A Guide to Bayesian Modeling

Radboudumc

Author: Belias Michael

PhD in Biostatistics

Athens, 20 April 2017

Bayesian Analysis

Michael Belias

April 20, 2017

Contents

1 Introduction

According to the Oxford dictionary, statistics is a branch of mathematics dealing with the collection, analysis, interpretation, presentation, and organization of data. Data may be of any applied science field such as medical, finance, social, physics etc and they can be separated into 2 types quantitative and qualitative.

The typical steps of a statistical analysis are seven in general :

1) Define the problem
2) Data collection and manipulation
3) Explore the data
4) Using the above three decide the model that will be used
5) Fit the model
6) Check the model and develop if necessary
7) Make the final model and infere

The above step are not distinct and in some cases there are overlaps and more steps nested. The same principles can be applied in the Bayesian Framework too.

In this tutorial we will learn:

- The bayesian intuition
- Fit the bayesian methods in simple popular statistical approaches such as:
  - (M)ANOVA
  - Linear Regression
  - Poisson regression
- Multi-lelel modeling
  - Hierarchical models
  - Meta analysis

2 Common probability distributions

2.1 Discrete distributions

2.1.1 Uniform

The uniform distribution is the simplest discrete probability distribution. It assigns equal probability to N different outcomes, usually represented with numbers 1,2,. . . .,N .

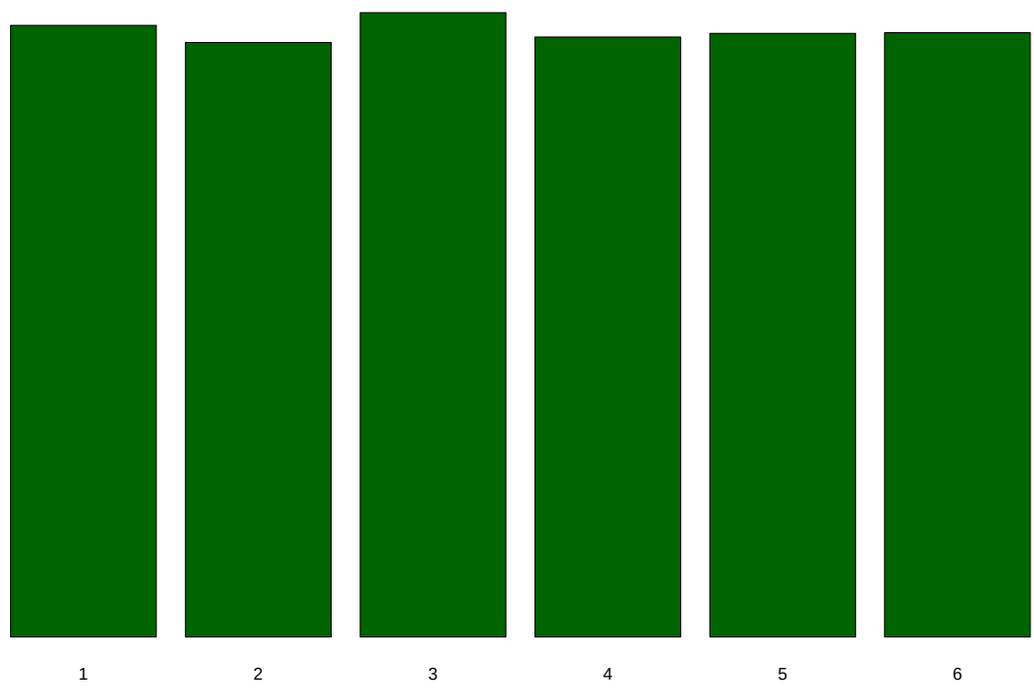
X ~ Uniform(N)

$P(X = x|N) = 1/N$ for x = 1,2,. . . N

$E[X] = \frac{N+1}{2}$

$Var[X] = \frac{N^2+1}{12}$

One common example is the outcome of throwing a fair six-sided die where N=6.



2.1.2 Bernoulli

The Bernoulli distribution is used for binary outcomes, such as 0 and 1. It has one parameter p, which is the probability of "success" frequently geting 1 (or any value we set). X ~ Bern(p)

$P(X = x|p) = p^x(1-p)^{1-x}$ for x = 0, 1

E[X] = p

Var[X] = p(1-p)

One common example is the outcome of flipping a fair coin (p = 0.5)

2.1.3 Binomial

The binomial distribution counts the number of "successes" in n independent Bernoulli trials (each with the same probability of success). Thus if $X_1, X_2, ..., X_n$ are independent Bernoulli(p) random variables, then Y = $\sum_{i=1}^{n} X_i$ is binomial distributed.

Y ~Binom(n, p)

P(Y= y|n,p) = $\binom{n}{y} p^y (1-p)^{(n-y)}$, for y = 0,1, . . . , n

E[Y] =np

Var[Y]= np(1-p)

where $\binom{n}{y} = \frac{n!}{y!(n-y)!}$.

2.1.4 Poisson

The Poisson distribution is used for counts, and arises in a variety of situations. The parameter $\lambda > 0$ is the rate at which we expect to observe the thing we are counting.

X~Pois(λ)

$P(X = x|\lambda) = \frac{\lambda^x exp(-\lambda)}{x!}$

E[X] = λ

Var[X] = λ

A Poisson process is a process wherein events occur on average at rate λ, events occur one at a time, and events occur independently of each other.

Example:

Significant earthquakes occur in the Western United States approximately following a Poisson process with rate of two earthquakes per week. What is the probability there will be at least 3 earthquakes in the next two weeks? Answer: the rate per two weeks is 2*2 = 4, so let X ~Pois(4) and we want to know $P(X \geq 3) = 1 - (X \leq 2) = 1 - P(X = 0) - P(X = 1) - P(X = 2) = 1 - e_4^{-4}e^{-4} - \frac{4^2e^{-4}}{2} = 1 - 13e^{-4} = 0.762$

Note that 0! = 1 by definition.

2.1.5 Geometric

The geometric distribution is the number of failures before obtaining the first success, i.e., the number of Bernoulli failures until a success is observed, such as the first head when flipping a coin. It takes values on the positive integers starting with 0 (alternatively, we could count total trials until first success, in which case we would start with 1).

X~Geo(p)

$P(X = x|p) = p(1-p)^x$, for x=1,2,...

$E[X] = \frac{1-p}{p}$

If the probability of getting a success is p, then the expected number of trials until the first success is 1=p and the expected number of failures until the first success is (1 - p)=p.

2.1.6 Negative Binomial

The negative binomial distribution extends the geometric distribution to model the number of failures before achieving the rth success. It takes values on the positive integers starting with 0.

Y~NegBinom(r,p)

P(Y= y|n,p) = $\binom{r+y-1}{y} p^r (1-p)^{(y)}$ for y=1,2,...

E[Y] = $\frac{r(1-p)}{p}$

Var[Y] = $\frac{r(1-p)}{p^2}$

Note that the geometric distribution is a special case of the negative binomial distribution where r = 1. Because 0 < p < 1, we have E[Y] < Var[Y]. This makes the negative binomial a popular alternative to the Poisson when modeling counts with high variance (recall, that the mean equals the variance for Poisson distributed variables).

2.1.7 Multinomial

Another generalization of the Bernoulli and the binomial is the multinomial distribution, which is like a binomial when there are more than two possible outcomes. Suppose we have n trials and there are k different possible outcomes which occur with probabilities $p_1, p_2, ...p_k$. For example, we are rolling a six-sided die that might be loaded so that the sides are not equally likely, then n is the total number of rolls, k = 6, p1 is the probability of rolling a one, and we denote by $x_1, x_2, ..., x_6$ a possible outcome for the number of times we observe rolls of each of one through six, where $\sum_{i=1}^{6} x_i = n$ and $\sum_{i=1}^{6} p_i = 1$

3 Bayesian Theory

3.1 History

Bayesian statistics are based on the homonymous Bayes' theorem or rule, invented by Thomas Bayes, which was a british reverend the 1740s . His primary field of studying was theology but Bayes was also "amateur" mathematician. He was influenced by David Hume a philosopher teacher while his studies in Edinburgh proposing that we can only rely on what we learn from experience. The probabilities as a mathematical field these days where just emerging being able to solve simple problems like *what is the probability of observing an effect given a cause?* but not the inverse P(cause | effect). Bayes gave a simple example of tossing balls on a table and recording where they stop (to the left or to the right side of the table), noting that the more balls are thrown, the better we may infere if the ball-tosing is bias to a side. This is nowadays called a learning process and although it was a remarkable finding Bayes forgot it in a drawer (!) until his death.Richard Price found it and after studing his papers for 2 years and making some corrections he finally published **An Essay toward solving a Problem in the Doctrine of Chances". 1763**.

Still the theorem was just an example not having the final form of today and even after this publication no-one really continued the development except of Laplace, who was trying to solve an astronomical problem , studied Price's paper developed a first version of what we now call Bayes theorem. The reception of Laplace's proposal was slightly hostile due to the inherent challenges such as the equal prior probabilities, being subjective and the serious technical computational problems in practice, which is still a great issue .

3.2 Bayes theorem

Bayes theorem is calculating the probability event given prior knowledge of conditions that might be related to the event. Bayes' theorem is stated mathematically as the following equation (Efron, 2013) :

$$P(A \mid B) = \frac{P(B|A)\,P(A)}{P(B)}$$

$$P(A \mid B) = \frac{P(B|A)\,P(A)}{P(B)}$$

where A and B are events and P(B) $\neq$ 0.

- P(A) and P(B) are the probabilities of observing A and B without regard to each other.
- P(A | B), a conditional probability, is the probability of observing event A given that B is true.
- P(B | A) is the probability of observing event B given that A is true.

This is the basis of Bayesian inference which is a particular approach to statistical inference, with it's own interpretation and When applied, the probabilities involved in Bayes' theorem may have different probability interpretations. With the Bayesian probability interpretation the theorem expresses how a subjective degree of belief should rationally change to account for availability of related evidence. Bayesian inference is fundamental to Bayesian statistics.

4 Monte Carlo estimation

4.1 Simulation and CLT

Before we learn how to simulate from complicated posterior distributions, let's review some of the basics of Monte Carlo estimation. Monte Carlo estimation refers to simulating hypothetical draws from a probability distribution in order to calculate important quantities. These quantities might include the mean, the variance, the probability of some event, or quantiles of the distribution. All of these calculations involve integration, which except for the simplest distributions, can be very difficult or impossible.

Suppose we have a random variable $\hat{\theta}$ that follows a Gamma(a,b). Let's say a=2 and b=1/3 , where b is the rate parameter. To calculate the mean of this distribution, we would need to compute the following integral

$$\mathrm{E}(\theta) = \int_0^\infty \theta f(\theta) d\theta = \int_0^\infty \theta \frac{b^a}{\Gamma(a)} \theta^{a-1} e^{-b\theta} d\theta$$

It is possible to compute this integral, and the answer is a/b (6 in this case). However, we could verify this answer through Monte Carlo estimation. To do so, we would simulate a large number of draws (call them θ_i^* for i = 1,2,...,m) from this gamma distribution and calculate their sample mean. Why can we do this? Recall from the previous course that if we have a random sample from a distribution, the average of those samples converges in probability to the true mean of that distribution by the Law of Large Numbers. Furthermore, by the Central Limit Theorem (CLT), this sample mean $\bar{\theta}^* = \frac{1}{m} \sum_{i=1}^m \theta_i^*$ approximately follows a normal distribution with mean E(θ) and variance Var(θ)/m. The theoretical variance of θ is the following integral:

$$\mathrm{Var}(\theta) = \int_0^\infty (\theta - E(\theta))^2 f(\theta) d\theta$$

Just like we did with the mean, we could approximate this variance with the sample variance $\frac{1}{m} \sum_{i=1}^m (\theta_i^* - \bar{\theta}^*)^2$

4.2 Calculating probabilities

This method can be used to calculate many different integrals. Say h(θ) is any function and we want to calculate $\int h(\theta)p(\theta)d\theta$ This integral is precisely what is meant by E(h(θ)), so we can conveniently approximate it by taking the sample mean of h(θ_i^*). That is, we apply the function hh to each simulated sample from the distribution, and take the average of all the results.

One extremely useful example of an hh function is is the indicator $I_A(\theta)$ where A is some logical condition about the value of θ. To demonstrate, suppose $h(\theta) = I_{[\theta<5]}(\theta)$, which will give a 1 if $\theta < 5$ and 0 otherwise.

The E(h(θ)) = $\int_0^\infty I_{[\theta<5]}(\theta)p(\theta)d\theta = \int_0^5 1 \cdot p(\theta)d\theta + \int_5^\infty 0 \cdot p(\theta)d\theta = P(\theta < 5)$. This means we can approximate the probability that $\theta < 5$ by drawing many samples θ_i^* , and approximating this integral with $\frac{1}{m}\sum_{i=1}^m I_{\theta^*<5}(\theta_i^*)$. This expression is simply counting how many of those samples come out to be less than 55, and dividing by the total number of simulated samples. So simple!

Likewise, we can approximate quantiles of a distribution. If we are looking for the value ζ such that P(θ < z) = 0.9 , we simply arrange the samples θ_i^* in ascending order and find the smallest drawn value that is greater than 90% of the others.

4.3 Monte Carlo error

How good is an approximation by Monte Carlo sampling? Again we can turn to the CLT, which tells us that the variance of our estimate is controlled in part by m . For a better estimate, we want larger m .

For example, if we seek E(θ), then the sample mean $\bar{\theta}^*$ approximately follows a normal distribution with mean E(θ) and variance Var(θ)/m. The variance tells us how far our estimate might be from the true value. One way to approximate Var(θ) is to replace it with the sample variance. The standard deviation of our Monte Carlo estimate is the square root of that, or the sample standard deviation divided by $\sqrt{m}$.

If m is large, it is reasonable to assume that the true value will likely be within about two standard deviations of your Monte Carlo estimate.

4.4 Marginalization

We can also obtain Monte Carlo samples from hierarchical models. As a simple example, let's consider a binomial random variable where $y \mid \phi \sim$ Bin$(10, \phi)$, and further suppose ϕ is random (as if it had a prior) and is distributed beta $\phi\ Beta(2, 2)$. Given any hierarchical model, we can always write the joint distribution of y and ϕ as $p(y, \phi) = p(y \mid \phi)p(\phi)$ using the chain rule of probability. To simulate from this joint distribution, repeat these steps for a large number m :

- Simulate ϕ_i^* rom its Beta(2,2)distribution
- Given the drawn ϕ_i^* ,simulate y_i^* from Bin$(10, \phi_i^*)$

This will produce m independent pairs of $(y^*, \phi^*)_i$ drawn from their joint distribution. One major advantage of Monte Carlo simulation is that marginalizing is easy. Calculating the marginal distribution of y, $p(y) = \int_0^1 p(y, \phi)d\phi$ might be challenging. But if we have draws from the joint distribution, we can just discard the ϕ_i^* raws and use the y_i^*as samples from their marginal distribution. This is also called the prior predictive distribution introduced in the previous course.

