 0.03072
Density plot for b9 5
Density
N = 333 Bandwidth = 0.09867
Density plot for b10 1
Density
N = 333 Bandwidth = 0.08316
Density plot for b10 2
Density
N = 333 Bandwidth = 0.02795
Density plot for b10 3
Density
N = 333 Bandwidth = 0.02174
Density plot for b10 4
Density
N = 333 Bandwidth = 0.03478
Density plot for b10 5
Density
N = 333 Bandwidth = 0.07178
Density plot for b11 1
Density
N = 333 Bandwidth = 0.1494
Density plot for b11 2
Density
N = 333 Bandwidth = 0.03111
Density plot for b11 3
Density
N = 333 Bandwidth = 0.02424
Density plot for b11 4
Density
N = 333 Bandwidth = 0.05066
Density plot for b11 5
Density
N = 333 Bandwidth = 0.09299
Density plot for deviance
Density
N = 333 Bandwidth = 3.832

And so, this model is of the following form:

$$ln\left(\frac{p_{i1}}{p_{i6}}\right) = -6.4 + 0.2 \cdot Z_{i1} + 1.2 \cdot Z_{i2} + 0.3 \cdot Z_{i5} - 0.9 \cdot Z_{i7} + 0.4 \cdot Z_{i8} + 0.4 \cdot Z_{i9} - \\ - 0.2 \cdot Z_{i10} - 0.5 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i2}}{p_{i6}}\right) = -2.9 + 0.1 \cdot Z_{i1} + 0.5 \cdot Z_{i2} + 0.2 \cdot Z_{i4} - 0.6 \cdot Z_{i7} - 0.2 \cdot Z_{i8} + 0.3 \cdot Z_{i9} + 0.1 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i3}}{p_{i6}}\right) = 0.1 - 0.4 \cdot Z_{i2} - 0.1 \cdot Z_{i3} - 0.1 \cdot Z_{i5} + 0.1 \cdot Z_{i6} - 0.5 \cdot Z_{i7} + 0.3 \cdot Z_{i10} + 0.8 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i4}}{p_{i6}}\right) = -2 + 0.3 \cdot Z_{i1} - 0.8 \cdot Z_{i2} + 0.2 \cdot Z_{i4} - 0.4 \cdot Z_{i5} - 0.7 \cdot Z_{i7} - 0.3 \cdot Z_{i8} + \\ + 0.8 \cdot Z_{i10} + 1.4 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i5}}{p_{i6}}\right) = -5.1 - 0.2 \cdot Z_{i2} + 0.4 \cdot Z_{i3} - 0.9 \cdot Z_{i5} - 0.1 \cdot Z_{i6} - 0.8 \cdot Z_{i7} - 0.3 \cdot Z_{i8} - 0.4 \cdot Z_{i9} + \\ + 0.9 \cdot Z_{i10} + 1.8 \cdot Z_{i11}$$

These are pretty similar results to the Uniform prior we calculated above.

Variable Selection through hyper-g on prior coefficients (using BAS package in R):

Similar method to the one above, meaning that once more:

$$\beta_{jk} \sim N(0, G \cdot (X^T \cdot X)^{-1} \cdot \sigma^2)$$

Only now, we have a different hyperprior, which is:

$$a \leftarrow 3$$

$$bw \leftarrow a/2 - 1$$

$$w \sim dbeta(1, bw)$$

$$G \leftarrow w/(1 - w)$$

And so, we repeat the procedure once again, by:

1. Using a Uniform Distribution on the priors:

Here we shall have:

For level 1 (category 1 with reference level 6):

```
Marginal Posterior Summaries of Coefficients:
```

```
 Using  BMA

 Based on the top  597 models
                     post mean  post SD   post p(B != 0)
Intercept             -7.52315    1.49508   1.00000
fixed.acidity          0.25773    0.64414   0.54600
volatile.acidity       0.84906    0.33209   0.99683
citric.acid            0.23838    0.46934   0.49570
residual.sugar        -0.01754    0.20251   0.42070
chlorides              0.28271    0.24538   0.78384
free.sulfur.dioxide    0.72020    0.71562   0.80117
total.sulfur.dioxide  -1.82046    1.24840   0.99258
density                0.37208    0.54611   0.59717
pH                     0.44995    0.52270   0.69106
sulphates             -0.02880    0.27377   0.36753
alcohol               -0.74725    0.72534   0.81191
> summary(res3.1)
                    P(B != 0 | Y)   model 1     model 2     model 3
Intercept               1.0000000   1.00000   1.0000000   1.0000000
fixed.acidity           0.5459961   1.00000   1.0000000   0.0000000
volatile.acidity        0.9968262   1.00000   1.0000000   1.0000000
citric.acid             0.4957031   0.00000   0.0000000   1.0000000
residual.sugar          0.4207031   0.00000   0.0000000   1.0000000
chlorides               0.7838379   1.00000   1.0000000   1.0000000
free.sulfur.dioxide     0.8011719   1.00000   1.0000000   1.0000000
total.sulfur.dioxide    0.9925781   1.00000   1.0000000   1.0000000
density                 0.5971680   0.00000   1.0000000   1.0000000
pH                      0.6910645   1.00000   1.0000000   1.0000000
sulphates               0.3675293   0.00000   0.0000000   0.0000000
alcohol                 0.8119141   1.00000   1.0000000   1.0000000
BF                             NA   1.00000   0.7122447   0.5627455
PostProbs                      NA   0.03990   0.0251000   0.0218000
R2                             NA   0.42060   0.4245000   0.4289000
dim                            NA   8.00000   9.0000000  10.0000000
logmarg                        NA -38.72036 -39.0596942 -39.2952883
                       model 4     model 5
Intercept            1.0000000   1.0000000
fixed.acidity        1.0000000   1.0000000
volatile.acidity     1.0000000   1.0000000
citric.acid          0.0000000   1.0000000
residual.sugar       0.0000000   0.0000000
chlorides            1.0000000   1.0000000
free.sulfur.dioxide  1.0000000   1.0000000
total.sulfur.dioxide 1.0000000   1.0000000
density              0.0000000   0.0000000
pH                   1.0000000   1.0000000
sulphates            1.0000000   0.0000000
```

```
alcohol                1.0000000   1.0000000
BF                     0.5837728   0.6661712
PostProbs              0.0195000   0.0184000
R2                     0.4211000   0.4239000
dim                    9.0000000   9.0000000
logmarg              -39.2586040 -39.1265691
```

For level 2 (category 2 with reference level 6):

```
 Marginal Posterior Summaries of Coefficients:

 Using  BMA

 Based on the top  839 models
                      post mean  post SD   post p(B != 0)
Intercept              -2.86369    0.20785   1.00000
fixed.acidity           0.10712    0.29831   0.42842
volatile.acidity        0.46271    0.14223   0.99438
citric.acid             0.05396    0.15641   0.35137
residual.sugar          0.19549    0.17493   0.69678
chlorides               0.02573    0.08736   0.32349
free.sulfur.dioxide    -0.06036    0.17042   0.36519
total.sulfur.dioxide   -0.44209    0.23341   0.91304
density                -0.21645    0.29171   0.56333
pH                      0.22298    0.23757   0.65645
sulphates               0.04183    0.11490   0.35405
alcohol                 0.06646    0.19198   0.43115
> summary(res3.2)
                     P(B != 0 | Y)   model 1      model 2      model 3
Intercept                1.0000000    1.0000    1.0000000    1.0000000
fixed.acidity            0.4284180    1.0000    1.0000000    1.0000000
volatile.acidity         0.9943848    1.0000    1.0000000    1.0000000
citric.acid              0.3513672    0.0000    0.0000000    0.0000000
residual.sugar           0.6967773    1.0000    1.0000000    1.0000000
chlorides                0.3234863    0.0000    0.0000000    1.0000000
free.sulfur.dioxide      0.3651855    0.0000    0.0000000    0.0000000
total.sulfur.dioxide     0.9130371    1.0000    1.0000000    1.0000000
density                  0.5633301    1.0000    1.0000000    1.0000000
pH                       0.6564453    1.0000    1.0000000    1.0000000
sulphates                0.3540527    0.0000    1.0000000    0.0000000
alcohol                  0.4311523    0.0000    0.0000000    0.0000000
BF                             NA     1.0000    0.7531036    0.7124517
PostProbs                      NA     0.0199    0.0142000    0.0132000
R2                             NA     0.1303    0.1344000    0.1342000
dim                            NA     7.0000    8.0000000    8.0000000
logmarg                        NA  -176.7948 -177.0783963 -177.1338871
                         model 4      model 5
Intercept              1.0000000    1.0000000
```

```
fixed.acidity            0.0000000    0.0000000
volatile.acidity         1.0000000    1.0000000
citric.acid              0.0000000    0.0000000
residual.sugar           1.0000000    1.0000000
chlorides                0.0000000    0.0000000
free.sulfur.dioxide      0.0000000    0.0000000
total.sulfur.dioxide     1.0000000    1.0000000
density                  1.0000000    1.0000000
pH                       1.0000000    0.0000000
sulphates                0.0000000    1.0000000
alcohol                  0.0000000    0.0000000
BF                       0.6853099    0.3956922
PostProbs                0.0115000    0.0103000
R2                       0.1218000    0.1184000
dim                      6.0000000    6.0000000
logmarg               -177.1727279 -177.7219624
```

For level 3 (category 3 with reference level 6):

```
 Marginal Posterior Summaries of Coefficients:

 Using  BMA

 Based on the top  125 models
                      post mean  post SD    post p(B != 0)
Intercept              0.141387   0.066142   1.000000
fixed.acidity          0.039904   0.087145   0.292480
volatile.acidity      -0.421703   0.098488   0.999707
citric.acid           -0.107425   0.133124   0.531250
residual.sugar         0.008734   0.033869   0.184473
chlorides             -0.081154   0.088043   0.574951
free.sulfur.dioxide    0.239161   0.108734   0.929688
total.sulfur.dioxide  -0.590159   0.112643   0.999951
density                0.006282   0.045699   0.161572
pH                    -0.005276   0.036958   0.161084
sulphates              0.367034   0.076644   0.998877
alcohol                0.797093   0.088799   0.999561
> summary(res3.3)
                     P(B != 0 | Y)   model 1      model 2      model 3
Intercept               1.0000000    1.0000    1.0000000    1.0000000
fixed.acidity           0.2924805    0.0000    1.0000000    0.0000000
volatile.acidity        0.9997070    1.0000    1.0000000    1.0000000
citric.acid             0.5312500    0.0000    1.0000000    0.0000000
residual.sugar          0.1844727    0.0000    0.0000000    0.0000000
chlorides               0.5749512    1.0000    0.0000000    0.0000000
free.sulfur.dioxide     0.9296875    1.0000    1.0000000    1.0000000
total.sulfur.dioxide    0.9999512    1.0000    1.0000000    1.0000000
density                 0.1615723    0.0000    0.0000000    0.0000000
```

```
pH                      0.1610840    0.0000    0.0000000    0.0000000
sulphates               0.9988770    1.0000    1.0000000    1.0000000
alcohol                 0.9995605    1.0000    1.0000000    1.0000000
BF                             NA    1.0000    0.4438559    0.4659427
PostProbs                      NA    0.1681    0.0775000    0.0694000
R2                             NA    0.1814    0.1825000    0.1784000
dim                            NA    7.0000    8.0000000    6.0000000
logmarg                        NA -765.9311 -766.7433254 -766.6947627
                         model 4      model 5
Intercept              1.0000000    1.0000000
fixed.acidity          0.0000000    0.0000000
volatile.acidity       1.0000000    1.0000000
citric.acid            1.0000000    1.0000000
residual.sugar         0.0000000    0.0000000
chlorides              0.0000000    1.0000000
free.sulfur.dioxide    1.0000000    1.0000000
total.sulfur.dioxide   1.0000000    1.0000000
density                0.0000000    0.0000000
pH                     0.0000000    0.0000000
sulphates              1.0000000    1.0000000
alcohol                1.0000000    1.0000000
BF                     0.3621207    0.3908064
PostProbs              0.0669000    0.0653000
R2                     0.1802000    0.1823000
dim                    7.0000000    8.0000000
logmarg             -766.9468479 -766.8706131
```

For level 4 (category 4 with reference level 6):

```
 Marginal Posterior Summaries of Coefficients:

 Using  BMA

 Based on the top  139 models
                      post mean  post SD   post p(B != 0)
Intercept             -1.96680    0.16004   1.00000
fixed.acidity          0.29379    0.29372   0.67393
volatile.acidity      -0.86458    0.18231   0.99985
citric.acid           -0.04244    0.14466   0.25996
residual.sugar         0.15445    0.18189   0.54561
chlorides             -0.22809    0.20644   0.70747
free.sulfur.dioxide    0.32244    0.23688   0.77344
total.sulfur.dioxide  -0.72767    0.23563   0.99893
density               -0.21311    0.31194   0.46133
pH                    -0.04897    0.14938   0.32085
sulphates              0.67686    0.13398   0.99961
alcohol                1.57815    0.21830   0.99917
> summary(res3.4)
```

	P(B != 0 | Y)	model 1	model 2	model 3
Intercept	1.0000000	1.0000	1.0000000	1.0000000
fixed.acidity	0.6739258	1.0000	1.0000000	0.0000000
volatile.acidity	0.9998535	1.0000	1.0000000	1.0000000
citric.acid	0.2599609	0.0000	0.0000000	0.0000000
residual.sugar	0.5456055	1.0000	1.0000000	0.0000000
chlorides	0.7074707	1.0000	0.0000000	1.0000000
free.sulfur.dioxide	0.7734375	1.0000	1.0000000	1.0000000
total.sulfur.dioxide	0.9989258	1.0000	1.0000000	1.0000000
density	0.4613281	1.0000	1.0000000	0.0000000
pH	0.3208496	0.0000	0.0000000	1.0000000
sulphates	0.9996094	1.0000	1.0000000	1.0000000
alcohol	0.9991699	1.0000	1.0000000	1.0000000
BF	NA	1.0000	0.3934193	0.4334659
PostProbs	NA	0.1100	0.0526000	0.0499000
R2	NA	0.5436	0.5381000	0.5350000
dim	NA	10.0000	9.0000000	8.0000000
logmarg	NA	-236.4561	-237.3889340	-237.2919970

	model 4	model 5
Intercept	1.0000000	1.0000000
fixed.acidity	1.0000000	1.0000000
volatile.acidity	1.0000000	1.0000000
citric.acid	0.0000000	1.0000000
residual.sugar	0.0000000	1.0000000
chlorides	1.0000000	1.0000000
free.sulfur.dioxide	1.0000000	1.0000000
total.sulfur.dioxide	1.0000000	1.0000000
density	0.0000000	1.0000000
pH	0.0000000	0.0000000
sulphates	1.0000000	1.0000000
alcohol	1.0000000	1.0000000
BF	0.4033167	0.2719545
PostProbs	0.0476000	0.0335000
R2	0.5348000	0.5440000
dim	8.0000000	11.0000000
logmarg	-237.3640880	-237.7581753

For level 5 (category 5 with reference level 6):

```
 Marginal Posterior Summaries of Coefficients:

 Using  BMA

 Based on the top  151 models
                     post mean  post SD   post p(B != 0)
Intercept             -8.18184    1.73365   1.00000
fixed.acidity         -0.56748    0.64713   0.73643
volatile.acidity      -0.03611    0.38643   0.45522
```

```
citric.acid             1.05922    0.70389   0.95942
residual.sugar         -0.83269    0.66020   0.93423
chlorides              -1.03723    0.96602   0.97109
free.sulfur.dioxide     0.17914    0.38303   0.54082
total.sulfur.dioxide   -1.24010    0.66914   0.99634
density                -0.08360    0.59972   0.50195
pH                     -0.57133    0.54758   0.82681
sulphates               0.81304    0.35972   0.99888
alcohol                 1.63723    0.68058   1.00000
> summary(res3.5)
                    P(B != 0 | Y)   model 1     model 2     model 3
Intercept               1.0000000   1.00000   1.0000000   1.0000000
fixed.acidity           0.7364258   1.00000   1.0000000   1.0000000
volatile.acidity        0.4552246   0.00000   0.0000000   1.0000000
citric.acid             0.9594238   1.00000   1.0000000   1.0000000
residual.sugar          0.9342285   1.00000   1.0000000   1.0000000
chlorides               0.9710937   1.00000   1.0000000   1.0000000
free.sulfur.dioxide     0.5408203   1.00000   0.0000000   1.0000000
total.sulfur.dioxide    0.9963379   1.00000   1.0000000   1.0000000
density                 0.5019531   0.00000   0.0000000   0.0000000
pH                      0.8268066   1.00000   1.0000000   1.0000000
sulphates               0.9988770   1.00000   1.0000000   1.0000000
alcohol                 1.0000000   1.00000   1.0000000   1.0000000
BF                             NA   1.00000   0.8888327   0.7741582
PostProbs                      NA   0.10440   0.1008000   0.0809000
R2                             NA   0.71730   0.7116000   0.7180000
dim                            NA  10.00000   9.0000000  11.0000000
logmarg                        NA -32.84751 -32.9653535 -33.1034863
                        model 4     model 5
Intercept             1.0000000   1.000000
fixed.acidity         1.0000000   1.000000
volatile.acidity      0.0000000   1.000000
citric.acid           1.0000000   1.000000
residual.sugar        1.0000000   1.000000
chlorides             1.0000000   1.000000
free.sulfur.dioxide   1.0000000   0.000000
total.sulfur.dioxide  1.0000000   1.000000
density               1.0000000   0.000000
pH                    1.0000000   1.000000
sulphates             1.0000000   1.000000
alcohol               1.0000000   1.000000
BF                    0.6939117   0.565809
PostProbs             0.0793000   0.069800
R2                    0.7178000   0.711700
dim                  11.0000000  10.000000
logmarg             -33.2129179 -33.417006
```

And so, the model will be:

$$ln\left(\frac{p_{i1}}{p_{i6}}\right) = -7.52 + 0.25 \cdot Z_{i1} + 0.85 \cdot Z_{i2} + 0.28 \cdot Z_{i5} + 0.72 \cdot Z_{i6} - 1.8 \cdot Z_{i7} + 0.3 \cdot Z_{i8} + \\ +0.44 \cdot Z_{i9} - 0.75 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i2}}{p_{i6}}\right) = -2.86 + 0.1 \cdot Z_{i1} + 0.46 \cdot Z_{i2} + 0.2 \cdot Z_{i4} - 0.44 \cdot Z_{i7} - 0.22 \cdot Z_{i8} + 0.22 \cdot Z_{i9}$$

$$ln\left(\frac{p_{i3}}{p_{i6}}\right) = 0.14 - 0.42 \cdot Z_{i2} - 0.08 \cdot Z_{i5} + 0.24 \cdot Z_{i6} - 0.6 \cdot Z_{i7} + 0.37 \cdot Z_{i10} + 0.8 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i4}}{p_{i6}}\right) = -1.97 + 0.29 \cdot Z_{i1} - 0.86 \cdot Z_{i2} + 0.15 \cdot Z_{i4} - 0.22 \cdot Z_{i5} + 0.32 \cdot Z_{i6} - \\ - 0.73 \cdot Z_{i7} - 0.21 \cdot Z_{i8} + 0.68 \cdot Z_{i10} + 1.58 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i5}}{p_{i6}}\right) = -8.18 - 0.57 \cdot Z_{i1} + 1.06 \cdot Z_{i3} - 0.83 \cdot Z_{i4} - 1.04 \cdot Z_{i5} + 0.18 \cdot Z_{i6} - \\ - 1.24 \cdot Z_{i7} - 0.57 \cdot Z_{i9} + 0.81 \cdot Z_{i10} + 1.63 \cdot Z_{i11}$$

2. Using a Beta-Binomial Distribution on the priors:

Our results will be:

For level 1 (category 1 with reference level 6):

```
 Marginal Posterior Summaries of Coefficients:

 Using  BMA

 Based on the top  714 models
                      post mean   post SD     post p(B != 0)
Intercept             -6.8430230   1.5042012   1.0000000
fixed.acidity          0.1617056   0.5145016   0.3564453
volatile.acidity       0.8787058   0.3153759   0.9950684
citric.acid            0.1836641   0.4138383   0.3675293
residual.sugar        -0.0004211   0.1671873   0.2753418
chlorides              0.2022830   0.2381807   0.5709473
free.sulfur.dioxide    0.4963420   0.6921159   0.5653809
total.sulfur.dioxide  -1.5038692   1.2120729   0.9023438
density                0.3061322   0.4860572   0.4829102
pH                     0.2768490   0.4610304   0.4469727
sulphates             -0.0142461   0.2225155   0.2402344
alcohol               -0.5392789   0.6986689   0.5900391
> summary(res3.1)
                    P(B != 0 | Y)        model 1      model 2   model 3
Intercept               1.0000000   1.000000000   1.00000000   1.00000
fixed.acidity           0.3564453   0.000000000   0.00000000   1.00000
volatile.acidity        0.9950684   1.000000000   1.00000000   1.00000
```

```
citric.acid              0.3675293   0.000000000   0.00000000   0.00000
residual.sugar           0.2753418   0.000000000   0.00000000   0.00000
chlorides                0.5709473   0.000000000   0.00000000   1.00000
free.sulfur.dioxide      0.5653809   0.000000000   0.00000000   1.00000
total.sulfur.dioxide     0.9023438   0.000000000   1.00000000   1.00000
density                  0.4829102   0.000000000   0.00000000   0.00000
pH                       0.4469727   0.000000000   0.00000000   1.00000
sulphates                0.2402344   0.000000000   0.00000000   0.00000
alcohol                  0.5900391   0.000000000   0.00000000   1.00000
BF                              NA   0.001661367   0.01156175   1.00000
PostProbs                       NA   0.036800000   0.03670000   0.02450
R2                              NA   0.198700000   0.26110000   0.42060
dim                             NA   2.000000000   3.00000000   8.00000
logmarg                         NA -45.120475228 -43.18041339 -38.72036
                         model 4     model 5
Intercept              1.00000000   1.0000000
fixed.acidity          0.00000000   0.0000000
volatile.acidity       1.00000000   1.0000000
citric.acid            0.00000000   0.0000000
residual.sugar         0.00000000   0.0000000
chlorides              0.00000000   1.0000000
free.sulfur.dioxide    0.00000000   0.0000000
total.sulfur.dioxide   1.00000000   1.0000000
density                1.00000000   0.0000000
pH                     0.00000000   0.0000000
sulphates              0.00000000   0.0000000
alcohol                0.00000000   0.0000000
BF                     0.02169067   0.0180953
PostProbs              0.01430000   0.0134000
R2                     0.29790000   0.2905000
dim                    4.00000000   4.0000000
logmarg              -42.55123353 -42.7324634
```

For level 2 (category 2 with reference level 6):

```
 Marginal Posterior Summaries of Coefficients:

 Using  BMA

 Based on the top  736 models
                      post mean  post SD   post p(B != 0)
Intercept             -2.87874    0.20794   1.00000
fixed.acidity          0.03418    0.19330   0.19663
volatile.acidity       0.48114    0.15086   0.98267
citric.acid            0.02062    0.10388   0.15469
residual.sugar         0.10450    0.15533   0.39712
chlorides              0.01143    0.05975   0.15415
free.sulfur.dioxide   -0.04813    0.15391   0.21890
total.sulfur.dioxide  -0.46098    0.23781   0.89351
density               -0.09611    0.21153   0.27876
pH                     0.13334    0.19911   0.42666
```

```
sulphates               0.01528    0.07241   0.14238
alcohol                 0.07382    0.17420   0.29150
> summary(res3.2)
                     P(B != 0 | Y)      model 1      model 2
Intercept                1.0000000    1.0000000    1.0000000
fixed.acidity            0.1966309    0.0000000    0.0000000
volatile.acidity         0.9826660    1.0000000    1.0000000
citric.acid              0.1546875    0.0000000    0.0000000
residual.sugar           0.3971191    0.0000000    0.0000000
chlorides                0.1541504    0.0000000    0.0000000
free.sulfur.dioxide      0.2188965    0.0000000    0.0000000
total.sulfur.dioxide     0.8935059    1.0000000    1.0000000
density                  0.2787598    0.0000000    0.0000000
pH                       0.4266602    0.0000000    0.0000000
sulphates                0.1423828    0.0000000    0.0000000
alcohol                  0.2915039    0.0000000    1.0000000
BF                             NA    0.3970406    0.7446754
PostProbs                      NA    0.1388000    0.0623000
R2                             NA    0.0961000    0.1074000
dim                            NA    3.0000000    4.0000000
logmarg                        NA -178.0274924 -177.3985826
                         model 3      model 4   model 5
Intercept                1.0000000    1.0000000    1.0000
fixed.acidity            0.0000000    0.0000000    0.0000
volatile.acidity         1.0000000    1.0000000    1.0000
citric.acid              0.0000000    0.0000000    0.0000
residual.sugar           0.0000000    1.0000000    1.0000
chlorides                0.0000000    0.0000000    0.0000
free.sulfur.dioxide      0.0000000    0.0000000    0.0000
total.sulfur.dioxide     1.0000000    1.0000000    1.0000
density                  0.0000000    0.0000000    0.0000
pH                       1.0000000    0.0000000    1.0000
sulphates                0.0000000    0.0000000    0.0000
alcohol                  0.0000000    0.0000000    0.0000
BF                       0.6489227    0.4632912    1.0000
PostProbs                0.0552000    0.0343000    0.0272
R2                       0.1063000    0.1044000    0.1156
dim                      4.0000000    4.0000000    5.0000
logmarg               -177.5362175 -177.8731751 -177.1038
```

For level 3 (category 3 with reference level 6):

```
 Marginal Posterior Summaries of Coefficients:

 Using  BMA

 Based on the top  118 models
                      post mean  post SD    post p(B != 0)
```

```
Intercept              0.142465   0.066061   1.000000
fixed.acidity          0.027626   0.075117   0.203418
volatile.acidity      -0.413797   0.095587   0.999902
citric.acid           -0.082520   0.124862   0.406543
residual.sugar         0.006284   0.029072   0.133008
chlorides             -0.067377   0.086408   0.468799
free.sulfur.dioxide    0.228102   0.121249   0.870215
total.sulfur.dioxide  -0.583919   0.121366   0.998096
density                0.004795   0.039211   0.117285
pH                    -0.003445   0.031899   0.126123
sulphates              0.358153   0.077227   0.998193
alcohol                0.800701   0.087438   0.999707
> summary(res3.3)
                    P(B != 0 | Y)   model 1      model 2      model 3
Intercept               1.0000000    1.0000    1.0000000    1.0000000
fixed.acidity           0.2034180    0.0000    0.0000000    0.0000000
volatile.acidity        0.9999023    1.0000    1.0000000    1.0000000
citric.acid             0.4065430    0.0000    0.0000000    1.0000000
residual.sugar          0.1330078    0.0000    0.0000000    0.0000000
chlorides               0.4687988    1.0000    0.0000000    0.0000000
free.sulfur.dioxide     0.8702148    1.0000    1.0000000    1.0000000
total.sulfur.dioxide    0.9980957    1.0000    1.0000000    1.0000000
density                 0.1172852    0.0000    0.0000000    0.0000000
pH                      0.1261230    0.0000    0.0000000    0.0000000
sulphates               0.9981934    1.0000    1.0000000    1.0000000
alcohol                 0.9997070    1.0000    1.0000000    1.0000000
BF                             NA    1.0000    0.4659427    0.3621207
PostProbs                      NA    0.1854    0.1850000    0.0697000
R2                             NA    0.1814    0.1784000    0.1802000
dim                            NA    7.0000    6.0000000    7.0000000
logmarg                        NA -765.9311 -766.6947627 -766.9468479
                        model 4      model 5
Intercept             1.0000000    1.0000000
fixed.acidity         1.0000000    0.0000000
volatile.acidity      1.0000000    1.0000000
citric.acid           1.0000000    1.0000000
residual.sugar        0.0000000    0.0000000
chlorides             0.0000000    1.0000000
free.sulfur.dioxide   1.0000000    1.0000000
total.sulfur.dioxide  1.0000000    1.0000000
density               0.0000000    0.0000000
pH                    0.0000000    0.0000000
sulphates             1.0000000    1.0000000
alcohol               1.0000000    1.0000000
BF                    0.4438559    0.3908064
PostProbs             0.0528000    0.0443000
R2                    0.1825000    0.1823000
dim                   8.0000000    8.0000000
```

```
logmarg                -766.7433254 -766.8706131
```

For level 4 (category 4 with reference level 6):

```
 Marginal Posterior Summaries of Coefficients:

 Using  BMA

 Based on the top  138 models
                     post mean  post SD   post p(B != 0)
Intercept             -1.95849    0.15929   1.00000
fixed.acidity          0.21308    0.26791   0.53838
volatile.acidity      -0.89002    0.17810   0.99976
citric.acid           -0.02680    0.11912   0.17441
residual.sugar         0.10976    0.16537   0.40825
chlorides             -0.19564    0.20780   0.60122
free.sulfur.dioxide    0.25818    0.24706   0.62329
total.sulfur.dioxide  -0.68087    0.23985   0.99795
density               -0.13858    0.27207   0.32080
pH                    -0.04653    0.13498   0.26587
sulphates              0.65850    0.13567   0.99844
alcohol                1.62077    0.20550   0.99980
> summary(res3.4)
                    P(B != 0 | Y)   model 1       model 2       model 3
Intercept               1.0000000    1.0000    1.00000000    1.0000000
fixed.acidity           0.5383789    1.0000    0.00000000    0.0000000
volatile.acidity        0.9997559    1.0000    1.00000000    1.0000000
citric.acid             0.1744141    0.0000    0.00000000    0.0000000
residual.sugar          0.4082520    1.0000    0.00000000    0.0000000
chlorides               0.6012207    1.0000    0.00000000    1.0000000
free.sulfur.dioxide     0.6232910    1.0000    0.00000000    0.0000000
total.sulfur.dioxide    0.9979492    1.0000    1.00000000    1.0000000
density                 0.3208008    1.0000    0.00000000    0.0000000
pH                      0.2658691    0.0000    0.00000000    0.0000000
sulphates               0.9984375    1.0000    1.00000000    1.0000000
alcohol                 0.9998047    1.0000    1.00000000    1.0000000
BF                             NA    1.0000    0.05858977    0.1362797
PostProbs                      NA    0.0771    0.05530000    0.0479000
R2                             NA    0.5436    0.51920000    0.5250000
dim                            NA   10.0000    5.00000000    6.0000000
logmarg                        NA -236.4561 -239.29324998 -238.4491009
                        model 4      model 5
Intercept             1.0000000    1.0000000
fixed.acidity         1.0000000    0.0000000
volatile.acidity      1.0000000    1.0000000
citric.acid           0.0000000    0.0000000
residual.sugar        0.0000000    0.0000000
chlorides             1.0000000    1.0000000
free.sulfur.dioxide   1.0000000    1.0000000
total.sulfur.dioxide  1.0000000    1.0000000
density               0.0000000    0.0000000
pH                    0.0000000    1.0000000
```

```
sulphates               1.0000000    1.0000000
alcohol                 1.0000000    1.0000000
BF                      0.4033167    0.4334659
PostProbs               0.0451000    0.0421000
R2                      0.5348000    0.5350000
dim                     8.0000000    8.0000000
logmarg              -237.3640880 -237.2919970
```

For level 5 (category 5 with reference level 6):

```
 Marginal Posterior Summaries of Coefficients:

 Using  BMA

 Based on the top  169 models
                      post mean  post SD   post p(B != 0)
Intercept              -8.14219    1.73035   1.00000
fixed.acidity          -0.53744    0.64231   0.69438
volatile.acidity       -0.04779    0.38350   0.42656
citric.acid             1.05143    0.70281   0.94722
residual.sugar         -0.84141    0.66956   0.92119
chlorides              -1.04793    0.97491   0.96411
free.sulfur.dioxide     0.16790    0.37331   0.50361
total.sulfur.dioxide   -1.23927    0.66381   0.99512
density                -0.09844    0.60212   0.48501
pH                     -0.55161    0.54722   0.79897
sulphates               0.82627    0.36340   0.99741
alcohol                 1.64879    0.68175   0.99995
> summary(res3.5)
                     P(B != 0 | Y)     model 1   model 2     model 3
Intercept                1.0000000   1.0000000   1.00000   1.0000000
fixed.acidity            0.6943848   1.0000000   1.00000   1.0000000
volatile.acidity         0.4265625   0.0000000   0.00000   0.0000000
citric.acid              0.9472168   1.0000000   1.00000   1.0000000
residual.sugar           0.9211914   1.0000000   1.00000   1.0000000
chlorides                0.9641113   1.0000000   1.00000   1.0000000
free.sulfur.dioxide      0.5036133   0.0000000   1.00000   1.0000000
total.sulfur.dioxide     0.9951172   1.0000000   1.00000   1.0000000
density                  0.4850098   0.0000000   0.00000   1.0000000
pH                       0.7989746   1.0000000   1.00000   1.0000000
sulphates                0.9974121   1.0000000   1.00000   1.0000000
alcohol                  0.9999512   1.0000000   1.00000   1.0000000
BF                              NA   0.8888327   1.00000   0.6939117
PostProbs                       NA   0.1108000   0.09930   0.0729000
R2                              NA   0.7116000   0.71730   0.7178000
dim                             NA   9.0000000  10.00000  11.0000000
logmarg                         NA -32.9653535 -32.84751 -33.2129179
                        model 4     model 5
Intercept              1.0000000   1.0000000
```

```
fixed.acidity          1.0000000   1.0000000
volatile.acidity       1.0000000   0.0000000
citric.acid            1.0000000   1.0000000
residual.sugar         1.0000000   1.0000000
chlorides              1.0000000   1.0000000
free.sulfur.dioxide    1.0000000   0.0000000
total.sulfur.dioxide   1.0000000   1.0000000
density                0.0000000   1.0000000
pH                     1.0000000   1.0000000
sulphates              1.0000000   1.0000000
alcohol                1.0000000   1.0000000
BF                     0.7741582   0.6133596
PostProbs              0.0729000   0.0645000
R2                     0.7180000   0.7119000
dim                   11.0000000  10.0000000
logmarg              -33.1034863 -33.3363111
```

And therefore, the model will have the following form:

$$ln\left(\frac{p_{i1}}{p_{i6}}\right) = -6.84 + 0.16 \cdot Z_{i1} + 0.88 \cdot Z_{i2} + 0.2 \cdot Z_{i5} + 0.5 \cdot Z_{i6} - 1.5 \cdot Z_{i7} + 0.28 \cdot Z_{i9} + -0.54 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i2}}{p_{i6}}\right) = -2.88 + 0.98 \cdot Z_{i2} + 0.4 \cdot Z_{i4} - 0.89 \cdot Z_{i7} + 0.43 \cdot Z_{i9}$$

$$ln\left(\frac{p_{i3}}{p_{i6}}\right) = 0.14 - 0.41 \cdot Z_{i2} - 0.07 \cdot Z_{i5} + 0.23 \cdot Z_{i6} - 0.58 \cdot Z_{i7} + 0.36 \cdot Z_{i10} + 0.8 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i4}}{p_{i6}}\right) = -1.96 + 0.21 \cdot Z_{i1} - 0.89 \cdot Z_{i2} + 0.11 \cdot Z_{i4} - 0.2 \cdot Z_{i5} + 0.26 \cdot Z_{i6} - - 0.68 \cdot Z_{i7} - 0.14 \cdot Z_{i8} + 0.66 \cdot Z_{i10} + 1.62 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i5}}{p_{i6}}\right) = -8.14 - 0.54 \cdot Z_{i1} + 1.05 \cdot Z_{i3} - 0.84 \cdot Z_{i4} - 1.04 \cdot Z_{i5} + 0.17 \cdot Z_{i6} - - 1.24 \cdot Z_{i7} - 0.55 \cdot Z_{i9} + 0.83 \cdot Z_{i10} + 1.65 \cdot Z_{i11}$$

Which is at least more parsimonious than the last one.

