Linear regression done via MCMC

Welcome to the SAA for fitting a linear regression using MCMC

Firstly on this page you will need to specify the dataset required from the list of available datasets.

Which dataset do you wish to use?:

Submit

Next you need to choose the response and predictor variables from the chosen dataset. After choosing these variables the SAA will run and you will see a block of text describing how many observations are to be used at the bottom of this page. The rest of the analysis will appear in pages 2-7.

What is the response variable?: normexam

What is the predictor variable?: standlrt

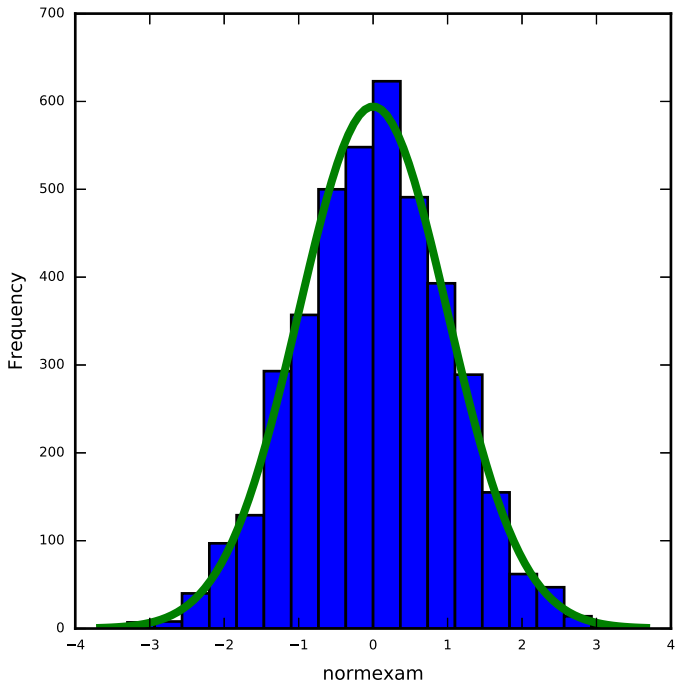
The Analysis Assistant you are currently using is designed to work on complete datasets only and so as a pre-processing step we have to remove any rows that contain missing data in columns used in the analysis that follows. For now the list of columns to be considered is: normexam, standlrt. There are 0 (0.0%) rows that get deleted This results in a dataset of 4059 rows.

We will begin our analysis of the dataset by doing some basic data exploration.

You have chosen normexam as your response variable and so a first step is to take a look at this variable and assess its suitability for a normal model. The summary statistics for the variable are in the table below:

Observations	4059
Mean	0.0
Standard Deviation	0.999
Median	0.004

We also look at a histogram of normexam to see if it is approximately normally distributed. Although in modelling the response in terms of a set of predictors it is what is unexplained (the model residuals) that need to be normally distributed, it is still useful to look at the response variable as a very skewed variable will often lead to very skewed residuals.



Here the distribution is reasonably symmetric with skewness value 0.004.

The values:

Row	normexam
88	3.13405
124	3.13405
1324	-3.05954
1785	-3.05954
1786	-3.05954
1826	-3.05954
2129	3.66609
2198	-3.05954
2310	-3.05954
3210	-3.05954
3376	3.13405
3386	3.37474
3510	-3.66607

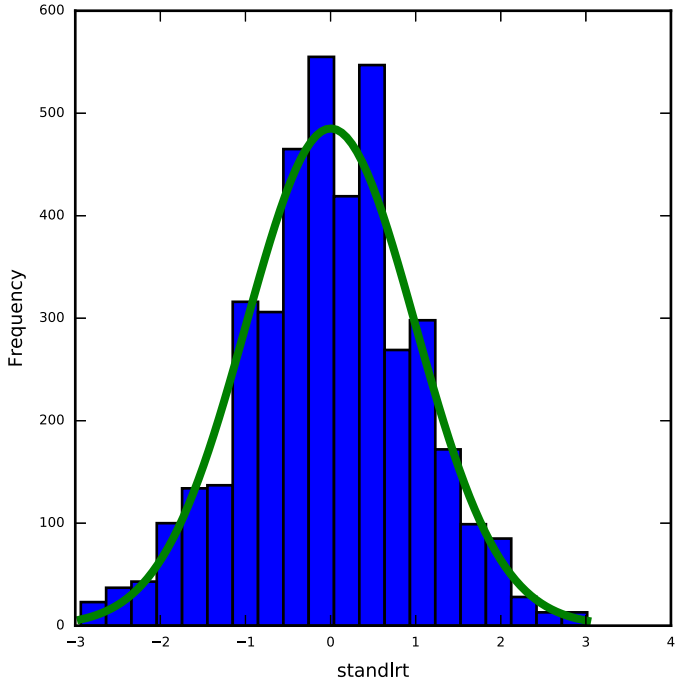
are greater than 3 sds from the mean. This might warrant investigating.

We can also look at the predictor variables that we have chosen.

For continuous predictors we are interested in looking at summary statistics, the shape of the distribution and any unusual values. If the distribution is skewed then we might want to transform the variable before fitting it in the model although it is more important to consider transformations of the response variable and remember what is important is whether the relationship between the response and predictor is linear. If there are unusual values we will want to check that the unusual values are correct and not errors and also whether we may want to treat the variable differently. Another possibility for unusual shaped distributions is to instead categorise the variable into ranges of values.

For predictor standlrt we see the following:

Name	standlrt
Observations	4059
Mean	0.002
Standard Deviation	0.993
Median	0.04



Here the median is larger than the mean and there is significant skew to the left. The skewness value is -0.128. Here the statistical significance may be to some degree due to the large sample size as from a practical perspective values of skew less than 2 in absolute magnitude are not considered too big a skew.

The values:

Row	standlrt
532	3.01595
1299	3.01595
1613	3.01595

are greater than 3 sds from the mean. This might warrant investigating.