In the next segment, we will demonstrate some of these principles. Remember, we do not yet know how to sample from the complicated posterior distributions introduced in the previous lesson. But once we learn that, we will be able to use the principles from this lesson to make approximate inferences from those posterior distributions.

5 Markov chains

Definition If we have a sequence of random variables $X_1, X_2, ..., X_n$ where the indices *1,2,...,n* represent successive points in time, we can use the chain rule of probability to calculate the probability of the entire sequence:

$$p(X_{t+1}|X_t, X_{t-1}, \ldots, X_2, X_1) = p(X_{t+1}|X_t)$$

Markov chains simplify this expression by using the *Markov assumption*. The assumption is that given the entire past history, the probability distribution for the random variable at the next time step only depends on the current variable. Mathematically, the assumption is written like this:

$$p(X_{t+1}|X_t, X_{t-1}, \ldots, X_2, X_1) = p(X_{t+1}|X_t)$$

for all *t=2,...,n*. Under this assumption, we can write the first expression as

$$p(X_1, X_2, \ldots X_n) = p(X_1) \cdot p(X_2|X_1) \cdot p(X_3|X_2) \cdot p(X_4|X_3) \cdot \ldots \cdot p(X_n|X_{n-1}) \,,$$

which is much simpler than the original. It consists of an initial distribution for the first variable, $P(X_1)$, and **n - 1** transition probabilities. We usually make one more assumption: that the transition probabilities do not change with time. Hence, the transition from time tt to time **t+1** depends only on the value of X_t.

5.1 Examples of Markov chains

5.1.1 Discrete Markov chain

Suppose you have a secret number (make it an integer) between 1 and 5. We will call it your initial number at *step 1*. Now for each time step, your secret number will change according to the following rules:

1. Flip a coin.
2.

- If the coin turns up heads, then increase your secret number by one (5 increases to 1).
- If the coin turns up tails, then decrease your secret number by one (1 decreases to 5).

3. Repeat *n** times, and record the evolving history of your secret number.

Before the experiment, we can think of the sequence of secret numbers as a sequence of random variables, each taking on a value in **{1,2,3,4,5}**. Assume that the coin is fair, so that with each flip, the probability of heads and tails are both 0.5.

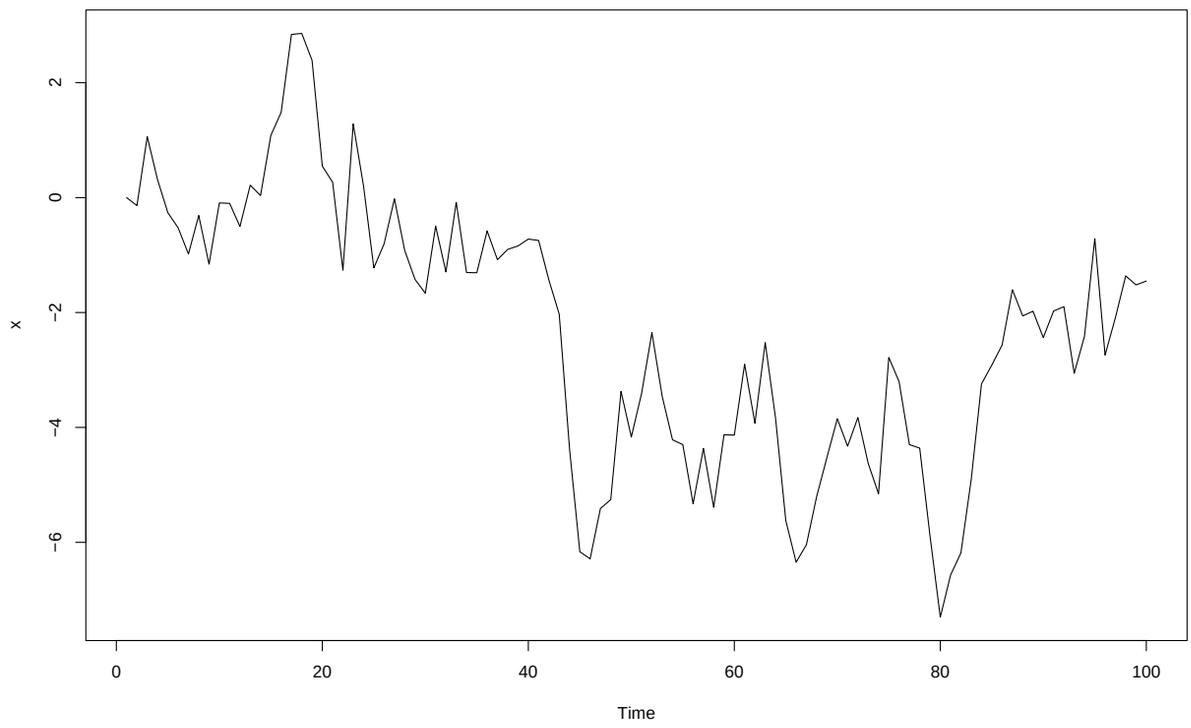
Does this game qualify as a true Markov chain? Suppose your secret number is currently 4 and that the history of your secret numbers is **(2,1,2,3)**. What is the probability that on the next step, your secret number will be 5? What about the other four possibilities? Because of the rules of this game, the probability of the next transition will depend only on the fact that your current number is 4. The numbers further back in your history are irrelevant, so this is a Markov chain.

This is an example of a discrete Markov chain, where the possible values of the random variables come from a discrete set. Those possible values (secret numbers in this example) are called states of the chain. The states are usually numbers, as in this example, but they can represent anything. In one common example, the states describe the weather on a particular day, which could be labeled as 1-fair, 2-poor.

5.1.2 Random walk (continuous)

Now let's look at a continuous example of a Markov chain. Say Xt=0Xt=0 and we have the following transition model: $p(X_{t+1}|X_t = x_t) = \text{N}(x_t, 1)$,. That is, the probability distribution for the next state is Normal with variance 1 and mean equal to the current state. This is often referred to as a "random walk." Clearly, it is a Markov chain because the transition to the next state X_{t+1}only depends on the current state X_t.

R-code



5.1.3 Transition matrix

Let's return to our example of the discrete Markov chain. If we assume that transition probabilities do not change with time, then there are a total of $5^2 = 25$ potential transition probabilities. Potential transition probabilities would be from *State 1* to *State 2*, *State 1* to *State 3*, and so forth. These transition probabilities can be arranged into a matrix Q:

$$Q = \begin{pmatrix} 0 & .5 & 0 & 0 & .5 \\ .5 & 0 & .5 & 0 & 0 \\ 0 & .5 & 0 & .5 & 0 \\ 0 & 0 & .5 & 0 & .5 \\ .5 & 0 & 0 & .5 & 0 \end{pmatrix}$$

where the transitions from *State 1*are in the first row, the transitions from *State 2* are in the second row, etc. For example, the probability $p(X_{t+1} = 5 \mid X_t = 4)$ can be found in the fourth row, fifth column.

The transition matrix is especially useful if we want to find the probabilities associated with multiple steps of the chain. For example, we might want to know $p(X_{t+2} = 3 \mid X_t = 1)$, the probability of your secret number being 3 two steps from now, given that your number is currently 1. We can calculate this as $\sum_{k=1}^{5} p(X_{t+2} = 3 \mid X_{t+1} = k) \cdot p(X_{t+1} = k \mid X_t = 1)$, which conveniently is found in the first row and third column of Q^2.

R-code

```
     [,1] [,2] [,3] [,4] [,5]
[1,] 0.50 0.00 0.25 0.25 0.00
[2,] 0.00 0.50 0.00 0.25 0.25
[3,] 0.25 0.00 0.50 0.00 0.25
[4,] 0.25 0.25 0.00 0.50 0.00
[5,] 0.00 0.25 0.25 0.00 0.50
```

5.1.4 Stationary distribution

Suppose we want to know the probability distribution of the your secret number in the distant future, say $p(X_{t+h}|X_t)$ where *h* is a large number. Let's calculate this for a few different values of *h*.

```
      [,1]  [,2]  [,3]  [,4]  [,5]
[1,] 0.062 0.312 0.156 0.156 0.312
[2,] 0.312 0.062 0.312 0.156 0.156
[3,] 0.156 0.312 0.062 0.312 0.156
[4,] 0.156 0.156 0.312 0.062 0.312
[5,] 0.312 0.156 0.156 0.312 0.062
```

```
      [,1]  [,2]  [,3]  [,4]  [,5]
[1,] 0.248 0.161 0.215 0.215 0.161
[2,] 0.161 0.248 0.161 0.215 0.215
[3,] 0.215 0.161 0.248 0.161 0.215
[4,] 0.215 0.215 0.161 0.248 0.161
[5,] 0.161 0.215 0.215 0.161 0.248
```

```
      [,1]  [,2]  [,3]  [,4]  [,5]
[1,] 0.201 0.199 0.200 0.200 0.199
[2,] 0.199 0.201 0.199 0.200 0.200
[3,] 0.200 0.199 0.201 0.199 0.200
[4,] 0.200 0.200 0.199 0.201 0.199
[5,] 0.199 0.200 0.200 0.199 0.201
```

Notice that as the future horizon gets more distant, the transition distributions appear to converge. The state you are currently in becomes less important in determining the more distant future. If we let hh get really large, and take it to the limit, all the rows of the long-range transition matrix will become equal to *(.2,.2,.2,.2,.2)*. That is, if you run the Markov chain for a very long time, the probability that you will end up in any particular state is *1/5=.2* for each of the five states. These long-range probabilities are equal to what is called the stationary distribution of the Markov chain.

The stationary distribution of a chain is the initial state distribution for which performing a transition will not change the probability of ending up in any given state. That is,

```
     [,1] [,2] [,3] [,4] [,5]
[1,]  0.2  0.2  0.2  0.2  0.2
```

One consequence of this property is that once a chain reaches its stationary distribution, the stationary distribution will remain the distribution of the states thereafter.

We can also demonstrate the stationary distribution by simulating a long chain from

this example.

Now that we have simulated the chain, let's look at the distribution of visits to the five states.

```
x
     1      2      3      4      5
0.1996 0.2020 0.1980 0.1994 0.2010
```

The overall distribution of the visits to the states is approximately equal to the stationary distribution.

As we have just seen, if you simulate a Markov chain for many iterations, the samples can be used as a Monte Carlo sample from the stationary distribution. This is exactly how we are going to use Markov chains for Bayesian inference. In order to simulate from a complicated posterior distribution, we will set up and run a Markov chain whose stationary distribution is the posterior distribution.

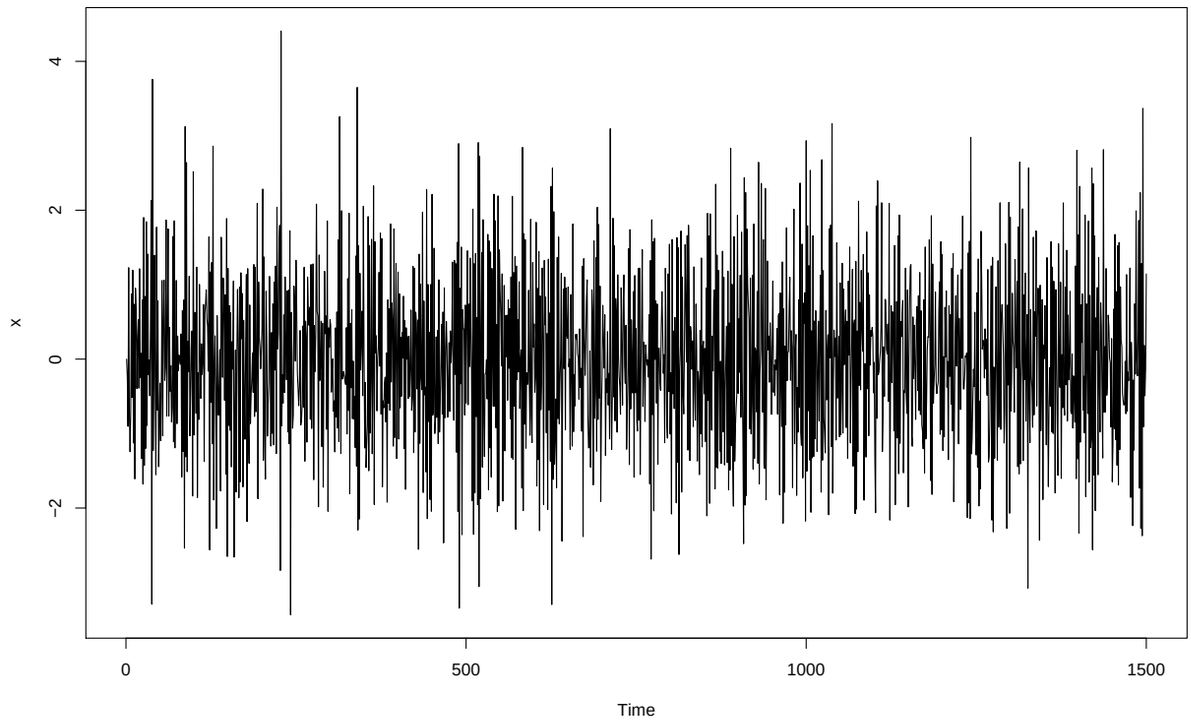
It is important to note that the stationary distribution doesn't always exist for any given Markov chain. The Markov chain must have certain properties, which we won't discuss here. However, the Markov chain algorithms we'll use in future lessons for Monte Carlo estimation are guaranteed to produce stationary distributions.

5.1.5 Continuous example

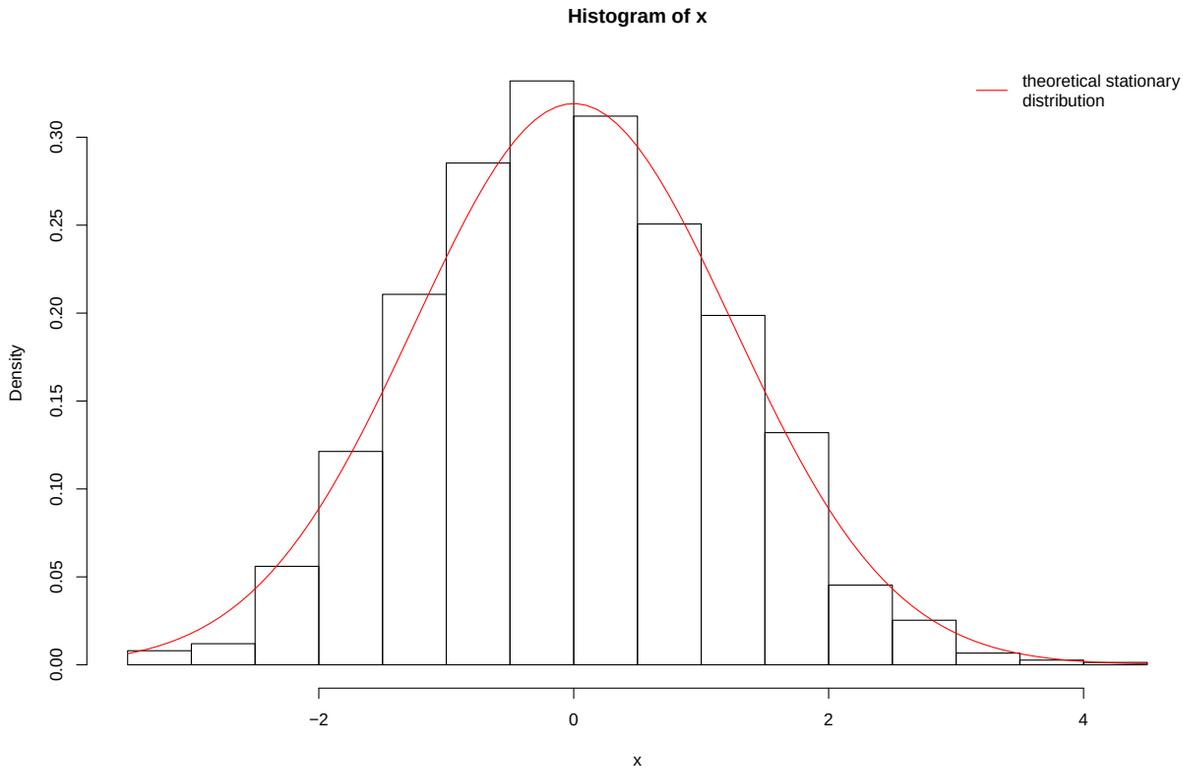
The continuous random walk example we gave earlier does not have a stationary distribution. However, we can modify it so that it does have a stationary distribution.