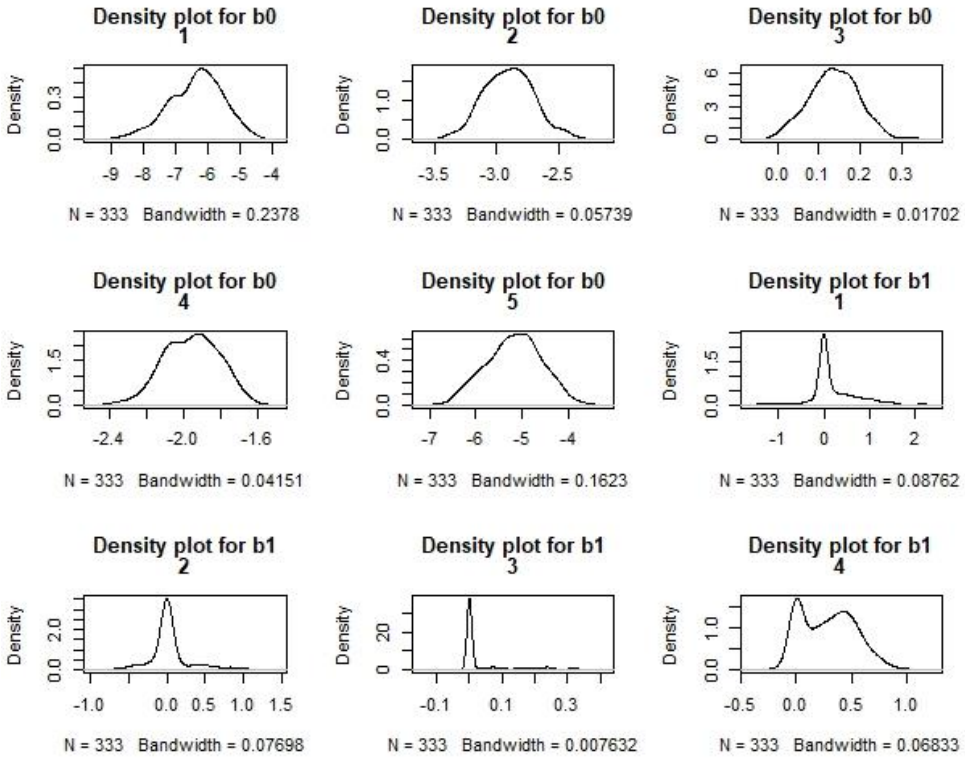
Variable Selection through (Liang et al.) hyper-g (using OpenBUGS):

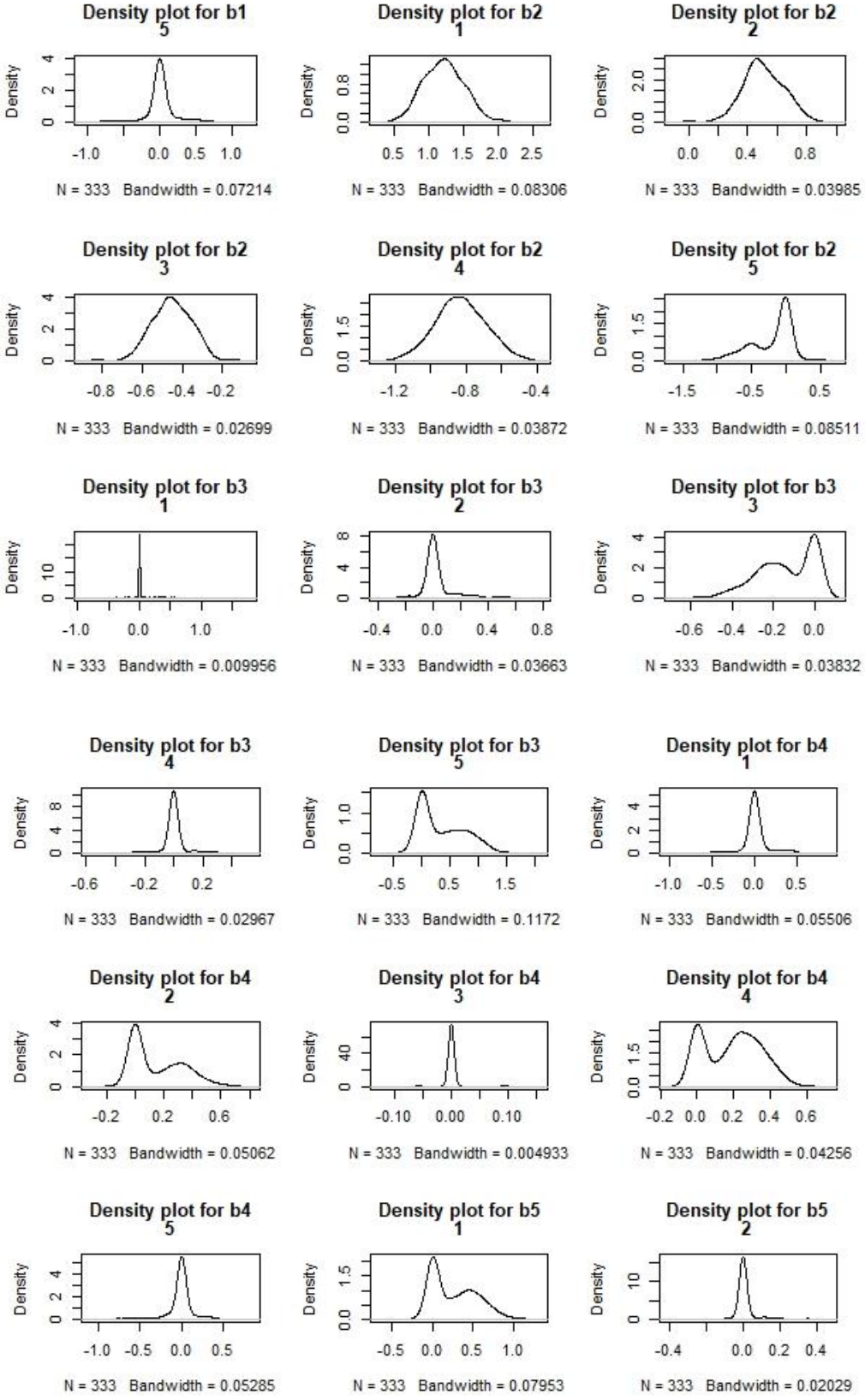
Just like before, only this time in OpenBUGS. So, as always, we will select our most important variables by:

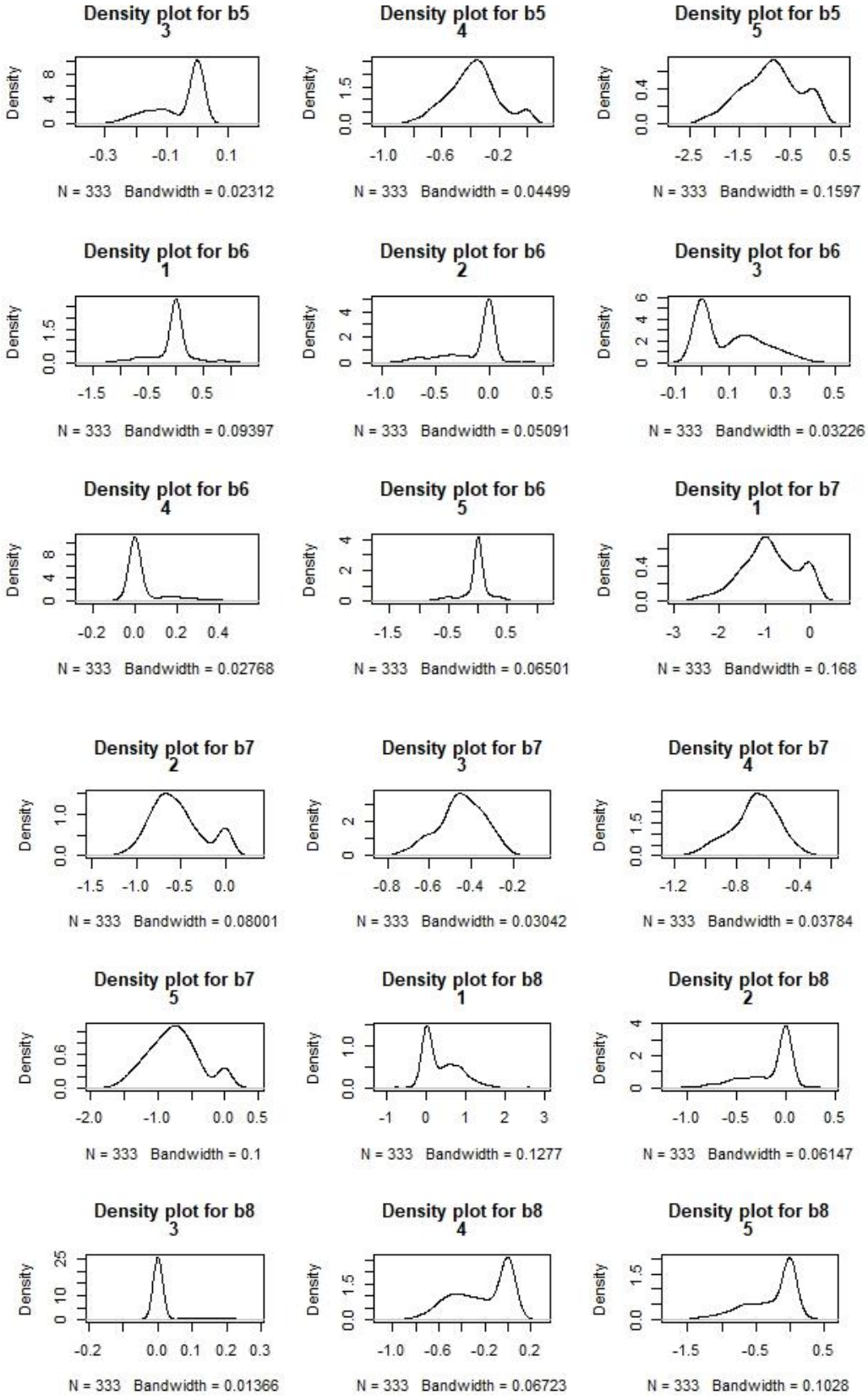
1. Using a Uniform Distribution on the priors:

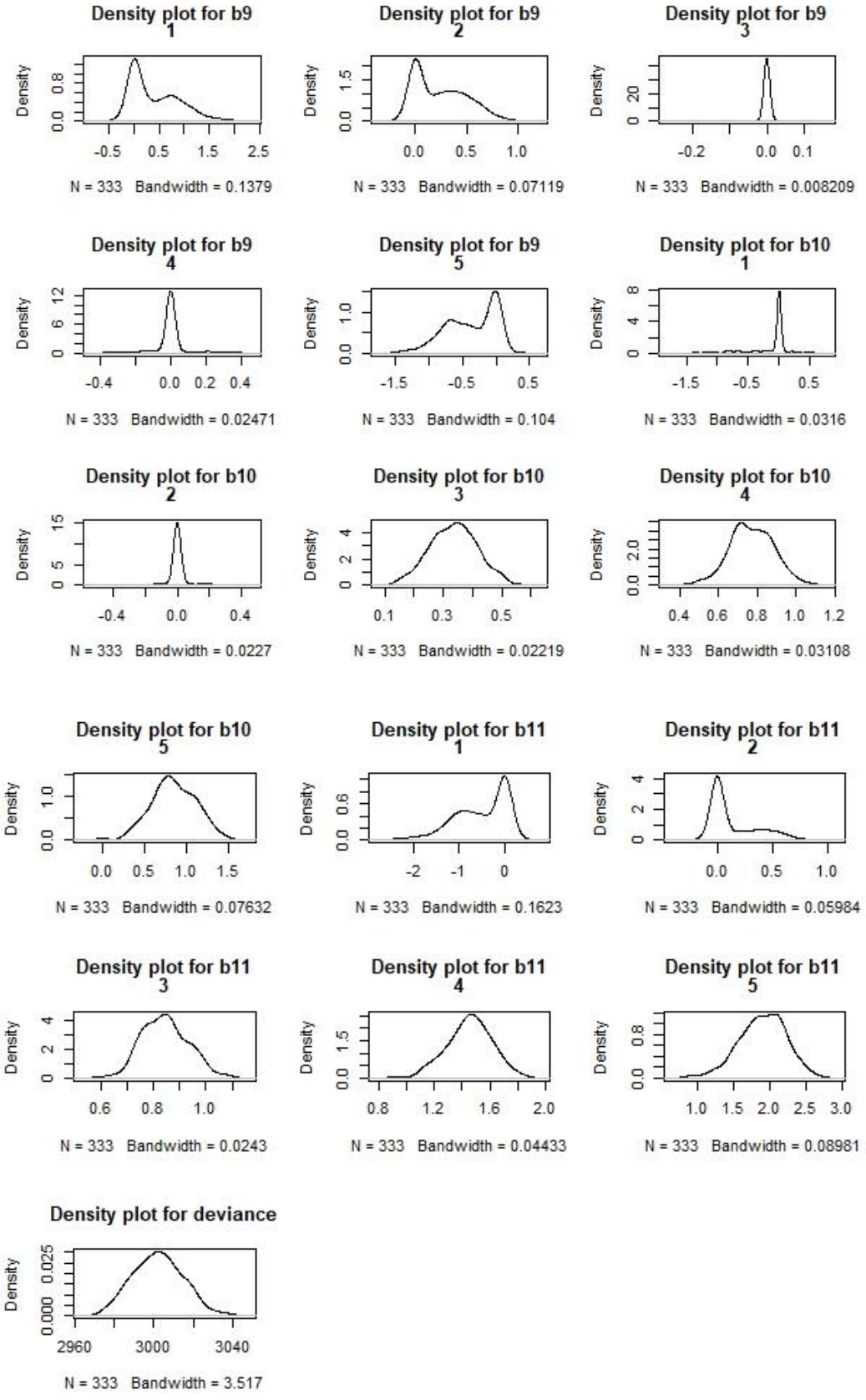
```
Inference for Bugs model at "C:\Users\30697\Desktop\hyperg.txt", fit
using OpenBUGS,
 1 chains, each with 11000 iterations (first 1000 discarded), n.thin =
30
 n.sims = 333 iterations saved
           mean   sd   2.5%    25%    50%    75%  97.5%
alpha[1]   -6.4  0.8   -8.2   -7.0   -6.3   -5.8   -4.9
alpha[2]   -2.9  0.2   -3.3   -3.0   -2.9   -2.8   -2.5
alpha[3]    0.1  0.1    0.0    0.1    0.1    0.2    0.2
alpha[4]   -2.0  0.1   -2.2   -2.1   -2.0   -1.8   -1.7
alpha[5]   -5.2  0.6   -6.3   -5.6   -5.2   -4.8   -4.2
gb1[1]      0.2  0.5   -0.6    0.0    0.0    0.4    1.4
gb1[2]      0.0  0.3   -0.5    0.0    0.0    0.0    0.9
gb1[3]      0.0  0.1    0.0    0.0    0.0    0.0    0.3
gb1[4]      0.3  0.2    0.0    0.0    0.3    0.5    0.8
gb1[5]      0.0  0.3   -0.7    0.0    0.0    0.0    0.7
gb2[1]      1.2  0.3    0.7    1.0    1.2    1.4    1.8
gb2[2]      0.5  0.1    0.2    0.4    0.5    0.6    0.8
gb2[3]     -0.5  0.1   -0.6   -0.5   -0.5   -0.4   -0.3
gb2[4]     -0.8  0.1   -1.1   -0.9   -0.8   -0.7   -0.6
gb2[5]     -0.2  0.3   -0.9   -0.4    0.0    0.0    0.1
gb3[1]      0.1  0.3   -0.4    0.0    0.0    0.0    1.0
gb3[2]      0.0  0.1   -0.2    0.0    0.0    0.0    0.5
gb3[3]     -0.1  0.1   -0.4   -0.2   -0.1    0.0    0.0
gb3[4]      0.0  0.1   -0.3    0.0    0.0    0.0    0.3
gb3[5]      0.4  0.4   -0.1    0.0    0.2    0.7    1.2
gb4[1]      0.0  0.2   -0.5    0.0    0.0    0.0    0.4
gb4[2]      0.2  0.2    0.0    0.0    0.0    0.3    0.5
gb4[3]      0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb4[4]      0.2  0.2    0.0    0.0    0.2    0.3    0.5
gb4[5]      0.0  0.2   -0.6    0.0    0.0    0.0    0.3
gb5[1]      0.3  0.3    0.0    0.0    0.2    0.5    0.8
gb5[2]      0.0  0.1   -0.1    0.0    0.0    0.0    0.2
gb5[3]     -0.1  0.1   -0.2   -0.1    0.0    0.0    0.0
gb5[4]     -0.4  0.2   -0.7   -0.5   -0.4   -0.3    0.0
gb5[5]     -0.9  0.6   -2.1   -1.3   -0.9   -0.5    0.0
gb6[1]     -0.1  0.3   -0.9    0.0    0.0    0.0    0.8
gb6[2]     -0.1  0.2   -0.7   -0.2    0.0    0.0    0.0
gb6[3]      0.1  0.1    0.0    0.0    0.1    0.2    0.4
gb6[4]      0.0  0.1    0.0    0.0    0.0    0.0    0.4
gb6[5]      0.0  0.2   -0.6    0.0    0.0    0.0    0.4
gb7[1]     -0.9  0.6   -2.1   -1.3   -0.9   -0.4    0.0
gb7[2]     -0.5  0.3   -1.0   -0.7   -0.6   -0.4    0.0
gb7[3]     -0.4  0.1   -0.7   -0.5   -0.4   -0.4   -0.3
gb7[4]     -0.7  0.2   -1.0   -0.8   -0.7   -0.6   -0.4
gb7[5]     -0.8  0.4   -1.4   -1.0   -0.8   -0.5    0.0
```

```
gb8[1]      0.4  0.5   -0.1    0.0    0.2    0.7    1.4
gb8[2]     -0.2  0.2   -0.8   -0.3    0.0    0.0    0.0
gb8[3]      0.0  0.0    0.0    0.0    0.0    0.0    0.2
gb8[4]     -0.2  0.2   -0.7   -0.4   -0.2    0.0    0.0
gb8[5]     -0.3  0.4   -1.2   -0.5    0.0    0.0    0.1
gb9[1]      0.4  0.5   -0.1    0.0    0.2    0.8    1.6
gb9[2]      0.2  0.3    0.0    0.0    0.2    0.4    0.8
gb9[3]      0.0  0.0   -0.1    0.0    0.0    0.0    0.0
gb9[4]      0.0  0.1   -0.2    0.0    0.0    0.0    0.2
gb9[5]     -0.4  0.4   -1.1   -0.7   -0.3    0.0    0.0
gb10[1]    -0.1  0.4   -1.2   -0.2    0.0    0.0    0.3
gb10[2]     0.0  0.1   -0.2    0.0    0.0    0.0    0.2
gb10[3]     0.3  0.1    0.2    0.3    0.3    0.4    0.5
gb10[4]     0.8  0.1    0.6    0.7    0.8    0.9    1.0
gb10[5]     0.8  0.3    0.3    0.7    0.8    1.0    1.4
gb11[1]    -0.5  0.6   -1.9   -0.9   -0.4    0.0    0.1
gb11[2]     0.1  0.2    0.0    0.0    0.0    0.3    0.7
gb11[3]     0.8  0.1    0.7    0.8    0.8    0.9    1.0
gb11[4]     1.5  0.2    1.1    1.4    1.5    1.6    1.8
gb11[5]     1.9  0.3    1.2    1.7    1.9    2.1    2.5
deviance 3002.1 12.5 2979.2 2993.4 3001.8 3010.6 3026.4

DIC info (using the rule, pD = var(deviance)/2)
pD = 77.9 and DIC = 3080.0
DIC is an estimate of expected predictive error (lower deviance is
better).
```




Density plot for b1 5
Density plot for b2 1
Density plot for b2 2
Density plot for b2 3
Density plot for b2 4
Density plot for b2 5
Density plot for b3 1
Density plot for b3 2
Density plot for b3 3
Density plot for b3 4
Density plot for b3 5
Density plot for b4 1
Density plot for b4 2
Density plot for b4 3
Density plot for b4 4
Density plot for b4 5
Density plot for b5 1
Density plot for b5 2
Density
N = 333 Bandwidth = 0.07214
N = 333 Bandwidth = 0.08306
N = 333 Bandwidth = 0.03985
N = 333 Bandwidth = 0.02699
N = 333 Bandwidth = 0.03872
N = 333 Bandwidth = 0.08511
N = 333 Bandwidth = 0.009956
N = 333 Bandwidth = 0.03663
N = 333 Bandwidth = 0.03832
N = 333 Bandwidth = 0.02967
N = 333 Bandwidth = 0.1172
N = 333 Bandwidth = 0.05506
N = 333 Bandwidth = 0.05062
N = 333 Bandwidth = 0.004933
N = 333 Bandwidth = 0.04256
N = 333 Bandwidth = 0.05285
N = 333 Bandwidth = 0.07953
N = 333 Bandwidth = 0.02029


Density plot for b5 3
Density plot for b5 4
Density plot for b5 5
Density plot for b6 1
Density plot for b6 2
Density plot for b6 3
Density plot for b6 4
Density plot for b6 5
Density plot for b7 1
Density plot for b7 2
Density plot for b7 3
Density plot for b7 4
Density plot for b7 5
Density plot for b8 1
Density plot for b8 2
Density plot for b8 3
Density plot for b8 4
Density plot for b8 5
Density
N = 333 Bandwidth = 0.02312
N = 333 Bandwidth = 0.04499
N = 333 Bandwidth = 0.1597
N = 333 Bandwidth = 0.09397
N = 333 Bandwidth = 0.05091
N = 333 Bandwidth = 0.03226
N = 333 Bandwidth = 0.02768
N = 333 Bandwidth = 0.06501
N = 333 Bandwidth = 0.168
N = 333 Bandwidth = 0.08001
N = 333 Bandwidth = 0.03042
N = 333 Bandwidth = 0.03784
N = 333 Bandwidth = 0.1
N = 333 Bandwidth = 0.1277
N = 333 Bandwidth = 0.06147
N = 333 Bandwidth = 0.01366
N = 333 Bandwidth = 0.06723
N = 333 Bandwidth = 0.1028


Density plot for b9 1
Density
N = 333 Bandwidth = 0.1379
Density plot for b9 2
Density
N = 333 Bandwidth = 0.07119
Density plot for b9 3
Density
N = 333 Bandwidth = 0.008209
Density plot for b9 4
Density
N = 333 Bandwidth = 0.02471
Density plot for b9 5
Density
N = 333 Bandwidth = 0.104
Density plot for b10 1
Density
N = 333 Bandwidth = 0.0316
Density plot for b10 2
Density
N = 333 Bandwidth = 0.0227
Density plot for b10 3
Density
N = 333 Bandwidth = 0.02219
Density plot for b10 4
Density
N = 333 Bandwidth = 0.03108
Density plot for b10 5
Density
N = 333 Bandwidth = 0.07632
Density plot for b11 1
Density
N = 333 Bandwidth = 0.1623
Density plot for b11 2
Density
N = 333 Bandwidth = 0.05984
Density plot for b11 3
Density
N = 333 Bandwidth = 0.0243
Density plot for b11 4
Density
N = 333 Bandwidth = 0.04433
Density plot for b11 5
Density
N = 333 Bandwidth = 0.08981
Density plot for deviance
Density
N = 333 Bandwidth = 3.517

And so, this model will look like this:

$$ln\left(\frac{p_{i1}}{p_{i6}}\right) = -6.4 + 0.2 \cdot Z_{i1} + 1.2 \cdot Z_{i2} + 0.1 \cdot Z_{i3} + 0.3 \cdot Z_{i5} - 0.1 \cdot Z_{i6} - 0.9 \cdot Z_{i7} + \\ + 0.4 \cdot Z_{i8} + \ 0.4 \cdot Z_{i9} - 0.1 \cdot Z_{i10} - 1.9 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i2}}{p_{i6}}\right) = -2.9 + 0.5 \cdot Z_{i2} + 0.2 \cdot Z_{i4} - 0.1 \cdot Z_{i6} - 0.5 \cdot Z_{i7} - 0.2 \cdot Z_{i8} + 0.2 \cdot Z_{i9} - 0.1 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i3}}{p_{i6}}\right) = 0.1 - 0.5 \cdot Z_{i2} - 0.1 \cdot Z_{i3} - 0.1 \cdot Z_{i5} + 0.1 \cdot Z_{i6} - 0.4 \cdot Z_{i7} + 0.3 \cdot Z_{i10} + 0.8 \cdot Z_{i11}$$

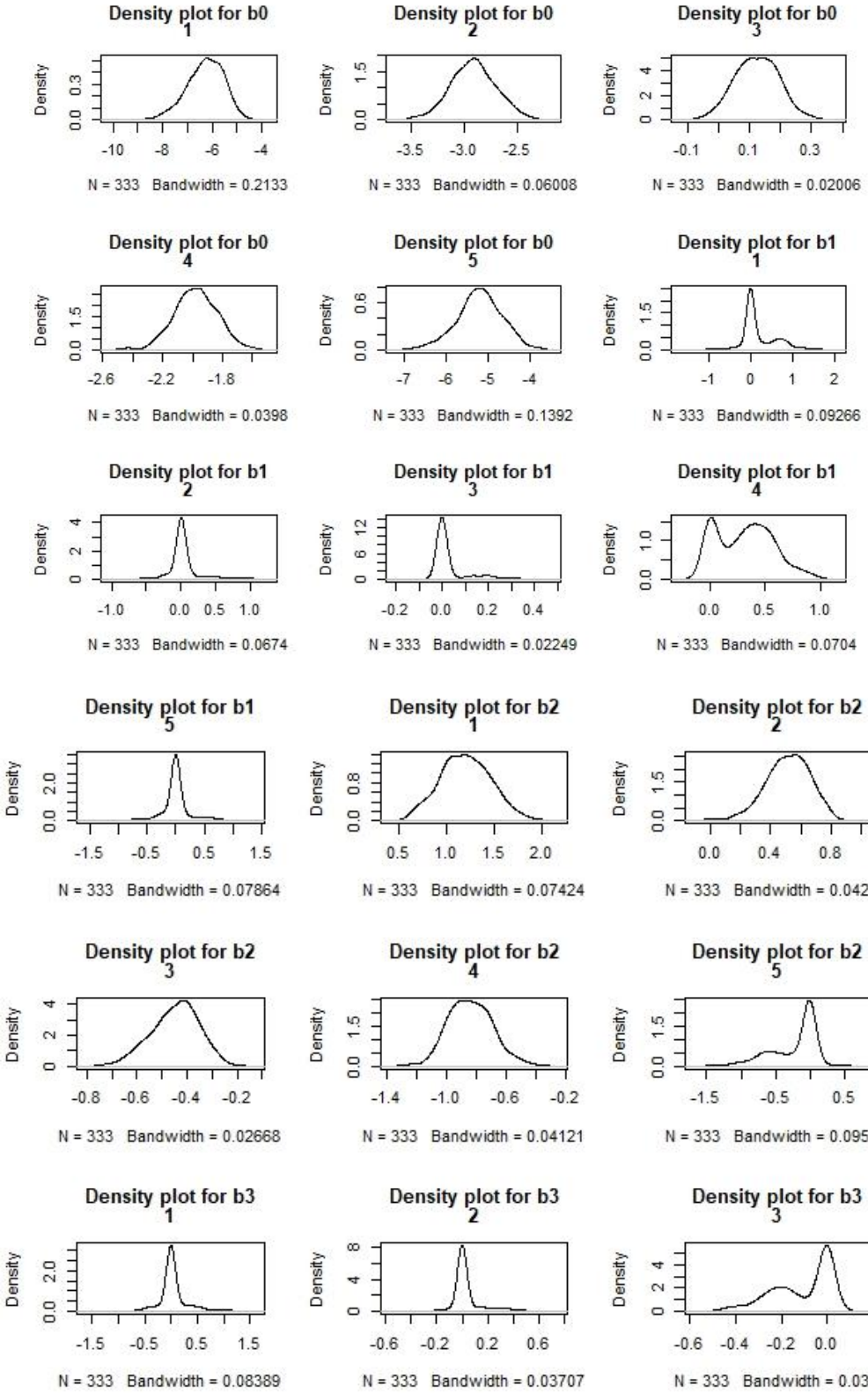
$$ln\left(\frac{p_{i4}}{p_{i6}}\right) = -2 + 0.3 \cdot Z_{i1} - 0.8 \cdot Z_{i2} + 0.2 \cdot Z_{i4} - 0.4 \cdot Z_{i5} - 0.7 \cdot Z_{i7} - 0.2 \cdot Z_{i8} + \\ + 0.8 \cdot Z_{i10} + 1.5 \cdot Z_{i11}$$

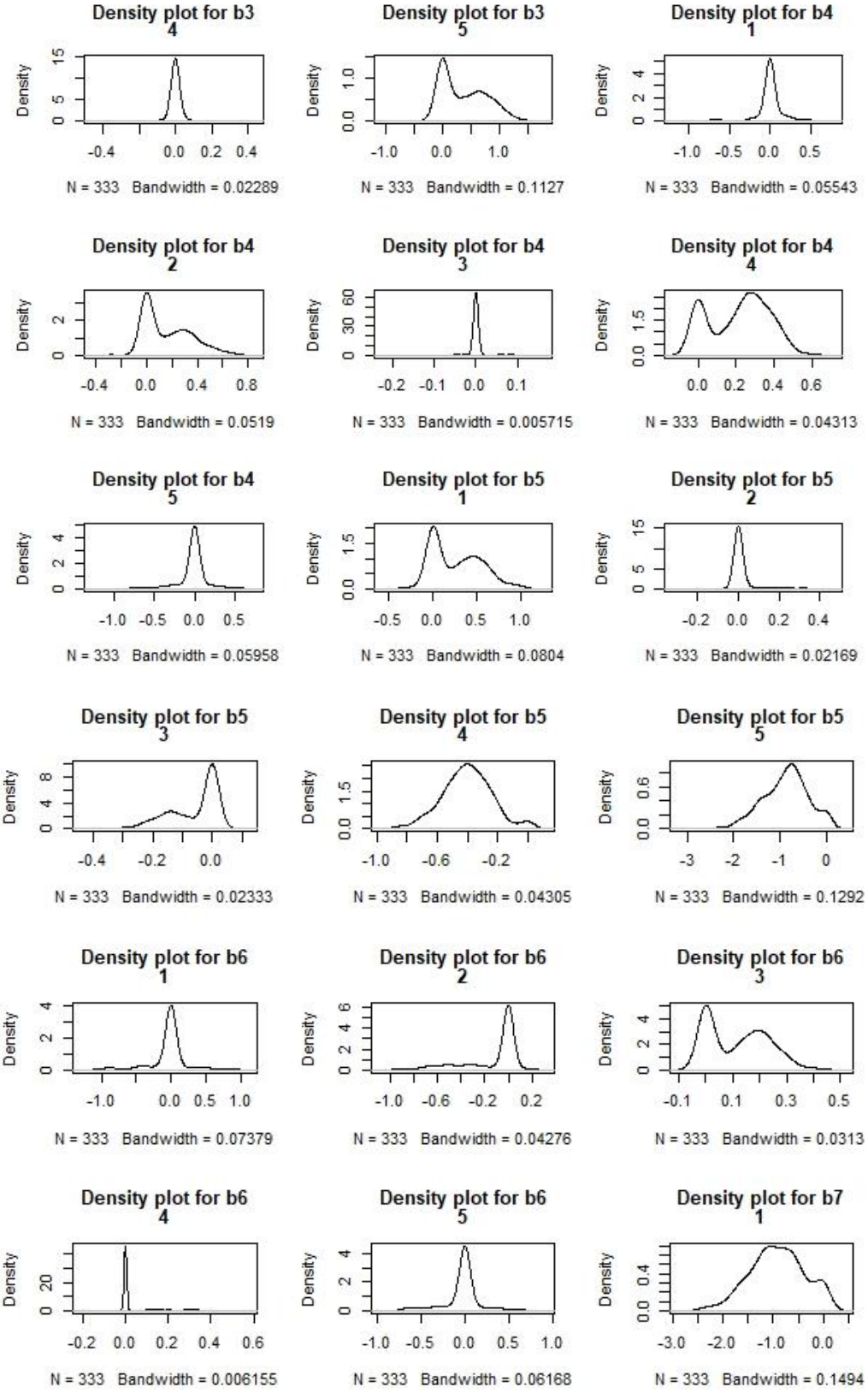
$$ln\left(\frac{p_{i5}}{p_{i6}}\right) = -5.2 - 0.2 \cdot Z_{i2} + 0.4 \cdot Z_{i3} - 0.9 \cdot Z_{i5} - 0.8 \cdot Z_{i7} - 0.3 \cdot Z_{i8} - 0.4 \cdot Z_{i9} + \\ + 0.8 \cdot Z_{i10} + 1.9 \cdot Z_{i11}$$

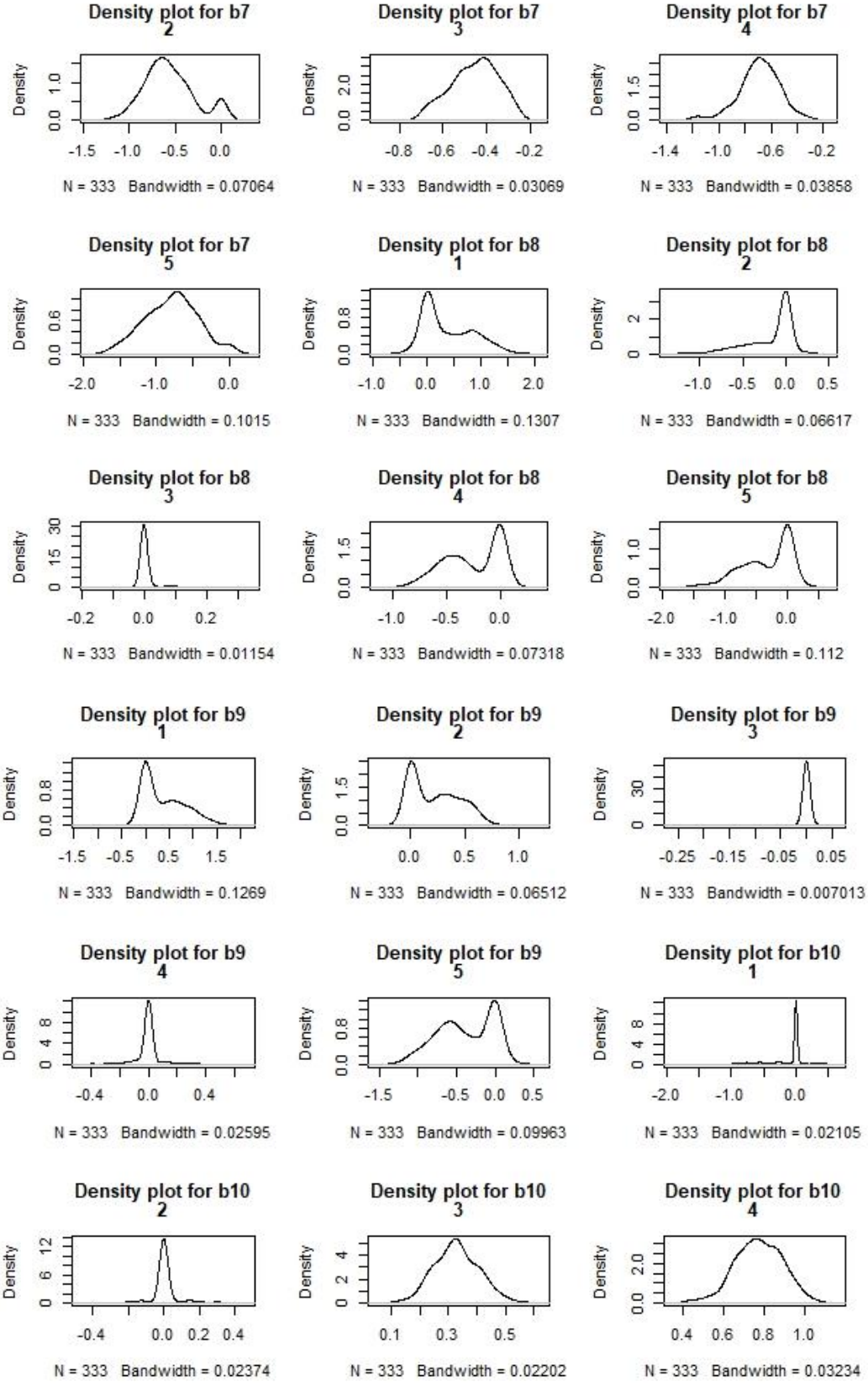
2. Using a Beta-Binomial Distribution on the priors:

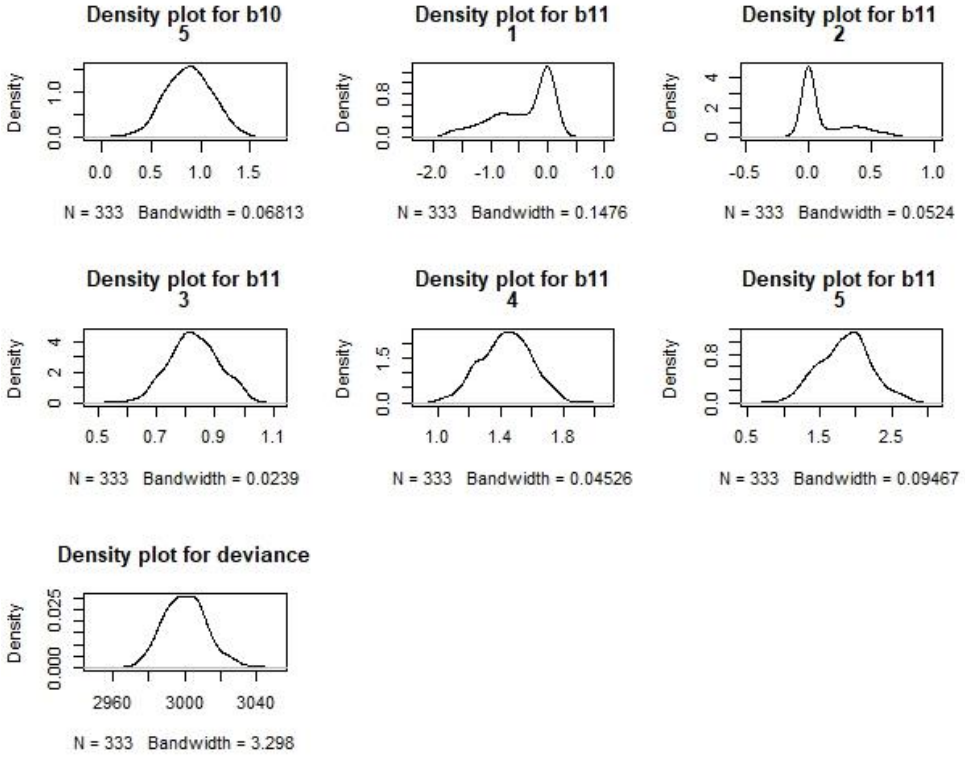
```
Inference for Bugs model at "C:\Users\30697\Desktop\hyperg.txt", fit
using OpenBUGS,
 1 chains, each with 11000 iterations (first 1000 discarded), n.thin =
30
 n.sims = 333 iterations saved
           mean   sd   2.5%    25%    50%    75%  97.5%
alpha[1]   -6.3  0.8   -7.9   -6.8   -6.2   -5.7   -5.0
alpha[2]   -2.9  0.2   -3.3   -3.1   -2.9   -2.8   -2.5
alpha[3]    0.1  0.1    0.0    0.1    0.1    0.2    0.3
alpha[4]   -2.0  0.1   -2.3   -2.1   -2.0   -1.9   -1.7
alpha[5]   -5.2  0.5   -6.3   -5.5   -5.2   -4.9   -4.2
gb1[1]      0.2  0.4   -0.4    0.0    0.0    0.4    1.2
gb1[2]      0.0  0.2   -0.3    0.0    0.0    0.0    0.8
gb1[3]      0.0  0.1    0.0    0.0    0.0    0.0    0.3
gb1[4]      0.3  0.2    0.0    0.0    0.3    0.5    0.8
gb1[5]      0.0  0.3   -0.6    0.0    0.0    0.0    0.7
gb2[1]      1.2  0.3    0.7    1.0    1.2    1.4    1.7
gb2[2]      0.5  0.2    0.2    0.4    0.5    0.6    0.8
gb2[3]     -0.4  0.1   -0.6   -0.5   -0.4   -0.4   -0.3
gb2[4]     -0.8  0.1   -1.1   -0.9   -0.8   -0.7   -0.5
gb2[5]     -0.2  0.3   -1.1   -0.5    0.0    0.0    0.1
gb3[1]      0.0  0.3   -0.5    0.0    0.0    0.0    0.9
gb3[2]      0.0  0.1   -0.1    0.0    0.0    0.0    0.4
gb3[3]     -0.1  0.1   -0.4   -0.2    0.0    0.0    0.0
```

```
gb3[4]      0.0  0.1   -0.2    0.0    0.0    0.0    0.3
gb3[5]      0.4  0.4    0.0    0.0    0.3    0.7    1.2
gb4[1]      0.0  0.2   -0.7    0.0    0.0    0.0    0.4
gb4[2]      0.2  0.2    0.0    0.0    0.1    0.3    0.6
gb4[3]      0.0  0.0    0.0    0.0    0.0    0.0    0.0
gb4[4]      0.2  0.2    0.0    0.0    0.2    0.3    0.5
gb4[5]      0.0  0.2   -0.7    0.0    0.0    0.0    0.4
gb5[1]      0.3  0.3    0.0    0.0    0.2    0.5    0.9
gb5[2]      0.0  0.1   -0.1    0.0    0.0    0.0    0.3
gb5[3]     -0.1  0.1   -0.2   -0.1    0.0    0.0    0.0
gb5[4]     -0.4  0.2   -0.7   -0.5   -0.4   -0.3    0.0
gb5[5]     -0.9  0.5   -1.9   -1.2   -0.8   -0.6    0.0
gb6[1]      0.0  0.3   -0.8    0.0    0.0    0.0    0.5
gb6[2]     -0.1  0.2   -0.7   -0.2    0.0    0.0    0.1
gb6[3]      0.1  0.1    0.0    0.0    0.1    0.2    0.3
gb6[4]      0.1  0.1    0.0    0.0    0.0    0.0    0.4
gb6[5]      0.0  0.2   -0.6    0.0    0.0    0.0    0.5
gb7[1]     -0.9  0.5   -2.0   -1.3   -0.9   -0.6    0.0
gb7[2]     -0.6  0.3   -1.0   -0.7   -0.6   -0.4    0.0
gb7[3]     -0.5  0.1   -0.7   -0.5   -0.4   -0.4   -0.3
gb7[4]     -0.7  0.2   -1.1   -0.8   -0.7   -0.6   -0.4
gb7[5]     -0.8  0.4   -1.5   -1.0   -0.8   -0.5    0.0
gb8[1]      0.4  0.5   -0.2    0.0    0.2    0.8    1.3
gb8[2]     -0.2  0.3   -0.8   -0.3    0.0    0.0    0.1
gb8[3]      0.0  0.0    0.0    0.0    0.0    0.0    0.1
gb8[4]     -0.3  0.3   -0.8   -0.5   -0.2    0.0    0.0
gb8[5]     -0.3  0.4   -1.2   -0.6   -0.2    0.0    0.2
gb9[1]      0.4  0.5   -0.1    0.0    0.2    0.7    1.4
gb9[2]      0.2  0.2    0.0    0.0    0.2    0.4    0.7
gb9[3]      0.0  0.0   -0.1    0.0    0.0    0.0    0.0
gb9[4]      0.0  0.1   -0.2    0.0    0.0    0.0    0.2
gb9[5]     -0.4  0.4   -1.1   -0.7   -0.4    0.0    0.0
gb10[1]    -0.1  0.3   -0.9   -0.1    0.0    0.0    0.3
gb10[2]     0.0  0.1   -0.2    0.0    0.0    0.0    0.2
gb10[3]     0.3  0.1    0.2    0.3    0.3    0.4    0.5
gb10[4]     0.8  0.1    0.5    0.7    0.8    0.9    1.0
gb10[5]     0.9  0.2    0.4    0.7    0.9    1.0    1.3
gb11[1]    -0.4  0.5   -1.6   -0.8   -0.2    0.0    0.1
gb11[2]     0.1  0.2   -0.1    0.0    0.0    0.2    0.6
gb11[3]     0.8  0.1    0.7    0.8    0.8    0.9    1.0
gb11[4]     1.4  0.2    1.1    1.3    1.4    1.5    1.7
gb11[5]     1.8  0.4    1.2    1.6    1.9    2.1    2.6
deviance 3000.3 12.3 2978.0 2991.9 2999.6 3007.6 3025.5

DIC info (using the rule, pD = var(deviance)/2)
pD = 76.2 and DIC = 3076.6
DIC is an estimate of expected predictive error (lower deviance is
better).
```


Density plot for b0 1
Density
N = 333 Bandwidth = 0.2133
Density plot for b0 2
Density
N = 333 Bandwidth = 0.06008
Density plot for b0 3
Density
N = 333 Bandwidth = 0.02006
Density plot for b0 4
Density
N = 333 Bandwidth = 0.0398
Density plot for b0 5
Density
N = 333 Bandwidth = 0.1392
Density plot for b1 1
Density
N = 333 Bandwidth = 0.09266
Density plot for b1 2
Density
N = 333 Bandwidth = 0.0674
Density plot for b1 3
Density
N = 333 Bandwidth = 0.02249
Density plot for b1 4
Density
N = 333 Bandwidth = 0.0704
Density plot for b1 5
Density
N = 333 Bandwidth = 0.07864
Density plot for b2 1
Density
N = 333 Bandwidth = 0.07424
Density plot for b2 2
Density
N = 333 Bandwidth = 0.04226
Density plot for b2 3
Density
N = 333 Bandwidth = 0.02668
Density plot for b2 4
Density
N = 333 Bandwidth = 0.04121
Density plot for b2 5
Density
N = 333 Bandwidth = 0.09508
Density plot for b3 1
Density
N = 333 Bandwidth = 0.08389
Density plot for b3 2
Density
N = 333 Bandwidth = 0.03707
Density plot for b3 3
Density
N = 333 Bandwidth = 0.036


Density plot for b3 4
Density
N = 333 Bandwidth = 0.02289
Density plot for b3 5
Density
N = 333 Bandwidth = 0.1127
Density plot for b4 1
Density
N = 333 Bandwidth = 0.05543
Density plot for b4 2
Density
N = 333 Bandwidth = 0.0519
Density plot for b4 3
Density
N = 333 Bandwidth = 0.005715
Density plot for b4 4
Density
N = 333 Bandwidth = 0.04313
Density plot for b4 5
Density
N = 333 Bandwidth = 0.05958
Density plot for b5 1
Density
N = 333 Bandwidth = 0.0804
Density plot for b5 2
Density
N = 333 Bandwidth = 0.02169
Density plot for b5 3
Density
N = 333 Bandwidth = 0.02333
Density plot for b5 4
Density
N = 333 Bandwidth = 0.04305
Density plot for b5 5
Density
N = 333 Bandwidth = 0.1292
Density plot for b6 1
Density
N = 333 Bandwidth = 0.07379
Density plot for b6 2
Density
N = 333 Bandwidth = 0.04276
Density plot for b6 3
Density
N = 333 Bandwidth = 0.0313
Density plot for b6 4
Density
N = 333 Bandwidth = 0.006155
Density plot for b6 5
Density
N = 333 Bandwidth = 0.06168
Density plot for b7 1
Density
N = 333 Bandwidth = 0.1494


Density plot for b7 2
Density
N = 333 Bandwidth = 0.07064
Density plot for b7 3
Density
N = 333 Bandwidth = 0.03069
Density plot for b7 4
Density
N = 333 Bandwidth = 0.03858
Density plot for b7 5
Density
N = 333 Bandwidth = 0.1015
Density plot for b8 1
Density
N = 333 Bandwidth = 0.1307
Density plot for b8 2
Density
N = 333 Bandwidth = 0.06617
Density plot for b8 3
Density
N = 333 Bandwidth = 0.01154
Density plot for b8 4
Density
N = 333 Bandwidth = 0.07318
Density plot for b8 5
Density
N = 333 Bandwidth = 0.112
Density plot for b9 1
Density
N = 333 Bandwidth = 0.1269
Density plot for b9 2
Density
N = 333 Bandwidth = 0.06512
Density plot for b9 3
Density
N = 333 Bandwidth = 0.007013
Density plot for b9 4
Density
N = 333 Bandwidth = 0.02595
Density plot for b9 5
Density
N = 333 Bandwidth = 0.09963
Density plot for b10 1
Density
N = 333 Bandwidth = 0.02105
Density plot for b10 2
Density
N = 333 Bandwidth = 0.02374
Density plot for b10 3
Density
N = 333 Bandwidth = 0.02202
Density plot for b10 4
Density
N = 333 Bandwidth = 0.03234



And so, this model will be something like this:

$$ln\left(\frac{p_{i1}}{p_{i6}}\right) = -6.3 + 0.2 \cdot Z_{i1} + 1.2 \cdot Z_{i2} + 0.3 \cdot Z_{i5} - 0.9 \cdot Z_{i7} + 0.4 \cdot Z_{i8} + 0.4 \cdot Z_{i9} - \\ - 0.1 \cdot Z_{i10} - 0.4 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i2}}{p_{i6}}\right) = -2.9 + 0.5 \cdot Z_{i2} + 0.2 \cdot Z_{i4} - 0.1 \cdot Z_{i6} - 0.6 \cdot Z_{i7} - 0.2 \cdot Z_{i8} + 0.2 \cdot Z_{i9} + 0.1 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i3}}{p_{i6}}\right) = 0.1 - 0.4 \cdot Z_{i2} - 0.1 \cdot Z_{i3} - 0.1 \cdot Z_{i5} + 0.1 \cdot Z_{i6} - 0.5 \cdot Z_{i7} + 0.3 \cdot Z_{i10} + 0.8 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i4}}{p_{i6}}\right) = -2 + 0.3 \cdot Z_{i1} - 0.8 \cdot Z_{i2} + 0.2 \cdot Z_{i4} - 0.4 \cdot Z_{i5} - 0.7 \cdot Z_{i7} - 0.3 \cdot Z_{i8} + \\ + 0.8 \cdot Z_{i10} + 1.4 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i5}}{p_{i6}}\right) = -5.2 - 0.2 \cdot Z_{i2} + 0.4 \cdot Z_{i3} - 0.9 \cdot Z_{i5} - 0.8 \cdot Z_{i7} - 0.3 \cdot Z_{i8} - 0.4 \cdot Z_{i9} + \\ + 0.9 \cdot Z_{i10} + 1.8 \cdot Z_{i11}$$

Conclusion (of Variable Selection):

We saw different methods and their results, as far as variable selection is concerned. The idea is simple, and their implementation differs in the form of the prior distribution that we choose each time for the parameters β.

What we would like to do, is to pick only the most necessary variables for our final model. While other methods (such as g-prior or BIC prior) are worth considering, they get rid of too many variables to the point of making our new model redundant when compared to the original. The table below is a synopsis of their Deviance Information Criterion values:

MODEL	DIC VALUE
Original Model	**3036.0**
Reduced Model (Variable Selection through BIC prior on betas with beta-binomial on the model)	**3063.0**
Reduced Model (Variable Selection through g-prior (of Liang et al.) prior on betas with beta-binomial on the model)	**3058.0**

And thus, we conclude that none of these models are better than the original. Therefore, we shall choose the *hyper-g prior* on our βs, which utilizes the *beta-binomial* for the model so as to at least have a modicum of parsimony in the final result. We will be using the following variables on each level:

1. *Level 1 (Category 1 with reference level of 6):* used the intercept (β_0), fixed acidity (X_1), volatile acidity (X_2), chlorides (X_5), free sulfur dioxide (X_6), total sulfur dioxide (X_7), pH (X_9) and alcohol (X_{11}).

2. *Level 2 (Category 2 with reference level of 6):* used the intercept (β_0), volatile acidity (X_2), residual sugar (X_4), total sulfur dioxide (X_7) and pH (X_9).

3. *Level 3 (Category 3 with reference level of 6):* used the intercept (β_0), volatile acidity (X_2), chlorides (X_5), free sulfur dioxide (X_6), total sulfur dioxide (X_7), sulphates (X_{10}) and alcohol (X_{11}).

4. *Level 4 (Category 4 with reference level of 6):* used the intercept (β_0), fixed acidity (X_1), volatile acidity (X_2), residual sugar (X_4), chlorides (X_5), free sulfur dioxide (X_6), total sulfur dioxide (X_7), density (X_8), sulphates (X_{10}) and alcohol (X_{11}).

5. *Level 5 (Category 5 with reference level of 6):* used the intercept (β_0), fixed acidity (X_1),citric acid (X_3), residual sugar (X_4), chlorides (X_5), free sulfur dioxide (X_6), total sulfur dioxide (X_7), pH (X_9), sulphates (X_{10}) and alcohol (X_{11}).

It looks like all the physiochemical characteristics are used in this model, although not on all levels. Time to run our model one more time, without all the superfluous variables that were included in the original.

Posterior analysis and interpretation of the results:

Should we run this much simpler model, which contains only the statistically significant variables for each level, we would get the following convergence diagnostics and graphs:

```
GEWEKE CONVERGENCE DIAGNOSTIC (Z-score)
=======================================

Iterations used = 1001:50951
Thinning interval = 50
Sample size per chain = 1000

$chain1

Fraction in 1st window = 0.1
Fraction in 2nd window = 0.5

     b01      B01      b02      B02      b03      B03      b04      B04
-0.20476 -0.26875  1.91301  1.59428 -0.15938 -0.18603  0.56523  0.50987
     b05      B05     b103     B103     b104     B104     b105     B105
 0.06036  0.27307  1.65632  1.61372  0.77819  0.76217  0.85338  1.05632
     b11      B11     b111     B111     b113     B113     b114     B114
 0.39281  0.03356  0.16221  0.27720 -1.30230 -1.20170 -0.65869 -0.61384
    b115     B115      b14      B14      b15      B15      b21      B21
-0.95878 -0.71558  0.77899  0.66690  0.60275  0.20317  0.21143 -0.00659
     b22      B22      b23      B23      b24      B24      b35      B35
-1.80271 -2.03212  0.45908  0.46769  0.77868  0.61238 -0.21261 -0.33850
     b42      B42      b44      B44      b45      B45      b51      B51
 0.25064 -0.00200  1.30299  1.12780 -0.05491  0.03058 -0.21426 -0.37251
     b53      B53      b54      B54      b55      B55      b61      B61
-0.42969 -0.50951 -0.06167 -0.09627 -0.12096  0.05724  0.32938  0.82647
     b63      B63      b64      B64      b65      B65      b71      B71
 1.61524  1.66812  0.61970  0.70173  0.67484  1.20563 -0.25496  0.14864
     b72      B72      b73      B73      b74      B74      b75      B75
 0.48214  0.30838 -0.08973 -0.00715 -0.89042 -0.85021 -0.26975  0.15530
     b84      B84      b91      B91      b92      B92      b95      B95
-0.20844 -0.18607 -1.15582 -1.55224  0.64378  0.44617  0.33552  0.51150
deviance
-1.10274
```

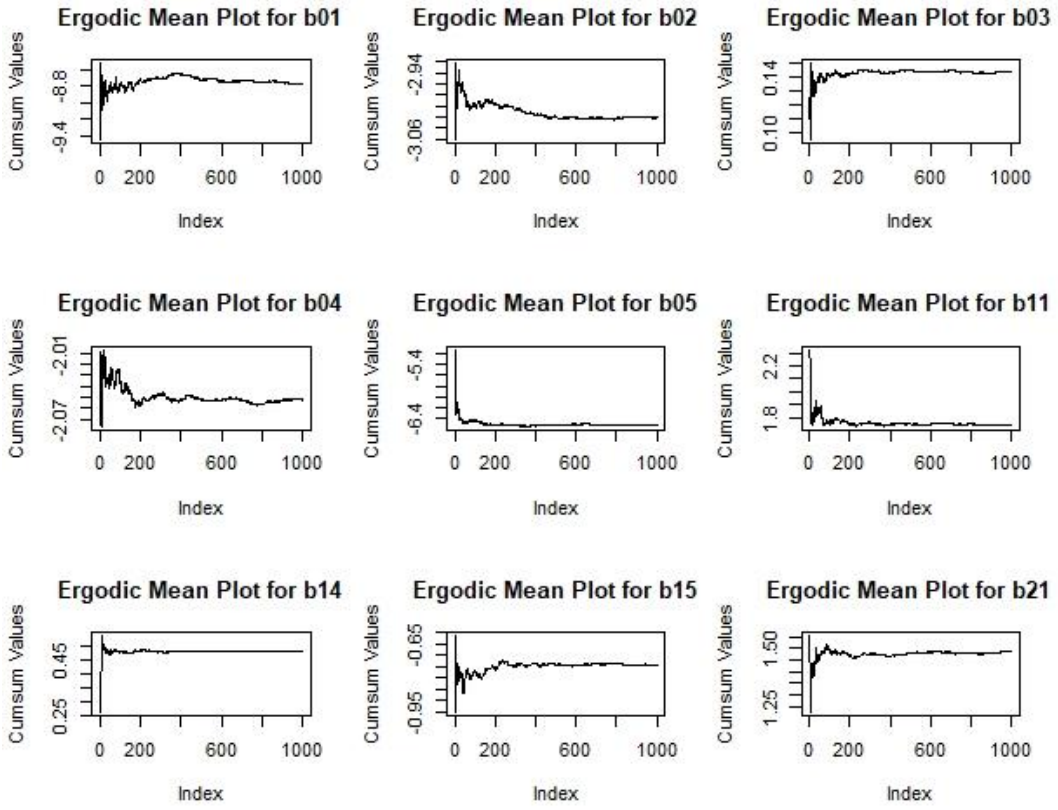
```
HEIDELBERGER AND WELCH STATIONARITY AND INTERVAL HALFWIDTH TESTS
===============================================================

Iterations used = 1001:50951
```

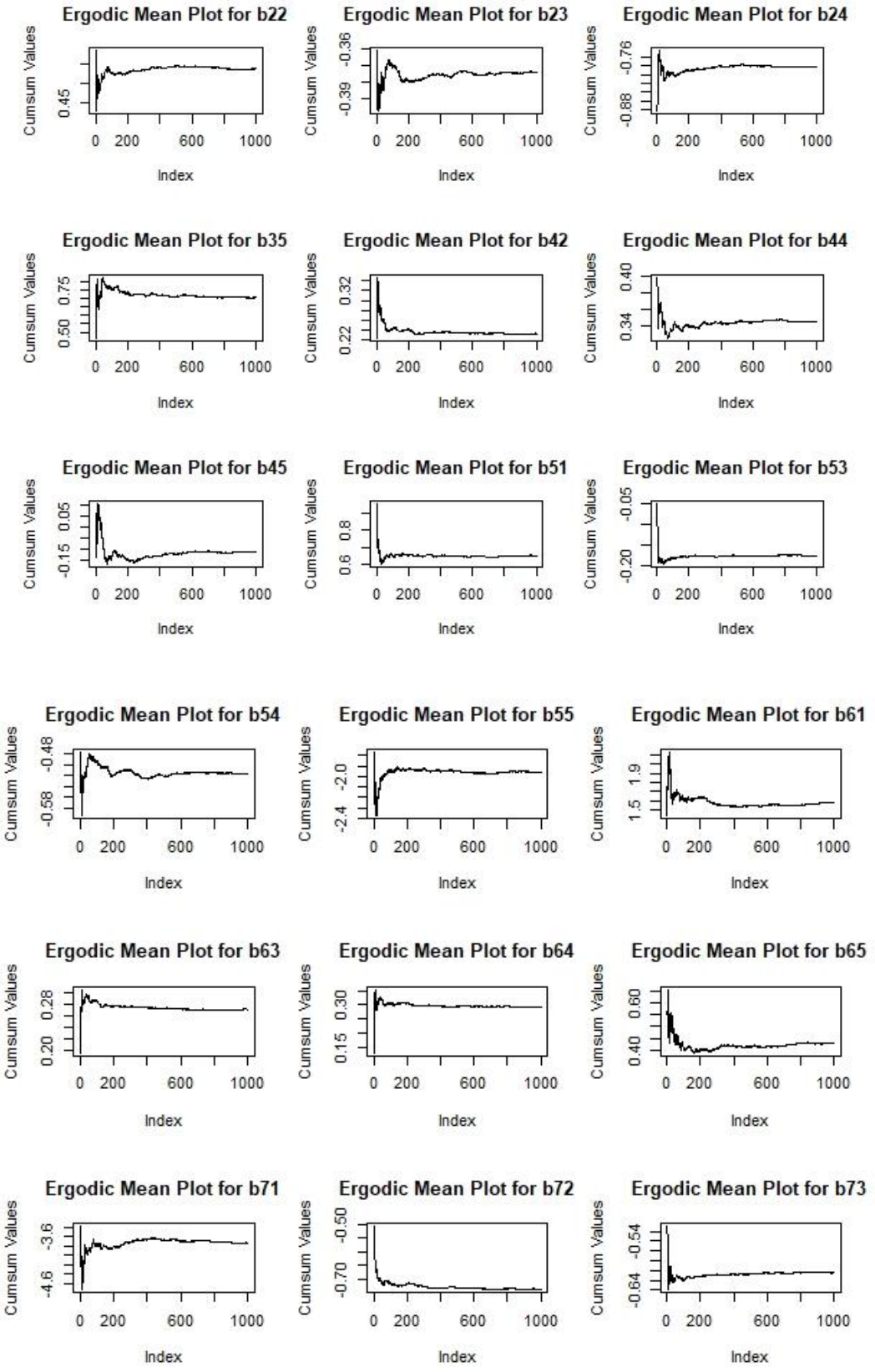
```
Thinning interval = 50
Sample size per chain = 1000

Precision of halfwidth test = 0.1

$chain1

         Stationarity start     p-value
         test         iteration
b01      passed         1       0.7111
B01      passed         1       0.7551
b02      passed         1       0.2048
B02      passed         1       0.2323
b03      passed         1       0.8287
B03      passed         1       0.8587
b04      passed         1       0.3420
B04      passed         1       0.4746
b05      passed         1       0.9057
B05      passed         1       0.7010
b103     passed         1       0.5418
B103     passed         1       0.5634
b104     passed         1       0.7611
B104     passed         1       0.7623
b105     passed         1       0.5258
B105     passed         1       0.4630
b11      passed         1       0.9614
B11      passed         1       0.9472
b111     passed         1       0.4913
B111     passed         1       0.3787
b113     passed         1       0.7410
B113     passed         1       0.7564
b114     passed         1       0.9030
B114     passed         1       0.9311
b115     passed         1       0.9153
B115     passed         1       0.9494
b14      passed         1       0.7338
B14      passed         1       0.6890
b15      passed         1       0.9916
B15      passed         1       0.9115
b21      passed         1       0.3615
B21      passed         1       0.3369
b22      passed         1       0.1210
B22      passed         1       0.0957
b23      passed         1       0.9838
B23      passed         1       0.9843
b24      passed         1       0.8962
B24      passed         1       0.9292
b35      passed         1       0.8978
B35      passed         1       0.9563
b42      passed         1       0.5880
B42      passed         1       0.5401
b44      passed         1       0.6320
B44      passed         1       0.6705
```

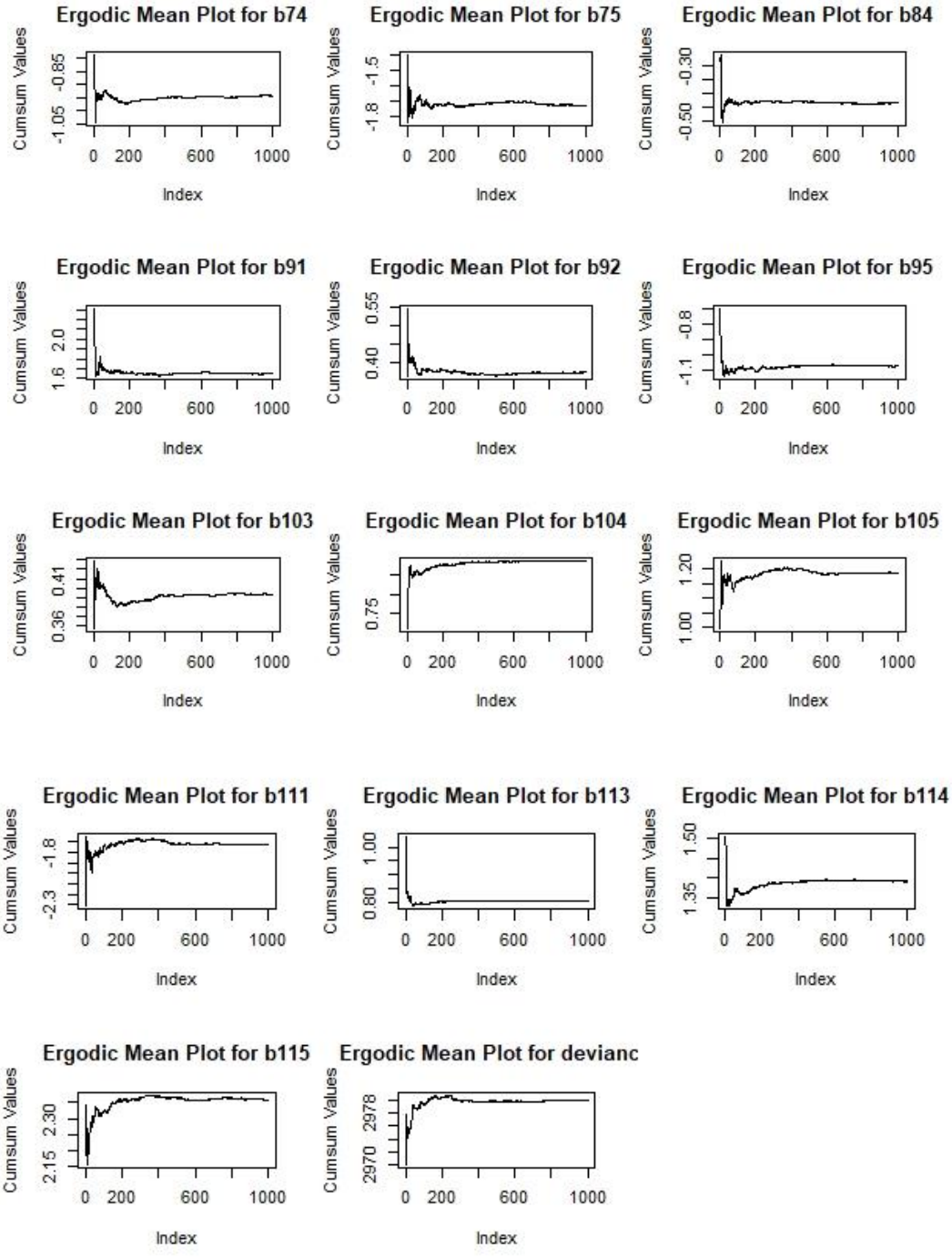
```
b45      passed         1       0.1784
B45      passed         1       0.1814
b51      passed         1       0.1671
B51      passed         1       0.2293
b53      passed         1       0.9044
B53      passed         1       0.8696
b54      passed         1       0.7560
B54      passed         1       0.7141
b55      passed         1       0.6099
B55      passed         1       0.6156
b61      passed         1       0.4318
B61      passed         1       0.0534
b63      passed         1       0.3285
B63      passed         1       0.3292
b64      passed         1       0.5903
B64      passed         1       0.5913
b65      passed         1       0.8239
B65      passed         1       0.8248
b71      passed         1       0.2836
B71      passed         1       0.8461
b72      passed         1       0.1399
B72      passed         1       0.3212
b73      passed         1       0.2287
B73      passed         1       0.1963
b74      passed         1       0.4157
B74      passed         1       0.4185
b75      passed         1       0.7521
B75      passed         1       0.8142
b84      passed         1       0.9514
B84      passed         1       0.9771
b91      passed         1       0.3171
B91      passed         1       0.2509
b92      passed         1       0.7585
B92      passed         1       0.7838
b95      passed         1       0.8984
B95      passed         1       0.9168
deviance passed       201       0.2551

         Halfwidth Mean      Halfwidth
         test
b01      passed    -8.77e+00 1.88e-01
B01      failed     3.33e-04 4.79e-05
b02      passed    -3.02e+00 1.25e-02
B02      passed     4.97e-02 6.18e-04
b03      passed     1.43e-01 3.92e-03
B03      passed     1.16e+00 4.55e-03
b04      passed    -2.05e+00 1.01e-02
B04      passed     1.30e-01 1.32e-03
b05      passed    -6.50e+00 6.65e-02
B05      passed     1.94e-03 1.13e-04
b103     passed     3.94e-01 4.76e-03
B103     passed     1.49e+00 7.15e-03
b104     passed     8.86e-01 7.02e-03
```

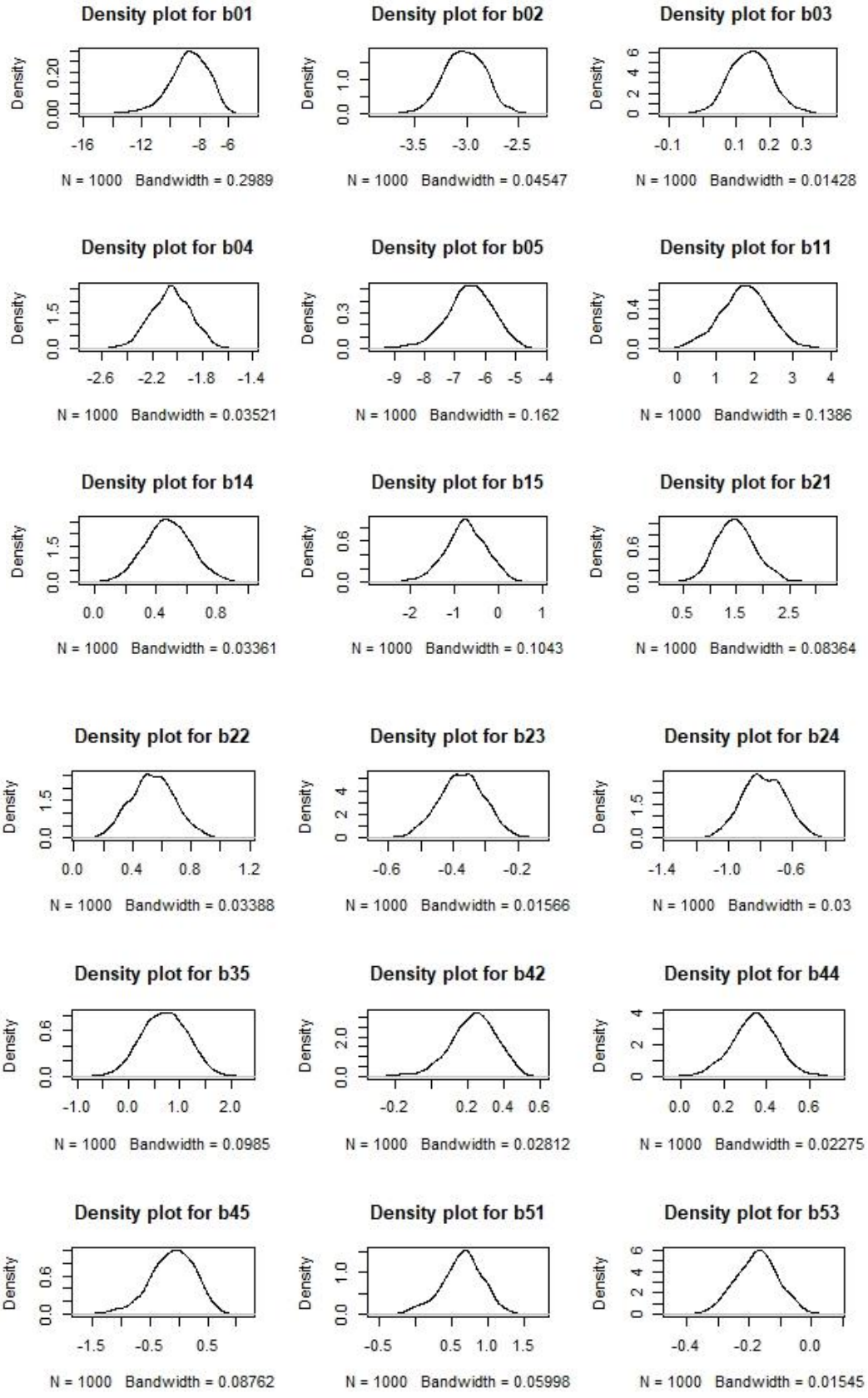
```
B104      passed     2.44e+00 1.71e-02
b105      passed     1.19e+00 1.91e-02
B105      passed     3.39e+00 6.90e-02
b11       passed     1.75e+00 4.21e-02
B11       passed     6.91e+00 3.02e-01
b111      passed    -1.72e+00 7.76e-02
B111      passed     2.34e-01 1.49e-02
b113      passed     8.04e-01 5.02e-03
B113      passed     2.24e+00 1.13e-02
b114      passed     1.39e+00 9.49e-03
B114      passed     4.06e+00 3.90e-02
b115      passed     2.36e+00 2.67e-02
B115      passed     1.11e+01 3.09e-01
b14       passed     4.78e-01 1.09e-02
B14       passed     1.63e+00 1.78e-02
b15       passed    -7.43e-01 3.46e-02
B15       passed     5.28e-01 1.67e-02
b21       passed     1.49e+00 3.48e-02
B21       passed     4.76e+00 1.83e-01
b22       passed     5.39e-01 1.27e-02
B22       passed     1.73e+00 2.22e-02
b23       passed    -3.74e-01 4.38e-03
B23       passed     6.90e-01 3.02e-03
b24       passed    -7.83e-01 8.23e-03
B24       passed     4.61e-01 3.78e-03
b35       passed     7.04e-01 2.70e-02
B35       passed     2.22e+00 6.17e-02
b42       passed     2.33e-01 7.79e-03
B42       passed     1.27e+00 9.68e-03
b44       passed     3.45e-01 7.33e-03
B44       passed     1.42e+00 1.04e-02
b45       failed    -1.12e-01 2.40e-02
B45       passed     9.60e-01 2.19e-02
b51       passed     6.48e-01 1.90e-02
B51       passed     1.99e+00 3.68e-02
b53       passed    -1.74e-01 4.24e-03
B53       passed     8.42e-01 3.56e-03
b54       passed    -5.18e-01 1.09e-02
B54       passed     6.04e-01 6.46e-03
b55       passed    -1.96e+00 4.62e-02
B55       passed     1.80e-01 7.64e-03
b61       passed     1.58e+00 8.66e-02
B61       failed     7.48e+00 1.43e+00
b63       passed     2.71e-01 5.28e-03
B63       passed     1.32e+00 6.98e-03
b64       passed     2.91e-01 8.85e-03
B64       passed     1.35e+00 1.21e-02
b65       passed     4.29e-01 2.62e-02
B65       passed     1.67e+00 4.43e-02
b71       passed    -3.74e+00 1.78e-01
B71       failed     5.82e-02 7.55e-03
b72       passed    -7.38e-01 1.17e-02
B72       passed     4.87e-01 5.64e-03
```