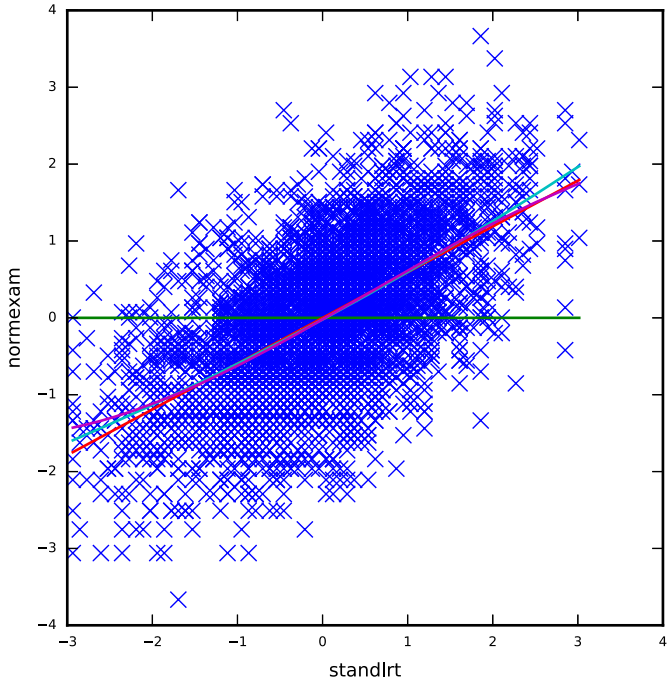
Once we are happy with our response variable and our predictor variable we now want to have a preliminary look at them together before progressing to the linear regression.

For the predictor we can look at correlations with the response and scatterplots with best fitting curves to see if there is a linear relationship.

Predictor: standlrt

The Pearson correlation between normexam and standlrt is 0.592 (p value < 0.001).

The Spearman rank correlation between normexam and standlrt is 0.58 (p value < 0.001).

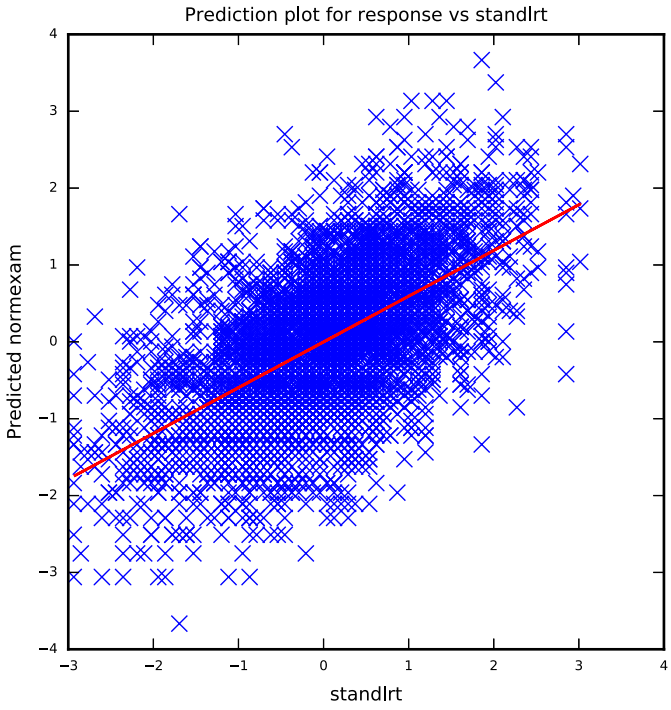


The graph includes best fitting curves for a constant, linear, quadratic and cubic relationship between normexam and standlrt. In this case a quadratic relationship is most appropriate and you might consider including a squared term in the predictor list.

Here we simply fit the linear regression model using MCMC for our chosen predictor.

Variable	Coefficient	SE	ESS
standlrt	0.595	0.0127	5960
Intercept	-0.00129	0.0127	6129
sigmasq	0.649	0.0143	5784

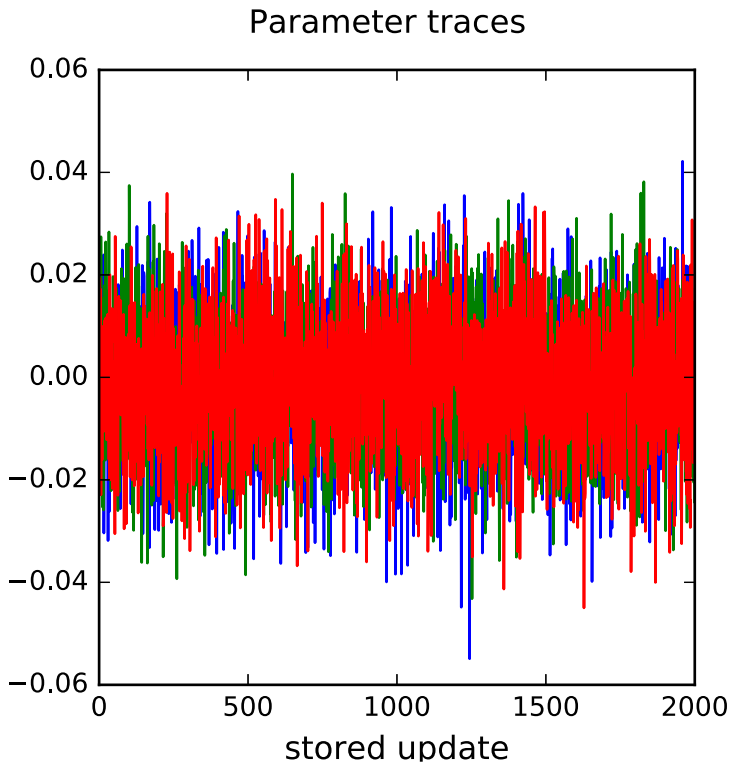
We can plot a predicted regression line to describe the model. This is shown below:

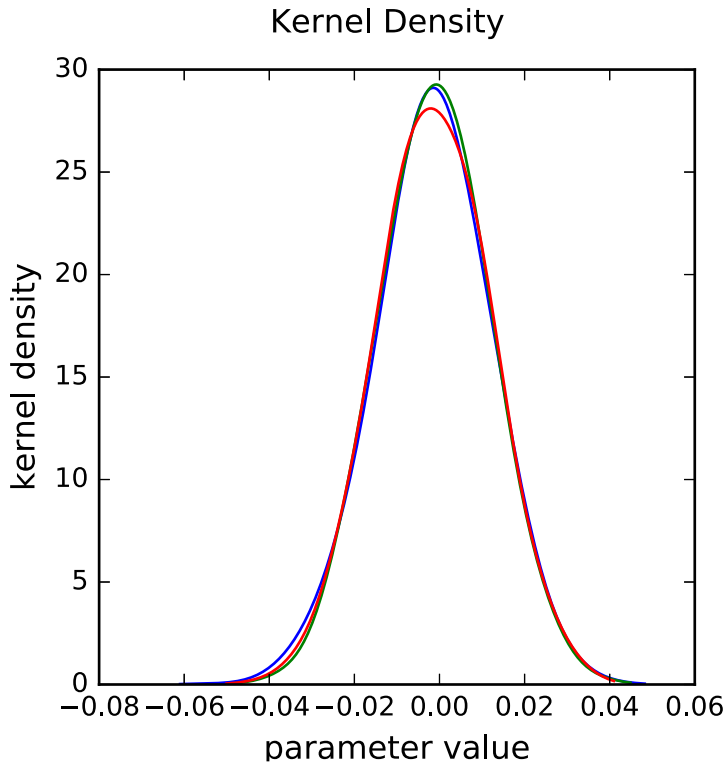


We will next look at the MCMC diagnostics produced for the parameters in our model.