Let the transition distribution be $p(X_{t+1}|X_t = x_t) = \text{N}(\phi x_t, 1)$ where *-1<ϕ<1* .That is, the probability distribution for the next state is Normal with variance 1 and mean equal to ϕ times the current state. As long as ϕ is between *-1* and* 1*, then the stationary distribution will exist for this model.

Let's simulate this chain for $\phi = -0.6$.



The theoretical stationary distribution for this chain is normal with mean 0 and variance 1/(1-ϕ^2) which in our example approximately equals *1.562*. Let's look at a histogram of our chain and compare that with the theoretical stationary distribution.



It appears that the chain has reached the stationary distribution. Therefore, we could treat this simulation from the chain like a Monte Carlo sample from the stationary distribution, a normal with mean *0* and variance *1.562*.

Because most posterior distributions we will look at are continuous, our Monte Carlo simulations with Markov chains will be similar to this example.

5.2 Monte Carlo Example

6 Metropolis-Hastings

Metropolis-Hastings is an algorithm that allows us to sample from a generic probability distribution (which we will call the target distribution), even if we do not know the normalizing constant. To do this, we construct and sample from a Markov chain whose stationary distribution is the target distribution. It consists of picking an arbitrary starting value, and iteratively accepting or rejecting candidate samples drawn from another distribution, one that is easy to sample.

Let's say we wish to produce samples from a target distribution $p(\theta) \propto g(\theta)$ where we don't know the normalizing constant (since $\int g(\theta)d\theta$ is hard or impossible to compute), so we only have $g(\theta)$ to work with. The Metropolis-Hastings algorithm proceeds as follows.

1. Select an initial value θ_0 .
2. For i=1,. . . ,m repeat the following steps:

- Draw a candidate sample θ^* from a proposal distribution $q(\theta^* \mid \theta_{i-1})$ (more on this later). *Compute the ratio $\alpha = \frac{g(\theta^*)/q(\theta^*|\theta_{i-1})}{g(\theta_{i-1})/q(\theta_{i-1}|\theta^*)} = \frac{g(\theta^*)q(\theta_{i-1}|\theta^*)}{g(\theta_{i-1})q(\theta^*|\theta_{i-1})}$.

If $\alpha \geq 1$, then set $\theta_i = \theta^*$ If $\alpha < 1$, then set $\theta_i = \theta^*$ with probability α, or $\theta_i = \theta_{i-1}$ with probability 1-α.

Steps 2b and 2c act as a correction since the proposal distribution is not the target distribution. At each step in the chain, we draw a candidate and decide whether to "move" the chain there or remain where we are. If the proposed move to the candidate is "advantageous," ($\alpha \geq 1$) we "move" there and if it is not "advantageous," we still might move there, but only with probability α. Since our decision to "move" to the candidate only depends on where the chain currently is, this is a Markov chain.

6.1 Proposal distribution

One careful choice we must make is the candidate generating distribution$q(\theta^* \mid \theta_{i-1})$ It may or may not depend on the previous iteration's value of θ. One example where it doesn't depend on the previous value would be if $q(\theta)$is always the same distribution. If we use this option, $q(\theta)$ should be as similar as possible to $p(\theta)$.

Another popular option, one that does depend on the previous iteration, is Random-Walk Metropolis-Hastings. Here, the proposal distribution is centered on θ_{i-1}. For instance, it might be a normal distribution with mean θ_{i-1}. Because the normal distribution is symmetric, this example comes with another advantage: $q(\theta^* \mid \theta_{i-1}) = q(\theta_{i-1} \mid \theta^*)$, causing it to cancel out when we calculate α. Thus, in Random-Walk Metropolis-Hastings where the candidate is drawn from a normal with mean θ_{i-1} and constant variance, the acceptance ratio is $\alpha = \frac{g(\theta^*)}{g(\theta_{i-1})}$.

6.2 Acceptance rate

Clearly, not all candidate draws are accepted, so our Markov chain sometimes "stays" where it is, possibly for many iterations. How often you want the chain to accept candidates depends on the type of algorithm you use. If you approximate $p(\theta)$ with $q(\theta^*)$and always draw candidates from that, accepting candidates often is good; it means $q(\theta^*)$is approximating $p(\theta)$ well. However, you still may want qq to have a larger variance than pp and see some rejection of candidates as an assurance that qq is covering the space well.

As we will see in coming examples, a high acceptance rate for the Random-Walk Metropolis-Hastings sampler is not a good thing. If the random walk is taking too small of steps, it will accept often, but will take a very long time to fully explore the posterior. If the random walk is taking too large of steps, many of its proposals will have low probability and the acceptance rate will be low, wasting many draws. Ideally, a random walk sampler should accept somewhere between 23% and 50% of

the candidates proposed.

In the next segment, we will see a demonstration of this algorithm used in a discrete case, where we can show mathematically that the Markov chain converges to the target distribution. In the following segment, we will demonstrate coding a Random-Walk Metropolis-Hastings algorithm in R to solve one of the problems from the end of Lesson 2.

6.3 Random walk with normal likelihood, t prior

Recall the model from the last segment of Lesson 2 where the data are the percent change in total personnel from last year to this year for *n=10* companies. We used a normal likelihood with known variance and *t* distribution for the prior on the unknown mean. Suppose the values are *y=(1.2,1.4,-0.5,0.3,0.9,2.3,1.0,0.1,1.3,1.9)* . Because this model is not conjugate, the posterior distribution is not in a standard form that we can easily sample. To obtain posterior samples, we will set up a Markov chain whose stationary distribution is this posterior distribution.

Recall that the posterior distribution is:

$$p(\mu \mid y_1, \ldots, y_n) \propto \frac{\exp[n(\bar{y}\mu - \mu^2/2)]}{1+\mu^2}$$

The posterior distribution on the left is our target distribution and the expression on the right is our g(μ).

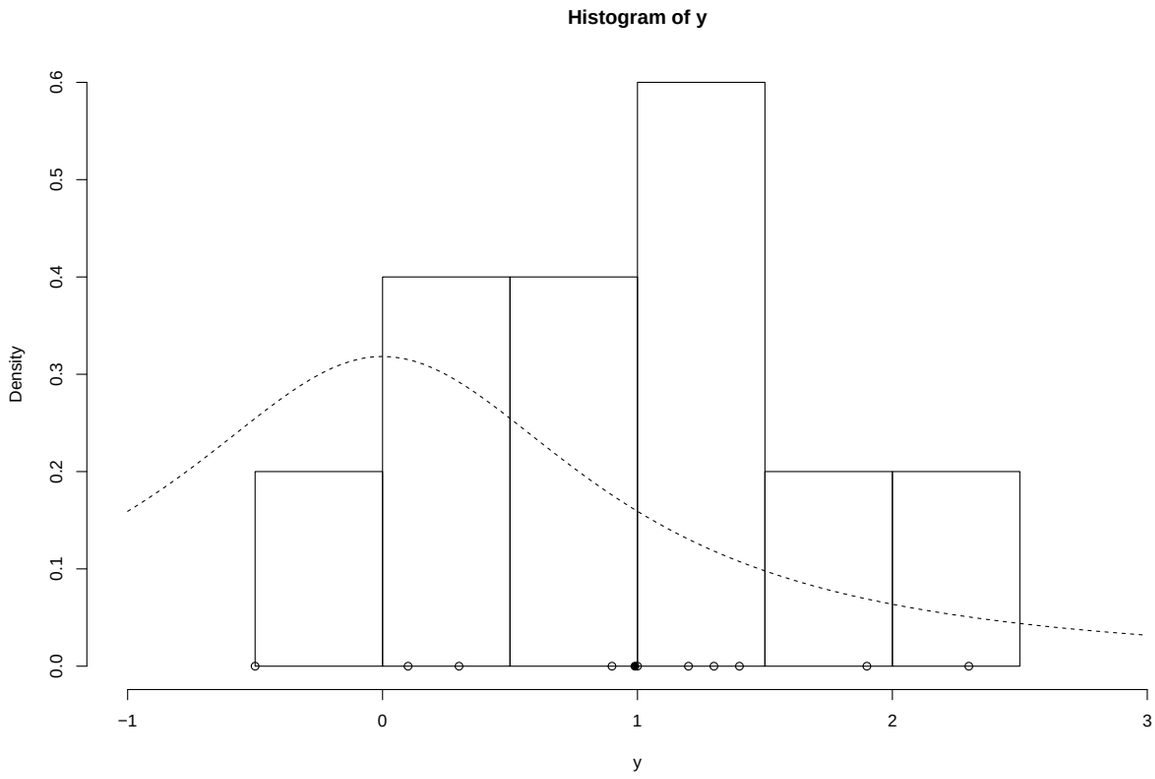
The first thing we can do in R is write a function to evaluate g(μ). Because posterior distributions include likelihoods (the product of many numbers that are potentially small), g(μ) might evaluate to such a small number that to the computer, it effectively zero. This will cause a problem when we evaluate the acceptance ratio α. To avoid this problem, we can work on the log scale, which will be more numerically stable. Thus, we will write a function to evaluate

$$\log(g(\mu)) = n(\bar{y}\mu - \mu^2/2) - \log(1 + \mu^2)$$

This function will require three arguments, μ , $\bar{y}$ and n.

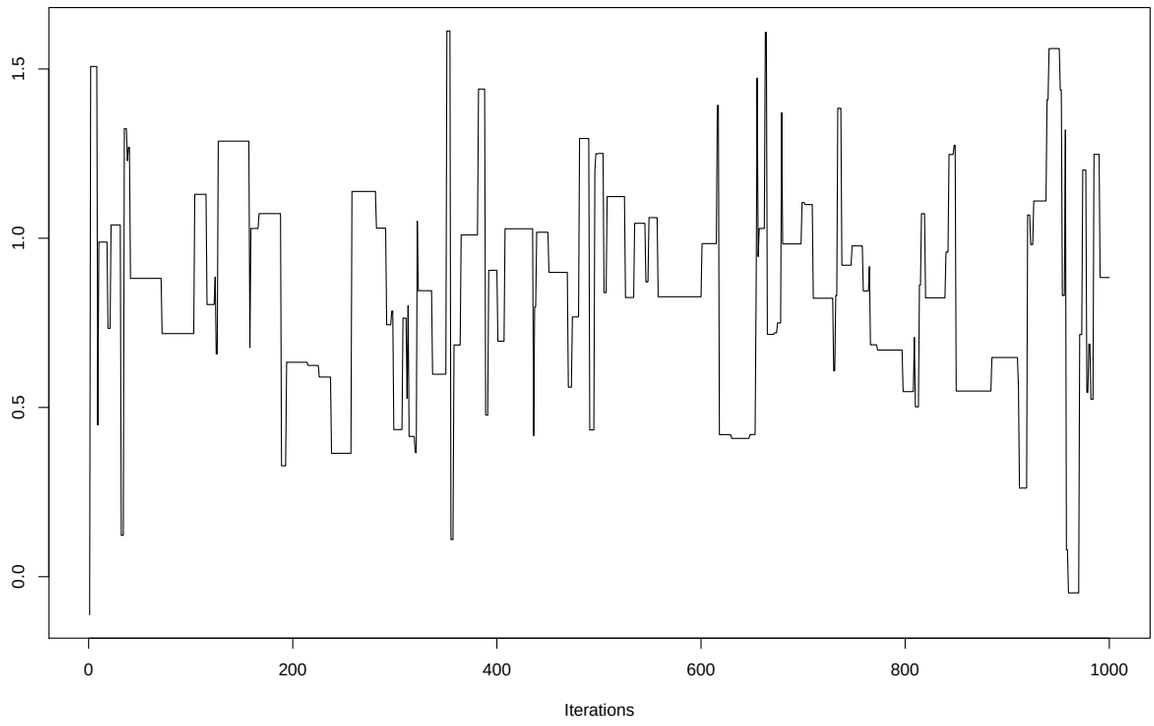
Next, let's write a function to execute the Random-Walk Metropolis-Hastings sampler with normal proposals.

Now, let's set up the problem.