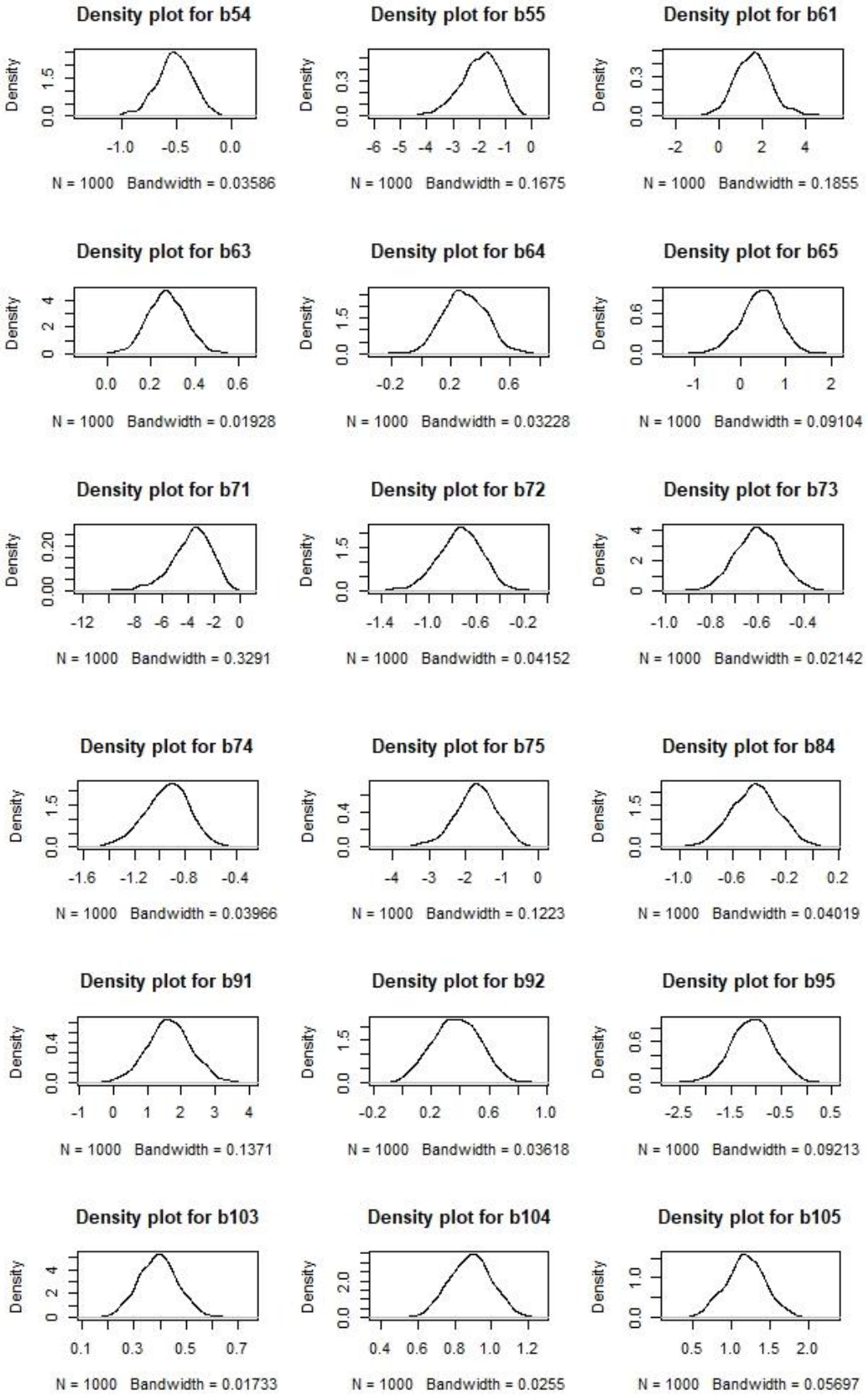
```
b73       passed    -6.05e-01 5.61e-03
B73       passed     5.49e-01 3.22e-03
b74       passed    -9.44e-01 1.09e-02
B74       passed     3.95e-01 4.25e-03
b75       passed    -1.73e+00 3.96e-02
B75       passed     2.08e-01 8.39e-03
b84       passed    -4.36e-01 1.34e-02
B84       passed     6.57e-01 8.86e-03
b91       passed     1.65e+00 4.78e-02
B91       passed     6.35e+00 3.41e-01
b92       passed     3.72e-01 9.92e-03
B92       passed     1.47e+00 1.46e-02
b95       passed    -1.07e+00 2.55e-02
B95       passed     3.73e-01 9.76e-03
deviance  passed     2.98e+03 7.54e-01
```

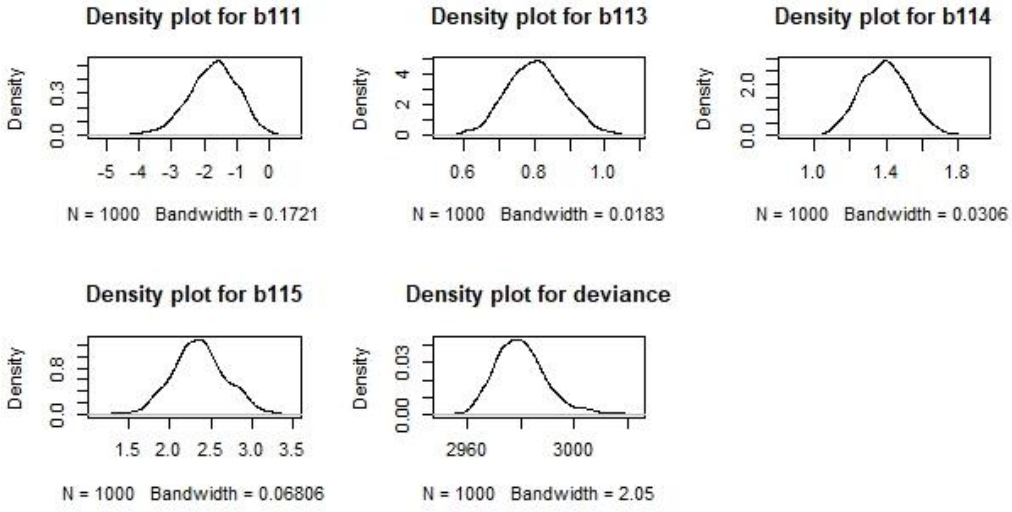


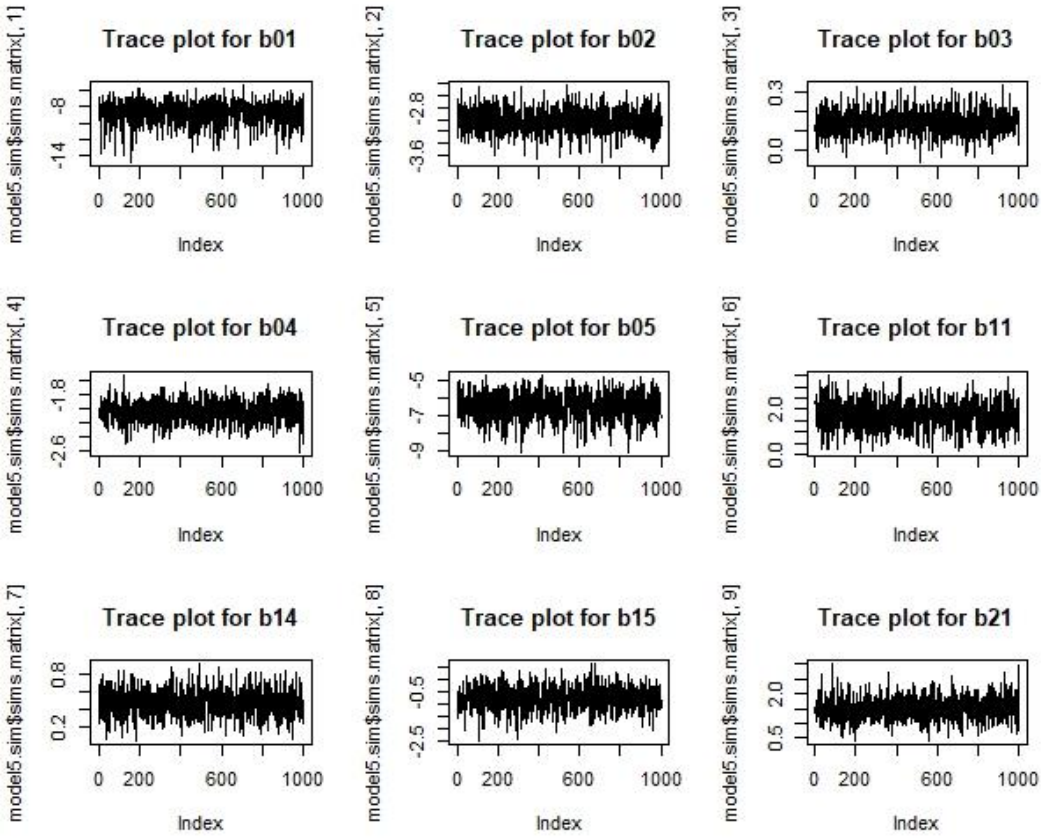

Ergodic Mean Plot for b22
Ergodic Mean Plot for b23
Ergodic Mean Plot for b24
Ergodic Mean Plot for b35
Ergodic Mean Plot for b42
Ergodic Mean Plot for b44
Ergodic Mean Plot for b45
Ergodic Mean Plot for b51
Ergodic Mean Plot for b53
Ergodic Mean Plot for b54
Ergodic Mean Plot for b55
Ergodic Mean Plot for b61
Ergodic Mean Plot for b63
Ergodic Mean Plot for b64
Ergodic Mean Plot for b65
Ergodic Mean Plot for b71
Ergodic Mean Plot for b72
Ergodic Mean Plot for b73
Cumsum Values
Index
0
200
600
1000

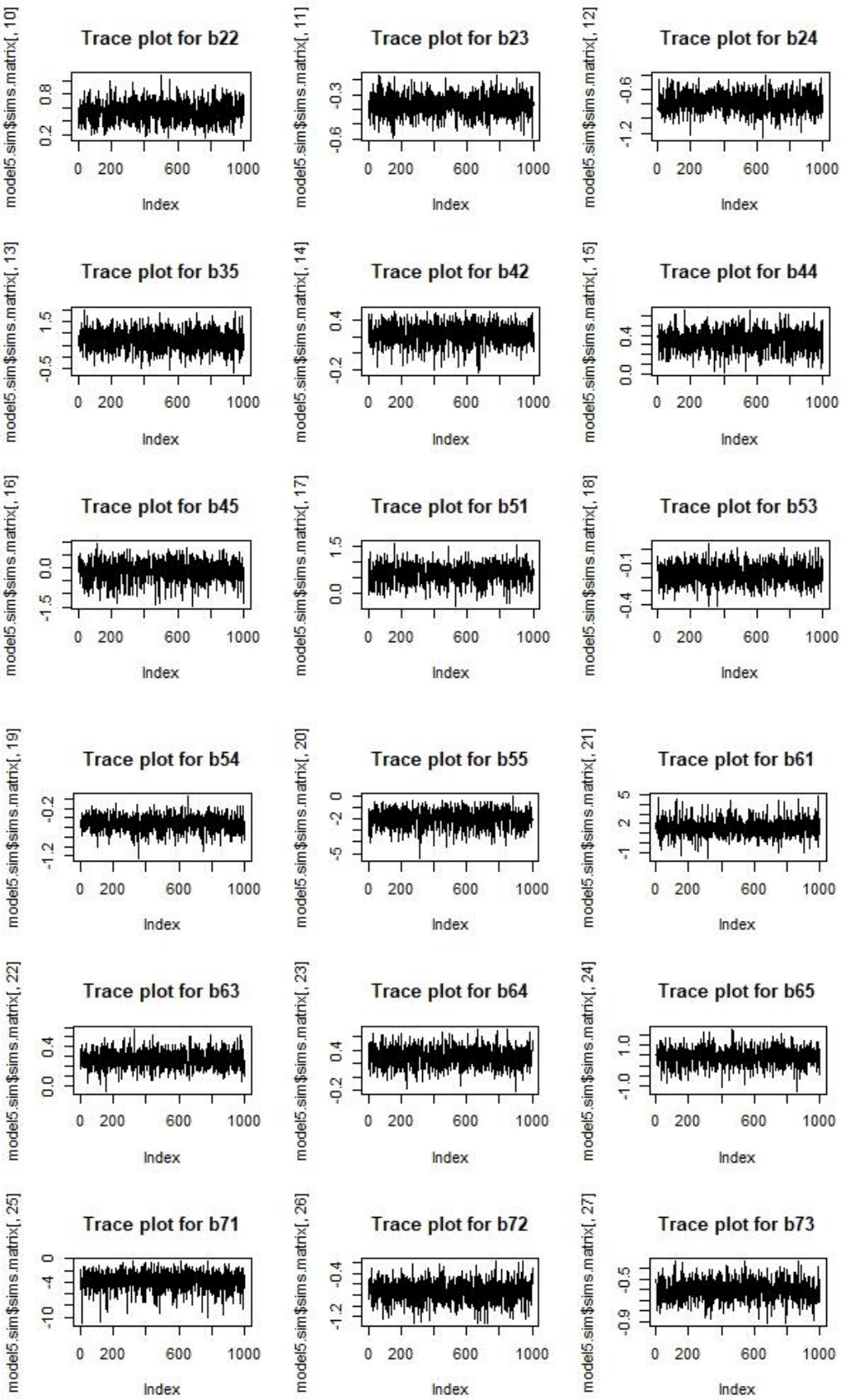

Ergodic Mean Plot for b74
Cumsum Values
-1.05 -0.85
0 200 600 1000
Index
Ergodic Mean Plot for b75
Cumsum Values
-1.8 -1.5
0 200 600 1000
Index
Ergodic Mean Plot for b84
Cumsum Values
-0.50 -0.30
0 200 600 1000
Index
Ergodic Mean Plot for b91
Cumsum Values
1.6 2.0
0 200 600 1000
Index
Ergodic Mean Plot for b92
Cumsum Values
0.40 0.55
0 200 600 1000
Index
Ergodic Mean Plot for b95
Cumsum Values
-1.1 -0.8
0 200 600 1000
Index
Ergodic Mean Plot for b103
Cumsum Values
0.36 0.41
0 200 600 1000
Index
Ergodic Mean Plot for b104
Cumsum Values
0.75
0 200 600 1000
Index
Ergodic Mean Plot for b105
Cumsum Values
1.00 1.20
0 200 600 1000
Index
Ergodic Mean Plot for b111
Cumsum Values
-2.3 -1.8
0 200 600 1000
Index
Ergodic Mean Plot for b113
Cumsum Values
0.80 1.00
0 200 600 1000
Index
Ergodic Mean Plot for b114
Cumsum Values
1.35 1.50
0 200 600 1000
Index
Ergodic Mean Plot for b115
Cumsum Values
2.15 2.30
0 200 600 1000
Index
Ergodic Mean Plot for devianc
Cumsum Values
2970 2978
0 200 600 1000
Index

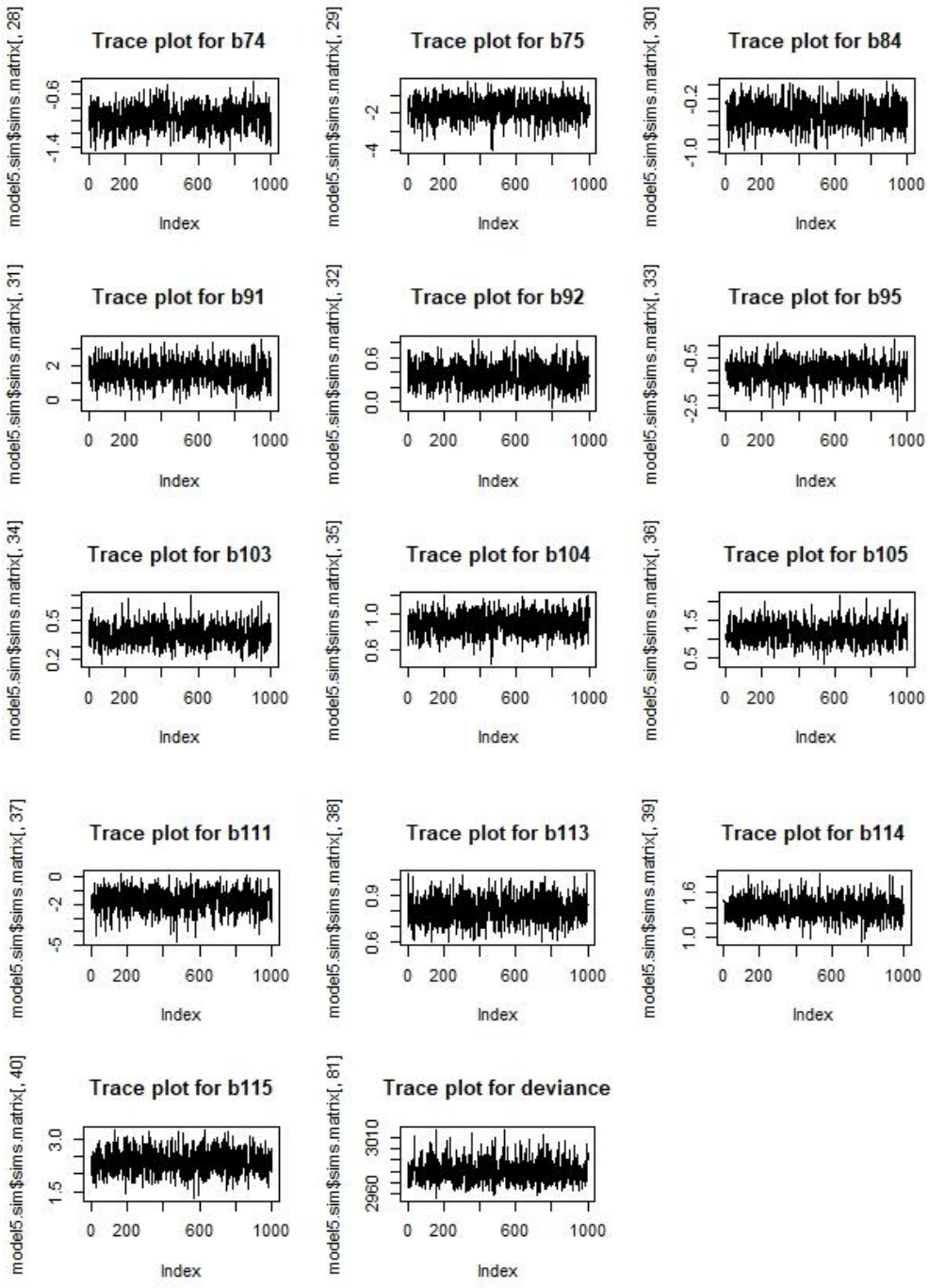

Density plot for b01
Density
N = 1000 Bandwidth = 0.2989
Density plot for b02
Density
N = 1000 Bandwidth = 0.04547
Density plot for b03
Density
N = 1000 Bandwidth = 0.01428
Density plot for b04
Density
N = 1000 Bandwidth = 0.03521
Density plot for b05
Density
N = 1000 Bandwidth = 0.162
Density plot for b11
Density
N = 1000 Bandwidth = 0.1386
Density plot for b14
Density
N = 1000 Bandwidth = 0.03361
Density plot for b15
Density
N = 1000 Bandwidth = 0.1043
Density plot for b21
Density
N = 1000 Bandwidth = 0.08364
Density plot for b22
Density
N = 1000 Bandwidth = 0.03388
Density plot for b23
Density
N = 1000 Bandwidth = 0.01566
Density plot for b24
Density
N = 1000 Bandwidth = 0.03
Density plot for b35
Density
N = 1000 Bandwidth = 0.0985
Density plot for b42
Density
N = 1000 Bandwidth = 0.02812
Density plot for b44
Density
N = 1000 Bandwidth = 0.02275
Density plot for b45
Density
N = 1000 Bandwidth = 0.08762
Density plot for b51
Density
N = 1000 Bandwidth = 0.05998
Density plot for b53
Density
N = 1000 Bandwidth = 0.01545

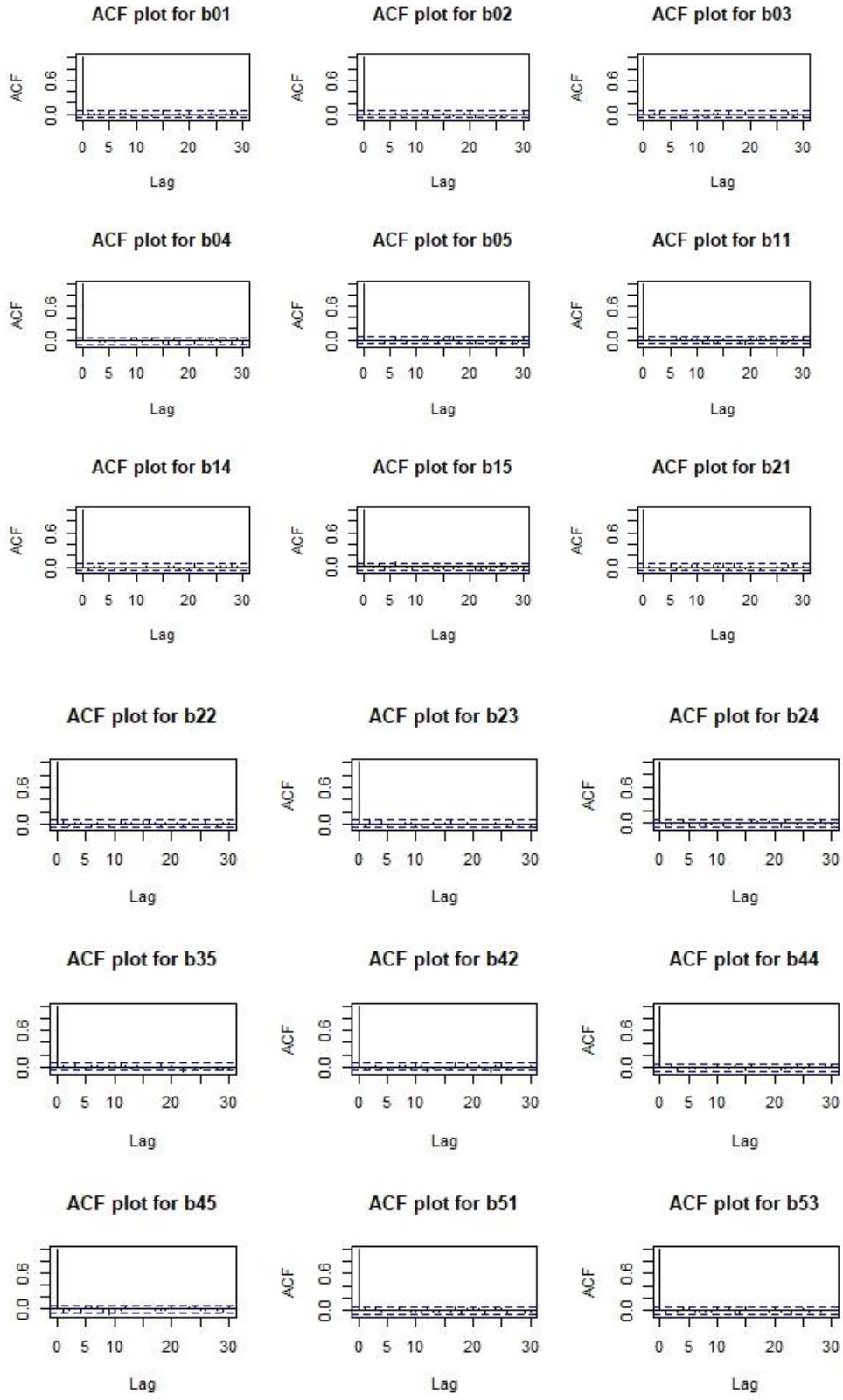

Density plot for b54
Density
N = 1000 Bandwidth = 0.03586
Density plot for b55
Density
N = 1000 Bandwidth = 0.1675
Density plot for b61
Density
N = 1000 Bandwidth = 0.1855
Density plot for b63
Density
N = 1000 Bandwidth = 0.01928
Density plot for b64
Density
N = 1000 Bandwidth = 0.03228
Density plot for b65
Density
N = 1000 Bandwidth = 0.09104
Density plot for b71
Density
N = 1000 Bandwidth = 0.3291
Density plot for b72
Density
N = 1000 Bandwidth = 0.04152
Density plot for b73
Density
N = 1000 Bandwidth = 0.02142
Density plot for b74
Density
N = 1000 Bandwidth = 0.03966
Density plot for b75
Density
N = 1000 Bandwidth = 0.1223
Density plot for b84
Density
N = 1000 Bandwidth = 0.04019
Density plot for b91
Density
N = 1000 Bandwidth = 0.1371
Density plot for b92
Density
N = 1000 Bandwidth = 0.03618
Density plot for b95
Density
N = 1000 Bandwidth = 0.09213
Density plot for b103
Density
N = 1000 Bandwidth = 0.01733
Density plot for b104
Density
N = 1000 Bandwidth = 0.0255
Density plot for b105
Density
N = 1000 Bandwidth = 0.05697

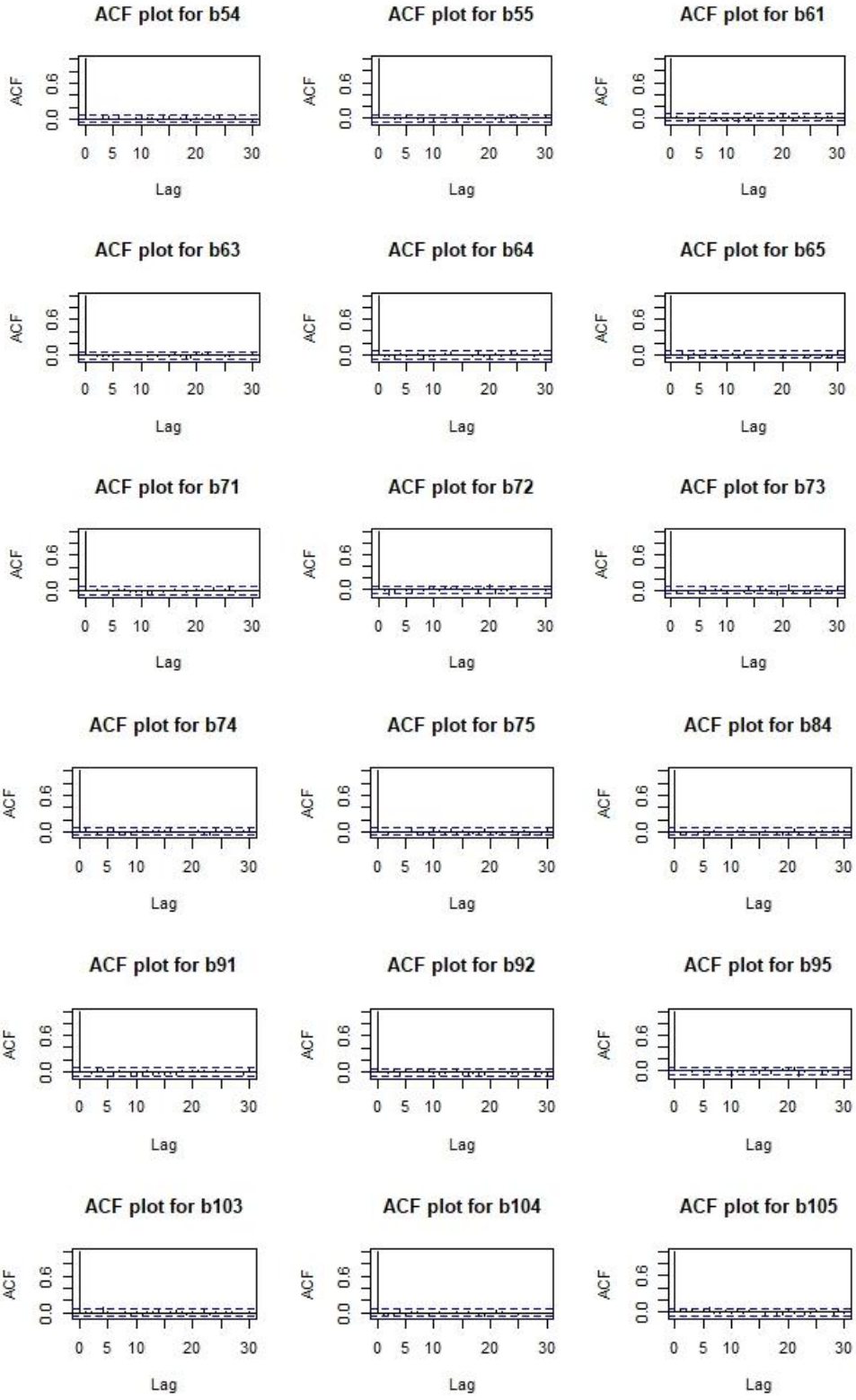

Density plot for b111
Density
N = 1000 Bandwidth = 0.1721
Density plot for b113
Density
N = 1000 Bandwidth = 0.0183
Density plot for b114
Density
N = 1000 Bandwidth = 0.0306
Density plot for b115
Density
N = 1000 Bandwidth = 0.06806
Density plot for deviance
Density
N = 1000 Bandwidth = 2.05

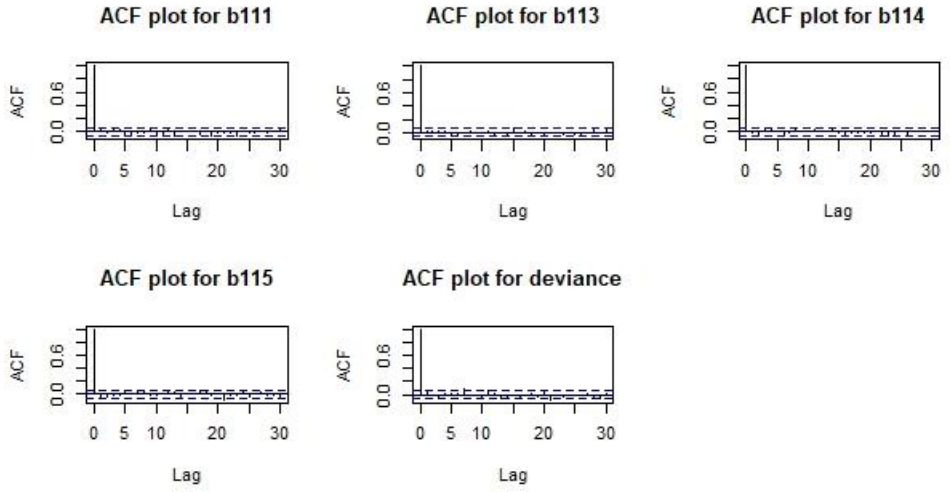

Trace plot for b01
model5.sim$sims.matrix[, 1]
Index
Trace plot for b02
model5.sim$sims.matrix[, 2]
Index
Trace plot for b03
model5.sim$sims.matrix[, 3]
Index
Trace plot for b04
model5.sim$sims.matrix[, 4]
Index
Trace plot for b05
model5.sim$sims.matrix[, 5]
Index
Trace plot for b11
model5.sim$sims.matrix[, 6]
Index
Trace plot for b14
model5.sim$sims.matrix[, 7]
Index
Trace plot for b15
model5.sim$sims.matrix[, 8]
Index
Trace plot for b21
model5.sim$sims.matrix[, 9]
Index


Trace plot for b22
model5.sim$sims.matrix[, 10]
Trace plot for b23
model5.sim$sims.matrix[, 11]
Trace plot for b24
model5.sim$sims.matrix[, 12]
Trace plot for b35
model5.sim$sims.matrix[, 13]
Trace plot for b42
model5.sim$sims.matrix[, 14]
Trace plot for b44
model5.sim$sims.matrix[, 15]
Trace plot for b45
model5.sim$sims.matrix[, 16]
Trace plot for b51
model5.sim$sims.matrix[, 17]
Trace plot for b53
model5.sim$sims.matrix[, 18]
Trace plot for b54
model5.sim$sims.matrix[, 19]
Trace plot for b55
model5.sim$sims.matrix[, 20]
Trace plot for b61
model5.sim$sims.matrix[, 21]
Trace plot for b63
model5.sim$sims.matrix[, 22]
Trace plot for b64
model5.sim$sims.matrix[, 23]
Trace plot for b65
model5.sim$sims.matrix[, 24]
Trace plot for b71
model5.sim$sims.matrix[, 25]
Trace plot for b72
model5.sim$sims.matrix[, 26]
Trace plot for b73
model5.sim$sims.matrix[, 27]
Index


Trace plot for b74
model5.sim$sims.matrix[, 28]
Index
Trace plot for b75
model5.sim$sims.matrix[, 29]
Index
Trace plot for b84
model5.sim$sims.matrix[, 30]
Index
Trace plot for b91
model5.sim$sims.matrix[, 31]
Index
Trace plot for b92
model5.sim$sims.matrix[, 32]
Index
Trace plot for b95
model5.sim$sims.matrix[, 33]
Index
Trace plot for b103
model5.sim$sims.matrix[, 34]
Index
Trace plot for b104
model5.sim$sims.matrix[, 35]
Index
Trace plot for b105
model5.sim$sims.matrix[, 36]
Index
Trace plot for b111
model5.sim$sims.matrix[, 37]
Index
Trace plot for b113
model5.sim$sims.matrix[, 38]
Index
Trace plot for b114
model5.sim$sims.matrix[, 39]
Index
Trace plot for b115
model5.sim$sims.matrix[, 40]
Index
Trace plot for deviance
model5.sim$sims.matrix[, 81]
Index


ACF plot for b01
ACF plot for b02
ACF plot for b03
ACF plot for b04
ACF plot for b05
ACF plot for b11
ACF plot for b14
ACF plot for b15
ACF plot for b21
ACF plot for b22
ACF plot for b23
ACF plot for b24
ACF plot for b35
ACF plot for b42
ACF plot for b44
ACF plot for b45
ACF plot for b51
ACF plot for b53
ACF
0.0 0.6
0 5 10 20 30
Lag


ACF plot for b54
ACF plot for b55
ACF plot for b61
ACF plot for b63
ACF plot for b64
ACF plot for b65
ACF plot for b71
ACF plot for b72
ACF plot for b73
ACF plot for b74
ACF plot for b75
ACF plot for b84
ACF plot for b91
ACF plot for b92
ACF plot for b95
ACF plot for b103
ACF plot for b104
ACF plot for b105
ACF
Lag
0.0
0.6
0
5
10
20
30



Even though admittedly the Geweke diagnostic could be a little better (maybe by taking a longer burn-in period) and the Heidelberger-Welch test could be improved by taking more iterations, all in all, the graphs and tests indicate a successful convergence for our model.