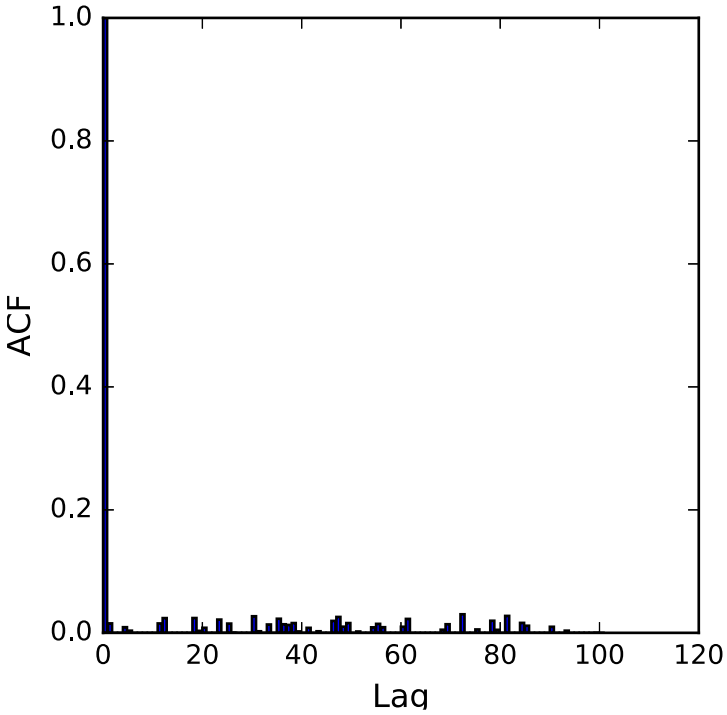
First for the intercept in the model:

MCMC estimation methods are simulation based which means that rather than a point estimate (and accompanying standard error) for each parameter they instead produce a (dependent) chain of values from the posterior distribution of the parameter. In fact in Stat-JR several chains are run from differing starting values/ random number seeds and so for each parameter we have several chains of values that can be combined to summarise the parameter. For parameter beta_1 we can first look at the posterior mean which has value -0.00129 and standard deviation of the chain which has value 0.0127 and plays the role of standard error for the parameter. We might also consider the posterior median which has value -0.00131 as an alternative if the distribution is not symmetric. Here the median is close to the mean as the posterior is reasonably symmetric. We can use the quantiles of the distribution and so we see a 95% credible interval for beta_1 is -0.0261 to 0.0235. We can look at the 3 chains for the parameter beta_1and we can also look at kernel density plots (which are like smoothed histograms) of the 3 chains on a single plot:



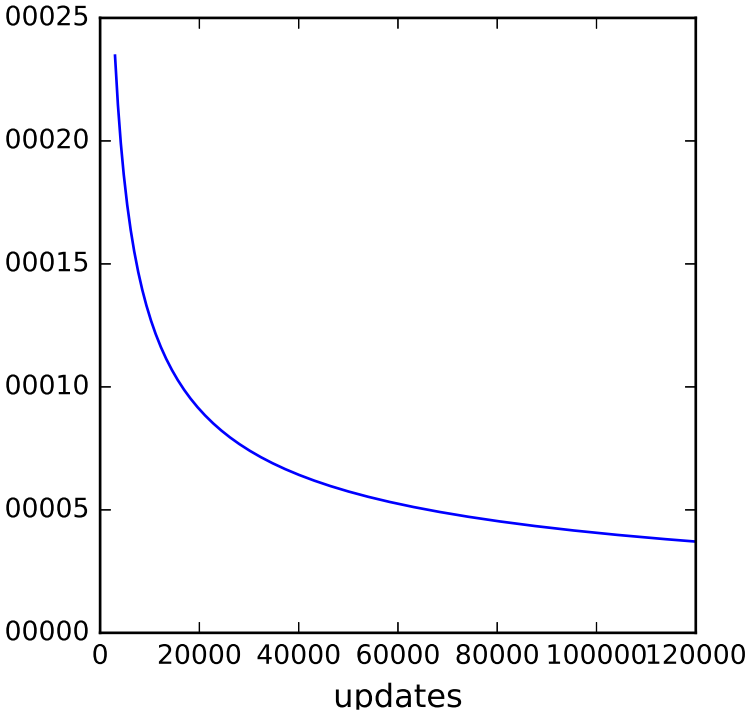


Due to the nature of MCMC algorithms updating parameters in separate steps there is some dependence in the parameter chains produced. One way of investigating this is to look at auto-correlation functions (acf) for the chains. Essentially an acf examines how correlated a chain of values is with a similar chain shifted by a number of iterations (the lag). We can plot such a function for a series of lags as shown below.