Finally, we're ready to run the sampler! Let's use *m=1000* iterations and proposal standard deviation (which controls the proposal step size) 3.0, and initial value at the prior median 0

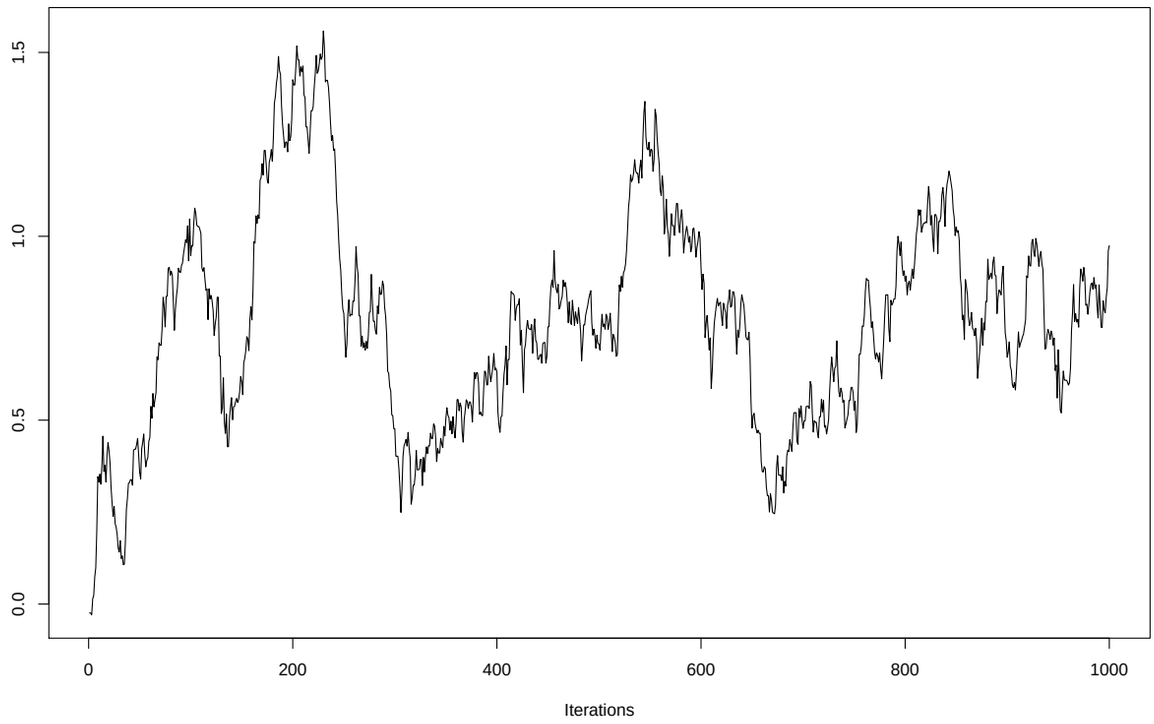
```
List of 2
 $ mu   : num [1:1000] -0.113 1.507 1.507 1.507 1.507 ...
 $ accpt: num 0.122
```



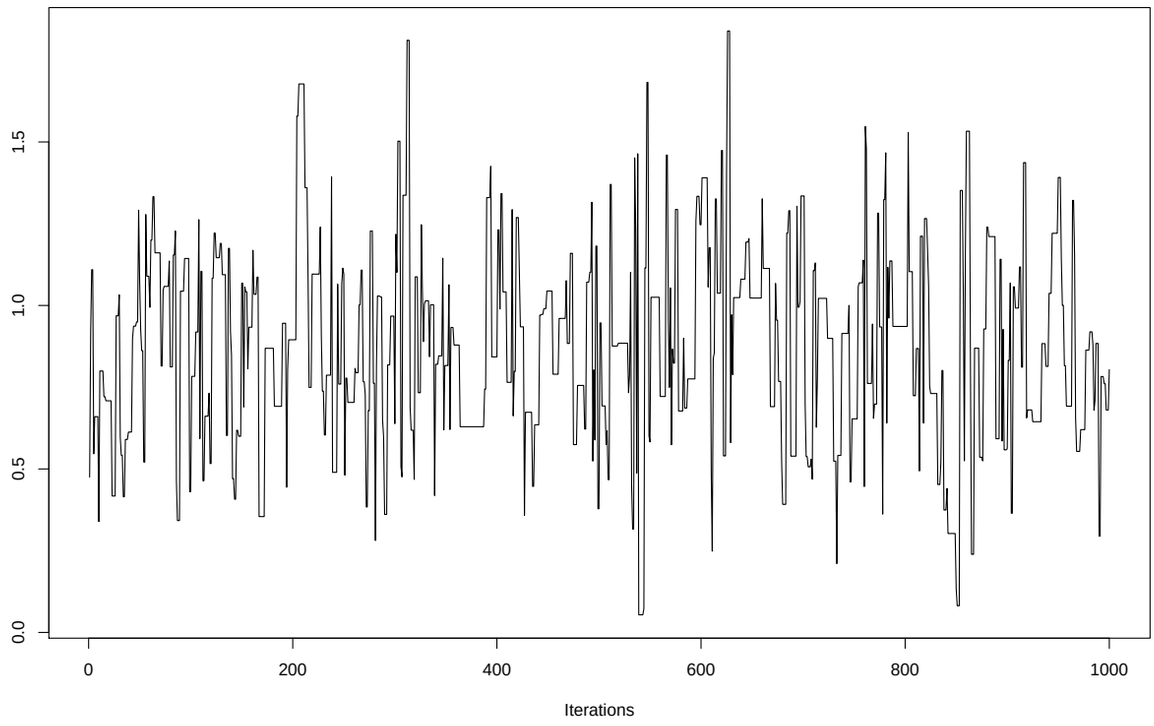
This last plot is called a trace plot. It shows the history of the chain and provides basic feedback about whether the chain has reached its stationary distribution.

It appears our proposal step size was too large (acceptance rate below 23%). Let's try another.

```
[1] 0.946
```

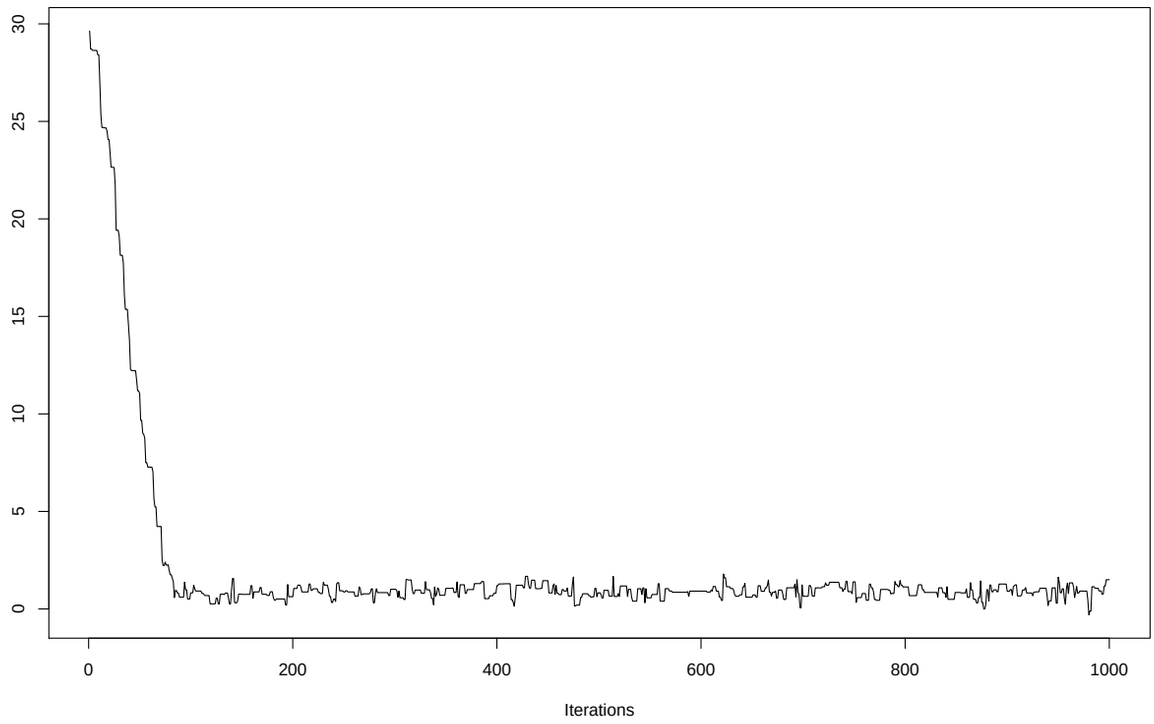



```
[1] 0.38
```

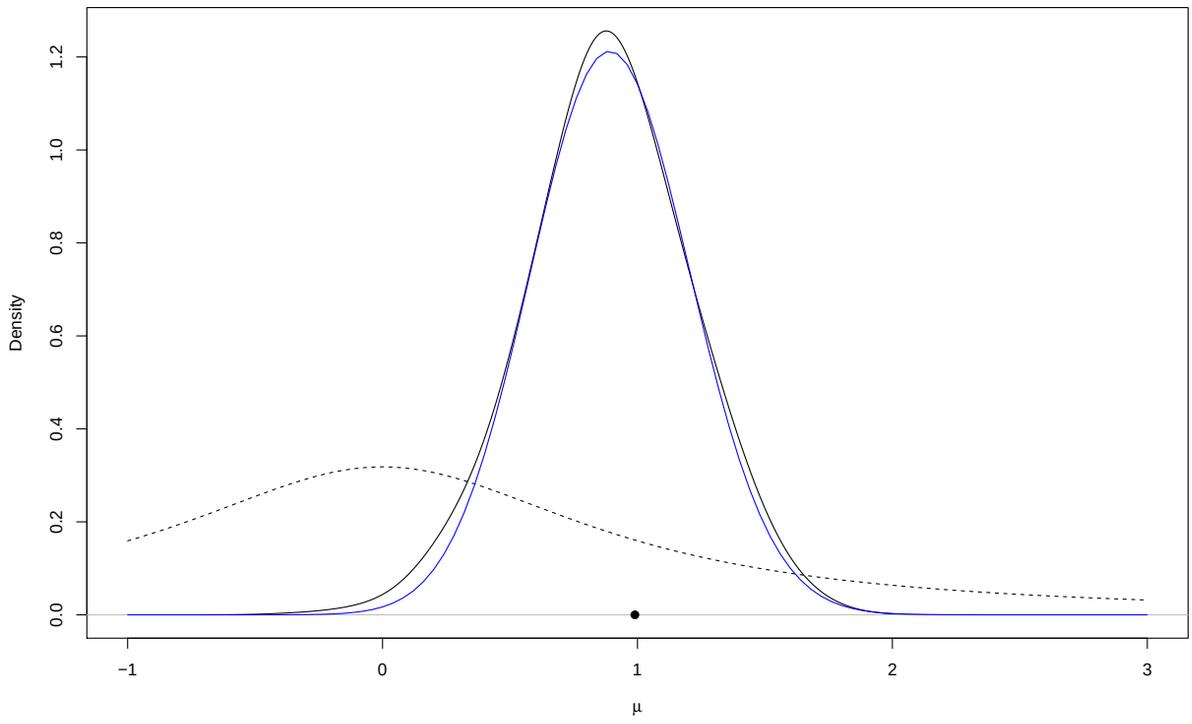


Hey, that looks pretty good. Just for fun, let's see what happens if we initialize the chain at some far-off value.

```
[1] 0.387
```



It took awhile to find the stationary distribution, but it looks like we succeeded! If we discard the first 100 or so values, it appears like the rest of the samples come from the stationary distribution, our posterior distribution! Let's plot the posterior density against the prior to see how the data updated our belief about μ.



These results are encouraging, but they are preliminary. We still need to investigate more formally whether our Markov chain has converged to the stationary distribution. We will explore this in a future lesson.

Obtaining posterior samples using the Metropolis-Hastings algorithm can be time-consuming and require some fine-tuning, as we've just seen. The good news is that we can rely on software to do most of the work for us. In the next couple of videos, we'll introduce a program that will make posterior sampling easy.

7 Gibbs sampling

So far, we have demonstrated MCMC for a single parameter. What if we seek the posterior distribution of multiple parameters, and that posterior distribution does not have a standard form? One option is to perform Metropolis-Hastings (M-H) by sampling candidates for all parameters at once, and accepting or rejecting all of those candidates together. While this is possible, it can get complicated. Another (simpler) option is to sample the parameters one at a time.

As a simple example, suppose we have a joint posterior distribution for two parameters θ and ϕ, written $P(\theta, \phi|y) \propto g(\theta, \phi)$. If we knew the value of ϕ, then we would just draw a candidate for θ and use $g(\theta, \phi)$ to compute our Metropolis-Hastings ratio, and possibly accept the candidate. Before moving on to the next iteration, if we don't know ϕ, then we can perform a similar update for it. Draw a candidate for ϕ using some proposal distribution and again use $g(\theta, \phi)$ to compute our Metropolis-Hastings ratio. Here we pretend we know the value of θ by substituting its current iteration from the Markov chain. Once we've drawn for both θ and ϕ, that completes one iteration and we begin the next iteration by drawing a new θ. In other words, we're just going back and forth, updating the parameters one at a time, plugging the current value of the other parameter into $g(\theta, \phi)$.

This idea of one-at-a-time updates is used in what we call Gibbs sampling, which also produces a stationary Markov chain (whose stationary distribution is the posterior).

7.1 Full conditional distributions

Before describing the full Gibbs sampling algorithm, there's one more thing we can do. Using the chain rule of probability, we have $p(\theta, \phi \mid y) = p(\theta \mid \phi, y) \cdot p(\phi \mid y)$. notice that the only difference between $p(\theta, \phi \mid y)$ and $p(\theta \mid \phi, y)$ is multiplication by a factor that doesn't involve θ . Since the $g(\theta, \phi)$ function above, when viewed as

a function of θ is proportional to both these expressions, we might as well have replaced it with $p(\theta \mid \phi, y)$ in our update for θ.This distribution $p(\theta \mid \phi, y)$ is called the full conditional distribution for θ. Why use it instead of $g(\theta, \phi)$? In some cases, the full conditional distribution is a standard distribution we know how to sample. If that happens, we no longer need to draw a candidate and decide whether to accept it. In fact, if we treat the full conditional distribution as a candidate proposal distribution, the resulting Metropolis-Hastings acceptance probability becomes exactly 1.

Gibbs samplers require a little more work up front because you need to find the full conditional distribution for each parameter. The good news is that all full conditional distributions have the same starting point: the full joint posterior distribution. Using the example above, we have $p(\theta \mid \phi, y) \propto p(\theta, \phi|y)$ where we simply now treat ϕ as a known number. Likewise, the other full conditional is $p(\phi \mid \theta, y) \propto p(\theta, \phi \mid y)$ where here, we consider θ to be a known number. We always start with the full posterior distribution. Thus, the process of finding full conditional distributions is the same as finding the posterior distribution of each parameter, pretending that all other parameters are known.

7.2 Gibbs sampler

The idea of Gibbs sampling is that we can update multiple parameters by sampling just one parameter at a time, cycling through all parameters and repeating. To perform the update for one particular parameter, we substitute in the current values of all other parameters.

Here is the algorithm. Suppose we have a joint posterior distribution for two parameters θ and ϕ, written $P(\theta, \phi \mid y)$. If we can find the distribution of each parameter at a time, i.e., $P(\theta \mid \phi, y)$ and $P(\phi \mid \theta, y)$, then we can take turns sampling these distributions like so:

1.Using ϕ_{i-1}draw θ_i from $P(\theta|\phi = \phi_{i-1}, y)$.

2.Using θ_i, draw ϕ_i from $P(\phi|\theta = \theta_i, y)$.

Together, steps 1 and 2 complete one cycle of the Gibbs sampler and produce the draw for (θ_i, ϕ_i) in one iteration of a MCMC sampler. If there are more than two parameters, we can handle that also. One Gibbs cycle would include an update for each of the parameters.

7.3 Example

7.3.1 Normal likelihood, unknown mean and variance

Let's make an example, where we have normal likelihood with unknown mean and unknown variance. The model is :

$$\begin{aligned} y_i \mid \mu, \sigma^2 &\overset{\text{iid}}{\sim} \text{N}(\mu, \sigma^2), \quad i = 1, \ldots, n \\ \mu &\sim \text{N}(\mu_0, \sigma_0^2) \\ \sigma^2 &\sim \text{IG}(\nu_0, \beta_0) \end{aligned}$$

.

We chose a normal prior for μ because, in the case where σ^2 is known, the normal is the conjugate prior for μ. Likewise, in the case where μ is known, the inverse-gamma is the conjugate prior for σ^2. This will give us convenient full conditional distributions in a Gibbs sampler.