The posterior results of our model are:

```
Inference for Bugs model at "C:\Users\30697\Desktop\multinomial2.txt", fit
using OpenBUGS,
 1 chains, each with 51000 iterations (first 1000 discarded), n.thin = 50
 n.sims = 1000 iterations saved
           mean   sd   2.5%    25%    50%    75%  97.5%
b01        -8.8  1.4  -12.1   -9.5   -8.6   -7.8   -6.5
b02        -3.0  0.2   -3.4   -3.2   -3.0   -2.9   -2.6
b03         0.1  0.1    0.0    0.1    0.1    0.2    0.3
b04        -2.1  0.2   -2.4   -2.2   -2.1   -1.9   -1.8
b05        -6.5  0.7   -8.1   -7.0   -6.5   -6.0   -5.1
b11         1.7  0.6    0.5    1.3    1.8    2.2    2.9
b14         0.5  0.1    0.2    0.4    0.5    0.6    0.8
b15        -0.7  0.5   -1.7   -1.0   -0.7   -0.4    0.1
b21         1.5  0.4    0.8    1.2    1.5    1.7    2.3
b22         0.5  0.2    0.2    0.4    0.5    0.6    0.8
b23        -0.4  0.1   -0.5   -0.4   -0.4   -0.3   -0.2
b24        -0.8  0.1   -1.0   -0.9   -0.8   -0.7   -0.5
b35         0.7  0.4   -0.1    0.4    0.7    1.0    1.5
b42         0.2  0.1    0.0    0.2    0.2    0.3    0.5
b44         0.3  0.1    0.1    0.3    0.3    0.4    0.5
b45        -0.1  0.4   -1.0   -0.4   -0.1    0.2    0.6
```

b51	0.6	0.3	0.0	0.5	0.7	0.8	1.2
b53	-0.2	0.1	-0.3	-0.2	-0.2	-0.1	0.0
b54	-0.5	0.2	-0.9	-0.6	-0.5	-0.4	-0.2
b55	-2.0	0.7	-3.6	-2.4	-1.9	-1.4	-0.7
b61	1.6	0.9	-0.1	1.0	1.6	2.1	3.5
b63	0.3	0.1	0.1	0.2	0.3	0.3	0.4
b64	0.3	0.1	0.0	0.2	0.3	0.4	0.6
b65	0.4	0.4	-0.4	0.2	0.5	0.7	1.2
b71	-3.7	1.6	-7.5	-4.6	-3.5	-2.7	-1.3
b72	-0.7	0.2	-1.1	-0.9	-0.7	-0.6	-0.4
b73	-0.6	0.1	-0.8	-0.7	-0.6	-0.5	-0.4
b74	-0.9	0.2	-1.3	-1.1	-0.9	-0.8	-0.6
b75	-1.7	0.6	-3.0	-2.1	-1.7	-1.3	-0.7
b84	-0.4	0.2	-0.8	-0.6	-0.4	-0.3	-0.1
b91	1.6	0.6	0.4	1.2	1.6	2.0	2.9
b92	0.4	0.2	0.1	0.3	0.4	0.5	0.7
b95	-1.1	0.4	-1.9	-1.3	-1.1	-0.8	-0.3
b103	0.4	0.1	0.2	0.3	0.4	0.4	0.5
b104	0.9	0.1	0.7	0.8	0.9	1.0	1.1
b105	1.2	0.3	0.7	1.0	1.2	1.4	1.7
b111	-1.7	0.8	-3.4	-2.2	-1.7	-1.2	-0.3
b113	0.8	0.1	0.7	0.7	0.8	0.9	1.0
b114	1.4	0.1	1.1	1.3	1.4	1.5	1.7
b115	2.4	0.3	1.8	2.1	2.4	2.6	3.0
B01	0.0	0.0	0.0	0.0	0.0	0.0	0.0
B02	0.0	0.0	0.0	0.0	0.0	0.1	0.1
B03	1.2	0.1	1.0	1.1	1.2	1.2	1.3
B04	0.1	0.0	0.1	0.1	0.1	0.1	0.2
B05	0.0	0.0	0.0	0.0	0.0	0.0	0.0
B11	6.9	4.5	1.6	3.8	5.8	8.7	18.4
B14	1.6	0.2	1.2	1.5	1.6	1.8	2.2
B15	0.5	0.2	0.2	0.4	0.5	0.7	1.1
B21	4.8	2.0	2.2	3.4	4.4	5.6	9.7
B22	1.7	0.3	1.3	1.5	1.7	1.9	2.3
B23	0.7	0.0	0.6	0.7	0.7	0.7	0.8
B24	0.5	0.1	0.4	0.4	0.5	0.5	0.6
B35	2.2	1.0	0.9	1.5	2.0	2.8	4.6
B42	1.3	0.2	1.0	1.2	1.3	1.4	1.6
B44	1.4	0.1	1.2	1.3	1.4	1.5	1.7
B45	1.0	0.4	0.4	0.7	0.9	1.2	1.7
B51	2.0	0.6	1.0	1.6	2.0	2.3	3.3
B53	0.8	0.1	0.7	0.8	0.8	0.9	1.0
B54	0.6	0.1	0.4	0.5	0.6	0.7	0.8
B55	0.2	0.1	0.0	0.1	0.2	0.2	0.5
B61	7.5	10.6	0.9	2.7	4.8	8.1	32.9
B63	1.3	0.1	1.1	1.2	1.3	1.4	1.5
B64	1.4	0.2	1.0	1.2	1.3	1.5	1.8
B65	1.7	0.7	0.7	1.2	1.6	2.0	3.3
B71	0.1	0.1	0.0	0.0	0.0	0.1	0.3
B72	0.5	0.1	0.3	0.4	0.5	0.5	0.7
B73	0.5	0.1	0.5	0.5	0.5	0.6	0.7
B74	0.4	0.1	0.3	0.3	0.4	0.4	0.5
B75	0.2	0.1	0.1	0.1	0.2	0.3	0.5

```
B84          0.7  0.1    0.5    0.6    0.6    0.7    0.9
B91          6.3  4.4    1.5    3.4    5.2    7.8   17.6
B92          1.5  0.2    1.1    1.3    1.4    1.6    2.0
B95          0.4  0.2    0.2    0.3    0.3    0.4    0.8
B103         1.5  0.1    1.3    1.4    1.5    1.6    1.7
B104         2.4  0.3    2.0    2.2    2.4    2.6    3.0
B105         3.4  0.9    2.0    2.8    3.3    3.9    5.5
B111         0.2  0.2    0.0    0.1    0.2    0.3    0.7
B113         2.2  0.2    1.9    2.1    2.2    2.4    2.6
B114         4.1  0.6    3.1    3.6    4.0    4.4    5.3
B115        11.1  3.7    5.8    8.6   10.5   12.8   19.6
deviance 2979.9  9.4 2964.1 2973.1 2979.1 2985.3 3002.2

DIC info (using the rule, pD = Dbar-Dhat)
pD = 37.4 and DIC = 3017.0
DIC is an estimate of expected predictive error (lower deviance is better).
```

Immediately we see that the DIC value (3017.0) is far smaller than that of the full model (3036.0), and so we conclude that this is indeed a better model than the original. Therefore, according to the table above, the final model will look like this:

$$ln\left(\frac{p_{i1}}{p_{i6}}\right) = -8.8 + 1.7 \cdot Z_{i1} + 1.5 \cdot Z_{i2} + 0.6 \cdot Z_{i5} + 1.6 \cdot Z_{i6} - 3.7 \cdot Z_{i7} + 1.6 \cdot Z_{i9} + \\ -1.7 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i2}}{p_{i6}}\right) = -3 + 0.5 \cdot Z_{i2} + 0.2 \cdot Z_{i4} - 0.7 \cdot Z_{i7} + 0.4 \cdot Z_{i9}$$

$$ln\left(\frac{p_{i3}}{p_{i6}}\right) = 0.1 - 0.4 \cdot Z_{i2} - 0.2 \cdot Z_{i5} + 0.3 \cdot Z_{i6} - 0.6 \cdot Z_{i7} + 0.4 \cdot Z_{i10} + 0.8 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i4}}{p_{i6}}\right) = -2.1 + 0.5 \cdot Z_{i1} - 0.8 \cdot Z_{i2} + 0.3 \cdot Z_{i4} - 0.5 \cdot Z_{i5} + 0.3 \cdot Z_{i6} - \\ - 0.9 \cdot Z_{i7} - 0.4 \cdot Z_{i8} + 0.9 \cdot Z_{i10} + 1.4 \cdot Z_{i11}$$

$$ln\left(\frac{p_{i5}}{p_{i6}}\right) = -6.5 - 0.7 \cdot Z_{i1} + 0.7 \cdot Z_{i3} - 0.1 \cdot Z_{i4} - 2 \cdot Z_{i5} + 0.4 \cdot Z_{i6} - \\ - 1.7 \cdot Z_{i7} - 1.1 \cdot Z_{i9} + 1.2 \cdot Z_{i10} + 2.4 \cdot Z_{i11}$$

More iterations would improve this model significantly (by reducing the MCMC error), but unfortunately, we have run out of time. However, levels with more observations (like for instance level 3: Category 3 with reference level 6) are generally better off as far as results are concerned.

Now to interpret them, but first, we have to remember that in order to center our values, we subtracted the mean and divided by the standard deviation of each variable. Therefore, we have no units of measurement for our variables.

Using OpenBUGS, we have calculated the exponentiated values of our parameters, as they will be used to interpret our results in the following manner:

For level 1:

$B01 = exp\{\beta_{01}\} \cong 0$: If we do not include any of our physiochemical characteristics in this level at all, then there is almost (it is not exactly zero) no chance that our wine will take a score of 3, but rather that of 5. It is kind of nonsensical as a result, but maybe the other factors that are not included in this level are enough to make this distinction between these two scores.

$B11 = exp\{\beta_{11}\} = 6.9$: Is saying to us that should we keep all the other variables of this level fixed and simultaneously increase the value of fixed acidity by 1 (no units of measurement, remember?), then our odds of receiving a score of 3, rather than 5 will increase by a huge 590%.

$B21 = exp\{\beta_{21}\} = 4.8$: Tells us that should we increase the value of volatile acidity by 1 unit (no units of measurement), all while keeping the other variables constant, then the odds of being in category 1 that gets a median score of 3, will increase drastically by 380%, instead of being in category 6, which gets a median score of 5.

$B51 = exp\{\beta_{51}\} = 2$: Tells us that by keeping all other variables of this level fixed, but also increasing the value of chlorides by 1 (no units of measurement), our odds of receiving a score of 3 instead of 5 will increase by 100%.

$B61 = exp\{\beta_{61}\} = 7.5$: Is an indication that should we keep all other variables of this level constant in their values, while increasing the value of free sulfur dioxide by 1 (no units of measurement), our odds of receiving a score of 3 instead of 5 will increase by an enormous 650%.

$B71 = exp\{\beta_{71}\} = 0.1$: This value points out that if we keep all the other variables of this level fixed and simultaneously increase the value of total sulfur dioxide by 1 (no units of measurement), then the odds of getting a score of 3 rather than 5 will decrease by 90%.

$B91 = exp\{\beta_{91}\} = 6.3$: Tells us that if we keep all the other variables of this level fixed, all the while increasing the value of pH by 1 (no units of measurement), then the odds of getting a score of 3 rather than 5 will increase by an astonishing 530%.

$B111 = exp\{\beta_{111}\} = 0.2$: Is saying to us that should we keep all the other variables of this level fixed, but also increase the value alcohol by 1 (no units of measurement), then the odds of getting a score of 3 rather than 5 will diminish by 80%.

For level 2:

$B02 = exp\{\beta_{02}\} \cong 0$: Should all the physiochemical characteristics be gone from this level, then there is almost (it is not exactly zero) no chance at all of receiving a score of 4, but instead, we will receive that of 5.

$B22 = exp\{\beta_{22}\} = 1.7$: Keeping all other variables constant, this value is an indication that should we increase the volatile acidity by 1 (again, no units of measurement), then the odds of receiving a score of 4, rather than 5 will increase by 70%.

$B42 = exp\{\beta_{42}\} = 1.3$: Is telling us that if we keep all other variables constant in their values and increase the value of residual sugar by 1 (no units of measurement), then the odds of getting a score of 4, and not 5 will increase by 30%.

$B72 = exp\{\beta_{72}\} = 0.5$: Once more, should other variables remain unchanged and we increase the total sulfur dioxide by 1 (no units of measurement), then our odds of receiving a score of 4 instead of 5 will decrease by 50%.

$B92 = exp\{\beta_{92}\} = 1.5$: Says that should all other variables remain unchanged in their values and we increase the value of pH by 1 (no units of measurement), then then our odds of receiving a score of 4 rather than 5 will increase by 50%.

For level 3:

$B03 = exp\{\beta_{03}\} = 1.2$: If all the other variables of this level are zero, then the probability of receiving a score of 6, rather than 5 is increased by 20%. This is what this value means.

$B23 = exp\{\beta_{23}\} = 0.7$: Should all other variables remain fixed, then if we increase volatile acidity by 1 (no units of measurement), then the odds of scoring 6, rather than 5 with the judges will decrease by 30%.

$B53 = exp\{\beta_{53}\} = 0.8$: Is saying that should all other variables remain fixed and we increase chlorides by 1 (no units of measurement), then the odds of receiving a score of 6, rather than 5 will diminish by 20%.

$B63 = exp\{\beta_{63}\} = 1.3$: If we keep all the other variables of this level fixed at their values and increase the free sulfur dioxide by 1 (no units of measurement), then the odds of getting a score of 6 instead of 5 will increase by 30%.

$B73 = exp\{\beta_{73}\} = 0.5$: Keeping all the other variables at fixed values, should we increase the total sulfur dioxide by 1 (no units of measurement yet again), then the odds of receiving a score of 6 and not 5 will decrease by 50%.

$B103 = exp\{\beta_{103}\} = 1.5$: If all other variables remain unchanged and we increase the sulphates by 1 (no units of measurement), then the odds of receiving a score of 6, rather than 5 will increase 50%.

$B113 = exp\{\beta_{113}\} = 2.2$: Is telling us that should we keep all other variables fixed and increase the alcohol by 1 (no units of measurement), then the odds of getting a score of 6 rather than 5 will increase by a whopping 120%.

For level 4:

$B04 = exp\{\beta_{04}\} = 0.1$: Should we equate all the other variables of this level to zero, then the probability of receiving a score of 7 rather than 5 will decrease by 90%.

$B14 = exp\{\beta_{14}\} = 1.6$: This value tells us that if we keep all other physiochemical characteristics of this level at fixed values, all the while increasing the fixed acidity by 1 (no units of measurement), then our odds of scoring a 7 instead of 5 with the experts will increase by 60%.

$B24 = exp\{\beta_{24}\} = 0.5$: If we keep all other variables constant in their values and increase volatile acidity by 1 (there are no units of measurement), then our odds of receiving a score of 7 instead of 5 will decrease by 50%.

$B44 = exp\{\beta_{44}\} = 1.4$: This value tells us that should we keep all other variables of this level fixed and increase the value of residual sugar by 1 (no units of measurement), then our odds of getting a score of 7 and not 5 will increase by 40%.

$B54 = exp\{\beta_{54}\} = 0.6$: This indicates that if we keep all other variables of this level fixed and increase the value of chlorides by 1 (no units of measurement), then our odds of getting a score of 7 instead of 5 will decrease by 40%.

$B64 = exp\{\beta_{64}\} = 1.4$: This says to us that if we keep all other variables of this level fixed, while increasing the value of free sulfur dioxide by 1 (no units of measurement), then our odds of getting a score of 7 rather than 5 will increase by 40%.

$B74 = exp\{\beta_{74}\} = 0.4$: Keeping all other variables fixed in their values and increasing total sulfur dioxide by 1 (no units of measurement), will decrease our odds of receiving a grade of 7 instead of 5 by 60%.

$B84 = exp\{\beta_{84}\} = 0.7$: If we keep all other physiochemical characteristics fixed and increase the value of density by 1 (no units of measurement), then our odds of getting a score of 7 rather than 5 will decrease by 30%.

$B104 = exp\{\beta_{104}\} = 2.4$: If all other variables remain unchanged and we increase the sulphates by 1 (once again, no units of measurement), then the odds of receiving a score of 7, instead of a 5 will increase 140%.

$B114 = exp\{\beta_{114}\} = 4.1$: Should we keep all other variables fixed and increase the alcohol by 1 unit (no units of measurement), then the odds of receiving a score of 7 rather than 5 will increase by a whole 310%.

For level 5:

$B05 = exp\{\beta_{05}\} \cong 0$: Should we equate all the other variables of this level to zero, then the probability of receiving a score of 8, rather than 5 is nearly impossible (it is not exactly zero).

$B15 = exp\{\beta_{15}\} = 0.5$: This result is telling us that should we increase fixed acidity by 1 (no units of measurement), while keeping all other variables of this level fixed, then our odds of getting a score of 8 rather than 5 will decrease by 50%

$B35 = exp\{\beta_{35}\} = 2.2$: Should all other variables remain fixed, then if we increase citric acid by 1 (again, no units of measurement), then the odds of receiving a score of 8, rather than 5 will increase by 120%.

$B45 = exp\{\beta_{45}\} = 1$: If we keep all the other variables of this level fixed at their values and increase the residual sugar by 1 (and again, no units of measurement), then the odds of receiving a score of 8 and 5 are exactly the same. Well not exactly, plus this is bound to change if we take more iterations.

$B55 = exp\{\beta_{55}\} = 0.2$: This value tells us that if we keep all the other variables of this level fixed, while increasing the value of chlorides by 1 (no units of measurement), then our odds of getting a score of 8 rather than 5 will decrease by 80%.

$B65 = exp\{\beta_{65}\} = 1.7$: Having all the other variables at fixed values, should we increase the value of free sulfur dioxide by 1 (no units of measurement yet again), then the odds of receiving a score of 8 and not 5 will increase by 70%.

$B75 = exp\{\beta_{75}\} = 0.2$: Keeping all the other variables at fixed values, should we increase the total sulfur dioxide by 1 (no units of measurement yet again), then the odds of receiving a score of 8 and not 5 will decrease by 80%.

$B95 = exp\{\beta_{95}\} = 0.4$: Is telling us that should we keep all the other physiochemical characteristics of this level constant in their values, while increasing pH by 1 (no units of measurement), then our odds of scoring an 8 instead of 5 with the judges will decrease by 60%.

$B105 = exp\{\beta_{105}\} = 3.4$: If all other variables remain unchanged and we increase the value of sulphates by 1 (no units of measurement), then the odds of receiving a score of 8, rather than 5 will increase drastically by 240%.

$B115 = exp\{\beta_{115}\} = 11.1$: Should we keep all other variables fixed and increase the value of alcohol by 1 (no units of measurement), then the odds of receiving a score of 8 rather than 5 will increase by an insane amount of 1010%.

So, what can we surmise from these results?

a. It appears that alcohol as well as sulphates appealed to the taste of our judges and in fact those were the two factors that could, in moderation, make or break our wine's score.
b. There was no good reason to keep a high level of volatile acid in our wine, as it usually made our wine's grade drop to a lower level. In fact, it is not even included in level 5.
c. The same thing can be said about chlorides, as they only make our wine's score drop to a lower value.

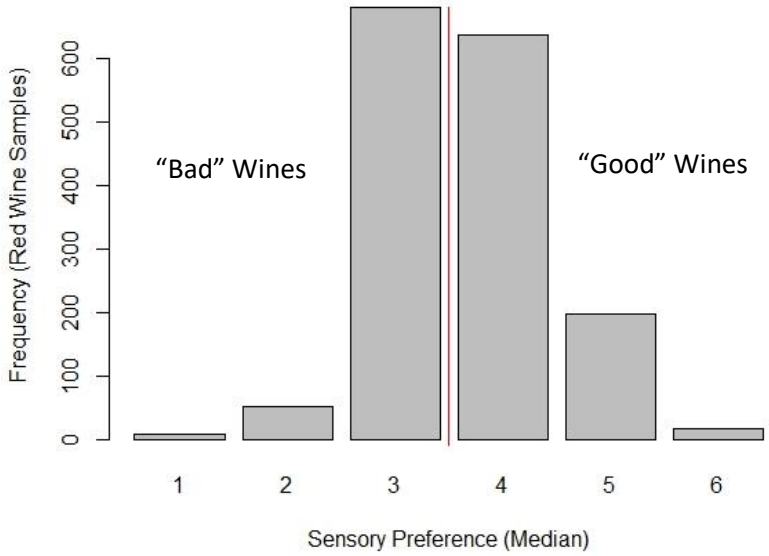
d. This can be also extended to pH. It is preferable to keep low values of this physiochemical characteristic in our wines.
e. Free sulfur dioxide is always tasteful, but should be kept in moderation in our wines as it can suddenly change our final score to the lowest.
f. Density appears only in level 4 and ruins the chances for our wine to get a higher score, so it should be kept at low values in our wine.
g. Total sulfur dioxide does not contribute to any grade whatsoever, apart from maybe that of 5. Thus, it should be kept at low levels in our wines.

And apart from those variables, the others of our model either contributed or took away from the scores of each level.

What if we wanted to produce only wines of scores 6-8?

Taste is a very fickle way of measurement. A wine that was an 8 today could be a 7 tomorrow and vice versa, except of course if these experts are infallible. As rational human beings, what we would like, should we be in the shoes of these winemakers, is to at least produce wines that would get high scores. So, we would love to see our wine get scores of 6, 7 or even 8, and not 5 or lower. What would we advise these people to look out for as far as physiochemical characteristics are concerned? The answer will be given by a simple binomial logistic regression. We shall split our data into two groups, below 5.5 and above 5.5 and try to determine the most important variables and how they will affect our final scores.

Graphically, this will look like so:



Therefore, if we run our variable selection package (BAS), for a BIC prior on betas and beta-binomial prior on the model, we will end up with the following results:

```
Marginal Posterior Summaries of Coefficients:

 Using  BMA

 Based on the top  66 models
                       post mean   post SD     post p(B != 0)
Intercept             -8.580e+00   6.374e+00   1.000e+00
fixed.acidity          9.771e-03   3.783e-02   8.374e-02
volatile.acidity      -3.016e+00   4.562e-01   9.985e-01
citric.acid           -1.291e-01   4.542e-01   1.003e-01
residual.sugar         4.875e-04   6.607e-03   1.421e-02
chlorides             -2.830e+00   2.447e+00   6.329e-01
free.sulfur.dioxide    1.224e+01   1.326e+01   5.115e-01
total.sulfur.dioxide  -1.471e+01   3.647e+00   9.994e-01
density                2.779e-01   6.254e+00   1.387e-02
pH                    -2.608e-02   1.648e-01   3.696e-02
sulphates              2.486e+00   5.058e-01   1.000e+00
alcohol                8.894e-01   7.855e-02   9.997e-01
> summary(res)
                    P(B != 0 | Y)   model 1      model 2      model 3
Intercept              1.00000000    1.0000    1.0000000    1.0000000
fixed.acidity          0.08374023    0.0000    0.0000000    0.0000000
volatile.acidity       0.99848633    1.0000    1.0000000    1.0000000
citric.acid            0.10034180    0.0000    0.0000000    0.0000000
residual.sugar         0.01420898    0.0000    0.0000000    0.0000000
chlorides              0.63291016    1.0000    1.0000000    0.0000000
free.sulfur.dioxide    0.51147461    1.0000    0.0000000    1.0000000
total.sulfur.dioxide   0.99941406    1.0000    1.0000000    1.0000000
density                0.01386719    0.0000    0.0000000    0.0000000
pH                     0.03696289    0.0000    0.0000000    0.0000000
sulphates              0.99995117    1.0000    1.0000000    1.0000000
alcohol                0.99970703    1.0000    1.0000000    1.0000000
BF                             NA    1.0000    0.4427971    0.2644016
PostProbs                      NA    0.2743    0.2653000    0.1504000
R2                             NA    0.2464    0.2423000    0.2418000
dim                            NA    7.0000    6.0000000    6.0000000
logmarg                        NA -856.4953 -857.3098941 -857.8255366
                       model 4      model 5
Intercept              1.00000000    1.0000000
fixed.acidity          0.00000000    1.0000000
volatile.acidity       1.00000000    1.0000000
citric.acid            0.00000000    1.0000000
residual.sugar         0.00000000    0.0000000
chlorides              0.00000000    0.0000000
free.sulfur.dioxide    0.00000000    0.0000000
```

```
total.sulfur.dioxide    1.00000000    1.0000000
density                 0.00000000    0.0000000
pH                      0.00000000    0.0000000
sulphates               1.00000000    1.0000000
alcohol                 1.00000000    1.0000000
BF                      0.09341534    0.0559124
PostProbs               0.13200000    0.0320000
R2                      0.23760000    0.2438000
dim                     5.00000000    7.0000000
logmarg              -858.86595013 -859.3792196
```

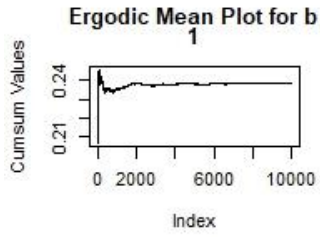
The variables that are unimportant in this model, are *Fixed Acidity*, *Citric Acid*, *Residual Sugar*, *Density* and *pH*. Thus, rerunning our model (*binomial2.txt*), without those superfluous variables, will give us the following results:

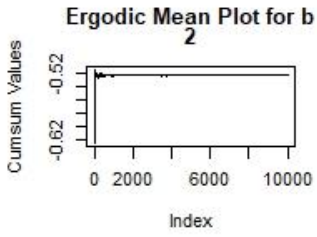
```
GEWEKE CONVERGENCE DIAGNOSTIC (Z-score)
=======================================

Iterations used = 1001:11000
Thinning interval = 1
Sample size per chain = 10000

$chain1

Fraction in 1st window = 0.1
Fraction in 2nd window = 0.5

    b[1]     B[1]     b[2]     B[2]     b[3]     B[3]     b[4]     B[4]
 -0.0163  -0.0777  -1.3401  -1.3972  -0.8335  -0.8397   0.3659   0.3377
    b[5]     B[5]     b[6]     B[6]     b[7]     B[7] deviance
 -0.5331  -0.6139  -0.0422  -0.0269   0.6717   0.6789  -1.4709
```

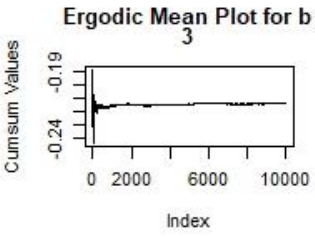
```
HEIDELBERGER AND WELCH STATIONARITY AND INTERVAL HALFWIDTH TESTS
===============================================================

Iterations used = 1001:11000
Thinning interval = 1
Sample size per chain = 10000

Precision of halfwidth test = 0.1

$chain1

         Stationarity start     p-value
         test         iteration
b[1]     passed          1      0.5710
B[1]     passed          1      0.6295
b[2]     passed          1      0.0818
```

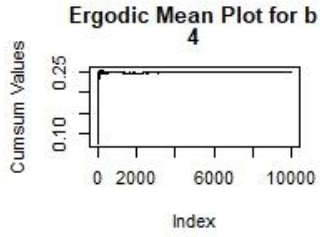
```
B[2]     passed          1      0.0814
b[3]     passed          1      0.8495
B[3]     passed          1      0.8663
b[4]     passed          1      0.6204
B[4]     passed          1      0.5536
b[5]     passed          1      0.1061
B[5]     passed          1      0.1318
b[6]     passed          1      0.3401
B[6]     passed          1      0.3277
b[7]     passed          1      0.2239
B[7]     passed          1      0.2310
deviance passed       2001      0.1183

         Halfwidth Mean     Halfwidth
         test
b[1]     passed       0.239 0.00149
B[1]     passed       1.272 0.00191
b[2]     passed      -0.522 0.00155
B[2]     passed       0.595 0.00092
b[3]     passed      -0.214 0.00169
B[3]     passed       0.809 0.00135
b[4]     passed       0.252 0.00205
B[4]     passed       1.291 0.00266
b[5]     passed      -0.580 0.00223
B[5]     passed       0.562 0.00125
b[6]     passed       0.465 0.00189
B[6]     passed       1.596 0.00292
b[7]     passed       0.920 0.00185
B[7]     passed       2.516 0.00473
deviance passed    1671.706 0.10683
```

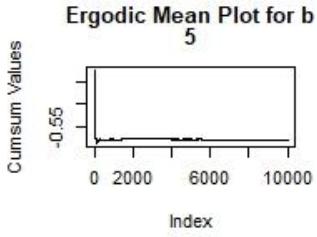
And as for plots, we get:

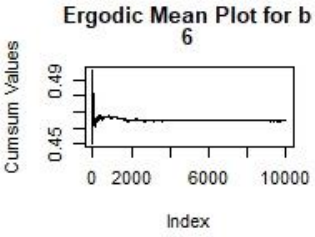

Ergodic Mean Plot for b 1
Cumsum Values
0.21
0.24
0
2000
6000
10000
Index

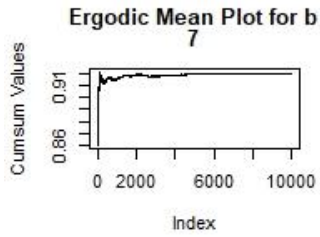

Ergodic Mean Plot for b 2
Cumsum Values
-0.62
-0.52
0
2000
6000
10000
Index

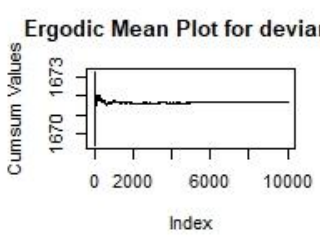

Ergodic Mean Plot for b 3
Cumsum Values
-0.24
-0.19
0
2000
6000
10000
Index

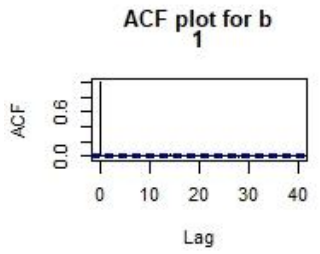

Ergodic Mean Plot for b 4
Cumsum Values
0.10
0.25
0
2000
6000
10000
Index

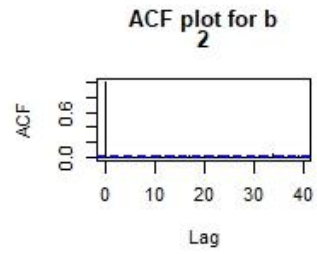

Ergodic Mean Plot for b 5
Cumsum Values
-0.55
0
2000
6000
10000
Index

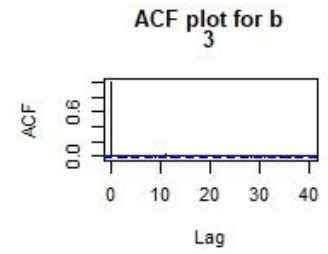

Ergodic Mean Plot for b 6
Cumsum Values
0.45
0.49
0
2000
6000
10000
Index

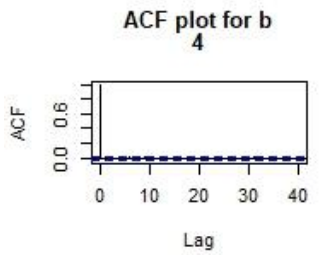

Ergodic Mean Plot for b 7
Cumsum Values
0.86
0.91
0
2000
6000
10000
Index

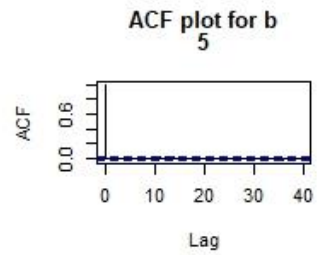

Ergodic Mean Plot for devianc
Cumsum Values
1670
1673
0
2000
6000
10000
Index

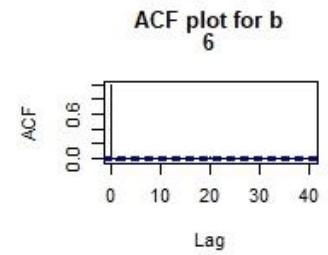

ACF plot for b 1
ACF
0.0
0.6
0
10
20
30
40
Lag

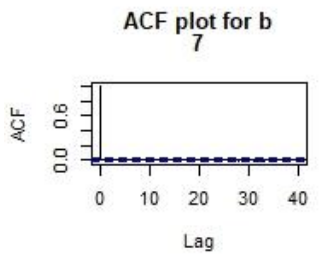

ACF plot for b 2
ACF
0.0
0.6
0
10
20
30
40
Lag

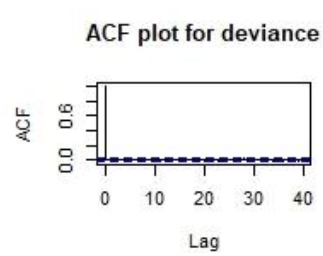

ACF plot for b 3
ACF
0.0
0.6
0
10
20
30
40
Lag

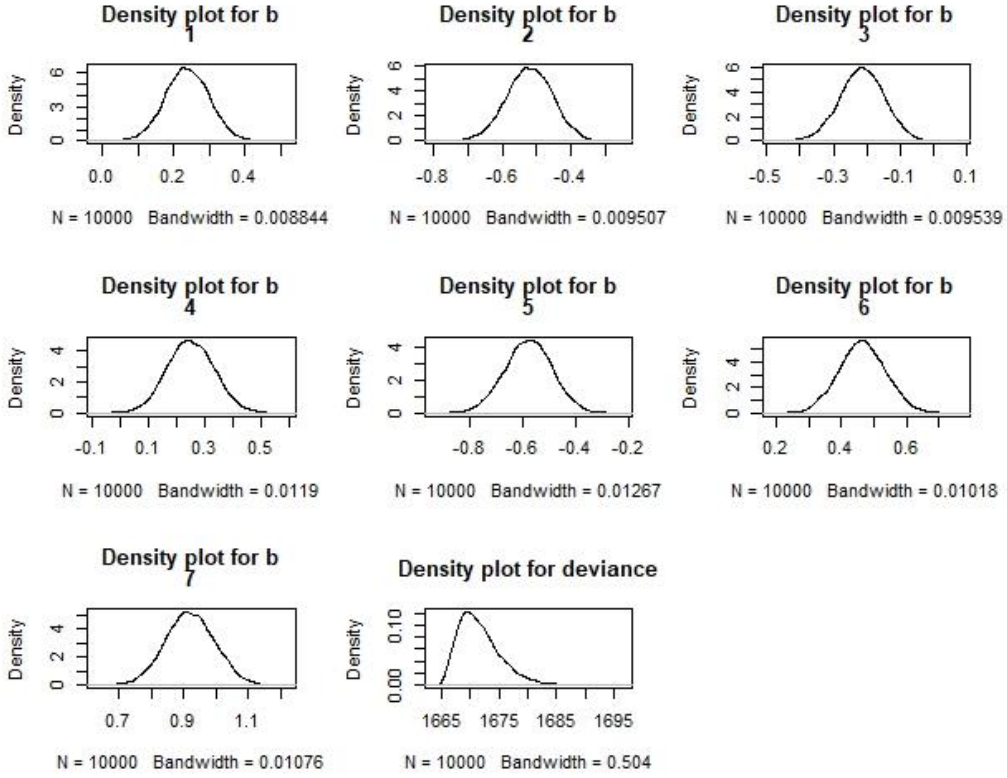

ACF plot for b 4
ACF
0.0
0.6
0
10
20
30
40
Lag

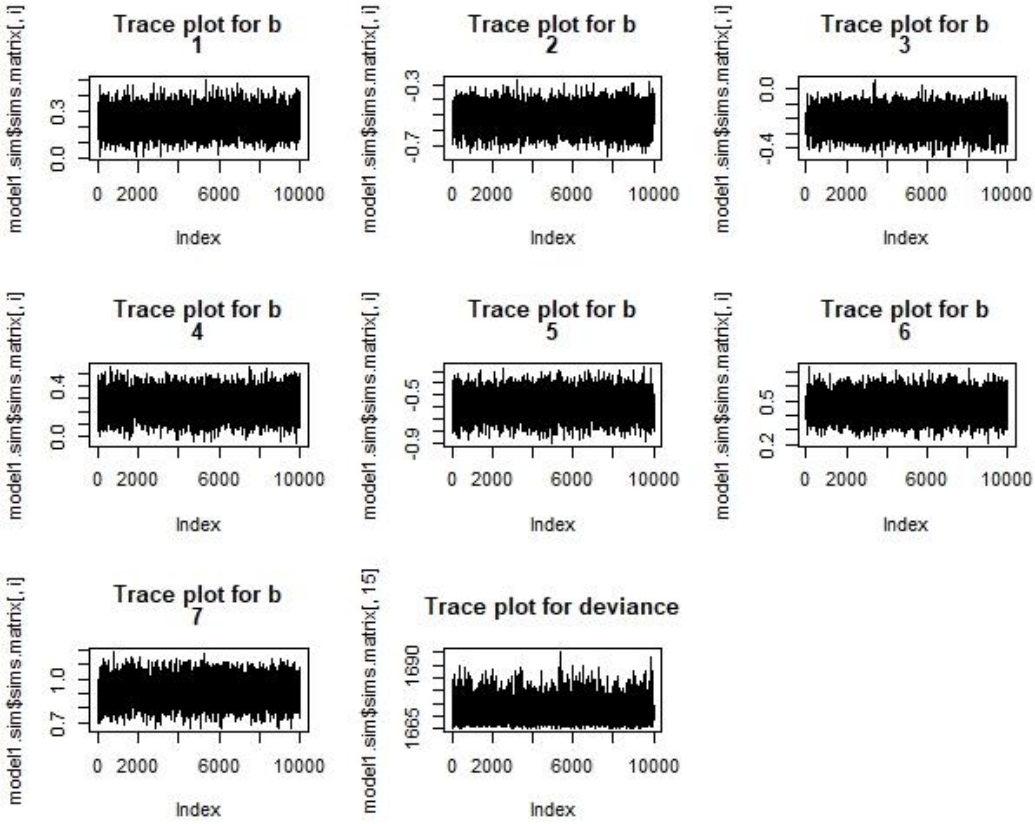

ACF plot for b 5
ACF
0.0
0.6
0
10
20
30
40
Lag


ACF plot for b 6
ACF
0.0
0.6
0
10
20
30
40
Lag


ACF plot for b 7
ACF
0.0
0.6
0
10
20
30
40
Lag


ACF plot for deviance
ACF
0.0
0.6
0
10
20
30
40
Lag


Density plot for b 1
Density
N = 10000 Bandwidth = 0.008844
Density plot for b 2
Density
N = 10000 Bandwidth = 0.009507
Density plot for b 3
Density
N = 10000 Bandwidth = 0.009539
Density plot for b 4
Density
N = 10000 Bandwidth = 0.0119
Density plot for b 5
Density
N = 10000 Bandwidth = 0.01267
Density plot for b 6
Density
N = 10000 Bandwidth = 0.01018
Density plot for b 7
Density
N = 10000 Bandwidth = 0.01076
Density plot for deviance
Density
N = 10000 Bandwidth = 0.504


Trace plot for b 1
model1.sim$sims.matrix[, i]
Index
Trace plot for b 2
model1.sim$sims.matrix[, i]
Index
Trace plot for b 3
model1.sim$sims.matrix[, i]
Index
Trace plot for b 4
model1.sim$sims.matrix[, i]
Index
Trace plot for b 5
model1.sim$sims.matrix[, i]
Index
Trace plot for b 6
model1.sim$sims.matrix[, i]
Index
Trace plot for b 7
model1.sim$sims.matrix[, i]
Index
Trace plot for deviance
model1.sim$sims.matrix[, 15]
Index

All of these indicate convergence. The results are:

```
Inference for Bugs model at "C:\Users\30697\Desktop\binomial2.txt",
fit using OpenBUGS,
 1 chains, each with 11000 iterations (first 1000 discarded)
 n.sims = 10000 iterations saved
           mean  sd   2.5%    25%    50%    75%  97.5%
b[1]        0.2 0.1    0.1    0.2    0.2    0.3    0.4
b[2]       -0.5 0.1   -0.7   -0.6   -0.5   -0.5   -0.4
b[3]       -0.2 0.1   -0.3   -0.3   -0.2   -0.2   -0.1
b[4]        0.3 0.1    0.1    0.2    0.3    0.3    0.4
b[5]       -0.6 0.1   -0.8   -0.6   -0.6   -0.5   -0.4
b[6]        0.5 0.1    0.3    0.4    0.5    0.5    0.6
b[7]        0.9 0.1    0.8    0.9    0.9    1.0    1.1
B[1]        1.3 0.1    1.1    1.2    1.3    1.3    1.4
B[2]        0.6 0.0    0.5    0.6    0.6    0.6    0.7
B[3]        0.8 0.1    0.7    0.8    0.8    0.8    0.9
B[4]        1.3 0.1    1.1    1.2    1.3    1.4    1.5
B[5]        0.6 0.0    0.5    0.5    0.6    0.6    0.7
B[6]        1.6 0.1    1.4    1.5    1.6    1.7    1.8
B[7]        2.5 0.2    2.2    2.4    2.5    2.6    2.9
deviance 1671.7 3.7 1666.4 1668.9 1671.0 1673.7 1680.5

DIC info (using the rule, pD = Dbar-Dhat)
pD = 6.9 and DIC = 1679.0
DIC is an estimate of expected predictive error (lower deviance is
better).
```

And so, our model will be:

$$logit(p_i) = 0.2 - 0.5 \cdot Z_{i2} - 0.2 \cdot Z_{i5} + 0.3 \cdot Z_{i6} - 0.6 \cdot Z_{i7} + 0.5 \cdot Z_{i10} + 0.9 \cdot Z_{i11}$$

The interpretation of those results is based on the exponentiated values of the parameters. Therefore:

$B[1] = 1.3$: If we equate all physiochemical parameters of the model above to zero, then the probability of producing a wine that will be in the upper echelon, which gets a score of 6-8, will be increased by 30%. So, if we do nothing, then our wine will supposedly get categorized as a good wine. This makes little to no sense, as a wine with alcohol of 0 is in reality, grape juice.