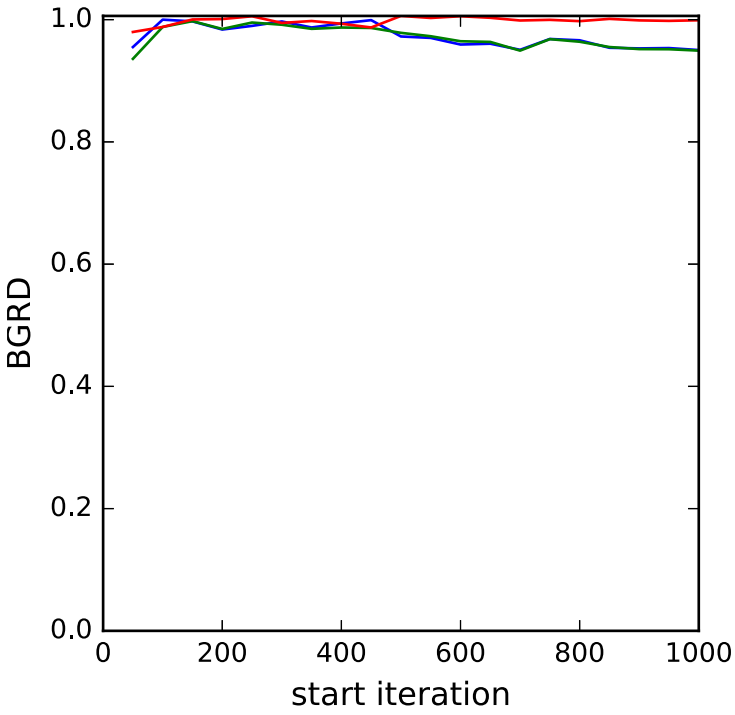
Here the acf value at lag 1 is ρ = 0.0154 and as MCMC algorithms should produce chains resembling an auto-regressive process of order 1 i.e. the value at the current iteration only depends on the last iteration, the value at lag 2 should be approximately ρ^2 (0.000236) and in reality it is -0.0153. Here the chain is only a little correlated.

The correlation in the chain results in the estimates containing possible Monte Carlo standard errors (MCSE) and these explain why running the chain from differing starting values results in estimates that are not exactly the same. The longer the chains are run the less Monte Carlo standard errors and the graph below plots how the MCSE decreases with further iterations. We have run for a total of 6000 and the MCSE is currently 0.000166



In MCMC estimation a question that needs answering is how long to run the chains for before we can rely on the estimates. There are many approaches here so for example we might want to run until the MCSE described above is below a particular value. Alternatively we might want to consider what number of iterations we would run for if we could guarantee they were independent draws from the posterior distribution. With this in mind a diagnostic the effective sample size (which is based on the autocorrelation in the chain) aims to be used in just this way as it estimates the equivalent number of independent iterations that the current (dependent) chains represent. The effective sample size for parameter beta_1 is 6129. Another diagnostic, the Brooks-Draper diagnostic aims to quote how many iterations are required to quote the posterior mean to a given number of significant figures with some certainty. We could quote the estimate -0.0013 with 253856 iterations whilst -0.00129 would require 25385548 iterations.

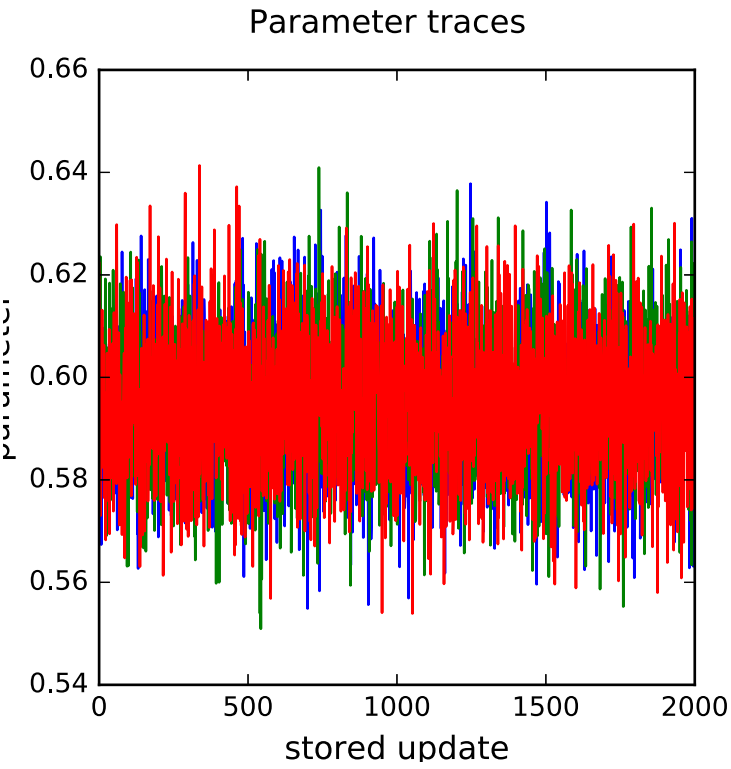
Finally if we have run for multiple chains from differing starting values then we might hope that those chains mix together and produce values from similar areas of the posterior showing that the chains have converged. One way of doing this is via a multiple chains diagnostic, the BGR (Brooks Gelman & Rubin) diagnostic which investigates the variability between the chains compared to that within the chains. If the three chains have converged together then the variability between the chains should be similar to that within the chains and so the hope is that this diagnostic converges to the value 1. We therefore plot this diagnostic against number of iterations.

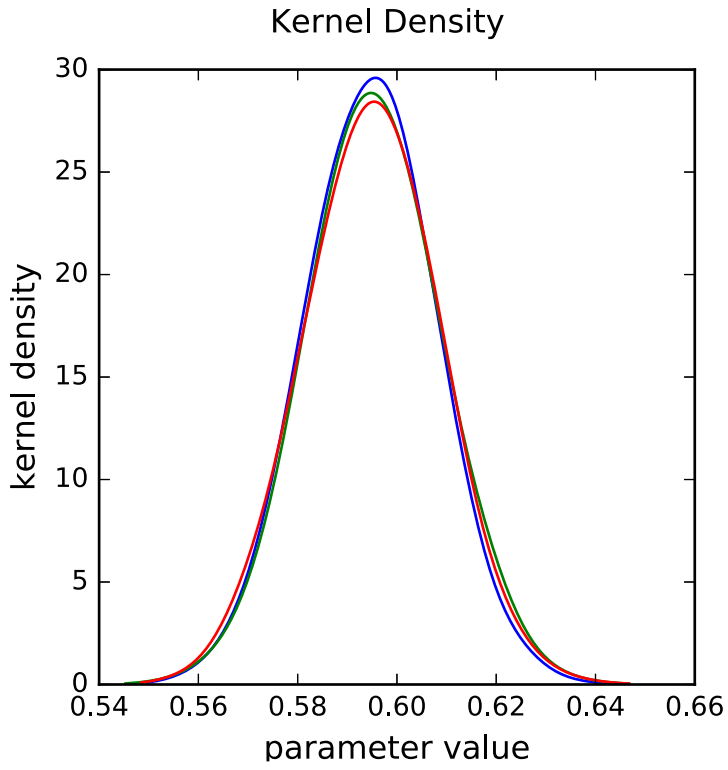


The BGR diagnostic reaches value 1 immediately therefore we are happy with convergence.