Let's first work out the form of the full posterior distribution. When we begin analyzing data, the JAGS software will complete this step for us. However, it is extremely valuable to see and understand how this works.

$$p(\mu, \sigma^2 \mid y_1, y_2, .$$

$$= \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{(y_i-\mu)^2}{2\sigma^2}\right] \times \frac{1}{\sqrt{2\pi\sigma_0^2}} \exp\left[-\frac{(\mu-\mu_0)^2}{2\sigma_0^2}\right] \times \frac{\beta_0^{\nu_0}}{\Gamma(\nu_0)} (\sigma^2)^{-(\nu_0+1)} \exp\left[-\frac{\beta_0}{\sigma^2}\right] I_{\sigma^2}$$

$$\propto (\sigma^2)^{-n/2} \exp\left[-\frac{\sum_{i=1}^{n}(y_i-\mu)^2}{2\sigma^2}\right] \exp\left[-\frac{(\mu-\mu_0)^2}{2\sigma_0^2}\right] (\sigma^2)^{-(\nu_0+1)} \exp\left[-\frac{\beta_0}{\sigma^2}\right] I_{\sigma^2}$$

From here, it is easy to continue on to find the two full conditional distributions we need. First let's look at μ, assuming σ^2 is known (in which case it becomes a constant and is absorbed into the normalizing constant):

$$\begin{aligned}
p(\mu \mid \sigma^2, y_1, \ldots, y_n) &\propto p(\mu, \sigma^2 \mid y_1, \ldots, y_n) \\
&\propto \exp\left[-\frac{\sum_{i=1}^{n}(y_i-\mu)^2}{2\sigma^2}\right] \exp\left[-\frac{(\mu-\mu_0)^2}{2\sigma_0^2}\right] \\
&\propto \exp\left[-\frac{1}{2}\left(\frac{\sum_{i=1}^{n}(y_i-\mu)^2}{2\sigma^2} + \frac{(\mu-\mu_0)^2}{2\sigma_0^2}\right)\right] \\
&\propto \mathrm{N}\left(\mu \mid \frac{n\bar{y}/\sigma^2 + \mu_0/\sigma_0^2}{n/\sigma^2 + 1/\sigma_0^2}, \frac{1}{n/\sigma^2 + 1/\sigma_0^2}\right),
\end{aligned}$$

which we derived in the supplementary material of the last course. So, given the data and σ^2, μ follows this normal distribution.

Now let's look at σ^2, assuming μ is known:

$$\begin{aligned}
p(\sigma^2 \mid \mu, y_1, \ldots, y_n) &\propto p(\mu, \sigma^2 \mid y_1, \ldots, y_n) \\
&\propto (\sigma^2)^{-n/2} \exp\left[-\frac{\sum_{i=1}^{n}(y_i-\mu)^2}{2\sigma^2}\right] (\sigma^2)^{-(\nu_0+1)} \exp\left[-\frac{\beta_0}{\sigma^2}\right] I_{\sigma^2>0}(\sigma^2) \\
&\propto (\sigma^2)^{-(\nu_0+n/2+1)} \exp\left[-\frac{1}{\sigma^2}\left(\beta_0 + \frac{\sum_{i=1}^{n}(y_i-\mu)^2}{2}\right)\right] I_{\sigma^2>0}(\sigma^2) \\
&\propto \mathrm{IG}\left(\sigma^2 \mid \nu_0 + \frac{n}{2}, \beta_0 + \frac{\sum_{i=1}^{n}(y_i-\mu)^2}{2}\right).
\end{aligned}$$

These two distributions provide the basis of a Gibbs sampler to simulate from a Markov chain whose stationary distribution is the full posterior of both μ and σ^2. We

simply alternate draws between these two parameters, using the most recent draw of one parameter to update the other.

We will do this in R in the next segment.

7.3.1.1 Gibbs sampler in R

To implement the Gibbs sampler we just described, let's return to our running example where the data are the percent change in total personnel from last year to this year for *n=10* companies. We'll still use a normal likelihood, but now we'll relax the assumption that we know the variance of growth between companies, σ^2, and estimate that variance. Instead of the t prior from earlier, we will use the conditionally conjugate priors, normal for μ and inverse-gamma for σ^2

The first step will be to write functions to simulate from the full conditional distributions we derived in the previous segment. The full conditional for μ, given σ^2 and data is

$$\text{N}\left(\mu \mid \frac{n\bar{y}/\sigma^2+\mu_0/\sigma_0^2}{n/\sigma^2+1/\sigma_0^2},\ \frac{1}{n/\sigma^2+1/\sigma_0^2}\right)$$

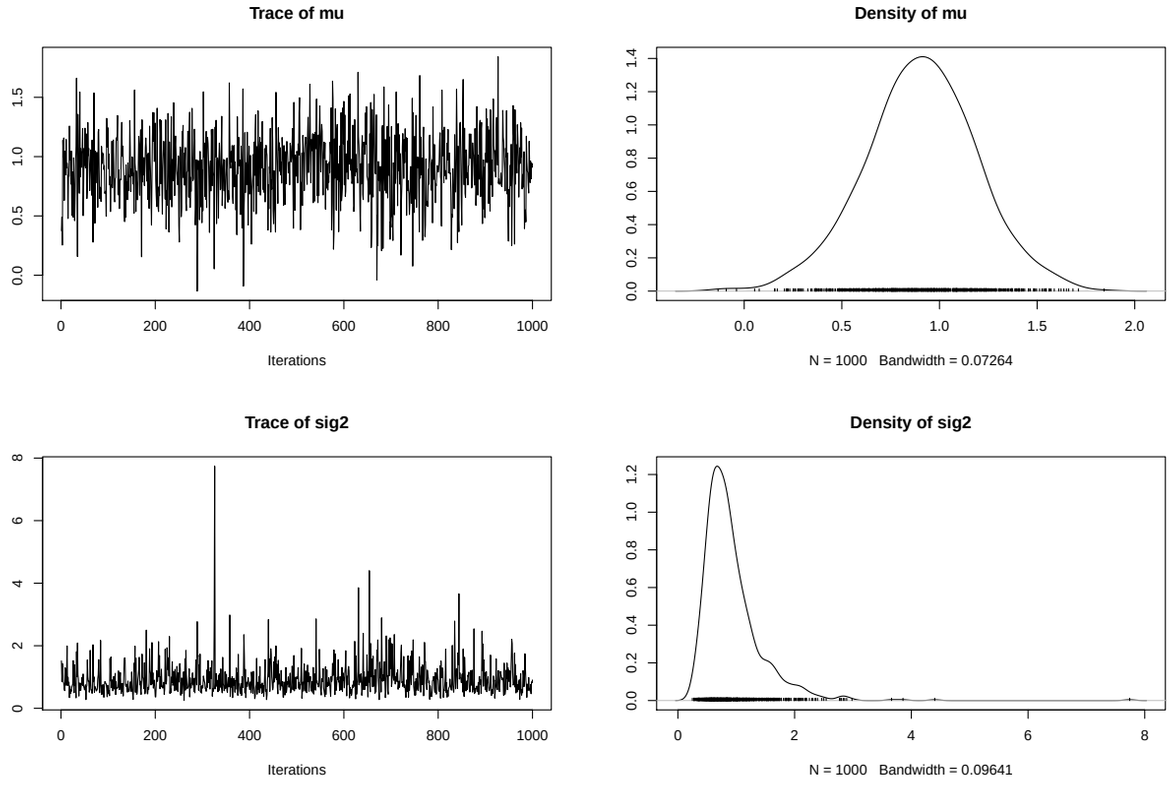
The full conditional for σ^2 given μ and data is:

$$\text{IG}\left(\sigma^2 \mid \nu_0+\frac{n}{2},\ \beta_0+\frac{\sum_{i=1}^{n}(y_i-\mu)^2}{2}\right)$$

With functions for drawing from the full conditionals, we are ready to write a function to perform Gibbs sampling.

Now we are ready to set up the problem in R.


```
            mu      sig2
[1,] 0.3746992 1.5179144
[2,] 0.4900277 0.8532821
[3,] 0.2536817 1.4325174
[4,] 1.1378504 1.2337821
[5,] 1.0016641 0.8409815
[6,] 1.1576873 0.7926196
```




```
Iterations = 1:1000
Thinning interval = 1
Number of chains = 1
Sample size per chain = 1000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

       Mean     SD Naive SE Time-series SE
mu   0.9051 0.2868  0.00907        0.00907
sig2 0.9282 0.5177  0.01637        0.01810

2. Quantiles for each variable:
```

```
        2.5%     25%     50%   75% 97.5%
mu    0.3024 0.7244 0.9089 1.090 1.481
sig2 0.3577 0.6084 0.8188 1.094 2.141
```

As with the Metropolis-Hastings example, these chains appear to have converged. In the next lesson, we will discuss convergence in more detail.

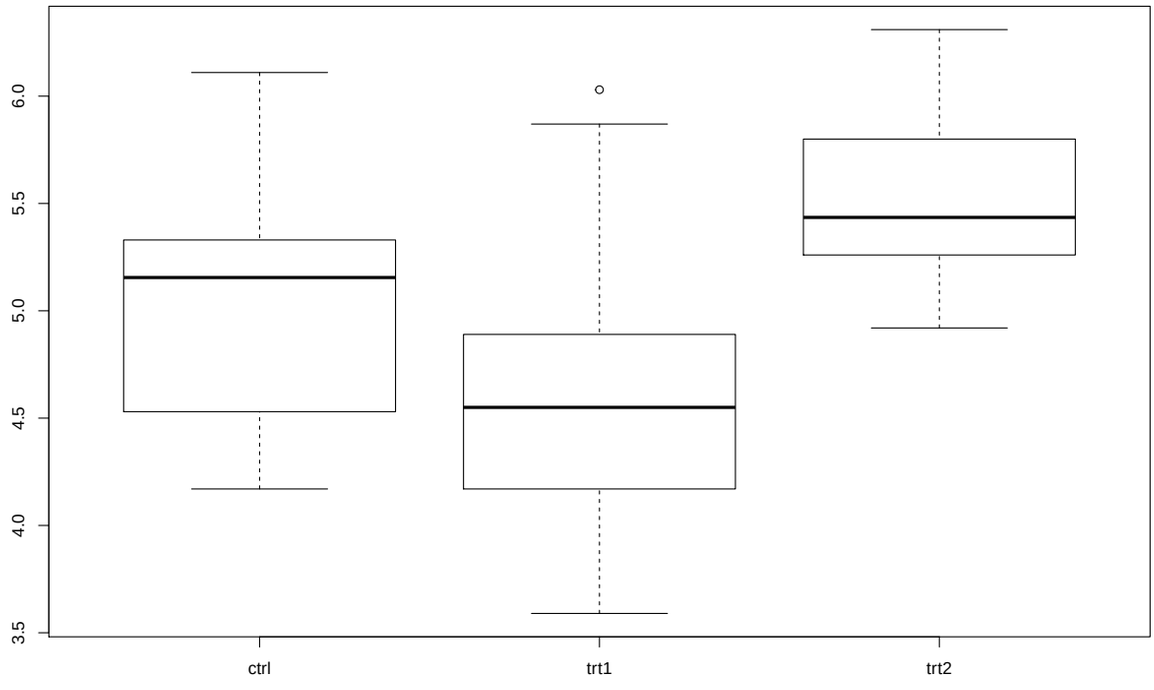
8 Popular one level models

8.1 ANOVA

As an example of a one-way ANOVA, we'll look at the Plant Growth data in R.

```
  weight group
1   4.17  ctrl
2   5.58  ctrl
3   5.18  ctrl
4   6.11  ctrl
5   4.50  ctrl
6   4.61  ctrl
```

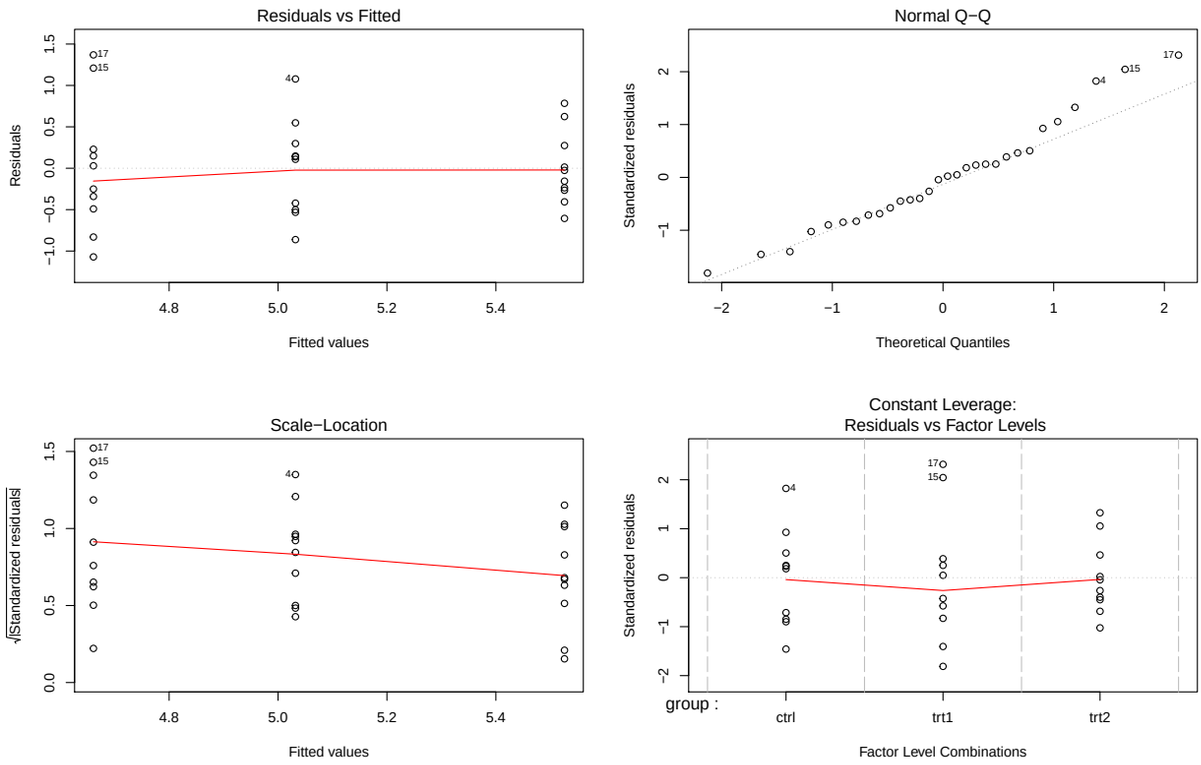
Because the explanatory variable group is a factor and not continuous, we choose to visualize the data with box plots rather than scatter plots.



The box plots summarize the distribution of the data for each of the three groups. It appears that treatment 2 has the highest mean yield. It might be questionable whether each group has the same variance, but we'll assume that is the case.