$B[2] = 0.6$: Keeping all other variables fixed, while increasing the volatile acidity by 1 (no units of measurement, because we've centered our values), then the odds of characterizing our wine as a "good" wine (i.e. the wine that gets a score of 6-8), will diminish by 40%.

$B[3] = 0.8$: Having kept all the other variables constant and at the same time increasing chlorides by 1 (no units of measurement), our odds of producing a "good" wine have been decreased by 20%.

$B[4] = 1.3$: Should we keep all other variables fixed and increase only free sulfur dioxide by 1 (no units of measurement), then our odds of scoring 6-8 for our wine will increase by 30%.

$B[5] = 0.6$: If we keep all other physiochemical parameters fixed at their values and increase only total sulfur dioxide by 1 (no units of measurement), then our odds for our wine to be categorized as a "good" wine will diminish by 40%.

$B[6] = 1.6$: Keeping all other parameters constant, while increasing sulphates by 1 (no unit of measurement), will increase our odds of having our wine score 6-8 points by 60%.

$B[7] = 2.5$: Should we keep all other parameters of our model fixed, while increasing alcohol by 1 (no unit of measurement), then the odds of producing a good wine will increase by a whole 150%.

Therefore, if we want to "fine tune" our wine and get those good ratings that we need, then we would have to increase, up to a logical extent (we do not want to produce poison) the physiochemical parameters of free sulfur dioxide, sulphates and alcohol, all while keeping a low value for volatile acidity, chlorides and total sulfur dioxide. These are the most important variables for the best wines in that region.

That is how we will theoretically produce a high scoring wine in the Vinho Verde region of Portugal.

Conclusion and final comments:

This was our suggestion, if we can get these physiochemical properties in a bottle of wine, we are almost certain that this bottle could achieve high scores among the judges. However, we have to remember that good wine does not depend exclusively on those physiochemical characteristics, but it needs so much more to succeed. The climate, the type of grapes, the soil, the way it is bottled, all these and more play an important role in the final product and its score.

Thus, these variables that we suggest should be a roadmap and not the destination, meaning that should a bottle of Vinho Verde contain these characteristics, then we have good indications that we are producing a good red wine that people can enjoy. And that is what this paper was all about.

APPENDIX 2

(Models and Algorithms)

All the models are advised to be put in a different .txt and named as the parentheses indicate.

1. *The Original Model: Multinomial Logistic Regression (multinomial.txt)*

```
model {
   for (i in 1:1599){                                ## loop over observations
     y[i,1] ~ dcat(p[i, 1:6])                        ## multinomial outcome
     for (j in 1:5){                                 ## loop over categories
      p[i,j] <- expeta[i,j]/(1 + expeta[i,1] +
                             expeta[i,2] + expeta[i,3] +
                             expeta[i,4] + expeta[i,5])

      expeta[i,j] <- exp(eta[i,j])
      eta[i,j] <- b0[j] +
                  b1[j]*x[i,1] +
                  b2[j]*x[i,2] +
                  b3[j]*x[i,3] +
                  b4[j]*x[i,4] +
                  b5[j]*x[i,5] +
                  b6[j]*x[i,6] +
                  b7[j]*x[i,7] +
                  b8[j]*x[i,8] +
                  b9[j]*x[i,9] +
                  b10[j]*x[i,10] +
                  b11[j]*x[i,11]

     }
     expeta[i,6] <- 1.0                        ## baseline (reference) category
     p[i,6] <- 1/(1 + expeta[i,1] +
                  expeta[i,2] + expeta[i,3] +
                  expeta[i,4] + expeta[i,5])
   }
## ------------
##    Priors
## ------------
   for (k in 1:5){
    b0[k] ~ dnorm(0.0, 1.0E-4)                        ## low-information priors
    b1[k] ~ dnorm(0.0, 1.0E-4)
    b2[k] ~ dnorm(0.0, 1.0E-4)
    b3[k] ~ dnorm(0.0, 1.0E-4)
    b4[k] ~ dnorm(0.0, 1.0E-4)
    b5[k] ~ dnorm(0.0, 1.0E-4)
    b6[k] ~ dnorm(0.0, 1.0E-4)
    b7[k] ~ dnorm(0.0, 1.0E-4)
    b8[k] ~ dnorm(0.0, 1.0E-4)
```

```
    b9[k] ~ dnorm(0.0, 1.0E-4)
    b10[k] ~ dnorm(0.0, 1.0E-4)
    b11[k] ~ dnorm(0.0, 1.0E-4)
   }
}
############################## end of model ###################################
```

2. *Empirical Variable Selection through the Multinomial Logistic Regression (empirical.txt)*

```
model {
   for (i in 1:1599){                          ## loop over observations
     y[i,1] ~ dcat(p[i, 1:6])                  ## multinomial outcome

     for (j in 1:5){                           ## loop over categories
      p[i,j] <- expeta[i,j]/(1 + expeta[i,1] +
                             expeta[i,2] + expeta[i,3] +
                             expeta[i,4] + expeta[i,5])

      expeta[i,j] <- exp(eta[i,j])
      eta[i,j] <- gb0[j] +
                  gb1[j]*x[i,1] +
                  gb2[j]*x[i,2] +
                  gb3[j]*x[i,3] +
                  gb4[j]*x[i,4] +
                  gb5[j]*x[i,5] +
                  gb6[j]*x[i,6] +
                  gb7[j]*x[i,7] +
                  gb8[j]*x[i,8] +
                  gb9[j]*x[i,9] +
                  gb10[j]*x[i,10] +
                  gb11[j]*x[i,11]

     }
     expeta[i,6] <- 1.0                ## baseline (reference) category
     p[i,6] <- 1/(1 + expeta[i,1] +
                  expeta[i,2] + expeta[i,3] +
                  expeta[i,4] + expeta[i,5])
   }
## ------------
##    Priors
## ------------
   for (k in 1:5){
    b0[k] ~ dnorm(prop.mean.beta0[k], taub0[k])
    taub0[k] <- (gamma0[k]/1599 + (1-gamma0[k]))/pow(prop.sd.beta0[k], 2)
    gamma0[k] <- 1.0              ## constants are always in the model
    gb0[k] <- gamma0[k]*b0[k]

    b1[k] ~ dnorm(prop.mean.beta1[k], taub1[k])
    taub1[k] <- (gamma1[k]/1599 + (1-gamma1[k]))/pow(prop.sd.beta1[k], 2)
    gamma1[k] ~ dbern(0.5)          ## Uniform
#    gamma1[k] ~ dbern(p1)           ## Beta-binomial
    gb1[k] <- gamma1[k]*b1[k]

    b2[k] ~ dnorm(prop.mean.beta2[k], taub2[k])
    taub2[k] <- (gamma2[k]/1599 + (1-gamma2[k]))/pow(prop.sd.beta2[k], 2)
    gamma2[k] ~ dbern(0.5)          ## Uniform
#    gamma2[k] ~ dbern(p2)           ## Beta-binomial
    gb2[k] <- gamma2[k]*b2[k]

    b3[k] ~ dnorm(prop.mean.beta3[k], taub3[k])
```

```
    taub3[k] <- (gamma3[k]/1599 + (1-gamma3[k]))/pow(prop.sd.beta3[k], 2)
    gamma3[k] ~ dbern(0.5)          ## Uniform
#    gamma3[k] ~ dbern(p3)           ## Beta-binomial
    gb3[k] <- gamma3[k]*b3[k]

    b4[k] ~ dnorm(prop.mean.beta4[k], taub4[k])
    taub4[k] <- (gamma4[k]/1599 + (1-gamma4[k]))/pow(prop.sd.beta4[k], 2)
    gamma4[k] ~ dbern(0.5)          ## Uniform
#    gamma4[k] ~ dbern(p4)           ## Beta-binomial
    gb4[k] <- gamma4[k]*b4[k]

    b5[k] ~ dnorm(prop.mean.beta5[k], taub5[k])
    taub5[k] <- (gamma5[k]/1599 + (1-gamma5[k]))/pow(prop.sd.beta5[k], 2)
    gamma5[k] ~ dbern(0.5)          ## Uniform
#    gamma5[k] ~ dbern(p5)           ## Beta-binomial
    gb5[k] <- gamma5[k]*b5[k]

    b6[k] ~ dnorm(prop.mean.beta6[k], taub6[k])
    taub6[k] <- (gamma6[k]/1599 + (1-gamma6[k]))/pow(prop.sd.beta6[k], 2)
    gamma6[k] ~ dbern(0.5)          ## Uniform
#    gamma6[k] ~ dbern(p6)           ## Beta-binomial
    gb6[k] <- gamma6[k]*b6[k]

    b7[k] ~ dnorm(prop.mean.beta7[k], taub7[k])
    taub7[k] <- (gamma7[k]/1599 + (1-gamma7[k]))/pow(prop.sd.beta7[k], 2)
    gamma7[k] ~ dbern(0.5)          ## Uniform
#    gamma7[k] ~ dbern(p7)           ## Beta-binomial
    gb7[k] <- gamma7[k]*b7[k]

    b8[k] ~ dnorm(prop.mean.beta8[k], taub8[k])
    taub8[k] <- (gamma8[k]/1599 + (1-gamma8[k]))/pow(prop.sd.beta8[k], 2)
    gamma8[k] ~ dbern(0.5)          ## Uniform
#    gamma8[k] ~ dbern(p8)           ## Beta-binomial
    gb8[k] <- gamma8[k]*b8[k]

    b9[k] ~ dnorm(prop.mean.beta9[k], taub9[k])
    taub9[k] <- (gamma9[k]/1599 + (1-gamma9[k]))/pow(prop.sd.beta9[k], 2)
    gamma9[k] ~ dbern(0.5)          ## Uniform
#    gamma9[k] ~ dbern(p9)           ## Beta-binomial
    gb9[k] <- gamma9[k]*b9[k]

    b10[k] ~ dnorm(prop.mean.beta10[k], taub10[k])
    taub10[k] <- (gamma10[k]/1599 + (1-gamma10[k]))/pow(prop.sd.beta10[k], 2)
    gamma10[k] ~ dbern(0.5)         ## Uniform
#    gamma10[k] ~ dbern(p10)         ## Beta-binomial
    gb10[k] <- gamma10[k]*b10[k]

    b11[k] ~ dnorm(prop.mean.beta11[k], taub11[k])
    taub11[k] <- (gamma11[k]/1599 + (1-gamma11[k]))/pow(prop.sd.beta11[k], 2)
    gamma11[k] ~ dbern(0.5)         ## Uniform
#    gamma11[k] ~ dbern(p11)         ## Beta-binomial
    gb11[k] <- gamma11[k]*b11[k]
   }
```

```
#  For the Beta-Binomial
#   p1 ~ dbeta(2,10)
#   p2 ~ dbeta(2,10)
#   p3 ~ dbeta(2,10)
#   p4 ~ dbeta(2,10)
#   p5 ~ dbeta(2,10)
#   p6 ~ dbeta(2,10)
#   p7 ~ dbeta(2,10)
#   p8 ~ dbeta(2,10)
#   p9 ~ dbeta(2,10)
#   p10 ~ dbeta(2,10)
#   p11 ~ dbeta(2,10)
}
############################## end of model ###################################
```

3. *Variable Selection using g-prior through the Multinomial Logistic Regression (gprior.txt)*

```
model {
   for (i in 1:1599){                                  ## loop over observations
     y[i,1] ~ dcat(p[i, 1:6])                          ## multinomial outcome

     for (j in 1:5){                                   ## loop over categories
      p[i,j] <- expeta[i,j]/(1 + expeta[i,1] +
                             expeta[i,2] + expeta[i,3] +
                             expeta[i,4] + expeta[i,5])

      expeta[i,j] <- exp(eta[i,j])
      eta[i,j] <- alpha[j] +
                  gb1[j]*x[i,1] +
                  gb2[j]*x[i,2] +
                  gb3[j]*x[i,3] +
                  gb4[j]*x[i,4] +
                  gb5[j]*x[i,5] +
                  gb6[j]*x[i,6] +
                  gb7[j]*x[i,7] +
                  gb8[j]*x[i,8] +
                  gb9[j]*x[i,9] +
                  gb10[j]*x[i,10] +
                  gb11[j]*x[i,11]

     }
     expeta[i,6] <- 1.0                        ## baseline (reference) category
     p[i,6] <- 1/(1 + expeta[i,1] +
                  expeta[i,2] + expeta[i,3] +
                  expeta[i,4] + expeta[i,5])
   }
## ------------
##    Priors
## ------------
# Liang g-prior
 logtau ~ dflat()
 tau <- exp(logtau)
#####################################################################
 for (j in 1:5) {
  alpha[j] ~ dflat()
 }
#####################################################################
 B1[1:5] ~ dmnorm(mean.beta1[1:5], T1[1:5, 1:5])
 for (j in 1:5) {
  mean.beta1[j] <-  (1-gamma1[j])*prop.mean.beta1[j]
  for (k in 1:5) {
   T1[j,k] < - gamma1[j]*gamma1[k]*tau*XTX[j,k]/G + (1-
gamma1[j]*gamma1[k])*equals(j,k)*pow(prop.sd.beta1[k],-2)
  }
 }
 for (j in 1:5) {
   gamma1[j] ~ dbern(0.5) # Uniform
#   gamma1[j] ~ dbern(p1)  # Beta-Binomial
```

```
   gb1[j] <- gamma1[j]*B1[j]
 }
#####################################################################
 B2[1:5] ~ dmnorm(mean.beta2[1:5], T2[1:5, 1:5])
 for (j in 1:5) {
  mean.beta2[j] <-  (1-gamma2[j])*prop.mean.beta2[j]
  for (k in 1:5) {
   T2[j,k] < - gamma2[j]*gamma2[k]*tau*XTX[j,k]/G + (1-
gamma2[j]*gamma2[k])*equals(j,k)*pow(prop.sd.beta2[k],-2)
  }
 }
 for (j in 1:5) {
   gamma2[j] ~ dbern(0.5) # Uniform
#   gamma2[j] ~ dbern(p1)  # Beta-Binomial
   gb2[j] <- gamma2[j]*B2[j]
 }
#####################################################################
 B3[1:5] ~ dmnorm(mean.beta3[1:5], T3[1:5, 1:5])
 for (j in 1:5) {
  mean.beta3[j] <-  (1-gamma3[j])*prop.mean.beta3[j]
  for (k in 1:5) {
   T3[j,k] < - gamma3[j]*gamma3[k]*tau*XTX[j,k]/G + (1-
gamma3[j]*gamma3[k])*equals(j,k)*pow(prop.sd.beta3[k],-2)
  }
 }
 for (j in 1:5) {
   gamma3[j] ~ dbern(0.5) # Uniform
#   gamma3[j] ~ dbern(p1)  # Beta-Binomial
   gb3[j] <- gamma3[j]*B3[j]
 }
#####################################################################
 B4[1:5] ~ dmnorm(mean.beta4[1:5], T4[1:5, 1:5])
 for (j in 1:5) {
  mean.beta4[j] <-  (1-gamma4[j])*prop.mean.beta4[j]
  for (k in 1:5) {
   T4[j,k] < - gamma4[j]*gamma4[k]*tau*XTX[j,k]/G + (1-
gamma4[j]*gamma4[k])*equals(j,k)*pow(prop.sd.beta4[k],-2)
  }
 }
 for (j in 1:5) {
   gamma4[j] ~ dbern(0.5) # Uniform
#   gamma4[j] ~ dbern(p1)  # Beta-Binomial
   gb4[j] <- gamma4[j]*B4[j]
 }
#####################################################################
 B5[1:5] ~ dmnorm(mean.beta5[1:5], T5[1:5, 1:5])
 for (j in 1:5) {
  mean.beta5[j] <-  (1-gamma5[j])*prop.mean.beta5[j]
  for (k in 1:5) {
   T5[j,k] < - gamma5[j]*gamma5[k]*tau*XTX[j,k]/G + (1-
gamma5[j]*gamma5[k])*equals(j,k)*pow(prop.sd.beta5[k],-2)
  }
 }
```

```
 for (j in 1:5) {
   gamma5[j] ~ dbern(0.5) # Uniform
#   gamma5[j] ~ dbern(p1)  # Beta-Binomial
   gb5[j] <- gamma5[j]*B5[j]
 }
#####################################################################
 B6[1:5] ~ dmnorm(mean.beta6[1:5], T6[1:5, 1:5])
 for (j in 1:5) {
  mean.beta6[j] <-  (1-gamma6[j])*prop.mean.beta6[j]
  for (k in 1:5) {
   T6[j,k] < - gamma6[j]*gamma6[k]*tau*XTX[j,k]/G + (1-
gamma6[j]*gamma6[k])*equals(j,k)*pow(prop.sd.beta6[k],-2)
  }
 }
 for (j in 1:5) {
   gamma6[j] ~ dbern(0.5) # Uniform
#   gamma6[j] ~ dbern(p1)  # Beta-Binomial
   gb6[j] <- gamma6[j]*B6[j]
 }
#####################################################################
 B7[1:5] ~ dmnorm(mean.beta7[1:5], T7[1:5, 1:5])
 for (j in 1:5) {
  mean.beta7[j] <-  (1-gamma7[j])*prop.mean.beta7[j]
  for (k in 1:5) {
   T7[j,k] < - gamma7[j]*gamma7[k]*tau*XTX[j,k]/G + (1-
gamma7[j]*gamma7[k])*equals(j,k)*pow(prop.sd.beta7[k],-2)
  }
 }
 for (j in 1:5) {
   gamma7[j] ~ dbern(0.5) # Uniform
#   gamma7[j] ~ dbern(p1)  # Beta-Binomial
   gb7[j] <- gamma7[j]*B7[j]
 }
#####################################################################
 B8[1:5] ~ dmnorm(mean.beta8[1:5], T8[1:5, 1:5])
 for (j in 1:5) {
  mean.beta8[j] <-  (1-gamma8[j])*prop.mean.beta8[j]
  for (k in 1:5) {
   T8[j,k] < - gamma8[j]*gamma8[k]*tau*XTX[j,k]/G + (1-
gamma8[j]*gamma8[k])*equals(j,k)*pow(prop.sd.beta8[k],-2)
  }
 }
 for (j in 1:5) {
   gamma8[j] ~ dbern(0.5) # Uniform
#   gamma8[j] ~ dbern(p1)  # Beta-Binomial
   gb8[j] <- gamma8[j]*B8[j]
 }
#####################################################################
 B9[1:5] ~ dmnorm(mean.beta9[1:5], T9[1:5, 1:5])
 for (j in 1:5) {
  mean.beta9[j] <-  (1-gamma9[j])*prop.mean.beta9[j]
  for (k in 1:5) {
```

```
   T9[j,k] < - gamma9[j]*gamma9[k]*tau*XTX[j,k]/G + (1-
gamma9[j]*gamma9[k])*equals(j,k)*pow(prop.sd.beta9[k],-2)
  }
 }
 for (j in 1:5) {
   gamma9[j] ~ dbern(0.5) # Uniform
#   gamma9[j] ~ dbern(p1)  # Beta-Binomial
   gb9[j] <- gamma9[j]*B9[j]
 }
#######################################################################
 B10[1:5] ~ dmnorm(mean.beta10[1:5], T10[1:5, 1:5])
 for (j in 1:5) {
  mean.beta10[j] <-  (1-gamma10[j])*prop.mean.beta10[j]
  for (k in 1:5) {
   T10[j,k] < - gamma10[j]*gamma10[k]*tau*XTX[j,k]/G + (1-
gamma10[j]*gamma10[k])*equals(j,k)*pow(prop.sd.beta10[k],-2)
  }
 }
 for (j in 1:5) {
   gamma10[j] ~ dbern(0.5) # Uniform
#   gamma10[j] ~ dbern(p1)  # Beta-Binomial
   gb10[j] <- gamma10[j]*B10[j]
 }
#######################################################################
 B11[1:5] ~ dmnorm(mean.beta11[1:5], T11[1:5, 1:5])
 for (j in 1:5) {
  mean.beta11[j] <-  (1-gamma11[j])*prop.mean.beta11[j]
  for (k in 1:5) {
   T11[j,k] < - gamma11[j]*gamma11[k]*tau*XTX[j,k]/G + (1-
gamma11[j]*gamma11[k])*equals(j,k)*pow(prop.sd.beta11[k],-2)
  }
 }
 for (j in 1:5) {
   gamma11[j] ~ dbern(0.5) # Uniform
#   gamma11[j] ~ dbern(p1)  # Beta-Binomial
   gb11[j] <- gamma11[j]*B11[j]
 }
#######################################################################
#  For the Beta-Binomial
#   p1 ~ dbeta(2,10)
#   p2 ~ dbeta(2,10)
#   p3 ~ dbeta(2,10)
#   p4 ~ dbeta(2,10)
#   p5 ~ dbeta(2,10)
#   p6 ~ dbeta(2,10)
#   p7 ~ dbeta(2,10)
#   p8 ~ dbeta(2,10)
#   p9 ~ dbeta(2,10)
#   p10 ~ dbeta(2,10)
#   p11 ~ dbeta(2,10)

# hyper-prior
G <- 1599
```

```
}
############################## end of model ###################################
```

4. Variable Selection using hyper-g through the Multinomial Logistic Regression (hyperg.txt)

```
model {
   for (i in 1:1599){                                  ## loop over observations
     y[i,1] ~ dcat(p[i, 1:6])                          ## multinomial outcome

     for (j in 1:5){                                   ## loop over categories
      p[i,j] <- expeta[i,j]/(1 + expeta[i,1] +
                             expeta[i,2] + expeta[i,3] +
                             expeta[i,4] + expeta[i,5])

      expeta[i,j] <- exp(eta[i,j])
      eta[i,j] <- alpha[j] +
                  gb1[j]*x[i,1] +
                  gb2[j]*x[i,2] +
                  gb3[j]*x[i,3] +
                  gb4[j]*x[i,4] +
                  gb5[j]*x[i,5] +
                  gb6[j]*x[i,6] +
                  gb7[j]*x[i,7] +
                  gb8[j]*x[i,8] +
                  gb9[j]*x[i,9] +
                  gb10[j]*x[i,10] +
                  gb11[j]*x[i,11]

     }
     expeta[i,6] <- 1.0                        ## baseline (reference) category
     p[i,6] <- 1/(1 + expeta[i,1] +
                  expeta[i,2] + expeta[i,3] +
                  expeta[i,4] + expeta[i,5])
   }
## ------------
##    Priors
## ------------
# Liang hyper-g
 logtau ~ dflat()
 tau <- exp(logtau)
#####################################################################
 for (j in 1:5) {
  alpha[j] ~ dflat()
 }
#####################################################################
 B1[1:5] ~ dmnorm(mean.beta1[1:5], T1[1:5, 1:5])
 for (j in 1:5) {
  mean.beta1[j] <-  (1-gamma1[j])*prop.mean.beta1[j]
  for (k in 1:5) {
   T1[j,k] < - gamma1[j]*gamma1[k]*tau*XTX[j,k]/G + (1-
gamma1[j]*gamma1[k])*equals(j,k)*pow(prop.sd.beta1[k],-2)
  }
 }
 for (j in 1:5) {
   gamma1[j] ~ dbern(0.5) # Uniform
#   gamma1[j] ~ dbern(p1)  # Beta-Binomial
```

```
   gb1[j] <- gamma1[j]*B1[j]
 }
#####################################################################
 B2[1:5] ~ dmnorm(mean.beta2[1:5], T2[1:5, 1:5])
 for (j in 1:5) {
  mean.beta2[j] <-  (1-gamma2[j])*prop.mean.beta2[j]
  for (k in 1:5) {
   T2[j,k] < - gamma2[j]*gamma2[k]*tau*XTX[j,k]/G + (1-
gamma2[j]*gamma2[k])*equals(j,k)*pow(prop.sd.beta2[k],-2)
  }
 }
 for (j in 1:5) {
   gamma2[j] ~ dbern(0.5) # Uniform
#   gamma2[j] ~ dbern(p1)  # Beta-Binomial
   gb2[j] <- gamma2[j]*B2[j]
 }
#####################################################################
 B3[1:5] ~ dmnorm(mean.beta3[1:5], T3[1:5, 1:5])
 for (j in 1:5) {
  mean.beta3[j] <-  (1-gamma3[j])*prop.mean.beta3[j]
  for (k in 1:5) {
   T3[j,k] < - gamma3[j]*gamma3[k]*tau*XTX[j,k]/G + (1-
gamma3[j]*gamma3[k])*equals(j,k)*pow(prop.sd.beta3[k],-2)
  }
 }
 for (j in 1:5) {
   gamma3[j] ~ dbern(0.5) # Uniform
#   gamma3[j] ~ dbern(p1)  # Beta-Binomial
   gb3[j] <- gamma3[j]*B3[j]
 }
#####################################################################
 B4[1:5] ~ dmnorm(mean.beta4[1:5], T4[1:5, 1:5])
 for (j in 1:5) {
  mean.beta4[j] <-  (1-gamma4[j])*prop.mean.beta4[j]
  for (k in 1:5) {
   T4[j,k] < - gamma4[j]*gamma4[k]*tau*XTX[j,k]/G + (1-
gamma4[j]*gamma4[k])*equals(j,k)*pow(prop.sd.beta4[k],-2)
  }
 }
 for (j in 1:5) {
   gamma4[j] ~ dbern(0.5) # Uniform
#   gamma4[j] ~ dbern(p1)  # Beta-Binomial
   gb4[j] <- gamma4[j]*B4[j]
 }
#####################################################################
 B5[1:5] ~ dmnorm(mean.beta5[1:5], T5[1:5, 1:5])
 for (j in 1:5) {
  mean.beta5[j] <-  (1-gamma5[j])*prop.mean.beta5[j]
  for (k in 1:5) {
   T5[j,k] < - gamma5[j]*gamma5[k]*tau*XTX[j,k]/G + (1-
gamma5[j]*gamma5[k])*equals(j,k)*pow(prop.sd.beta5[k],-2)
  }
 }
```

```
 for (j in 1:5) {
   gamma5[j] ~ dbern(0.5) # Uniform
#   gamma5[j] ~ dbern(p1)  # Beta-Binomial
   gb5[j] <- gamma5[j]*B5[j]
 }
#####################################################################
 B6[1:5] ~ dmnorm(mean.beta6[1:5], T6[1:5, 1:5])
 for (j in 1:5) {
  mean.beta6[j] <-  (1-gamma6[j])*prop.mean.beta6[j]
  for (k in 1:5) {
   T6[j,k] < - gamma6[j]*gamma6[k]*tau*XTX[j,k]/G + (1-
gamma6[j]*gamma6[k])*equals(j,k)*pow(prop.sd.beta6[k],-2)
  }
 }
 for (j in 1:5) {
   gamma6[j] ~ dbern(0.5) # Uniform
#   gamma6[j] ~ dbern(p1)  # Beta-Binomial
   gb6[j] <- gamma6[j]*B6[j]
 }
#####################################################################
 B7[1:5] ~ dmnorm(mean.beta7[1:5], T7[1:5, 1:5])
 for (j in 1:5) {
  mean.beta7[j] <-  (1-gamma7[j])*prop.mean.beta7[j]
  for (k in 1:5) {
   T7[j,k] < - gamma7[j]*gamma7[k]*tau*XTX[j,k]/G + (1-
gamma7[j]*gamma7[k])*equals(j,k)*pow(prop.sd.beta7[k],-2)
  }
 }
 for (j in 1:5) {
   gamma7[j] ~ dbern(0.5) # Uniform
#   gamma7[j] ~ dbern(p1)  # Beta-Binomial
   gb7[j] <- gamma7[j]*B7[j]
 }
#####################################################################
 B8[1:5] ~ dmnorm(mean.beta8[1:5], T8[1:5, 1:5])
 for (j in 1:5) {
  mean.beta8[j] <-  (1-gamma8[j])*prop.mean.beta8[j]
  for (k in 1:5) {
   T8[j,k] < - gamma8[j]*gamma8[k]*tau*XTX[j,k]/G + (1-
gamma8[j]*gamma8[k])*equals(j,k)*pow(prop.sd.beta8[k],-2)
  }
 }
 for (j in 1:5) {
   gamma8[j] ~ dbern(0.5) # Uniform
#   gamma8[j] ~ dbern(p1)  # Beta-Binomial
   gb8[j] <- gamma8[j]*B8[j]
 }
#####################################################################
 B9[1:5] ~ dmnorm(mean.beta9[1:5], T9[1:5, 1:5])
 for (j in 1:5) {
  mean.beta9[j] <-  (1-gamma9[j])*prop.mean.beta9[j]
  for (k in 1:5) {
```

```
   T9[j,k] < - gamma9[j]*gamma9[k]*tau*XTX[j,k]/G + (1-
gamma9[j]*gamma9[k])*equals(j,k)*pow(prop.sd.beta9[k],-2)
  }
 }
 for (j in 1:5) {
   gamma9[j] ~ dbern(0.5) # Uniform
#   gamma9[j] ~ dbern(p1)  # Beta-Binomial
   gb9[j] <- gamma9[j]*B9[j]
 }
#######################################################################
 B10[1:5] ~ dmnorm(mean.beta10[1:5], T10[1:5, 1:5])
 for (j in 1:5) {
  mean.beta10[j] <-  (1-gamma10[j])*prop.mean.beta10[j]
  for (k in 1:5) {
   T10[j,k] < - gamma10[j]*gamma10[k]*tau*XTX[j,k]/G + (1-
gamma10[j]*gamma10[k])*equals(j,k)*pow(prop.sd.beta10[k],-2)
  }
 }
 for (j in 1:5) {
   gamma10[j] ~ dbern(0.5) # Uniform
#   gamma10[j] ~ dbern(p1)  # Beta-Binomial
   gb10[j] <- gamma10[j]*B10[j]
 }
#######################################################################
 B11[1:5] ~ dmnorm(mean.beta11[1:5], T11[1:5, 1:5])
 for (j in 1:5) {
  mean.beta11[j] <-  (1-gamma11[j])*prop.mean.beta11[j]
  for (k in 1:5) {
   T11[j,k] < - gamma11[j]*gamma11[k]*tau*XTX[j,k]/G + (1-
gamma11[j]*gamma11[k])*equals(j,k)*pow(prop.sd.beta11[k],-2)
  }
 }
 for (j in 1:5) {
   gamma11[j] ~ dbern(0.5) # Uniform
#   gamma11[j] ~ dbern(p1)  # Beta-Binomial
   gb11[j] <- gamma11[j]*B11[j]
 }
#######################################################################
#  For the Beta-Binomial
#   p1 ~ dbeta(2,10)
#   p2 ~ dbeta(2,10)
#   p3 ~ dbeta(2,10)
#   p4 ~ dbeta(2,10)
#   p5 ~ dbeta(2,10)
#   p6 ~ dbeta(2,10)
#   p7 ~ dbeta(2,10)
#   p8 ~ dbeta(2,10)
#   p9 ~ dbeta(2,10)
#   p10 ~ dbeta(2,10)
#   p11 ~ dbeta(2,10)
#######################################################################
# hyper-prior
   a <- 3
```

```
   bw <- a/2-1
   w ~ dbeta( 1, bw )
   G <- w/(1-w)
}
############################## end of model ###################################
```

5. The Reduced Model: Multinomial Logistic Regression Without the Unnecessary Variables
hyper-g, beta-binomial case (multinomial2.txt)

```
model {
   for (i in 1:1599){                                  ## loop over observations
     y[i,1] ~ dcat(p[i, 1:6])                          ## multinomial outcome

     for (j in 1:5){                                   ## loop over categories
      p[i,j] <- expeta[i,j]/(1 + expeta[i,1] +
                             expeta[i,2] + expeta[i,3] +
                             expeta[i,4] + expeta[i,5])

      expeta[i,j] <- exp(eta[i,j])
      }
      eta[i,1] <- b01 + b11*x[i,1] + b21*x[i,2] + b51*x[i,5] + b61*x[i,6] +
                  b71*x[i,7] + b91*x[i,9] + b111*x[i,11]
      eta[i,2] <- b02 + b22*x[i,2] + b42*x[i,4] + b72*x[i,7] + b92*x[i,9]
      eta[i,3] <- b03 + b23*x[i,2] + b53*x[i,5] + b63*x[i,6] + b73*x[i,7] +
                  b103*x[i,10] + b113*x[i,11]
      eta[i,4] <- b04 + b14*x[i,1] + b24*x[i,2] + b44*x[i,4] + b54*x[i,5] +
                  b64*x[i,6] + b74*x[i,7] + b84*x[i,8] + b104*x[i,10] +
                  b114*x[i,11]
      eta[i,5] <- b05 + b15*x[i,1] + b35*x[i,3] + b45*x[i,4] + b55*x[i,5] +
                  b65*x[i,6] + b75*x[i,7] + b95*x[i,9] + b105*x[i,10] +
                  b115*x[i,11]

     expeta[i,6] <- 1.0                        ## baseline (reference) category
     p[i,6] <- 1/(1 + expeta[i,1] +
                  expeta[i,2] + expeta[i,3] +
                  expeta[i,4] + expeta[i,5])
   }
## ------------
##    Priors
## ------------

    b01 ~ dnorm(0.0, 1.0E-4)                          ## low-information priors
    b02 ~ dnorm(0.0, 1.0E-4)
    b03 ~ dnorm(0.0, 1.0E-4)
    b04 ~ dnorm(0.0, 1.0E-4)
    b05 ~ dnorm(0.0, 1.0E-4)
    b11 ~ dnorm(0.0, 1.0E-4)
    b14 ~ dnorm(0.0, 1.0E-4)
    b15 ~ dnorm(0.0, 1.0E-4)
    b21 ~ dnorm(0.0, 1.0E-4)
    b22 ~ dnorm(0.0, 1.0E-4)
    b23 ~ dnorm(0.0, 1.0E-4)
    b24 ~ dnorm(0.0, 1.0E-4)
    b35 ~ dnorm(0.0, 1.0E-4)
    b42 ~ dnorm(0.0, 1.0E-4)
    b44 ~ dnorm(0.0, 1.0E-4)
    b45 ~ dnorm(0.0, 1.0E-4)
    b51 ~ dnorm(0.0, 1.0E-4)
```

```
    b53 ~ dnorm(0.0, 1.0E-4)
    b54 ~ dnorm(0.0, 1.0E-4)
    b55 ~ dnorm(0.0, 1.0E-4)
    b61 ~ dnorm(0.0, 1.0E-4)
    b63 ~ dnorm(0.0, 1.0E-4)
    b64 ~ dnorm(0.0, 1.0E-4)
    b65 ~ dnorm(0.0, 1.0E-4)
    b71 ~ dnorm(0.0, 1.0E-4)
    b72 ~ dnorm(0.0, 1.0E-4)
    b73 ~ dnorm(0.0, 1.0E-4)
    b74 ~ dnorm(0.0, 1.0E-4)
    b75 ~ dnorm(0.0, 1.0E-4)
    b84 ~ dnorm(0.0, 1.0E-4)
    b91 ~ dnorm(0.0, 1.0E-4)
    b92 ~ dnorm(0.0, 1.0E-4)
    b95 ~ dnorm(0.0, 1.0E-4)
    b103 ~ dnorm(0.0, 1.0E-4)
    b104 ~ dnorm(0.0, 1.0E-4)
    b105 ~ dnorm(0.0, 1.0E-4)
    b111 ~ dnorm(0.0, 1.0E-4)
    b113 ~ dnorm(0.0, 1.0E-4)
    b114 ~ dnorm(0.0, 1.0E-4)
    b115 ~ dnorm(0.0, 1.0E-4)

## -------------------------
##    Exponentiated Values
## -------------------------
    B01 <- exp(b01)
    B02 <- exp(b02)
    B03 <- exp(b03)
    B04 <- exp(b04)
    B05 <- exp(b05)
    B11 <- exp(b11)
    B14 <- exp(b14)
    B15 <- exp(b15)
    B21 <- exp(b21)
    B22 <- exp(b22)
    B23 <- exp(b23)
    B24 <- exp(b24)
    B35 <- exp(b35)
    B42 <- exp(b42)
    B44 <- exp(b44)
    B45 <- exp(b45)
    B51 <- exp(b51)
    B53 <- exp(b53)
    B54 <- exp(b54)
    B55 <- exp(b55)
    B61 <- exp(b61)
    B63 <- exp(b63)
    B64 <- exp(b64)
    B65 <- exp(b65)
    B71 <- exp(b71)
    B72 <- exp(b72)
```

```
    B73 <- exp(b73)
    B74 <- exp(b74)
    B75 <- exp(b75)
    B84 <- exp(b84)
    B91 <- exp(b91)
    B92 <- exp(b92)
    B95 <- exp(b95)
    B103 <- exp(b103)
    B104 <- exp(b104)
    B105 <- exp(b105)
    B111 <- exp(b111)
    B113 <- exp(b113)
    B114 <- exp(b114)
    B115 <- exp(b115)
}
############################## end of model ###################################
```