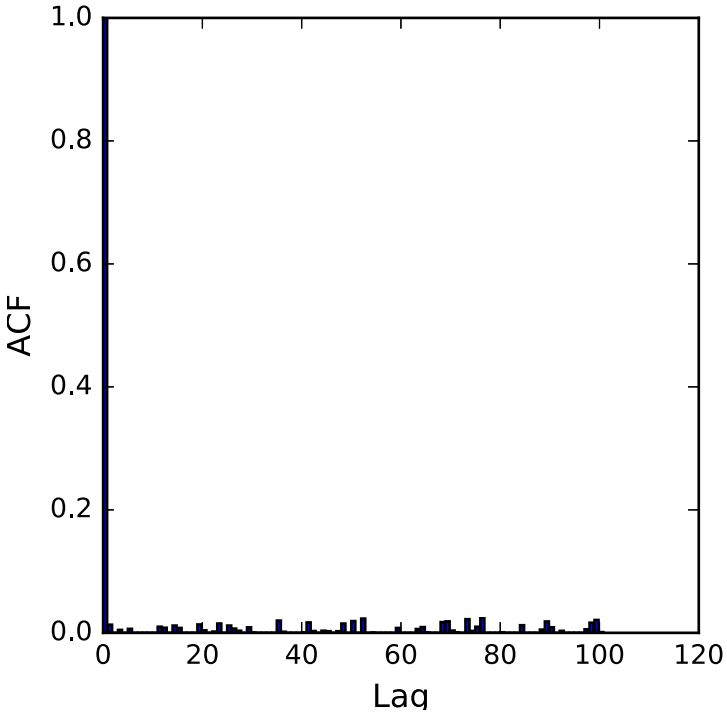
Next for the slope:

MCMC estimation methods are simulation based which means that rather than a point estimate (and accompanying standard error) for each parameter they instead produce a (dependent) chain of values from the posterior distribution of the parameter. In fact in Stat-JR several chains are run from differing starting values/ random number seeds and so for each parameter we have several chains of values that can be combined to summarise the parameter. For parameter beta_0 we can first look at the posterior mean which has value 0.595 and standard deviation of the chain which has value 0.0127 and plays the role of standard error for the parameter. We might also consider the posterior median which has value 0.595 as an alternative if the distribution is not symmetric. Here the median is close to the mean as the posterior is reasonably symmetric. We can use the quantiles of the distribution and so we see a 95% credible interval for beta_0 is 0.57 to 0.62. We can look at the 3 chains for the parameter beta_0and we can also look at kernel density plots (which are like smoothed histograms) of the 3 chains on a single plot:



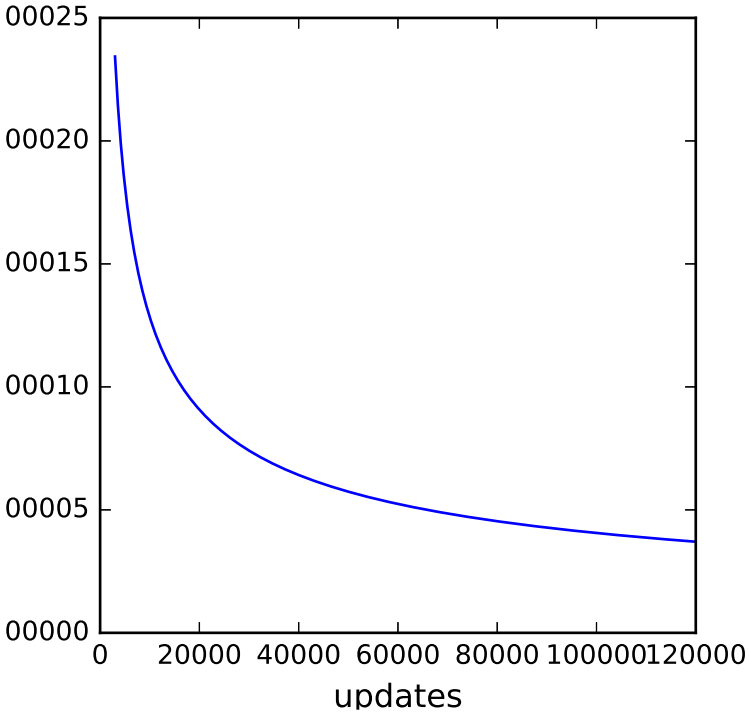


Due to the nature of MCMC algorithms updating parameters in separate steps there is some dependence in the parameter chains produced. One way of investigating this is to look at auto-correlation functions (acf) for the chains. Essentially an acf examines how correlated a chain of values is with a similar chain shifted by a number of iterations (the lag). We can plot such a function for a series of lags as shown below.



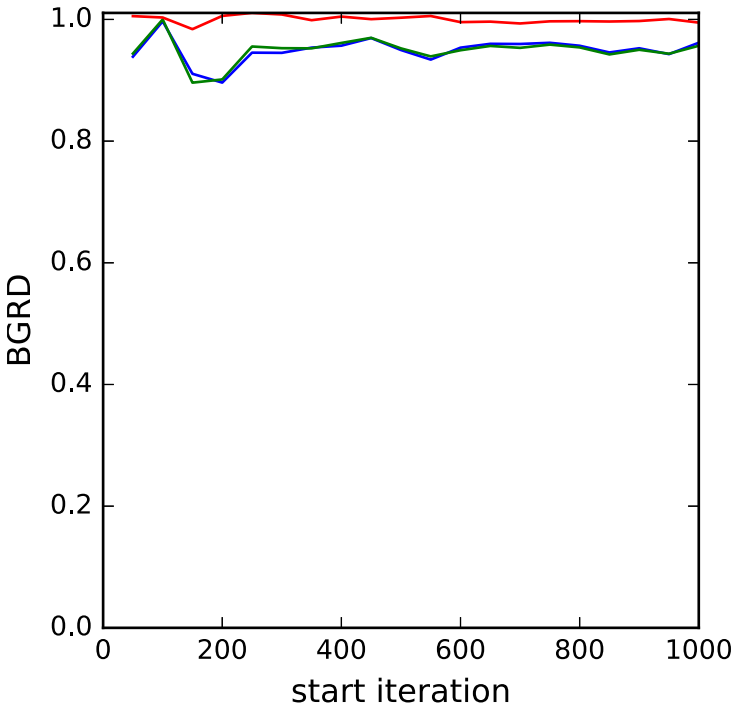
Here the acf value at lag 1 is ρ = 0.0132 and as MCMC algorithms should produce chains resembling an auto-regressive process of order 1 i.e. the value at the current iteration only depends on the last iteration, the value at lag 2 should be approximately ρ^2 (0.000175) and in reality it is -0.00353. Here the chain is only a little correlated.