Modeling Again, we can start with the reference analysis (with a noninformative prior) with a linear model in R.

```
Call:
lm(formula = weight ~ group, data = PlantGrowth)

Residuals:
    Min      1Q  Median      3Q     Max
-1.0710 -0.4180 -0.0060  0.2627  1.3690

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   5.0320     0.1971  25.527   <2e-16 ***
grouptrt1    -0.3710     0.2788  -1.331   0.1944
grouptrt2     0.4940     0.2788   1.772   0.0877 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.6234 on 27 degrees of freedom
Multiple R-squared:  0.2641,    Adjusted R-squared:  0.2096
F-statistic: 4.846 on 2 and 27 DF,  p-value: 0.01591
```




```
Analysis of Variance Table

Response: weight
          Df  Sum Sq Mean Sq F value  Pr(>F)
group      2  3.7663  1.8832  4.8461 0.01591 *
Residuals 27 10.4921  0.3886
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

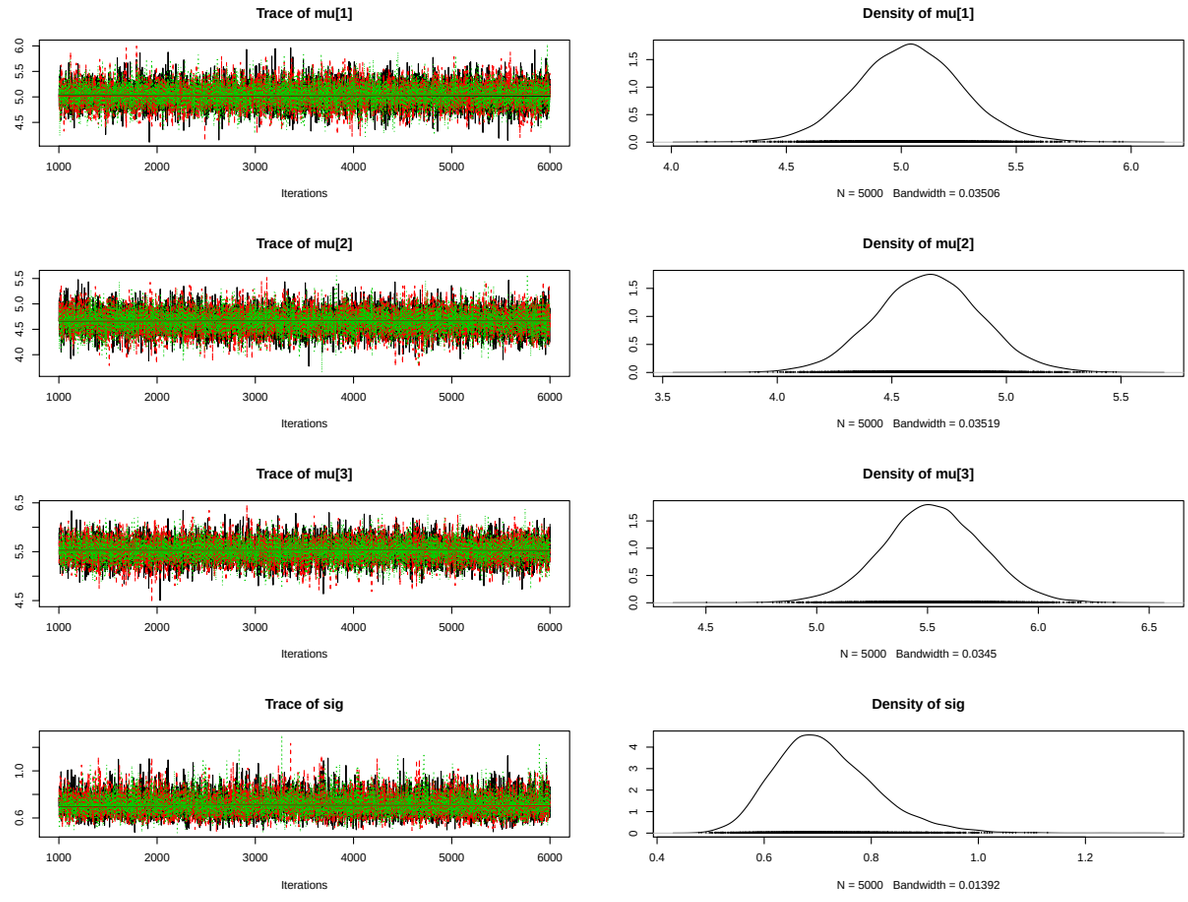
The default model structure in R is the linear model with dummy indicator variables. Hence, the "intercept" in this model is the mean yield for the control group. The two other parameters are the estimated effects of treatments 1 and 2. To recover the mean yield in treatment group 1, you would add the intercept term and the treatment 1 effect. To see how R sets the model up, use the model.matrix(lmod) function to extract the X matrix.

The anova() function in R compares variability of observations between the treatment groups to variability within the treatment groups to test whether all means are equal or whether at least one is different. The small p-value here suggests that the means are not all equal.

Let's fit the cell means model in JAGS.

```
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 30
   Unobserved stochastic nodes: 4
   Total graph size: 85

Initializing model
```

Model checking As usual, we check for convergence of our MCMC.



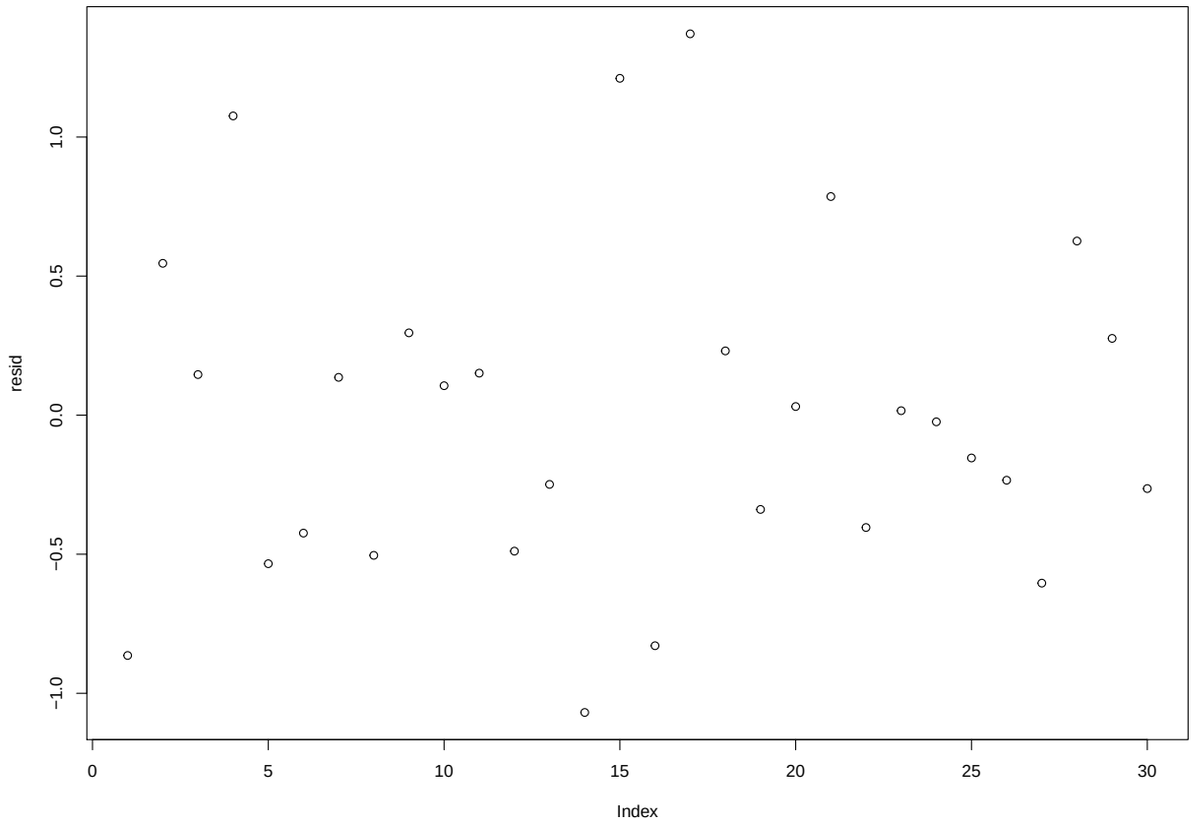

```
Potential scale reduction factors:

      Point est. Upper C.I.
mu[1]          1          1
mu[2]          1          1
mu[3]          1          1
sig            1          1

Multivariate psrf

1
```

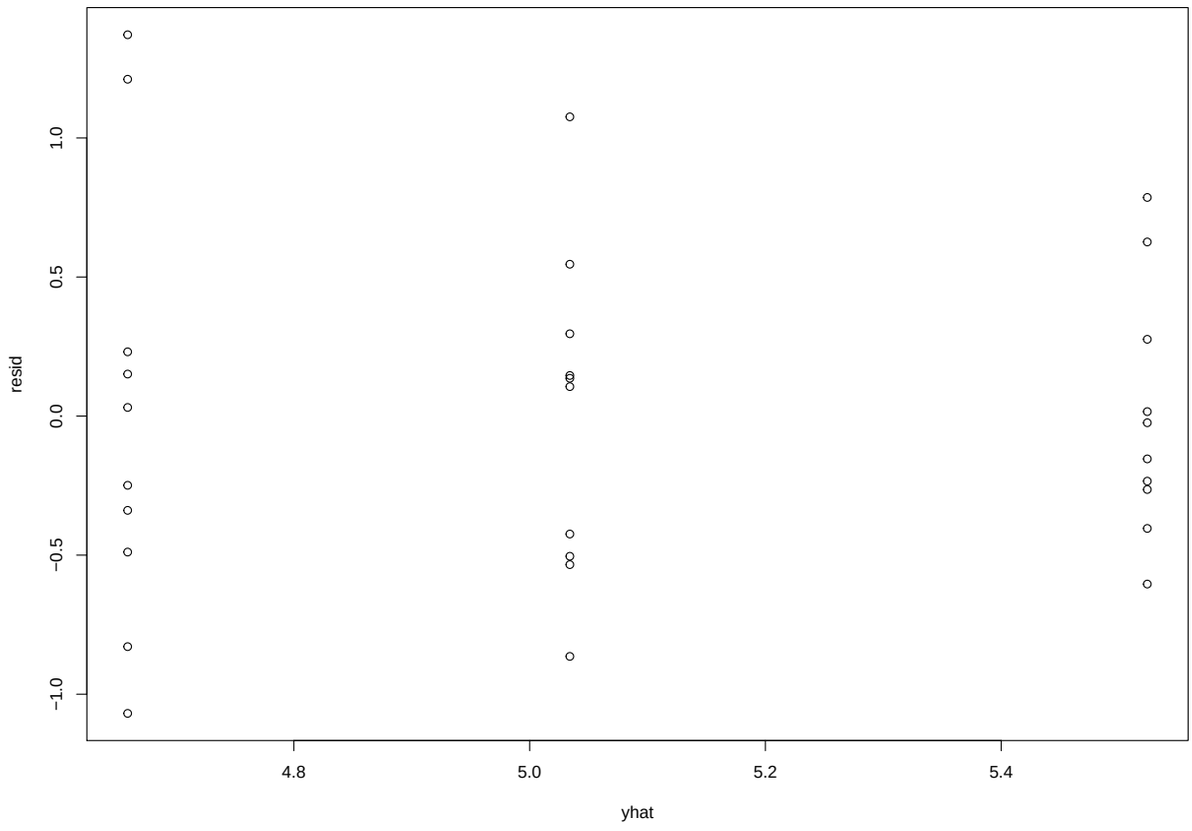
```
            mu[1]       mu[2]       mu[3]        sig
```

```
Lag 0   1.000000000  1.000000000  1.000000000 1.000000000
Lag 1   0.001280097  0.009162200 -0.003860081 0.102625112
Lag 5  -0.008247193  0.006167418 -0.005483792 0.006026036
Lag 10  0.009692229 -0.008475887 -0.001071425 0.017419677
Lag 50 -0.006672748  0.001876296  0.007534024 0.008478319
```

```
   mu[1]    mu[2]    mu[3]      sig
14377.21 15000.00 15000.00 12404.40
```

```
    mu[1]     mu[2]     mu[3]       sig
5.0340823 4.6589550 5.5239668 0.7134754
```





Again, it might be appropriate to have a separate variance for each group. We will have you do that as an exercise.

Results

Let's look at the posterior summary of the parameters.

```
Iterations = 1001:6000
Thinning interval = 1
Number of chains = 3
Sample size per chain = 5000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:
```

```
        Mean      SD  Naive SE Time-series SE
mu[1] 5.0341 0.23033 0.0018807      0.0019214
mu[2] 4.6590 0.22959 0.0018746      0.0018747
mu[3] 5.5240 0.22574 0.0018431      0.0018433
sig   0.7135 0.09274 0.0007572      0.0008327

2. Quantiles for each variable:

        2.5%    25%    50%   75%  97.5%
mu[1] 4.5843 4.8814 5.0334 5.185 5.4967
mu[2] 4.2096 4.5065 4.6595 4.811 5.1131
mu[3] 5.0757 5.3766 5.5215 5.675 5.9696
sig   0.5605 0.6486 0.7037 0.769 0.9234

          lower     upper
mu[1] 4.5708556 5.4784773
mu[2] 4.2185234 5.1201296
mu[3] 5.0672079 5.9583427
sig   0.5504343 0.9051048
attr(,"Probability")
[1] 0.95
```

The HPDinterval() function in the coda package calculates intervals of highest posterior density for each parameter.

We are interested to know if one of the treatments increases mean yield. It is clear that treatment 1 does not. What about treatment 2?

```
[1] 0.9372
```

There is a high posterior probability that the mean yield for treatment 2 is greater than the mean yield for the control group.

It may be the case that treatment 2 would be costly to put into production. Suppose that to be worthwhile, this treatment must increase mean yield by 10%. What is the posterior probability that the increase is at least that?