6. The Binomial Logistic Regression (binomial.txt)

```
model {
  for (i in 1:1599){              ## loop over observations
    y[i,1] ~ dbern(p[i])          ## binary outcome
    logit(p[i]) <- ystar[i]       ## logit link
    ystar[i] <- b[1] +            ## regression structure for covariates
                b[2]*x[i,1] +
                b[3]*x[i,2] +
                b[4]*x[i,3] +
                b[5]*x[i,4] +
                b[6]*x[i,5] +
                b[7]*x[i,6] +
                b[8]*x[i,7] +
                b[9]*x[i,8] +
                b[10]*x[i,9] +
                b[11]*x[i,10] +
                b[12]*x[i,11]
  }
## ------------
##    Priors
## ------------
 for (k in 1:12){
  b[k] ~ dnorm(0.0, 1.0E-4)       ## low information priors
 }
}
############################## end of model ###################################
```

7. The Binomial Logistic Regression Without the Unnecessary Variables (binomial2.txt)

```
model {
  for (i in 1:1599){              ## loop over observations
    y[i,1] ~ dbern(p[i])          ## binary outcome
    logit(p[i]) <- ystar[i]       ## logit link
    ystar[i] <- b[1] +            ## regression structure for covariates
                b[2]*x[i,2] +
                b[3]*x[i,5] +
                b[4]*x[i,6] +
                b[5]*x[i,7] +
                b[6]*x[i,10] +
                b[7]*x[i,11]
  }
## ------------
##    Priors
## ------------
 for (k in 1:7){
  b[k] ~ dnorm(0.0, 1.0E-4)        ## low information priors
 }

## ----------------------
##  Exponentiated Values
## ----------------------
 for (k in 1:7){
  B[k] <- exp(b[k])
 }
}
############################## end of model ###################################
```