The correlation in the chain results in the estimates containing possible Monte Carlo standard errors (MCSE) and these explain why running the chain from differing starting values results in estimates that are not exactly the same. The longer the chains are run the less Monte Carlo standard errors and the graph below plots how the MCSE decreases with further iterations. We have run for a total of 6000 and the MCSE is currently 0.000166



In MCMC estimation a question that needs answering is how long to run the chains for before we can rely on the estimates. There are many approaches here so for example we might want to run until the MCSE described above is below a particular value. Alternatively we might want to consider what number of iterations we would run for if we could guarantee they were independent draws from the posterior distribution. With this in mind a diagnostic the effective sample size (which is based on the autocorrelation in the chain) aims to be used in just this way as it estimates the equivalent number of independent iterations that the current (dependent) chains represent. The effective sample size for parameter beta_0 is 5960. Another diagnostic, the Brooks-Draper diagnostic aims to quote how many iterations are required to quote the posterior mean to a given number of significant figures with some certainty. We could quote the estimate 0.59 with 26 iterations whilst 0.595 would require 2531 iterations.

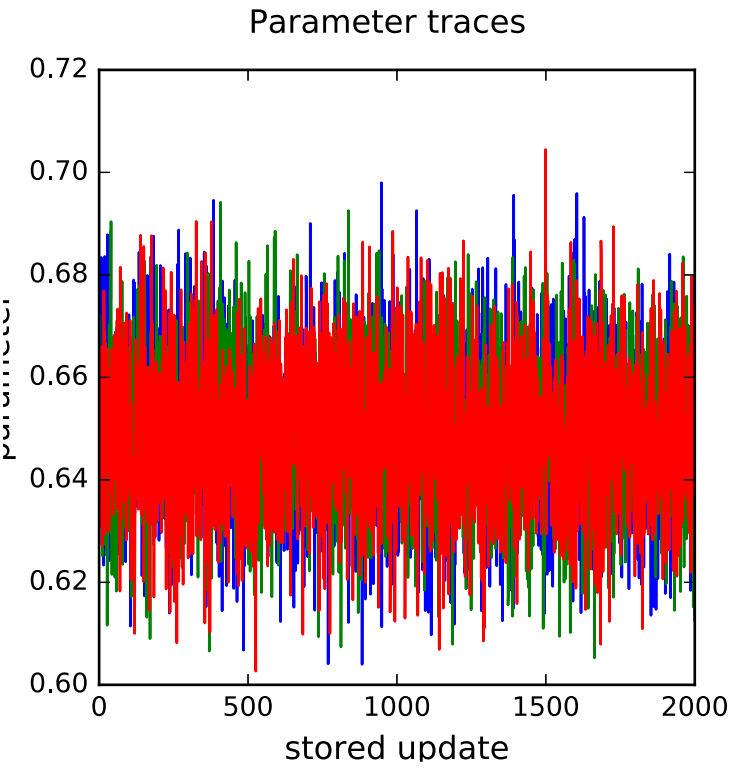
Finally if we have run for multiple chains from differing starting values then we might hope that those chains mix together and produce values from similar areas of the posterior showing that the chains have converged. One way of doing this is via a multiple chains diagnostic, the BGR (Brooks Gelman & Rubin) diagnostic which investigates the variability between the chains compared to that within the chains. If the three chains have converged together then the variability between the chains should be similar to that within the chains and so the hope is that this diagnostic converges to the value 1. We therefore plot this diagnostic against number of iterations.

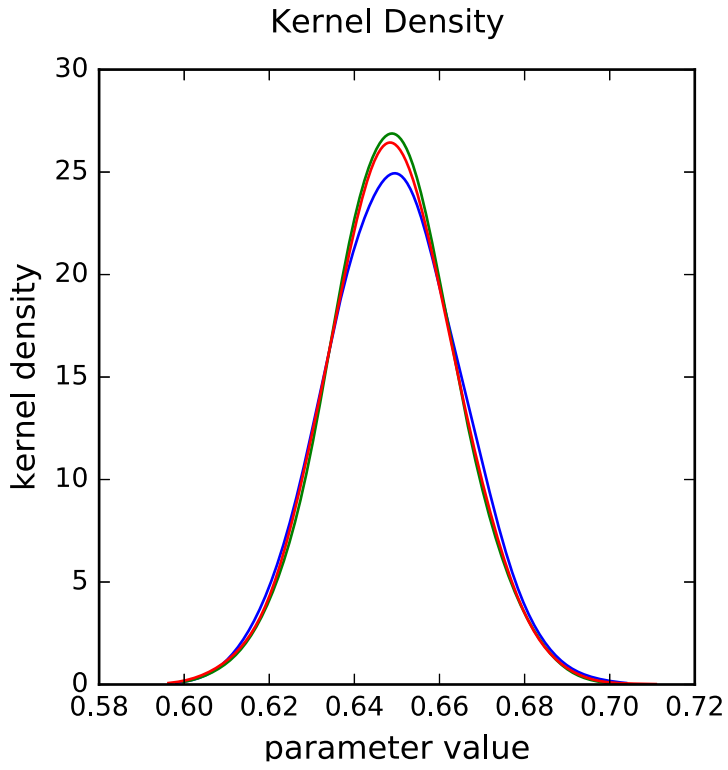


The BGR diagnostic reaches value 1 immediately therefore we are happy with convergence.