```
[1] 0.485
```

We have about 50/50 odds that adopting treatment 2 would increase mean yield by at least 10%.

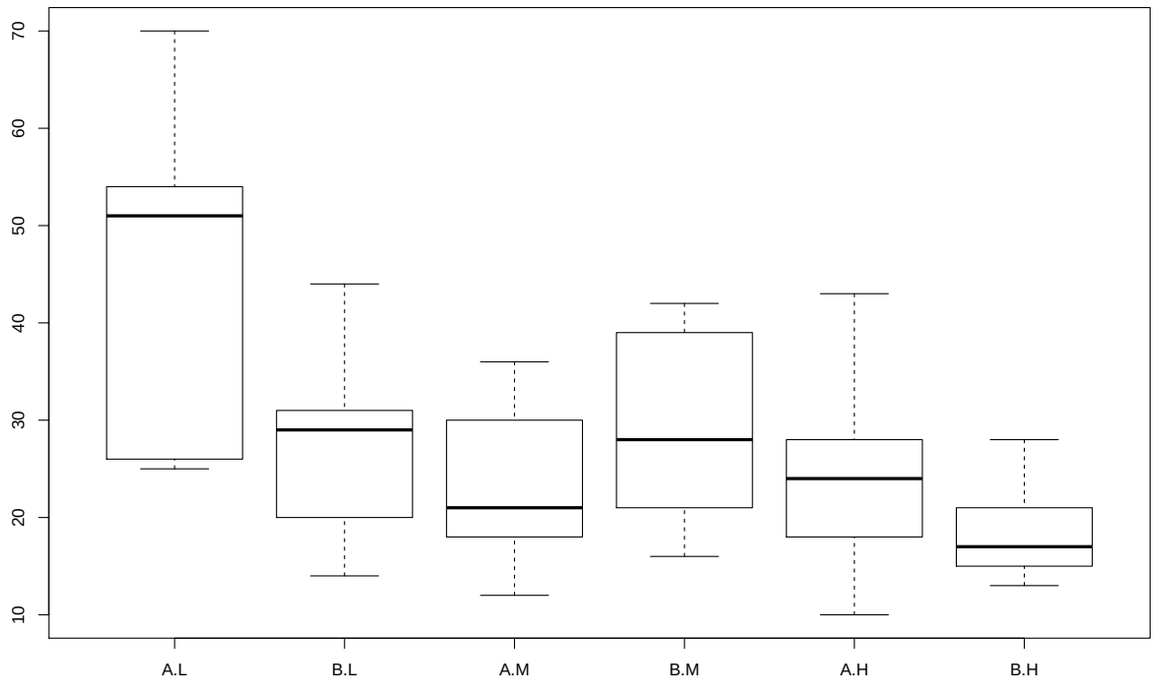
8.2 MANOVA

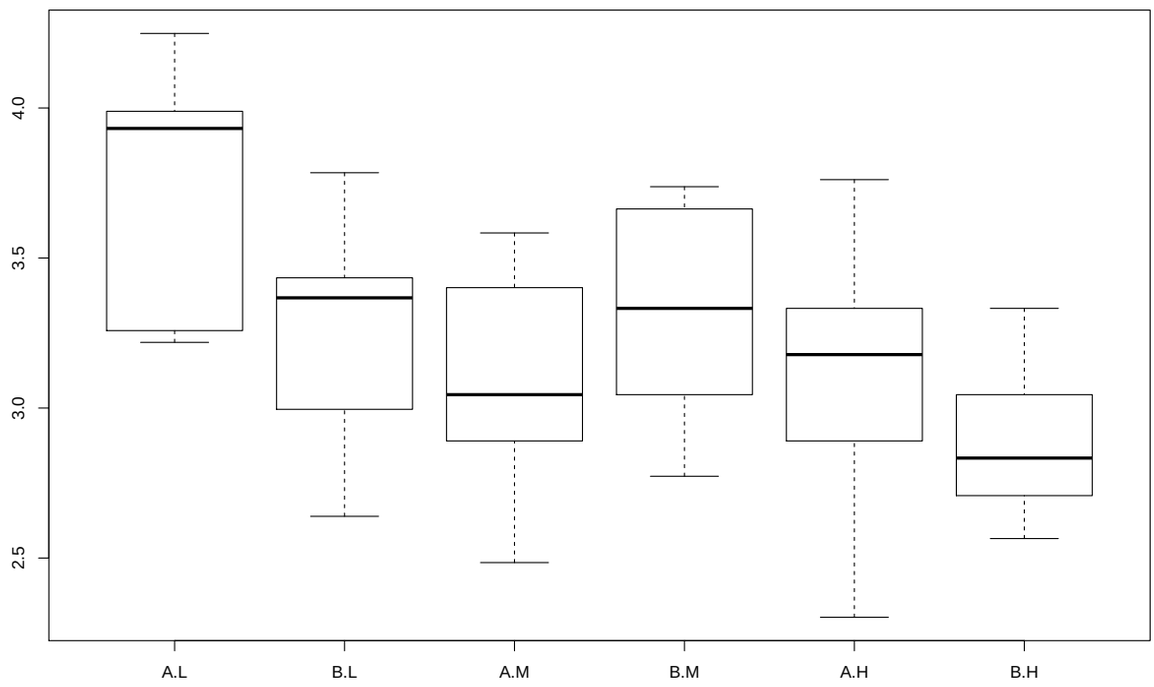
Let's explore an example with two factors. We'll use the Warpbreaks data set in R. Check the documentation for a description of the data by typing ?warpbreaks.

```
  breaks wool tension
1     26    A       L
2     30    A       L
3     54    A       L
4     25    A       L
5     70    A       L
6     52    A       L
```

Again, we visualize the data with box plots.

```
   L M H
A 9 9 9
B 9 9 9
```


70
60
50
40
30
20
10
A.L
B.L
A.M
B.M
A.H
B.H

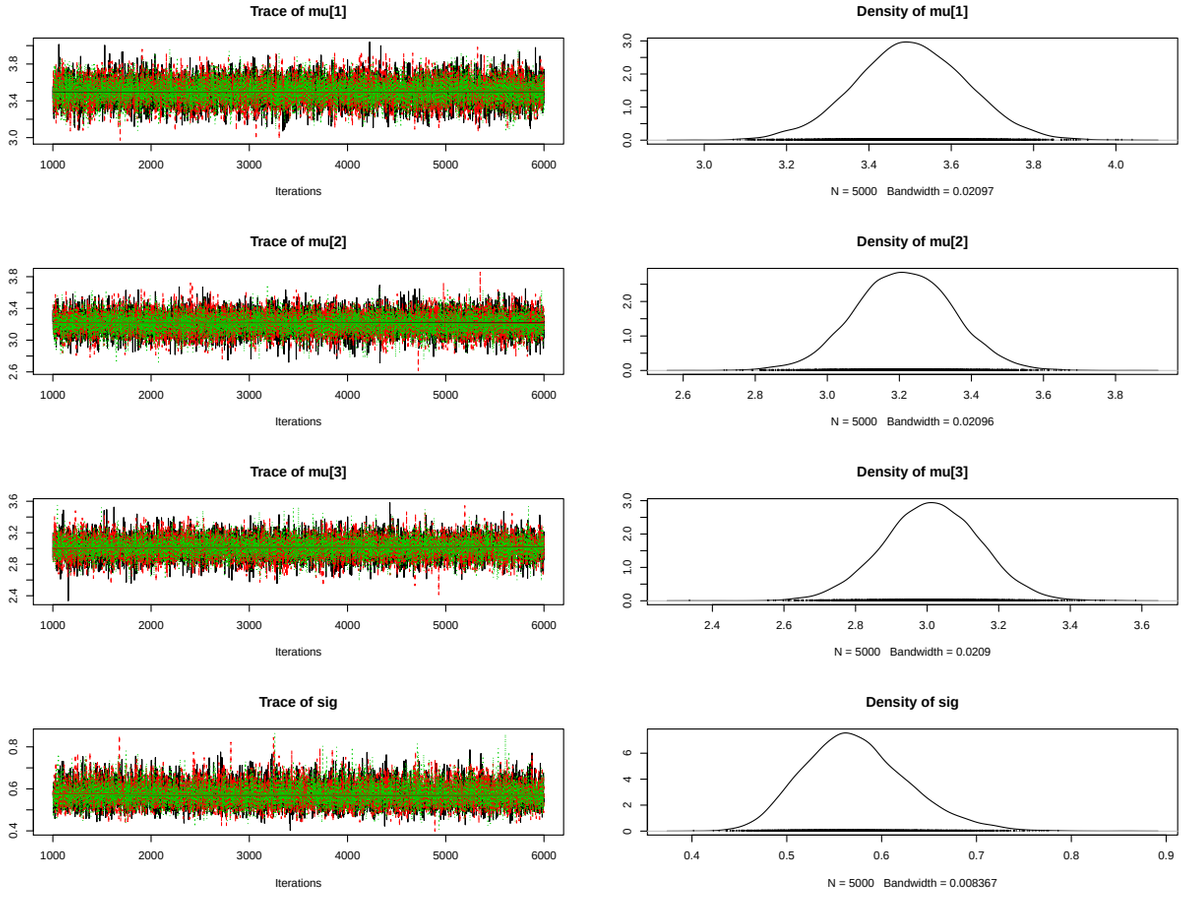

4.0
3.5
3.0
2.5
A.L
B.L
A.M
B.M
A.H
B.H

The different groups have more similar variance if we use the logarithm of breaks. From this visualization, it looks like both factors may play a role in the number of breaks. It appears that there is a general decrease in breaks as we move from low to medium to high tension. Let's start with a one-way model using tension only.

8.2.1 One-way model

```
'data.frame':   54 obs. of  3 variables:
 $ breaks : num  26 30 54 25 70 52 51 26 67 18 ...
 $ wool   : Factor w/ 2 levels "A","B": 1 1 1 1 1 1 1 1 1 1 ...
 $ tension: Factor w/ 3 levels "L","M","H": 1 1 1 1 1 1 1 1 1 2 ...
```

```
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 54
   Unobserved stochastic nodes: 4
   Total graph size: 134

Initializing model
```




```
Potential scale reduction factors:

      Point est. Upper C.I.
mu[1]          1          1
mu[2]          1          1
mu[3]          1          1
sig            1          1

Multivariate psrf

1
```

```
            mu[1]       mu[2]       mu[3]         sig
```

```
Lag 0   1.000000000  1.000000000  1.000000000  1.0000000000
Lag 1  -0.013580883 -0.009456398 -0.002265484  0.0458843311
Lag 5   0.003131747  0.007275436 -0.011125300 -0.0039197598
Lag 10 -0.004521612 -0.011855835 -0.008779452 -0.0059212032
Lag 50  0.007855755 -0.001667093 -0.003429943  0.0001826969
```

```
   mu[1]    mu[2]    mu[3]      sig
15028.66 15079.94 15000.00 13425.51
```

Here are the results.

```
Iterations = 1001:6000
Thinning interval = 1
Number of chains = 3
Sample size per chain = 5000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

        Mean      SD  Naive SE Time-series SE
mu[1] 3.5022 0.13538 0.0011054      0.0011044
mu[2] 3.2124 0.13528 0.0011045      0.0011016
mu[3] 3.0126 0.13489 0.0011014      0.0011014
sig   0.5732 0.05509 0.0004498      0.0004755

2. Quantiles for each variable:

        2.5%    25%    50%    75%  97.5%
mu[1] 3.2318 3.4119 3.5012 3.5934 3.7692
mu[2] 2.9479 3.1218 3.2126 3.3042 3.4784
mu[3] 2.7458 2.9233 3.0132 3.1042 3.2732
```

```
sig   0.4782 0.5349 0.5692 0.6073 0.6932
```

The 95% posterior interval for the mean of group 2 (medium tension) overlaps with both the low and high groups, but the low and high groups do not have overlapping intervals. That is a pretty strong indication that the means for low and high tension are different. Let's collect the DIC for this model and move on to the two-way model.

8.2.2 Two-way additive model

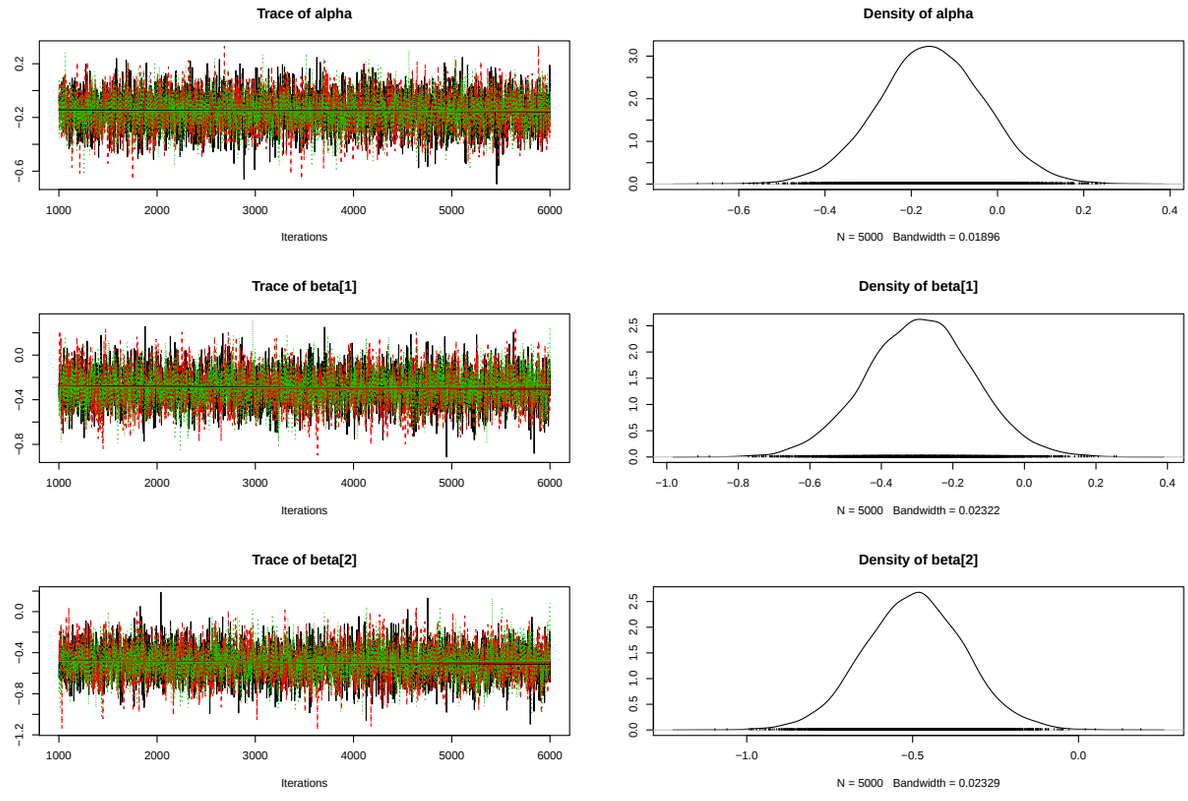
With two factors, one with two levels and the other with three, we have six treatment groups, which is the same situation we discussed when introducing multiple factor ANOVA. We will first fit the additive model which treats the two factors separately with no interaction. To get the XX matrix (or design matrix) for this model, we can extract it from a linear model in R.

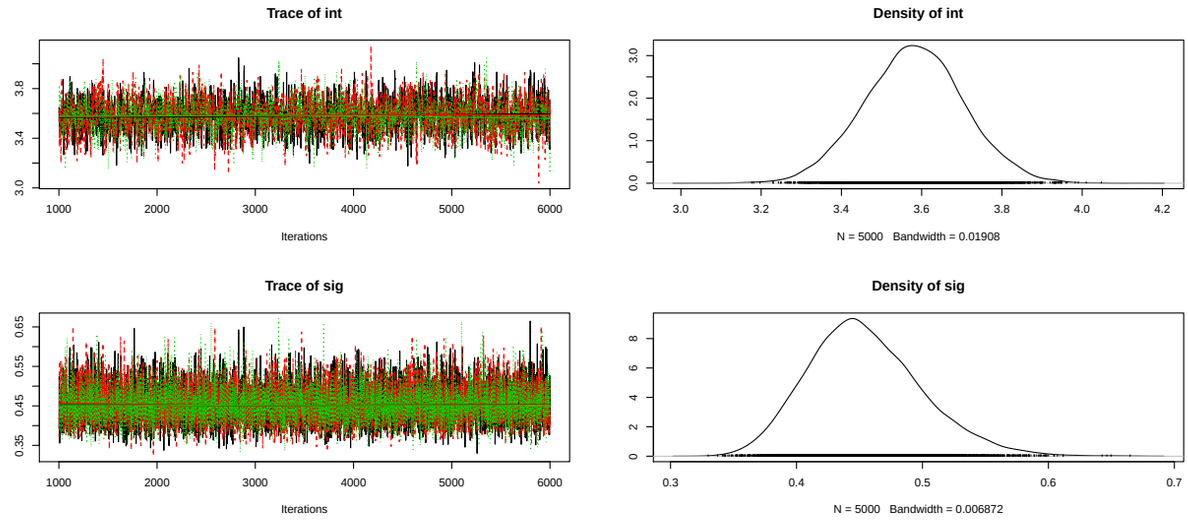
```
  (Intercept) woolB tensionM tensionH
1           1     0        0        0
2           1     0        0        0
3           1     0        0        0
4           1     0        0        0
5           1     0        0        0
6           1     0        0        0

   (Intercept) woolB tensionM tensionH
49           1     1        0        1
50           1     1        0        1
51           1     1        0        1
52           1     1        0        1
53           1     1        0        1
54           1     1        0        1
```

By default, R has chosen the mean for wool A and low tension to be the intercept. Then, there is an effect for wool B, and effects for medium tension and high tension,

each associated with dummy indicator variables.