8. *The R Code*:

```
#########################################################################
# Initial Statistics & Plots for our Analysis.
#========================================================================
## Load the data into R.
wines <- read.csv("C:\\Users\\30697\\Desktop\\06_winequality-red.csv",
header = T, sep=";", stringsAsFactors = FALSE)

## Check for missing values (no missing values).
any(is.na(wines))

## Change the values from mg/dm^3 to g/dm^3 for free and total sulfur
## dioxide.
wines$free.sulfur.dioxide <- wines$free.sulfur.dioxide*0.001
wines$total.sulfur.dioxide <- wines$total.sulfur.dioxide*0.001

## Get some statistics on our parameters.
lapply(wines, mean)
lapply(wines, min)
lapply(wines, max)

## Plot of the sensory data.
plot(factor(wines$quality), xlab = "Sensory Preference (Median)", ylab
= "Frequency (Red Wine Samples)")

## Quality with 6 Categories (3-8).
table(wines$quality)

## Shift the response variable from 3 until 8, to 1 until 6, placing
## the third category (our reference category) to the last position.
wines[wines[,12]==3,12] <- 1
wines[wines[,12]==4,12] <- 2
wines[wines[,12]==6,12] <- 3
wines[wines[,12]==7,12] <- 4
wines[wines[,12]==5,12] <- 6  # our reference level
wines[wines[,12]==8,12] <- 5

# Used for BAS later.
wines.bas <- wines

## Check if it worked.
table(wines$quality)
plot(factor(wines$quality), xlab = "Sensory Preference (Median)", ylab
= "Frequency (Red Wine Samples)")

## Turn our data frame into a matrix and then into a list.
y <- as.matrix(wines$quality)
```

```
wines <- scale(wines[,-12], center=TRUE, scale=TRUE) # centering our
values (reduces run time by ~ 5-10 minutes)
wines1 <- as.matrix(wines)
lwines <- list(y=y,x=wines1[,1:11])


#########################################################################
# Run our Model in R.
#========================================================================
## Load libraries.
library(R2WinBUGS)
library(BRugs)

## Set the directory.
openbugs.dir <- "D:\\OpenBUGS323"

## A list of initials (one chain).
inits1 <- list(
     list(b0=rep(0, times=5), b1=rep(0, times=5), b2=rep(0, times=5),
          b3=rep(0, times=5), b4=rep(0, times=5), b5=rep(0, times=5),
          b6=rep(0, times=5), b7=rep(0, times=5), b8=rep(0, times=5),
          b9=rep(0, times=5), b10=rep(0, times=5),
          b11=rep(0, times=5)) )

## The parameters of interest.
parameter.names <- c( 'b0', 'b1', 'b2', 'b3', 'b4', 'b5', 'b6', 'b7',
'b8', 'b9', 'b10', 'b11')

## The model (in OpenBUGS). (Run time for 1 chain ~ 85 minutes)
model1.sim <- bugs( lwines, inits1, model.file =
"C:\\Users\\30697\\Desktop\\multinomial.txt", parameters =
parameter.names,
        n.chains = 1, n.iter = 31000, n.burnin=1000, n.thin=30,
bugs.directory = openbugs.dir, debug=F, program="OpenBUGS")


#########################################################################
# Check for convergence using plots and CODA
#========================================================================

library(coda)
temp<-as.mcmc.list(model1.sim)
summary(temp)
summary(temp)$statistics[,"Naive SE"]
summary(temp)$statistics[,"Time-series SE"]
batchSE(temp)
effectiveSize(temp)
```

```
## Ineffective size, some are < 200, but it's okay.

## 1000 samples (iterations) & 1 chain allow for the following test:
geweke.diag(temp)
heidel.diag(temp)

## Imperfect convergence.
pairs(wines.bas)
pairs(wines.bas[c(1,8,9)])
pairs(wines.bas[c(6,7)])


#------------------------------------------------------------------------
## Convergence plots for the OpenBUGS program.
#------------------------------------------------------------------------
## Ergodic Plots for betas.
#------------------------------------------------------------------------
# dev.off() # delete all previous plots
par(mfrow=c(3,3)) # 9 plots per picture

for (i in 1:5){
plot(cumsum(model1.sim$sims.matrix[,i])/1:length(model1.sim$sims.matri
x[,i]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b0",i)))
}

for (i in 6:10){
plot(cumsum(model1.sim$sims.matrix[,i])/1:length(model1.sim$sims.matri
x[,i]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b1",i-5)))
}

for (i in 11:15){
plot(cumsum(model1.sim$sims.matrix[,i])/1:length(model1.sim$sims.matri
x[,i]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b2",i-10)))
}

for (i in 16:20){
plot(cumsum(model1.sim$sims.matrix[,i])/1:length(model1.sim$sims.matri
x[,i]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b3",i-15)))
}
```

```
for (i in 21:25){
plot(cumsum(model1.sim$sims.matrix[,i])/1:length(model1.sim$sims.matri
x[,i]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b4",i-20)))
}

for (i in 26:30){
plot(cumsum(model1.sim$sims.matrix[,i])/1:length(model1.sim$sims.matri
x[,i]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b5",i-25)))
}

for (i in 31:35){
plot(cumsum(model1.sim$sims.matrix[,i])/1:length(model1.sim$sims.matri
x[,i]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b6",i-30)))
}

for (i in 36:40){
plot(cumsum(model1.sim$sims.matrix[,i])/1:length(model1.sim$sims.matri
x[,i]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b7",i-35)))
}

for (i in 41:45){
plot(cumsum(model1.sim$sims.matrix[,i])/1:length(model1.sim$sims.matri
x[,i]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b8",i-40)))
}

for (i in 46:50){
plot(cumsum(model1.sim$sims.matrix[,i])/1:length(model1.sim$sims.matri
x[,i]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b9",i-45)))
}

for (i in 51:55){
plot(cumsum(model1.sim$sims.matrix[,i])/1:length(model1.sim$sims.matri
x[,i]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b10",i-50)))
}
```

```
for (i in 56:60){
plot(cumsum(model1.sim$sims.matrix[,i])/1:length(model1.sim$sims.matri
x[,i]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b0",i-55)))
}

plot(cumsum(model1.sim$sims.matrix[,61])/1:length(model1.sim$sims.matr
ix[,61]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for deviance")))

#------------------------------------------------------------------------
## Density Plots for betas.
#------------------------------------------------------------------------
# dev.off() # delete all previous plots
par(mfrow=c(3,3)) # 9 plots per picture

for (i in 1:5){
   plot(density(model1.sim$sims.matrix[,i]), main = paste(c("Density
plot for b0",i)))
}

for (i in 6:10){
   plot(density(model1.sim$sims.matrix[,i]), main = paste(c("Density
plot for b1",i-5)))
}

for (i in 11:15){
   plot(density(model1.sim$sims.matrix[,i]), main = paste(c("Density
plot for b2",i-10)))
}

for (i in 16:20){
   plot(density(model1.sim$sims.matrix[,i]), main = paste(c("Density
plot for b3",i-15)))
}

for (i in 21:25){
   plot(density(model1.sim$sims.matrix[,i]), main = paste(c("Density
plot for b4",i-20)))
}

for (i in 26:30){
   plot(density(model1.sim$sims.matrix[,i]), main = paste(c("Density
plot for b5",i-25)))
}
```

```
for (i in 31:35){
   plot(density(model1.sim$sims.matrix[,i]), main = paste(c("Density
plot for b6",i-30)))
}

for (i in 36:40){
   plot(density(model1.sim$sims.matrix[,i]), main = paste(c("Density
plot for b7",i-35)))
}

for (i in 41:45){
   plot(density(model1.sim$sims.matrix[,i]), main = paste(c("Density
plot for b8",i-40)))
}

for (i in 46:50){
   plot(density(model1.sim$sims.matrix[,i]), main = paste(c("Density
plot for b9",i-45)))
}

for (i in 51:55){
   plot(density(model1.sim$sims.matrix[,i]), main = paste(c("Density
plot for b10",i-50)))
}

for (i in 56:60){
   plot(density(model1.sim$sims.matrix[,i]), main = paste(c("Density
plot for b11",i-55)))
}

plot(density(model1.sim$sims.matrix[,61]), main = paste(c("Density
plot for deviance")))

#------------------------------------------------------------------
## Trace Plots for betas.
#------------------------------------------------------------------
# dev.off() # delete all previous plots
par(mfrow=c(3,3)) # 9 plots per picture

for (i in 1:5){
  plot(model1.sim$sims.matrix[,i], type = "l", main = paste(c("Trace
plot for b0",i)))
}

for (i in 6:10){
  plot(model1.sim$sims.matrix[,i], type = "l", main = paste(c("Trace
plot for b1",i-5)))
```

```
}

for (i in 11:15){
  plot(model1.sim$sims.matrix[,i], type = "l", main = paste(c("Trace
plot for b2",i-10)))
}

for (i in 16:20){
  plot(model1.sim$sims.matrix[,i], type = "l", main = paste(c("Trace
plot for b3",i-15)))
}

for (i in 21:25){
  plot(model1.sim$sims.matrix[,i], type = "l", main = paste(c("Trace
plot for b4",i-20)))
}

for (i in 26:30){
  plot(model1.sim$sims.matrix[,i], type = "l", main = paste(c("Trace
plot for b5",i-25)))
}

for (i in 31:35){
  plot(model1.sim$sims.matrix[,i], type = "l", main = paste(c("Trace
plot for b6",i-30)))
}

for (i in 36:40){
  plot(model1.sim$sims.matrix[,i], type = "l", main = paste(c("Trace
plot for b7",i-35)))
}

for (i in 41:45){
  plot(model1.sim$sims.matrix[,i], type = "l", main = paste(c("Trace
plot for b8",i-40)))
}

for (i in 46:50){
  plot(model1.sim$sims.matrix[,i], type = "l", main = paste(c("Trace
plot for b9",i-45)))
}

for (i in 51:55){
  plot(model1.sim$sims.matrix[,i], type = "l", main = paste(c("Trace
plot for b10",i-50)))
}

for (i in 56:60){
```

```
  plot(model1.sim$sims.matrix[,i], type = "l", main = paste(c("Trace
plot for b11",i-55)))
}

plot(model1.sim$sims.matrix[,61], type = "l", main = paste(c("Trace
plot for deviance")))


#------------------------------------------------------------------------
## Autocorrelation Plots for betas.
#------------------------------------------------------------------------
# dev.off() # delete all previous plots
par(mfrow=c(3,3)) # 9 plots per picture

for (i in 1:5){
   acf(model1.sim$sims.matrix[,i], main = paste(c("ACF plot for
b0",i)))
}

for (i in 6:10){
   acf(model1.sim$sims.matrix[,i], main = paste(c("ACF plot for b1",i-
5)))
}

for (i in 11:15){
   acf(model1.sim$sims.matrix[,i], main = paste(c("ACF plot for b2",i-
10)))
}

for (i in 16:20){
   acf(model1.sim$sims.matrix[,i], main = paste(c("ACF plot for b3",i-
15)))
}

for (i in 21:25){
   acf(model1.sim$sims.matrix[,i], main = paste(c("ACF plot for b4",i-
20)))
}

for (i in 26:30){
   acf(model1.sim$sims.matrix[,i], main = paste(c("ACF plot for b5",i-
25)))
}

for (i in 31:35){
   acf(model1.sim$sims.matrix[,i], main = paste(c("ACF plot for b6",i-
30)))
}
```

```
for (i in 36:40){
   acf(model1.sim$sims.matrix[,i], main = paste(c("ACF plot for b7",i-
35)))
}

for (i in 41:45){
   acf(model1.sim$sims.matrix[,i], main = paste(c("ACF plot for b8",i-
40)))
}

for (i in 46:50){
   acf(model1.sim$sims.matrix[,i], main = paste(c("ACF plot for b9",i-
45)))
}

for (i in 51:55){
   acf(model1.sim$sims.matrix[,i], main = paste(c("ACF plot for
b10",i-50)))
}

for (i in 56:60){
   acf(model1.sim$sims.matrix[,i], main = paste(c("ACF plot for
b11",i-55)))
}

acf(model1.sim$sims.matrix[,i], main = paste(c("ACF plot for
deviance")))


#####################################################################
# Variable Selection using BAS.
#====================================================================
# install.packages("BAS")
library(BAS)

#================================================
# Preparing our data before running BAS
#================================================
# We will separate the multinomial problem
# into 5 binomial logistic regresions, in
# order to use the package BAS.
# Our reference category will be 6.
#----------------------------------------

## Seperate our categories and make the response a binomial.
## E.g: Category 6 = 0 (baseline), Category 1 = 1
```

```
## For category 1, with reference level 6.
category1.6 <- wines.bas[wines.bas[,12]==1|wines.bas[,12]==6,]
category1.6[category1.6[,12]==6,12] <- 0

## For category 2, with reference level 6.
category2.6 <- wines.bas[wines.bas[,12]==2|wines.bas[,12]==6,]
category2.6[category2.6[,12]==6,12] <- 0
category2.6[category2.6[,12]==2,12] <- 1

## For category 3, with reference level 6.
category3.6 <- wines.bas[wines.bas[,12]==3|wines.bas[,12]==6,]
category3.6[category3.6[,12]==6,12] <- 0
category3.6[category3.6[,12]==3,12] <- 1

## For category 4, with reference level 6.
category4.6 <- wines.bas[wines.bas[,12]==4|wines.bas[,12]==6,]
category4.6[category4.6[,12]==6,12] <- 0
category4.6[category4.6[,12]==4,12] <- 1

## For category 5, with reference level 6.
category5.6 <- wines.bas[wines.bas[,12]==5|wines.bas[,12]==6,]
category5.6[category5.6[,12]==6,12] <- 0
category5.6[category5.6[,12]==5,12] <- 1

## Check that our datasets have the proper values
# table(wines.bas$quality)
# table(category1.6$quality)
# table(category2.6$quality)
# table(category3.6$quality)
# table(category4.6$quality)
# table(category5.6$quality)


#================================================
# Variable Selection through BIC, using BAS
#================================================
## For category 1, with reference level 6.
##---------------------------------------
n <- nrow(category1.6)

## Uniform Prior
#res1.1 <- bas.glm(quality~., family='binomial', data=category1.6,
betaprior=bic.prior(n), modelprior = uniform())

## Beta-Binomial Prior
res1.1 <- bas.glm(quality~., family='binomial', data=category1.6,
betaprior=bic.prior(n), modelprior = beta.binomial(2,10))
```

```
coef(res1.1)
summary(res1.1)

##---------------------------------------
## For category 2, with reference level 6.
##---------------------------------------
n <- nrow(category2.6)

## Uniform Prior
#res1.2 <- bas.glm(quality~., family='binomial', data=category2.6,
betaprior=bic.prior(n), modelprior = uniform())

## Beta-Binomial Prior
res1.2 <- bas.glm(quality~., family='binomial', data=category2.6,
betaprior=bic.prior(n), modelprior = beta.binomial(2,10))

coef(res1.2)
summary(res1.2)

##---------------------------------------
## For category 3, with reference level 6.
##---------------------------------------
n <- nrow(category3.6)

## Uniform Prior
#res1.3 <- bas.glm(quality~., family='binomial', data=category3.6,
betaprior=bic.prior(n), modelprior = uniform())

## Beta-Binomial Prior
res1.3 <- bas.glm(quality~., family='binomial', data=category3.6,
betaprior=bic.prior(n), modelprior = beta.binomial(2,10))

coef(res1.3)
summary(res1.3)

##---------------------------------------
## For category 4, with reference level 6.
##---------------------------------------
n <- nrow(category4.6)

## Uniform Prior
#res1.4 <- bas.glm(quality~., family='binomial', data=category4.6,
betaprior=bic.prior(n), modelprior = uniform())

## Beta-Binomial Prior
res1.4 <- bas.glm(quality~., family='binomial', data=category4.6,
betaprior=bic.prior(n), modelprior = beta.binomial(2,10))
```

```
coef(res1.4)
summary(res1.4)

##---------------------------------------
## For category 5, with reference level 6.
##---------------------------------------
n <- nrow(category5.6)

## Uniform Prior
#res1.5 <- bas.glm(quality~., family='binomial', data=category5.6,
betaprior=bic.prior(n), modelprior = uniform())

## Beta-Binomial Prior
res1.5 <- bas.glm(quality~., family='binomial', data=category5.6,
betaprior=bic.prior(n), modelprior = beta.binomial(2,10))

coef(res1.5)
summary(res1.5)


#=================================================
# Variable Selection through g-prior, using BAS
#=================================================
## For category 1, with reference level 6.
##---------------------------------------
n <- nrow(category1.6)

## Uniform Prior
#res2.1 <- bas.glm(quality~., family='binomial', data=category1.6,
betaprior=g.prior(n), modelprior = uniform())

## Beta-Binomial Prior
res2.1 <- bas.glm(quality~., family='binomial', data=category1.6,
betaprior=g.prior(n), modelprior = beta.binomial(2,10))

coef(res2.1)
summary(res2.1)

##---------------------------------------
## For category 2, with reference level 6.
##---------------------------------------
n <- nrow(category2.6)

## Uniform Prior
#res2.2 <- bas.glm(quality~., family='binomial', data=category2.6,
betaprior=g.prior(n), modelprior = uniform())

## Beta-Binomial Prior
```

```
res2.2 <- bas.glm(quality~., family='binomial', data=category2.6,
betaprior=g.prior(n), modelprior = beta.binomial(2,10))

coef(res2.2)
summary(res2.2)

##---------------------------------------
## For category 3, with reference level 6.
##---------------------------------------
n <- nrow(category3.6)

## Uniform Prior
#res2.3 <- bas.glm(quality~., family='binomial', data=category3.6,
betaprior=g.prior(n), modelprior = uniform())

## Beta-Binomial Prior
res2.3 <- bas.glm(quality~., family='binomial', data=category3.6,
betaprior=g.prior(n), modelprior = beta.binomial(2,10))

coef(res2.3)
summary(res2.3)

##---------------------------------------
## For category 4, with reference level 6.
##---------------------------------------
n <- nrow(category4.6)

## Uniform Prior
#res2.4 <- bas.glm(quality~., family='binomial', data=category4.6,
betaprior=g.prior(n), modelprior = uniform())

## Beta-Binomial Prior
res2.4 <- bas.glm(quality~., family='binomial', data=category4.6,
betaprior=g.prior(n), modelprior = beta.binomial(2,10))

coef(res2.4)
summary(res2.4)

##---------------------------------------
## For category 5, with reference level 6.
##---------------------------------------
n <- nrow(category5.6)

## Uniform Prior
#res2.5 <- bas.glm(quality~., family='binomial', data=category5.6,
betaprior=g.prior(n), modelprior = uniform())

## Beta-Binomial Prior
```

```
res2.5 <- bas.glm(quality~., family='binomial', data=category5.6,
betaprior=g.prior(n), modelprior = beta.binomial(2,10))

coef(res2.5)
summary(res2.5)



#================================================
# Variable Selection through hyper-g, using BAS
#================================================
## Here alpha = 3.
##---------------------------------------
## For category 1, with reference level 6.
##---------------------------------------
## Uniform Prior
#res3.1 <- bas.glm(quality~., family='binomial', data=category1.6,
betaprior=hyper.g(3), modelprior = uniform())

## Beta-Binomial
res3.1 <- bas.glm(quality~., family='binomial', data=category1.6,
betaprior=hyper.g(3), modelprior = beta.binomial(2,10))

coef(res3.1)
summary(res3.1)

##---------------------------------------
## For category 2, with reference level 6.
##---------------------------------------
## Uniform Prior
#res3.2 <- bas.glm(quality~., family='binomial', data=category2.6,
betaprior=hyper.g(3), modelprior = uniform())

## Beta-Binomial
res3.2 <- bas.glm(quality~., family='binomial', data=category2.6,
betaprior=hyper.g(3), modelprior = beta.binomial(2,10))

coef(res3.2)
summary(res3.2)

##---------------------------------------
## For category 3, with reference level 6.
##---------------------------------------
## Uniform Prior
#res3.3 <- bas.glm(quality~., family='binomial', data=category3.6,
betaprior=hyper.g(3), modelprior = uniform())
```

```
## Beta-Binomial
res3.3 <- bas.glm(quality~., family='binomial', data=category3.6,
betaprior=hyper.g(3), modelprior = beta.binomial(2,10))

coef(res3.3)
summary(res3.3)

##---------------------------------------
## For category 4, with reference level 6.
##---------------------------------------
## Uniform Prior
#res3.4 <- bas.glm(quality~., family='binomial', data=category4.6,
betaprior=hyper.g(3), modelprior = uniform())

## Beta-Binomial
res3.4 <- bas.glm(quality~., family='binomial', data=category4.6,
betaprior=hyper.g(3), modelprior = beta.binomial(2,10))

coef(res3.4)
summary(res3.4)

##---------------------------------------
## For category 5, with reference level 6.
##---------------------------------------
## Uniform Prior
#res3.5 <- bas.glm(quality~., family='binomial', data=category5.6,
betaprior=hyper.g(3), modelprior = uniform())

## Beta-Binomial
res3.5 <- bas.glm(quality~., family='binomial', data=category5.6,
betaprior=hyper.g(3), modelprior = beta.binomial(2,10))

coef(res3.5)
summary(res3.5)




#########################################################################
# Variable Selection using OpenBUGS.
#========================================================================
# Preparing our data to run the Empirical Bayes model in OpenBUGS.
#========================================================================
## Load libraries.
library(R2WinBUGS)
library(BRugs)
```

```
## Set the directory.
openbugs.dir <- "D:\\OpenBUGS323"

## Posterior Mean Estimates
prop.mean.beta0 <- c(-10, -3, 0.2, -2, -6.6)
prop.mean.beta1 <- c(1.1, 1, 0.2, 0.7, -0.5)
prop.mean.beta2 <- c(1.8, 0.5, -0.5, -0.8, -0.1)
prop.mean.beta3 <- c(0.8, 0, -0.3, -0.1, 0.5)
prop.mean.beta4 <- c(-0.1, 0.6, 0.1, 0.4, 0)
prop.mean.beta5 <- c(0.7, 0.1, -0.1, -0.5, -1.9)
prop.mean.beta6 <- c(2, -0.3, 0.2, 0.2, 0.4)
prop.mean.beta7 <- c(-4.6, -0.5, -0.5, -0.9, -1.7)
prop.mean.beta8 <- c(0.7, -1.1, -0.1, -0.6, -0.2)
prop.mean.beta9 <- c(1.9, 0.8, 0, 0.1, -1)
prop.mean.beta10 <- c(-0.6, 0.1, 0.4, 0.9, 1.2)
prop.mean.beta11 <- c(-1.6, -0.3, 0.8, 1.4, 2.3)

## Posterior Standard Deviation Estimates
prop.sd.beta0 <- c(1.7, 0.2, 0.1, 0.2, 0.8)
prop.sd.beta1 <- c(1.3, 0.5, 0.2, 0.2, 0.7)
prop.sd.beta2 <- c(0.5, 0.2, 0.1, 0.2, 0.4)
prop.sd.beta3 <- c(0.9, 0.3, 0.1, 0.2, 0.5)
prop.sd.beta4 <- c(0.5, 0.2, 0.1, 0.1, 0.4)
prop.sd.beta5 <- c(0.4, 0.2, 0.1, 0.2, 0.7)
prop.sd.beta6 <- c(1.1, 0.3, 0.1, 0.1, 0.4)
prop.sd.beta7 <- c(1.9, 0.3, 0.1, 0.2, 0.5)
prop.sd.beta8 <- c(1.1, 0.4, 0.2, 0.2, 0.6)
prop.sd.beta9 <- c(0.8, 0.3, 0.1, 0.2, 0.5)
prop.sd.beta10 <- c(0.7, 0.2, 0.1, 0.1, 0.3)
prop.sd.beta11 <- c(1, 0.3, 0.1, 0.2, 0.5)


## A new list for our data.
lwines1 <- list(y=y, x=wines1[,1:11], prop.mean.beta0=prop.mean.beta0,
           prop.mean.beta1=prop.mean.beta1,
           prop.mean.beta2=prop.mean.beta2,
           prop.mean.beta3=prop.mean.beta3,
           prop.mean.beta4=prop.mean.beta4,
           prop.mean.beta5=prop.mean.beta5,
           prop.mean.beta6=prop.mean.beta6,
           prop.mean.beta7=prop.mean.beta7,
           prop.mean.beta8=prop.mean.beta8,
           prop.mean.beta9=prop.mean.beta9,
           prop.mean.beta10=prop.mean.beta10,
           prop.mean.beta11=prop.mean.beta11,
           prop.sd.beta0=prop.sd.beta0, prop.sd.beta1=prop.sd.beta1,
           prop.sd.beta2=prop.sd.beta2, prop.sd.beta3=prop.sd.beta3,
           prop.sd.beta4=prop.sd.beta4, prop.sd.beta5=prop.sd.beta5,
```

```
           prop.sd.beta6=prop.sd.beta6, prop.sd.beta7=prop.sd.beta7,
           prop.sd.beta8=prop.sd.beta8, prop.sd.beta9=prop.sd.beta9,
           prop.sd.beta10=prop.sd.beta10,
           prop.sd.beta11=prop.sd.beta11)

## A list of initials (one chain).
inits2 <- list(
     list(b0=rep(0, times=5), b1=rep(0, times=5), b2=rep(0, times=5),
            b3=rep(0, times=5), b4=rep(0, times=5),
            b5=rep(0, times=5), b6=rep(0, times=5),
            b7=rep(0, times=5), b8=rep(0, times=5),
            b9=rep(0, times=5), b10=rep(0, times=5),
            b11=rep(0, times=5),
            gamma1=rep(0, times=5),
            gamma2=rep(0, times=5), gamma3=rep(0, times=5),
            gamma4=rep(0, times=5), gamma5=rep(0, times=5),
            gamma6=rep(0, times=5), gamma7=rep(0, times=5),
            gamma8=rep(0, times=5), gamma9=rep(0, times=5),
            gamma10=rep(0, times=5), gamma11=rep(0, times=5)#,
#           p1=0.5, p2=0.5, p3=0.5, p4=0.5, p5=0.5, p6=0.5,
#           p7=0.5, p8=0.5, p9=0.5, p10=0.5, p11=0.5
            ) )

## The parameters of interest.
parameter.names1 <- c( 'gb0', 'gb1', 'gb2', 'gb3', 'gb4', 'gb5',
'gb6', 'gb7', 'gb8', 'gb9', 'gb10', 'gb11')

## The model for the empirical Bayes method (in OpenBUGS). (Run time ~
3:30 hours)
model2.sim <- bugs( lwines1, inits2, model.file =
"C:\\Users\\30697\\Desktop\\empirical.txt", parameters =
parameter.names1,
         n.chains = 1, n.iter = 11000, n.burnin=1000, n.thin=30,
bugs.directory = openbugs.dir, debug=F, program="OpenBUGS")

##------------------------------------------------------------------
## Density Plots for betas.
##------------------------------------------------------------------
# dev.off() # delete all previous plots
par(mfrow=c(3,3)) # 9 plots per picture

for (i in 1:5){
   plot(density(model2.sim$sims.matrix[,i]), main = paste(c("Density
plot for b0",i)))
}

for (i in 6:10){
```

```
   plot(density(model2.sim$sims.matrix[,i]), main = paste(c("Density
plot for b1",i-5)))
}

for (i in 11:15){
   plot(density(model2.sim$sims.matrix[,i]), main = paste(c("Density
plot for b2",i-10)))
}

for (i in 16:20){
   plot(density(model2.sim$sims.matrix[,i]), main = paste(c("Density
plot for b3",i-15)))
}

for (i in 21:25){
   plot(density(model2.sim$sims.matrix[,i]), main = paste(c("Density
plot for b4",i-20)))
}

for (i in 26:30){
   plot(density(model2.sim$sims.matrix[,i]), main = paste(c("Density
plot for b5",i-25)))
}

for (i in 31:35){
   plot(density(model2.sim$sims.matrix[,i]), main = paste(c("Density
plot for b6",i-30)))
}

for (i in 36:40){
   plot(density(model2.sim$sims.matrix[,i]), main = paste(c("Density
plot for b7",i-35)))
}

for (i in 41:45){
   plot(density(model2.sim$sims.matrix[,i]), main = paste(c("Density
plot for b8",i-40)))
}

for (i in 46:50){
   plot(density(model2.sim$sims.matrix[,i]), main = paste(c("Density
plot for b9",i-45)))
}

for (i in 51:55){
   plot(density(model2.sim$sims.matrix[,i]), main = paste(c("Density
plot for b10",i-50)))
}
```

```
for (i in 56:60){
   plot(density(model2.sim$sims.matrix[,i]), main = paste(c("Density
plot for b11",i-55)))
}

plot(density(model2.sim$sims.matrix[,61]), main = paste(c("Density
plot for deviance")))


#=========================================================================
# Preparing our data to run the g-prior (of Liang et al.) model in
# OpenBUGS.
#=========================================================================
## Load libraries.
library(R2WinBUGS)
library(BRugs)

## Set the directory.
openbugs.dir <- "D:\\OpenBUGS323"

## Posterior Mean Estimates
prop.mean.beta1 <- c(1.1, 1, 0.2, 0.7, -0.5)
prop.mean.beta2 <- c(1.8, 0.5, -0.5, -0.8, -0.1)
prop.mean.beta3 <- c(0.8, 0, -0.3, -0.1, 0.5)
prop.mean.beta4 <- c(-0.1, 0.6, 0.1, 0.4, 0)
prop.mean.beta5 <- c(0.7, 0.1, -0.1, -0.5, -1.9)
prop.mean.beta6 <- c(2, -0.3, 0.2, 0.2, 0.4)
prop.mean.beta7 <- c(-4.6, -0.5, -0.5, -0.9, -1.7)
prop.mean.beta8 <- c(0.7, -1.1, -0.1, -0.6, -0.2)
prop.mean.beta9 <- c(1.9, 0.8, 0, 0.1, -1)
prop.mean.beta10 <- c(-0.6, 0.1, 0.4, 0.9, 1.2)
prop.mean.beta11 <- c(-1.6, -0.3, 0.8, 1.4, 2.3)

## Posterior Standard Deviation Estimates
prop.sd.beta1 <- c(1.3, 0.5, 0.2, 0.2, 0.7)
prop.sd.beta2 <- c(0.5, 0.2, 0.1, 0.2, 0.4)
prop.sd.beta3 <- c(0.9, 0.3, 0.1, 0.2, 0.5)
prop.sd.beta4 <- c(0.5, 0.2, 0.1, 0.1, 0.4)
prop.sd.beta5 <- c(0.4, 0.2, 0.1, 0.2, 0.7)
prop.sd.beta6 <- c(1.1, 0.3, 0.1, 0.1, 0.4)
prop.sd.beta7 <- c(1.9, 0.3, 0.1, 0.2, 0.5)
prop.sd.beta8 <- c(1.1, 0.4, 0.2, 0.2, 0.6)
prop.sd.beta9 <- c(0.8, 0.3, 0.1, 0.2, 0.5)
prop.sd.beta10 <- c(0.7, 0.2, 0.1, 0.1, 0.3)
prop.sd.beta11 <- c(1, 0.3, 0.1, 0.2, 0.5)

## The wines data are already centered, so
```

```
## here we only calculate the XTX matrix
XTX <- t(wines1)%*%wines1

## A new list for our data.
lwines2 <- list(y=y, XTX=XTX, x=wines1[,1:11],
           prop.mean.beta1=prop.mean.beta1,
           prop.mean.beta2=prop.mean.beta2,
           prop.mean.beta3=prop.mean.beta3,
           prop.mean.beta4=prop.mean.beta4,
           prop.mean.beta5=prop.mean.beta5,
           prop.mean.beta6=prop.mean.beta6,
           prop.mean.beta7=prop.mean.beta7,
           prop.mean.beta8=prop.mean.beta8,
           prop.mean.beta9=prop.mean.beta9,
           prop.mean.beta10=prop.mean.beta10,
           prop.mean.beta11=prop.mean.beta11,
           prop.sd.beta1=prop.sd.beta1,
           prop.sd.beta2=prop.sd.beta2, prop.sd.beta3=prop.sd.beta3,
           prop.sd.beta4=prop.sd.beta4, prop.sd.beta5=prop.sd.beta5,
           prop.sd.beta6=prop.sd.beta6, prop.sd.beta7=prop.sd.beta7,
           prop.sd.beta8=prop.sd.beta8, prop.sd.beta9=prop.sd.beta9,
           prop.sd.beta10=prop.sd.beta10,
           prop.sd.beta11=prop.sd.beta11)

## A list of initials (one chain).
inits3 <- list(
     list(alpha=rep(0.6, times=5), B1=rep(0, times=5),
           B2=rep(0, times=5), B3=rep(0, times=5), B4=rep(0, times=5),
           B5=rep(0, times=5), B6=rep(0, times=5), B7=rep(0, times=5),
           B8=rep(0, times=5), B9=rep(0, times=5),
           B10=rep(0, times=5), B11=rep(0, times=5), logtau=0,
             gamma1=rep(0, times=5),
             gamma2=rep(0, times=5), gamma3=rep(0, times=5),
             gamma4=rep(0, times=5), gamma5=rep(0, times=5),
             gamma6=rep(0, times=5), gamma7=rep(0, times=5),
             gamma8=rep(0, times=5), gamma9=rep(0, times=5),
             gamma10=rep(0, times=5), gamma11=rep(0, times=5)#,
#            p1=0.5, p2=0.5, p3=0.5, p4=0.5, p5=0.5, p6=0.5,
#            p7=0.5, p8=0.5, p9=0.5, p10=0.5, p11=0.5
             ) )

## The parameters of interest.
parameter.names2 <- c( 'alpha', 'gb1', 'gb2', 'gb3', 'gb4', 'gb5',
'gb6', 'gb7', 'gb8', 'gb9', 'gb10', 'gb11')

## The model for the gprior method (in OpenBUGS). (WARNING!!! Run time
~ 13 hours)
```

```
model3.sim <- bugs( lwines2, inits3, model.file =
"C:\\Users\\30697\\Desktop\\gprior.txt", parameters =
parameter.names2,
         n.chains = 1, n.iter = 11000, n.burnin=1000, n.thin=30,
bugs.directory = openbugs.dir, debug=F, program="OpenBUGS")

##-------------------------------------------------------------------
## Density Plots for betas.
##-------------------------------------------------------------------
# dev.off() # delete all previous plots
par(mfrow=c(3,3)) # 9 plots per picture

for (i in 1:5){
   plot(density(model3.sim$sims.matrix[,i]), main = paste(c("Density
plot for b0",i)))
}

for (i in 6:10){
   plot(density(model3.sim$sims.matrix[,i]), main = paste(c("Density
plot for b1",i-5)))
}

for (i in 11:15){
   plot(density(model3.sim$sims.matrix[,i]), main = paste(c("Density
plot for b2",i-10)))
}

for (i in 16:20){
   plot(density(model3.sim$sims.matrix[,i]), main = paste(c("Density
plot for b3",i-15)))
}

for (i in 21:25){
   plot(density(model3.sim$sims.matrix[,i]), main = paste(c("Density
plot for b4",i-20)))
}

for (i in 26:30){
   plot(density(model3.sim$sims.matrix[,i]), main = paste(c("Density
plot for b5",i-25)))
}

for (i in 31:35){
   plot(density(model3.sim$sims.matrix[,i]), main = paste(c("Density
plot for b6",i-30)))
}

for (i in 36:40){
```

```
   plot(density(model3.sim$sims.matrix[,i]), main = paste(c("Density
plot for b7",i-35)))
}

for (i in 41:45){
   plot(density(model3.sim$sims.matrix[,i]), main = paste(c("Density
plot for b8",i-40)))
}

for (i in 46:50){
   plot(density(model3.sim$sims.matrix[,i]), main = paste(c("Density
plot for b9",i-45)))
}

for (i in 51:55){
   plot(density(model3.sim$sims.matrix[,i]), main = paste(c("Density
plot for b10",i-50)))
}

for (i in 56:60){
   plot(density(model3.sim$sims.matrix[,i]), main = paste(c("Density
plot for b11",i-55)))
}

plot(density(model3.sim$sims.matrix[,61]), main = paste(c("Density
plot for deviance")))
```

```
#=========================================================================
# Preparing our data to run the hyper-g (of Liang et al.) model in
# OpenBUGS.
#=========================================================================
## Load libraries.
library(R2WinBUGS)
library(BRugs)

## Set the directory.
openbugs.dir <- "D:\\OpenBUGS323"

## Posterior Mean Estimates
prop.mean.beta1 <- c(1.1, 1, 0.2, 0.7, -0.5)
prop.mean.beta2 <- c(1.8, 0.5, -0.5, -0.8, -0.1)
prop.mean.beta3 <- c(0.8, 0, -0.3, -0.1, 0.5)
```

```
prop.mean.beta4 <- c(-0.1, 0.6, 0.1, 0.4, 0)
prop.mean.beta5 <- c(0.7, 0.1, -0.1, -0.5, -1.9)
prop.mean.beta6 <- c(2, -0.3, 0.2, 0.2, 0.4)
prop.mean.beta7 <- c(-4.6, -0.5, -0.5, -0.9, -1.7)
prop.mean.beta8 <- c(0.7, -1.1, -0.1, -0.6, -0.2)
prop.mean.beta9 <- c(1.9, 0.8, 0, 0.1, -1)
prop.mean.beta10 <- c(-0.6, 0.1, 0.4, 0.9, 1.2)
prop.mean.beta11 <- c(-1.6, -0.3, 0.8, 1.4, 2.3)

## Posterior Standard Deviation Estimates
prop.sd.beta1 <- c(1.3, 0.5, 0.2, 0.2, 0.7)
prop.sd.beta2 <- c(0.5, 0.2, 0.1, 0.2, 0.4)
prop.sd.beta3 <- c(0.9, 0.3, 0.1, 0.2, 0.5)
prop.sd.beta4 <- c(0.5, 0.2, 0.1, 0.1, 0.4)
prop.sd.beta5 <- c(0.4, 0.2, 0.1, 0.2, 0.7)
prop.sd.beta6 <- c(1.1, 0.3, 0.1, 0.1, 0.4)
prop.sd.beta7 <- c(1.9, 0.3, 0.1, 0.2, 0.5)
prop.sd.beta8 <- c(1.1, 0.4, 0.2, 0.2, 0.6)
prop.sd.beta9 <- c(0.8, 0.3, 0.1, 0.2, 0.5)
prop.sd.beta10 <- c(0.7, 0.2, 0.1, 0.1, 0.3)
prop.sd.beta11 <- c(1, 0.3, 0.1, 0.2, 0.5)

## The wines data are already centered, so
## here we only calculate the XTX matrix
XTX <- t(wines1)%*%wines1

## A new list for our data.
lwines2 <- list(y=y, XTX=XTX, x=wines1[,1:11],
           prop.mean.beta1=prop.mean.beta1,
           prop.mean.beta2=prop.mean.beta2,
           prop.mean.beta3=prop.mean.beta3,
           prop.mean.beta4=prop.mean.beta4,
           prop.mean.beta5=prop.mean.beta5,
           prop.mean.beta6=prop.mean.beta6,
           prop.mean.beta7=prop.mean.beta7,
           prop.mean.beta8=prop.mean.beta8,
           prop.mean.beta9=prop.mean.beta9,
           prop.mean.beta10=prop.mean.beta10,
           prop.mean.beta11=prop.mean.beta11,
           prop.sd.beta1=prop.sd.beta1,
           prop.sd.beta2=prop.sd.beta2, prop.sd.beta3=prop.sd.beta3,
           prop.sd.beta4=prop.sd.beta4, prop.sd.beta5=prop.sd.beta5,
           prop.sd.beta6=prop.sd.beta6, prop.sd.beta7=prop.sd.beta7,
           prop.sd.beta8=prop.sd.beta8, prop.sd.beta9=prop.sd.beta9,
           prop.sd.beta10=prop.sd.beta10,
           prop.sd.beta11=prop.sd.beta11)

## A list of initials (one chain).
```

```
inits4 <- list(
     list(alpha=rep(0.6, times=5), B1=rep(0, times=5),
            B2=rep(0, times=5), B3=rep(0, times=5),
            B4=rep(0, times=5), B5=rep(0, times=5),
            B6=rep(0, times=5), B7=rep(0, times=5),
            B8=rep(0, times=5), B9=rep(0, times=5),
            B10=rep(0, times=5), B11=rep(0, times=5),
            logtau=0, w=0.7,
            gamma1=rep(0, times=5),
            gamma2=rep(0, times=5), gamma3=rep(0, times=5),
            gamma4=rep(0, times=5), gamma5=rep(0, times=5),
            gamma6=rep(0, times=5), gamma7=rep(0, times=5),
            gamma8=rep(0, times=5), gamma9=rep(0, times=5),
            gamma10=rep(0, times=5), gamma11=rep(0, times=5)#,
#            p1=0.5, p2=0.5, p3=0.5, p4=0.5, p5=0.5, p6=0.5,
#            p7=0.5, p8=0.5, p9=0.5, p10=0.5, p11=0.5
            ) )

## The parameters of interest.
parameter.names2 <- c( 'alpha', 'gb1', 'gb2', 'gb3', 'gb4', 'gb5',
'gb6', 'gb7', 'gb8', 'gb9', 'gb10', 'gb11')

## The model for the gprior method (in OpenBUGS). (WARNING!!! Run time
~ 13 hours)
model4.sim <- bugs( lwines2, inits4, model.file =
"C:\\Users\\30697\\Desktop\\hyperg.txt", parameters =
parameter.names2,
         n.chains = 1, n.iter = 11000, n.burnin=1000, n.thin=30,
bugs.directory = openbugs.dir, debug=F, program="OpenBUGS")

##-------------------------------------------------------------------
## Density Plots for betas.
##-------------------------------------------------------------------
# dev.off() # delete all previous plots
par(mfrow=c(3,3)) # 9 plots per picture

for (i in 1:5){
   plot(density(model4.sim$sims.matrix[,i]), main = paste(c("Density
plot for b0",i)))
}

for (i in 6:10){
   plot(density(model4.sim$sims.matrix[,i]), main = paste(c("Density
plot for b1",i-5)))
}

for (i in 11:15){
```

```
   plot(density(model4.sim$sims.matrix[,i]), main = paste(c("Density
plot for b2",i-10)))
}

for (i in 16:20){
   plot(density(model4.sim$sims.matrix[,i]), main = paste(c("Density
plot for b3",i-15)))
}

for (i in 21:25){
   plot(density(model4.sim$sims.matrix[,i]), main = paste(c("Density
plot for b4",i-20)))
}

for (i in 26:30){
   plot(density(model4.sim$sims.matrix[,i]), main = paste(c("Density
plot for b5",i-25)))
}

for (i in 31:35){
   plot(density(model4.sim$sims.matrix[,i]), main = paste(c("Density
plot for b6",i-30)))
}

for (i in 36:40){
   plot(density(model4.sim$sims.matrix[,i]), main = paste(c("Density
plot for b7",i-35)))
}

for (i in 41:45){
   plot(density(model4.sim$sims.matrix[,i]), main = paste(c("Density
plot for b8",i-40)))
}

for (i in 46:50){
   plot(density(model4.sim$sims.matrix[,i]), main = paste(c("Density
plot for b9",i-45)))
}

for (i in 51:55){
   plot(density(model4.sim$sims.matrix[,i]), main = paste(c("Density
plot for b10",i-50)))
}

for (i in 56:60){
   plot(density(model4.sim$sims.matrix[,i]), main = paste(c("Density
plot for b11",i-55)))
}
```

```
plot(density(model4.sim$sims.matrix[,61]), main = paste(c("Density
plot for deviance")))



#########################################################################
# Run our simpler (reduced) Model in R.
#========================================================================

## Load libraries.
library(R2WinBUGS)
library(BRugs)

## Set the directory.
openbugs.dir <- "D:\\OpenBUGS323"

## A list of initials (one chain).
inits5 <- list(
     list(b01=0, b02=0, b03=0, b04=0, b05=0, b11=0, b14=0, b15=0,
            b21=0, b22=0, b23=0, b24=0, b35=0, b42=0, b44=0, b45=0,
            b51=0, b53=0, b54=0, b55=0, b61=0, b63=0, b64=0, b65=0,
            b71=0, b72=0, b73=0, b74=0, b75=0, b84=0, b91=0, b92=0,
            b95=0, b103=0, b104=0, b105=0, b111=0, b113=0, b114=0,
            b115=0) )

## The parameters of interest.
parameter.names3 <- c( 'b01', 'b02', 'b03', 'b04', 'b05', 'b11',
                       'b14', 'b15', 'b21', 'b22', 'b23', 'b24',
                       'b35', 'b42', 'b44', 'b45', 'b51', 'b53',
                       'b54', 'b55', 'b61', 'b63', 'b64', 'b65',
                       'b71', 'b72', 'b73', 'b74', 'b75', 'b84',
                       'b91', 'b92', 'b95', 'b103', 'b104',
                       'b105', 'b111', 'b113', 'b114', 'b115',
                       'B01', 'B02', 'B03', 'B04', 'B05', 'B11',
                       'B14', 'B15', 'B21', 'B22', 'B23', 'B24',
                       'B35', 'B42', 'B44', 'B45', 'B51', 'B53',
                       'B54', 'B55', 'B61', 'B63', 'B64', 'B65',
                       'B71', 'B72', 'B73', 'B74', 'B75', 'B84',
                       'B91', 'B92', 'B95', 'B103', 'B104',
                       'B105', 'B111', 'B113', 'B114', 'B115')

## The model (in OpenBUGS). (Run time for 1 chain ~ 50 minutes)
model5.sim <- bugs( lwines, inits5, model.file =
"C:\\Users\\30697\\Desktop\\multinomial2.txt", parameters =
parameter.names3,
```

```
         n.chains = 1, n.iter = 51000, n.burnin=1000, n.thin=50,
bugs.directory = openbugs.dir, debug=F, program="OpenBUGS")


#########################################################################
# Check for convergence using plots and CODA
#========================================================================

library(coda)
temp<-as.mcmc.list(model5.sim)
summary(temp)
summary(temp)$statistics[,"Naive SE"]
summary(temp)$statistics[,"Time-series SE"]
batchSE(temp)
effectiveSize(temp)

## 1000 samples (iterations) allow for the following tests:
geweke.diag(temp)
heidel.diag(temp)


##------------------------------------------------------------------------
## Ergodic Plots for betas.
##------------------------------------------------------------------------
par(mfrow=c(3,3)) # 9 plots per picture

plot(cumsum(model5.sim$sims.matrix[,1])/1:length(model5.sim$sims.matri
x[,1]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b01")))
plot(cumsum(model5.sim$sims.matrix[,2])/1:length(model5.sim$sims.matri
x[,2]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b02")))
plot(cumsum(model5.sim$sims.matrix[,3])/1:length(model5.sim$sims.matri
x[,3]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b03")))
plot(cumsum(model5.sim$sims.matrix[,4])/1:length(model5.sim$sims.matri
x[,4]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b04")))
plot(cumsum(model5.sim$sims.matrix[,5])/1:length(model5.sim$sims.matri
x[,5]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b05")))
plot(cumsum(model5.sim$sims.matrix[,6])/1:length(model5.sim$sims.matri
x[,6]),
```

```
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b11")))
plot(cumsum(model5.sim$sims.matrix[,7])/1:length(model5.sim$sims.matri
x[,7]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b14")))
plot(cumsum(model5.sim$sims.matrix[,8])/1:length(model5.sim$sims.matri
x[,8]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b15")))
plot(cumsum(model5.sim$sims.matrix[,9])/1:length(model5.sim$sims.matri
x[,9]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b21")))
plot(cumsum(model5.sim$sims.matrix[,10])/1:length(model5.sim$sims.matr
ix[,10]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b22")))
plot(cumsum(model5.sim$sims.matrix[,11])/1:length(model5.sim$sims.matr
ix[,11]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b23")))
plot(cumsum(model5.sim$sims.matrix[,12])/1:length(model5.sim$sims.matr
ix[,12]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b24")))
plot(cumsum(model5.sim$sims.matrix[,13])/1:length(model5.sim$sims.matr
ix[,13]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b35")))
plot(cumsum(model5.sim$sims.matrix[,14])/1:length(model5.sim$sims.matr
ix[,14]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b42")))
plot(cumsum(model5.sim$sims.matrix[,15])/1:length(model5.sim$sims.matr
ix[,15]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b44")))
plot(cumsum(model5.sim$sims.matrix[,16])/1:length(model5.sim$sims.matr
ix[,16]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b45")))
plot(cumsum(model5.sim$sims.matrix[,17])/1:length(model5.sim$sims.matr
ix[,17]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b51")))
plot(cumsum(model5.sim$sims.matrix[,18])/1:length(model5.sim$sims.matr
ix[,18]),
```

```
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b53")))
plot(cumsum(model5.sim$sims.matrix[,19])/1:length(model5.sim$sims.matr
ix[,19]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b54")))
plot(cumsum(model5.sim$sims.matrix[,20])/1:length(model5.sim$sims.matr
ix[,20]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b55")))
plot(cumsum(model5.sim$sims.matrix[,21])/1:length(model5.sim$sims.matr
ix[,21]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b61")))
plot(cumsum(model5.sim$sims.matrix[,22])/1:length(model5.sim$sims.matr
ix[,22]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b63")))
plot(cumsum(model5.sim$sims.matrix[,23])/1:length(model5.sim$sims.matr
ix[,23]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b64")))
plot(cumsum(model5.sim$sims.matrix[,24])/1:length(model5.sim$sims.matr
ix[,24]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b65")))
plot(cumsum(model5.sim$sims.matrix[,25])/1:length(model5.sim$sims.matr
ix[,25]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b71")))
plot(cumsum(model5.sim$sims.matrix[,26])/1:length(model5.sim$sims.matr
ix[,26]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b72")))
plot(cumsum(model5.sim$sims.matrix[,27])/1:length(model5.sim$sims.matr
ix[,27]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b73")))
plot(cumsum(model5.sim$sims.matrix[,28])/1:length(model5.sim$sims.matr
ix[,28]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b74")))
plot(cumsum(model5.sim$sims.matrix[,29])/1:length(model5.sim$sims.matr
ix[,29]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b75")))
plot(cumsum(model5.sim$sims.matrix[,30])/1:length(model5.sim$sims.matr
ix[,30]),
```

```
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b84")))
plot(cumsum(model5.sim$sims.matrix[,31])/1:length(model5.sim$sims.matr
ix[,31]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b91")))
plot(cumsum(model5.sim$sims.matrix[,32])/1:length(model5.sim$sims.matr
ix[,32]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b92")))
plot(cumsum(model5.sim$sims.matrix[,33])/1:length(model5.sim$sims.matr
ix[,33]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b95")))
plot(cumsum(model5.sim$sims.matrix[,34])/1:length(model5.sim$sims.matr
ix[,34]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b103")))
plot(cumsum(model5.sim$sims.matrix[,35])/1:length(model5.sim$sims.matr
ix[,35]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b104")))
plot(cumsum(model5.sim$sims.matrix[,36])/1:length(model5.sim$sims.matr
ix[,36]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b105")))
plot(cumsum(model5.sim$sims.matrix[,37])/1:length(model5.sim$sims.matr
ix[,37]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b111")))
plot(cumsum(model5.sim$sims.matrix[,38])/1:length(model5.sim$sims.matr
ix[,38]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b113")))
plot(cumsum(model5.sim$sims.matrix[,39])/1:length(model5.sim$sims.matr
ix[,39]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b114")))
plot(cumsum(model5.sim$sims.matrix[,40])/1:length(model5.sim$sims.matr
ix[,40]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for b115")))
plot(cumsum(model5.sim$sims.matrix[,81])/1:length(model5.sim$sims.matr
ix[,81]),
type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean Plot
for deviance")))
```

```
##-----------------------------------------------------------------------
## Density Plots for betas.
##-----------------------------------------------------------------------
# dev.off() # delete all previous plots
par(mfrow=c(3,3)) # 9 plots per picture

plot(density(model5.sim$sims.matrix[,1]), main = paste(c("Density plot
for b01")))
plot(density(model5.sim$sims.matrix[,2]), main = paste(c("Density plot
for b02")))
plot(density(model5.sim$sims.matrix[,3]), main = paste(c("Density plot
for b03")))
plot(density(model5.sim$sims.matrix[,4]), main = paste(c("Density plot
for b04")))
plot(density(model5.sim$sims.matrix[,5]), main = paste(c("Density plot
for b05")))
plot(density(model5.sim$sims.matrix[,6]), main = paste(c("Density plot
for b11")))
plot(density(model5.sim$sims.matrix[,7]), main = paste(c("Density plot
for b14")))
plot(density(model5.sim$sims.matrix[,8]), main = paste(c("Density plot
for b15")))
plot(density(model5.sim$sims.matrix[,9]), main = paste(c("Density plot
for b21")))
plot(density(model5.sim$sims.matrix[,10]), main = paste(c("Density
plot for b22")))
plot(density(model5.sim$sims.matrix[,11]), main = paste(c("Density
plot for b23")))
plot(density(model5.sim$sims.matrix[,12]), main = paste(c("Density
plot for b24")))
plot(density(model5.sim$sims.matrix[,13]), main = paste(c("Density
plot for b35")))
plot(density(model5.sim$sims.matrix[,14]), main = paste(c("Density
plot for b42")))
plot(density(model5.sim$sims.matrix[,15]), main = paste(c("Density
plot for b44")))
plot(density(model5.sim$sims.matrix[,16]), main = paste(c("Density
plot for b45")))
plot(density(model5.sim$sims.matrix[,17]), main = paste(c("Density
plot for b51")))
plot(density(model5.sim$sims.matrix[,18]), main = paste(c("Density
plot for b53")))
plot(density(model5.sim$sims.matrix[,19]), main = paste(c("Density
plot for b54")))
plot(density(model5.sim$sims.matrix[,20]), main = paste(c("Density
plot for b55")))
plot(density(model5.sim$sims.matrix[,21]), main = paste(c("Density
plot for b61")))
```

```
plot(density(model5.sim$sims.matrix[,22]), main = paste(c("Density
plot for b63")))
plot(density(model5.sim$sims.matrix[,23]), main = paste(c("Density
plot for b64")))
plot(density(model5.sim$sims.matrix[,24]), main = paste(c("Density
plot for b65")))
plot(density(model5.sim$sims.matrix[,25]), main = paste(c("Density
plot for b71")))
plot(density(model5.sim$sims.matrix[,26]), main = paste(c("Density
plot for b72")))
plot(density(model5.sim$sims.matrix[,27]), main = paste(c("Density
plot for b73")))
plot(density(model5.sim$sims.matrix[,28]), main = paste(c("Density
plot for b74")))
plot(density(model5.sim$sims.matrix[,29]), main = paste(c("Density
plot for b75")))
plot(density(model5.sim$sims.matrix[,30]), main = paste(c("Density
plot for b84")))
plot(density(model5.sim$sims.matrix[,31]), main = paste(c("Density
plot for b91")))
plot(density(model5.sim$sims.matrix[,32]), main = paste(c("Density
plot for b92")))
plot(density(model5.sim$sims.matrix[,33]), main = paste(c("Density
plot for b95")))
plot(density(model5.sim$sims.matrix[,34]), main = paste(c("Density
plot for b103")))
plot(density(model5.sim$sims.matrix[,35]), main = paste(c("Density
plot for b104")))
plot(density(model5.sim$sims.matrix[,36]), main = paste(c("Density
plot for b105")))
plot(density(model5.sim$sims.matrix[,37]), main = paste(c("Density
plot for b111")))
plot(density(model5.sim$sims.matrix[,38]), main = paste(c("Density
plot for b113")))
plot(density(model5.sim$sims.matrix[,39]), main = paste(c("Density
plot for b114")))
plot(density(model5.sim$sims.matrix[,40]), main = paste(c("Density
plot for b115")))
plot(density(model5.sim$sims.matrix[,81]), main = paste(c("Density
plot for deviance")))


##-------------------------------------------------------------------
## Trace Plots for betas.
##-------------------------------------------------------------------
# dev.off() # delete all previous plots
par(mfrow=c(3,3)) # 9 plots per picture
```

```
plot(model5.sim$sims.matrix[,1], type = "l", main = paste(c("Trace
plot for b01")))
plot(model5.sim$sims.matrix[,2], type = "l", main = paste(c("Trace
plot for b02")))
plot(model5.sim$sims.matrix[,3], type = "l", main = paste(c("Trace
plot for b03")))
plot(model5.sim$sims.matrix[,4], type = "l", main = paste(c("Trace
plot for b04")))
plot(model5.sim$sims.matrix[,5], type = "l", main = paste(c("Trace
plot for b05")))
plot(model5.sim$sims.matrix[,6], type = "l", main = paste(c("Trace
plot for b11")))
plot(model5.sim$sims.matrix[,7], type = "l", main = paste(c("Trace
plot for b14")))
plot(model5.sim$sims.matrix[,8], type = "l", main = paste(c("Trace
plot for b15")))
plot(model5.sim$sims.matrix[,9], type = "l", main = paste(c("Trace
plot for b21")))
plot(model5.sim$sims.matrix[,10], type = "l", main = paste(c("Trace
plot for b22")))
plot(model5.sim$sims.matrix[,11], type = "l", main = paste(c("Trace
plot for b23")))
plot(model5.sim$sims.matrix[,12], type = "l", main = paste(c("Trace
plot for b24")))
plot(model5.sim$sims.matrix[,13], type = "l", main = paste(c("Trace
plot for b35")))
plot(model5.sim$sims.matrix[,14], type = "l", main = paste(c("Trace
plot for b42")))
plot(model5.sim$sims.matrix[,15], type = "l", main = paste(c("Trace
plot for b44")))
plot(model5.sim$sims.matrix[,16], type = "l", main = paste(c("Trace
plot for b45")))
plot(model5.sim$sims.matrix[,17], type = "l", main = paste(c("Trace
plot for b51")))
plot(model5.sim$sims.matrix[,18], type = "l", main = paste(c("Trace
plot for b53")))
plot(model5.sim$sims.matrix[,19], type = "l", main = paste(c("Trace
plot for b54")))
plot(model5.sim$sims.matrix[,20], type = "l", main = paste(c("Trace
plot for b55")))
plot(model5.sim$sims.matrix[,21], type = "l", main = paste(c("Trace
plot for b61")))
plot(model5.sim$sims.matrix[,22], type = "l", main = paste(c("Trace
plot for b63")))
plot(model5.sim$sims.matrix[,23], type = "l", main = paste(c("Trace
plot for b64")))
plot(model5.sim$sims.matrix[,24], type = "l", main = paste(c("Trace
plot for b65")))
```

```
plot(model5.sim$sims.matrix[,25], type = "l", main = paste(c("Trace
plot for b71")))
plot(model5.sim$sims.matrix[,26], type = "l", main = paste(c("Trace
plot for b72")))
plot(model5.sim$sims.matrix[,27], type = "l", main = paste(c("Trace
plot for b73")))
plot(model5.sim$sims.matrix[,28], type = "l", main = paste(c("Trace
plot for b74")))
plot(model5.sim$sims.matrix[,29], type = "l", main = paste(c("Trace
plot for b75")))
plot(model5.sim$sims.matrix[,30], type = "l", main = paste(c("Trace
plot for b84")))
plot(model5.sim$sims.matrix[,31], type = "l", main = paste(c("Trace
plot for b91")))
plot(model5.sim$sims.matrix[,32], type = "l", main = paste(c("Trace
plot for b92")))
plot(model5.sim$sims.matrix[,33], type = "l", main = paste(c("Trace
plot for b95")))
plot(model5.sim$sims.matrix[,34], type = "l", main = paste(c("Trace
plot for b103")))
plot(model5.sim$sims.matrix[,35], type = "l", main = paste(c("Trace
plot for b104")))
plot(model5.sim$sims.matrix[,36], type = "l", main = paste(c("Trace
plot for b105")))
plot(model5.sim$sims.matrix[,37], type = "l", main = paste(c("Trace
plot for b111")))
plot(model5.sim$sims.matrix[,38], type = "l", main = paste(c("Trace
plot for b113")))
plot(model5.sim$sims.matrix[,39], type = "l", main = paste(c("Trace
plot for b114")))
plot(model5.sim$sims.matrix[,40], type = "l", main = paste(c("Trace
plot for b115")))
plot(model5.sim$sims.matrix[,81], type = "l", main = paste(c("Trace
plot for deviance")))


##-------------------------------------------------------------------
## Autocorrelation Plots for betas.
##-------------------------------------------------------------------
# dev.off() # delete all previous plots
par(mfrow=c(3,3)) # 9 plots per picture

acf(model5.sim$sims.matrix[,1], main = paste(c("ACF plot for b01")))
acf(model5.sim$sims.matrix[,2], main = paste(c("ACF plot for b02")))
acf(model5.sim$sims.matrix[,3], main = paste(c("ACF plot for b03")))
acf(model5.sim$sims.matrix[,4], main = paste(c("ACF plot for b04")))
acf(model5.sim$sims.matrix[,5], main = paste(c("ACF plot for b05")))
acf(model5.sim$sims.matrix[,6], main = paste(c("ACF plot for b11")))
```

```
acf(model5.sim$sims.matrix[,7], main = paste(c("ACF plot for b14")))
acf(model5.sim$sims.matrix[,8], main = paste(c("ACF plot for b15")))
acf(model5.sim$sims.matrix[,9], main = paste(c("ACF plot for b21")))
acf(model5.sim$sims.matrix[,10], main = paste(c("ACF plot for b22")))
acf(model5.sim$sims.matrix[,11], main = paste(c("ACF plot for b23")))
acf(model5.sim$sims.matrix[,12], main = paste(c("ACF plot for b24")))
acf(model5.sim$sims.matrix[,13], main = paste(c("ACF plot for b35")))
acf(model5.sim$sims.matrix[,14], main = paste(c("ACF plot for b42")))
acf(model5.sim$sims.matrix[,15], main = paste(c("ACF plot for b44")))
acf(model5.sim$sims.matrix[,16], main = paste(c("ACF plot for b45")))
acf(model5.sim$sims.matrix[,17], main = paste(c("ACF plot for b51")))
acf(model5.sim$sims.matrix[,18], main = paste(c("ACF plot for b53")))
acf(model5.sim$sims.matrix[,19], main = paste(c("ACF plot for b54")))
acf(model5.sim$sims.matrix[,20], main = paste(c("ACF plot for b55")))
acf(model5.sim$sims.matrix[,21], main = paste(c("ACF plot for b61")))
acf(model5.sim$sims.matrix[,22], main = paste(c("ACF plot for b63")))
acf(model5.sim$sims.matrix[,23], main = paste(c("ACF plot for b64")))
acf(model5.sim$sims.matrix[,24], main = paste(c("ACF plot for b65")))
acf(model5.sim$sims.matrix[,25], main = paste(c("ACF plot for b71")))
acf(model5.sim$sims.matrix[,26], main = paste(c("ACF plot for b72")))
acf(model5.sim$sims.matrix[,27], main = paste(c("ACF plot for b73")))
acf(model5.sim$sims.matrix[,28], main = paste(c("ACF plot for b74")))
acf(model5.sim$sims.matrix[,29], main = paste(c("ACF plot for b75")))
acf(model5.sim$sims.matrix[,30], main = paste(c("ACF plot for b84")))
acf(model5.sim$sims.matrix[,31], main = paste(c("ACF plot for b91")))
acf(model5.sim$sims.matrix[,32], main = paste(c("ACF plot for b92")))
acf(model5.sim$sims.matrix[,33], main = paste(c("ACF plot for b95")))
acf(model5.sim$sims.matrix[,34], main = paste(c("ACF plot for b103")))
acf(model5.sim$sims.matrix[,35], main = paste(c("ACF plot for b104")))
acf(model5.sim$sims.matrix[,36], main = paste(c("ACF plot for b105")))
acf(model5.sim$sims.matrix[,37], main = paste(c("ACF plot for b111")))
acf(model5.sim$sims.matrix[,38], main = paste(c("ACF plot for b113")))
acf(model5.sim$sims.matrix[,39], main = paste(c("ACF plot for b114")))
acf(model5.sim$sims.matrix[,40], main = paste(c("ACF plot for b115")))
acf(model5.sim$sims.matrix[,81], main = paste(c("ACF plot for
deviance")))
```

```
#########################################################################
# Initial Statistics & Plots for our Analysis for the Binomial Case
#========================================================================
## Load the data into R.
wines <- read.csv("C:\\Users\\30697\\Desktop\\06_winequality-red.csv",
header = T, sep=";", stringsAsFactors = FALSE)
```

```
## Change the values from mg/dm^3 to g/dm^3 for free and total sulfur
## dioxide.
wines$free.sulfur.dioxide <- wines$free.sulfur.dioxide*0.001
wines$total.sulfur.dioxide <- wines$total.sulfur.dioxide*0.001

## Plot of the sensory data.
plot(factor(wines$quality), xlab = "Sensory Preference (Median)", ylab
= "Frequency (Red Wine Samples)")

## Quality with 6 Categories (3-8).
table(wines$quality)

## Shift the response variable from 3 until 8, to 1 until 6.
wines[wines[,12]==3,12] <- 1
wines[wines[,12]==4,12] <- 2
wines[wines[,12]==5,12] <- 3
wines[wines[,12]==6,12] <- 4
wines[wines[,12]==7,12] <- 5
wines[wines[,12]==8,12] <- 6

## Check if it worked.
table(wines$quality)
plot(factor(wines$quality), xlab = "Sensory Preference (Median)", ylab
= "Frequency (Red Wine Samples)")

## Split the data response into binary
wines$quality <- ifelse(wines$quality<=3, 0, 1)
y <- wines$quality
table(wines$quality)
# 10+53+681=744 zeros
# 638+199+18=855 ones

##==============================
## Variable Selection using BAS
##==============================
library(BAS)
wines.bas2 <- wines

n <- 1599
res <- bas.glm(quality~., family='binomial', data=wines.bas2,
betaprior=g.prior(n), modelprior = beta.binomial(2,10))

coef(res)
summary(res)


## Turn our data frame into a matrix and then into a list.
```

```
wines <- scale(wines[,-12], center=TRUE, scale=TRUE) # if we want to
center our values
wines1 <- as.matrix(wines)
y <- as.matrix(y)
lwines <- list(y=y,x=wines1[,1:11])


#########################################################################
# Run our Binomial Model in R.
#========================================================================
## Load libraries.
library(R2WinBUGS)
library(BRugs)

## Set the directory.
openbugs.dir <- "D:\\OpenBUGS323"

## A list of initials (one chain).
inits6 <- list(
     list(b=rep(0, times=7)) )

## The parameters of interest.
parameter.names6 <- c('b', 'B')

## The model (in OpenBUGS).
model6.sim <- bugs( lwines, inits6, model.file =
"C:\\Users\\30697\\Desktop\\binomial2.txt", parameters =
parameter.names6,
n.chains = 1, n.iter = 11000, n.burnin=1000, n.thin=1,  bugs.directory
= openbugs.dir, debug=F, program="OpenBUGS")


#########################################################################
# Check for convergence using plots and CODA
#========================================================================
library(coda)
temp<-as.mcmc.list(model5.sim)
summary(temp)
summary(temp)$statistics[,"Naive SE"]
summary(temp)$statistics[,"Time-series SE"]
batchSE(temp)
effectiveSize(temp)

## Automated Convergence Tests.
codamenu()
2
temp
2
```

```
1
4
3
2
4
2
8
4
y

## Ergodic Plots for bs.
for (i in 1:7){

plot(cumsum(model1.sim$sims.matrix[,i])/1:length(model1.sim$sims.matri
x[,i]), type="l", ylab = "Cumsum Values", main = paste(c("Ergodic Mean
Plot for b",i)))
}
plot(cumsum(model1.sim$sims.matrix[,15])/1:length(model1.sim$sims.matr
ix[,15]), type="l", ylab = "Cumsum Values", main = paste(c("Ergodic
Mean Plot for deviance")))

## Autocorrelation Plots for bs.
for (i in 1:7){
   acf(model1.sim$sims.matrix[,i], main = paste(c("ACF plot for
b",i)))
}
acf(model1.sim$sims.matrix[,15], main = paste(c("ACF plot for
deviance")))

## Density Plots for bs.
for (i in 1:7){
   plot(density(model1.sim$sims.matrix[,i]), main = paste(c("Density
plot for b",i)))
}
plot(density(model1.sim$sims.matrix[,15]), main = paste(c("Density
plot for deviance")))

## Trace Plots for b0s.
for (i in 1:7){
  plot(model1.sim$sims.matrix[,i], type = "l", main = paste(c("Trace
plot for b",i)))
}
plot(model1.sim$sims.matrix[,15], type = "l", main = paste(c("Trace
plot for deviance")))
```

BIBLIOGRAPHY

Cortez P., Cerdeira A., Almeida F., Matos T. and Reis J.. Modeling wine preferences by data mining from physicochemical properties. In Decision Support Systems, Elsevier, 47(4): 547-553, 2009.

Ntzoufras, I. 2009. Bayesian Modelling Using WinBUGS. Wiley.

Agresti, A. 2019. An introduction to categorical data analysis. Third Edition. Hoboken, NJ: Wiley-Interscience.

Korner-Nievergelt F., Roth T., Felten S., Guélat J., Almasi B., Korner-Nievergelt P. 2015. Bayesian data analysis in ecology using linear models with R, BUGS, and Stan. Academic Press; London: 2015.

Best, Nicky, et al. 2012. The BUGS Book - A Practical Introduction to Bayesian Analysis.