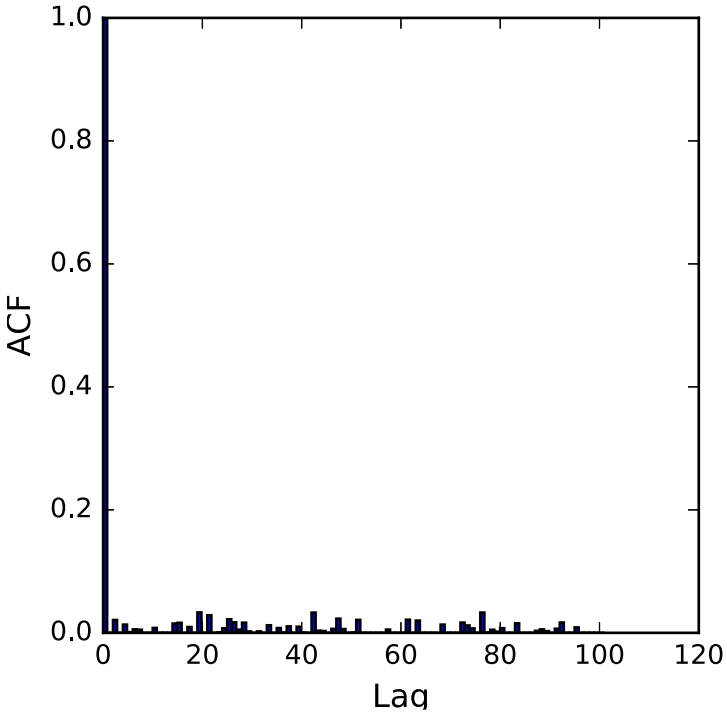
Finally for the residual variance:

MCMC estimation methods are simulation based which means that rather than a point estimate (and accompanying standard error) for each parameter they instead produce a (dependent) chain of values from the posterior distribution of the parameter. In fact in Stat-JR several chains are run from differing starting values/ random number seeds and so for each parameter we have several chains of values that can be combined to summarise the parameter. For parameter sigma2 we can first look at the posterior mean which has value 0.649 and standard deviation of the chain which has value 0.0143 and plays the role of standard error for the parameter. We might also consider the posterior median which has value 0.649 as an alternative if the distribution is not symmetric. Here the median is close to the mean as the posterior is reasonably symmetric. We can use the quantiles of the distribution and so we see a 95% credible interval for sigma2 is 0.621 to 0.677. We can look at the 3 chains for the parameter sigma2and we can also look at kernel density plots (which are like smoothed histograms) of the 3 chains on a single plot:



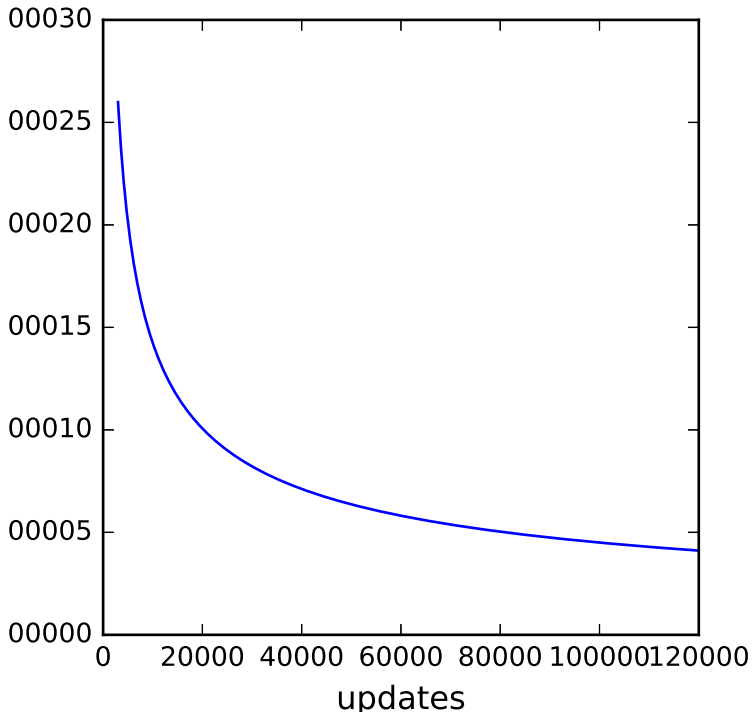


Due to the nature of MCMC algorithms updating parameters in separate steps there is some dependence in the parameter chains produced. One way of investigating this is to look at auto-correlation functions (acf) for the chains. Essentially an acf examines how correlated a chain of values is with a similar chain shifted by a number of iterations (the lag). We can plot such a function for a series of lags as shown below.



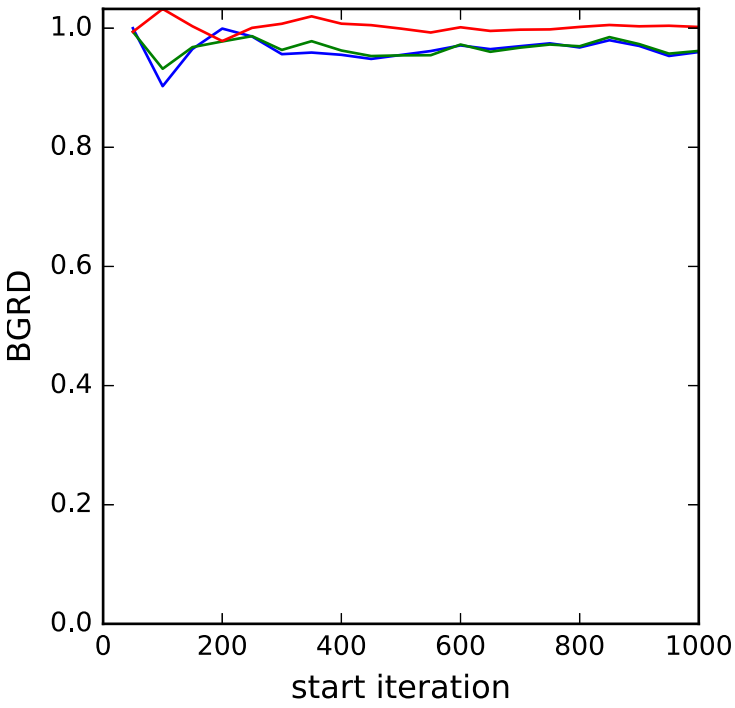
Here the acf value at lag 1 is ρ = -0.0051 and as MCMC algorithms should produce chains resembling an auto-regressive process of order 1 i.e. the value at the current iteration only depends on the last iteration, the value at lag 2 should be approximately ρ^2 (2.6e-05) and in reality it is 0.0208. Here the chain is only a little correlated.

The correlation in the chain results in the estimates containing possible Monte Carlo standard errors (MCSE) and these explain why running the chain from differing starting values results in estimates that are not exactly the same. The longer the chains are run the less Monte Carlo standard errors and the graph below plots how the MCSE decreases with further iterations. We have run for a total of 6000 and the MCSE is currently 0.000184



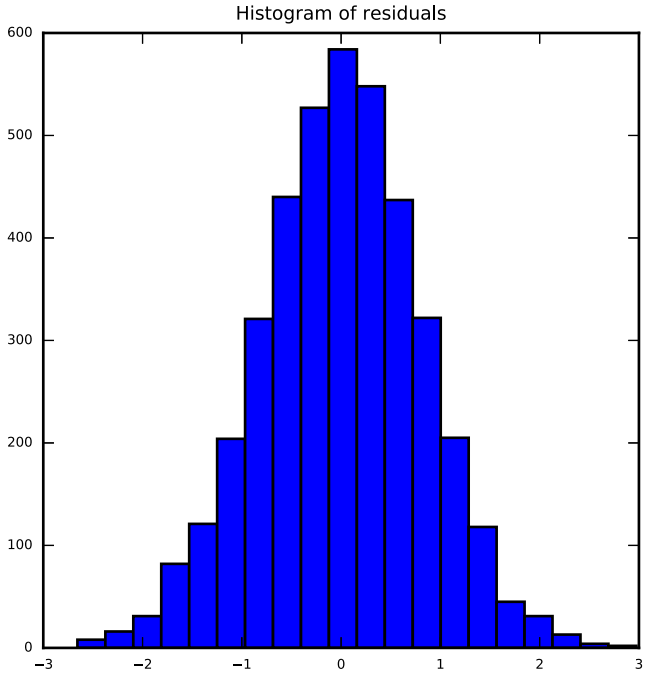
In MCMC estimation a question that needs answering is how long to run the chains for before we can rely on the estimates. There are many approaches here so for example we might want to run until the MCSE described above is below a particular value. Alternatively we might want to consider what number of iterations we would run for if we could guarantee they were independent draws from the posterior distribution. With this in mind a diagnostic the effective sample size (which is based on the autocorrelation in the chain) aims to be used in just this way as it estimates the equivalent number of independent iterations that the current (dependent) chains represent. The effective sample size for parameter sigma2 is 5784. Another diagnostic, the Brooks-Draper diagnostic aims to quote how many iterations are required to quote the posterior mean to a given number of significant figures with some certainty. We could quote the estimate 0.65 with 32 iterations whilst 0.649 would require 3114 iterations.

Finally if we have run for multiple chains from differing starting values then we might hope that those chains mix together and produce values from similar areas of the posterior showing that the chains have converged. One way of doing this is via a multiple chains diagnostic, the BGR (Brooks Gelman & Rubin) diagnostic which investigates the variability between the chains compared to that within the chains. If the three chains have converged together then the variability between the chains should be similar to that within the chains and so the hope is that this diagnostic converges to the value 1. We therefore plot this diagnostic against number of iterations.



The BGR diagnostic reaches value 1 immediately therefore we are happy with convergence.

Here we look at the residuals from the model and plot them in various ways.

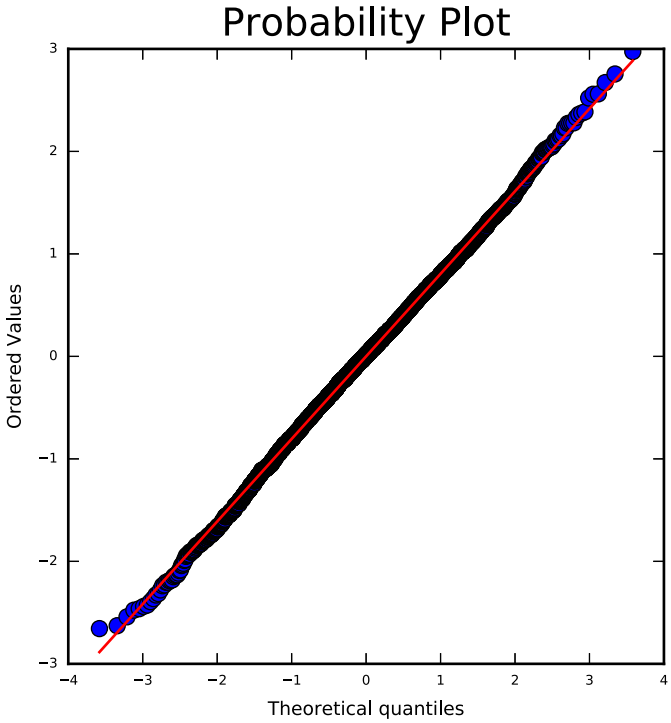


Here the distribution is reasonably symmetric with skewness value -0.039.

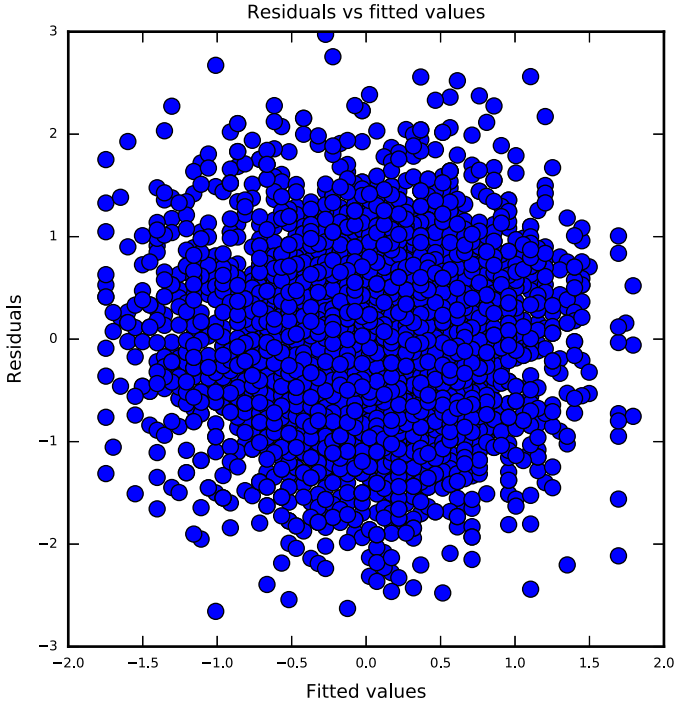
The values:

Row	normexam
82	2.97404400232
88	2.52114023915
428	2.67167398957
530	-2.47663354852
2129	2.56143270149
2310	-2.54142517158
2624	-2.46206364098
2930	-2.4399736105
3326	-2.42650142258
3374	2.55763415614
3420	2.75541987377
3510	-2.65620317431
3645	-2.62794224366

are greater than 3 sds from zero. This might warrant investigating.



If the residuals are fairly normally distributed then the points in this graph should be close to the red line.



Here you should consider whether there are any patterns in this plot. Ideally we would like to see similar variability of the residuals across the range of fitted values.