```
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 54
   Unobserved stochastic nodes: 5
   Total graph size: 257

Initializing model
```






```
Potential scale reduction factors:

      Point est. Upper C.I.
mu[1]          1          1
mu[2]          1          1
mu[3]          1          1
sig            1          1

Multivariate psrf

1
```

	mu[1]	mu[2]	mu[3]	sig
Lag 0	1.000000000	1.000000000	1.000000000	1.0000000000
Lag 1	-0.013580883	-0.009456398	-0.002265484	0.0458843311

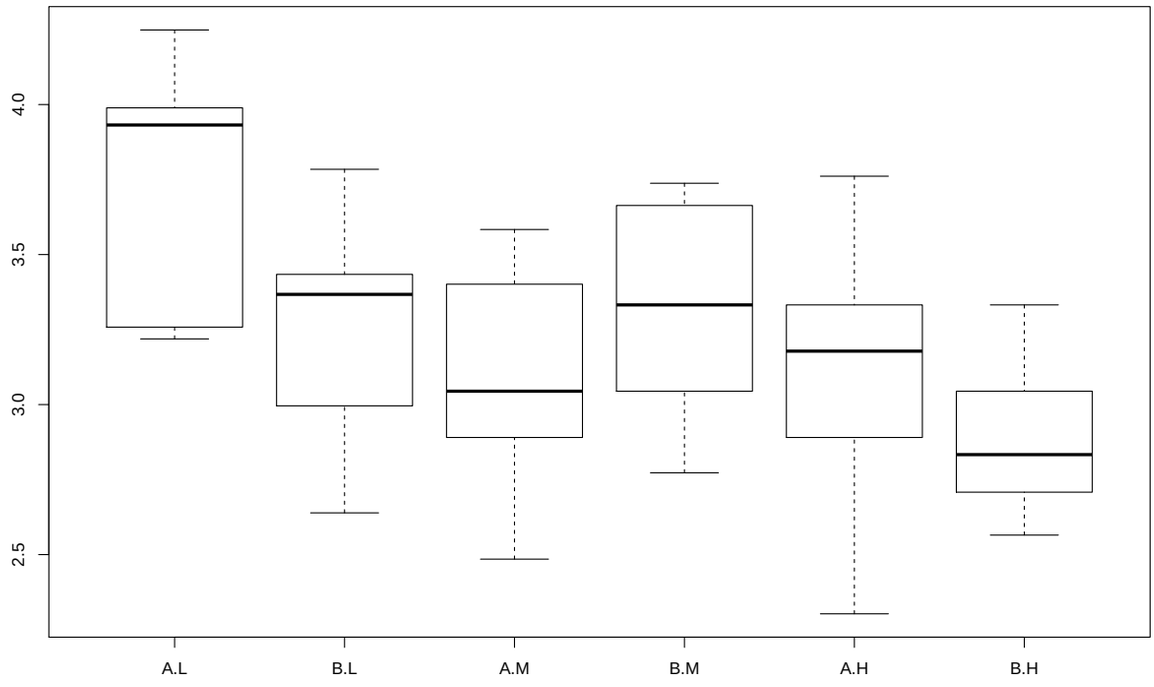
```
Lag 5   0.003131747  0.007275436 -0.011125300 -0.0039197598
Lag 10 -0.004521612 -0.011855835 -0.008779452 -0.0059212032
Lag 50  0.007855755 -0.001667093 -0.003429943  0.0001826969

   mu[1]    mu[2]    mu[3]      sig
15028.66 15079.94 15000.00 13425.51
```

Let's summarize the results, collect the DIC for this model, and compare it to the first one-way model.

```
Iterations = 1001:6000
Thinning interval = 1
Number of chains = 3
Sample size per chain = 5000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

           Mean      SD  Naive SE Time-series SE
alpha   -0.1538 0.12394 0.0010120      0.0017367
beta[1] -0.2914 0.15084 0.0012316      0.0024010
beta[2] -0.4951 0.15111 0.0012338      0.0023827
int      3.5810 0.12413 0.0010135      0.0025129
sig      0.4544 0.04518 0.0003689      0.0004093

2. Quantiles for each variable:

           2.5%     25%     50%      75%     97.5%
alpha   -0.3991 -0.2353 -0.1544 -0.07133  0.090617
beta[1] -0.5894 -0.3925 -0.2903 -0.19173  0.003711
beta[2] -0.7907 -0.5957 -0.4938 -0.39430 -0.195812
```

```
int      3.3372  3.4979  3.5816  3.66289  3.827580
sig      0.3760  0.4228  0.4504  0.48228  0.552900

Mean deviance:  55.66
penalty 5.204
Penalized deviance: 60.86

Mean deviance:  66.69
penalty 4.102
Penalized deviance: 70.79
```

This suggests we haven't gained much from adding the wool factor to the model. Before we discard wool, however, we should consider whether there is an interaction. Let's look again at the box plot with all six treatment groups.



Our two-way model has a single effect for wool B and the estimate is negative. If this is true, then we would expect wool B to be associated with fewer breaks than its

wool A counterpart on average. This is true for low and high tension, but it appears that breaks are higher for wool B when there is medium tension. That is, the effect for wool B is not consistent across tension levels, so it may appropriate to add an interaction term. In R, this would look like:

```
Call:
lm(formula = log(breaks) ~ .^2, data = warpbreaks)

Residuals:
     Min       1Q   Median       3Q      Max
-0.81504 -0.27885  0.04042  0.27319  0.64358

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)      3.7179     0.1247  29.824  < 2e-16 ***
woolB           -0.4356     0.1763  -2.471  0.01709 *
tensionM        -0.6012     0.1763  -3.410  0.00133 **
tensionH        -0.6003     0.1763  -3.405  0.00134 **
woolB:tensionM   0.6281     0.2493   2.519  0.01514 *
woolB:tensionH   0.2221     0.2493   0.891  0.37749
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.374 on 48 degrees of freedom
Multiple R-squared:  0.3363,    Adjusted R-squared:  0.2672
F-statistic: 4.864 on 5 and 48 DF,  p-value: 0.001116
```

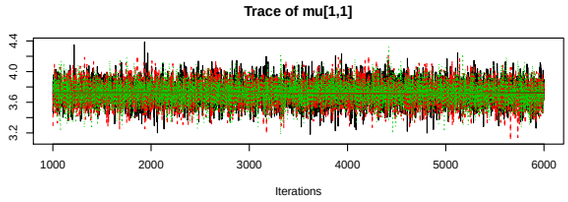
Adding the interaction, we get an effect for being in wool B and medium tension, as well as for being in wool B and high tension. There are now six parameters for the mean, one for each treatment group, so this model is equivalent to the full cell means

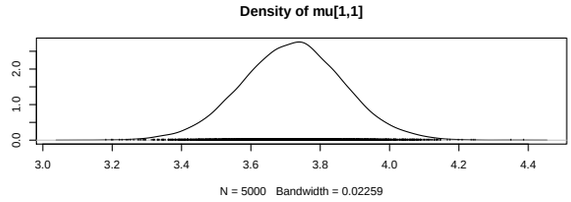
model. Let's use that.

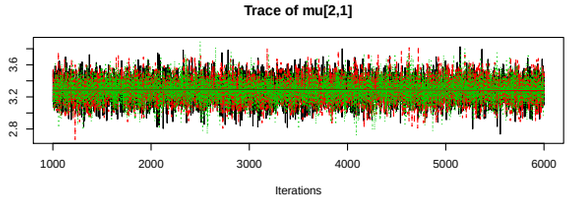
8.2.3 Two-way cell means model

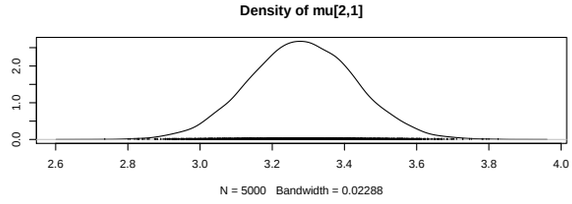
In this new model, μ will be a matrix with six entries, each corresponding to a treatment group.

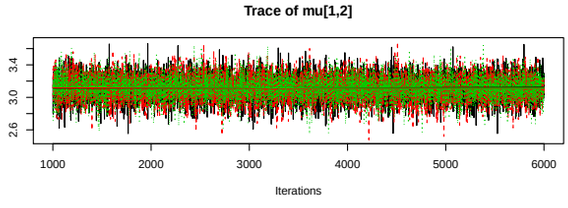
```
'data.frame':   54 obs. of  3 variables:
 $ breaks : num  26 30 54 25 70 52 51 26 67 18 ...
 $ wool   : Factor w/ 2 levels "A","B": 1 1 1 1 1 1 1 1 1 1 ...
 $ tension: Factor w/ 3 levels "L","M","H": 1 1 1 1 1 1 1 1 1 2 ...

Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 54
   Unobserved stochastic nodes: 7
   Total graph size: 199

Initializing model
```

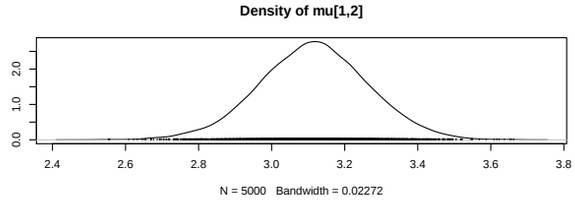

Trace of mu[1,1]
Iterations

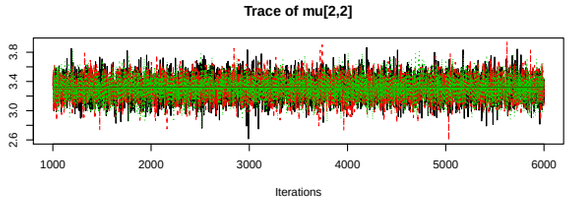

Density of mu[1,1]
N = 5000 Bandwidth = 0.02259

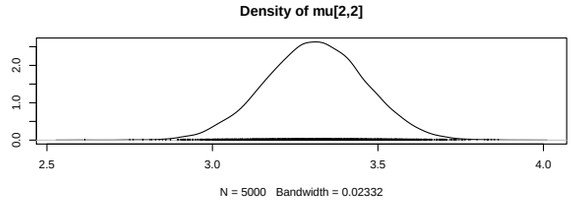

Trace of mu[2,1]
Iterations

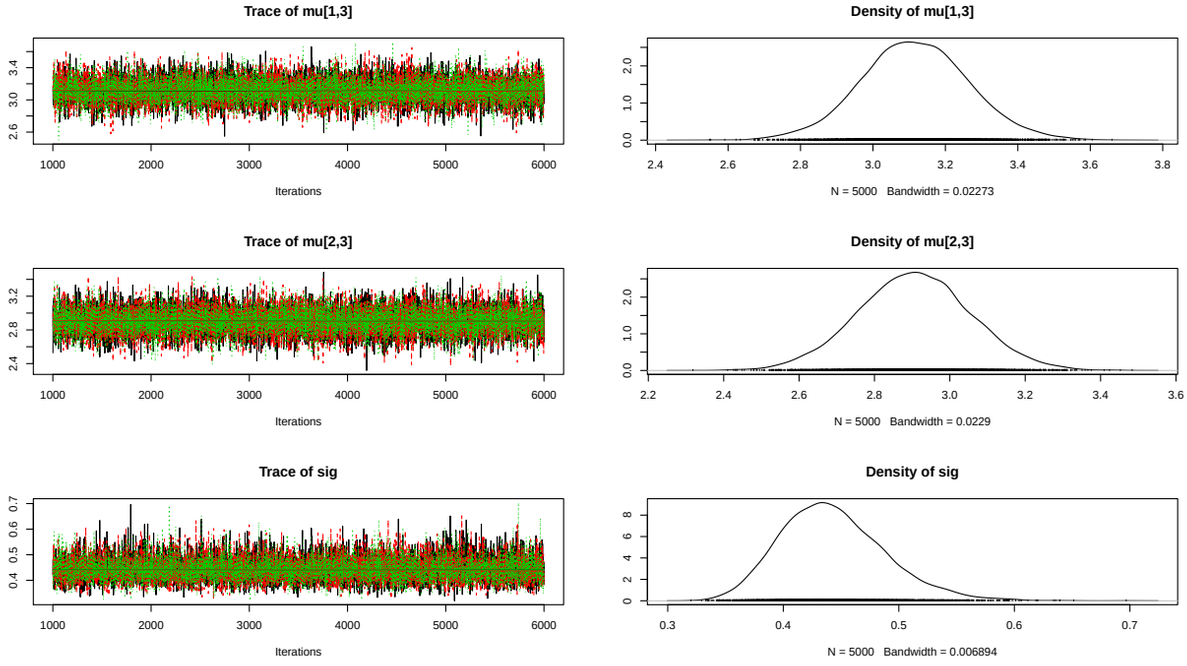

Density of mu[2,1]
N = 5000 Bandwidth = 0.02288


Trace of mu[1,2]
Iterations


Density of mu[1,2]
N = 5000 Bandwidth = 0.02272


Trace of mu[2,2]
Iterations


Density of mu[2,2]
N = 5000 Bandwidth = 0.02332




```
Potential scale reduction factors:

       Point est. Upper C.I.
mu[1,1]          1          1
mu[2,1]          1          1
mu[1,2]          1          1
mu[2,2]          1          1
mu[1,3]          1          1
mu[2,3]          1          1
sig              1          1

Multivariate psrf
```

```
1
```

	mu[1,1]	mu[2,1]	mu[1,2]	mu[2,2]	mu[1,3]
Lag 0	1.000000000	1.0000000000	1.000000000	1.0000000000	1.0000000000
Lag 1	-0.010840923	-0.0008654414	-0.006230629	0.0143324296	0.0045536780
Lag 5	0.000698421	0.0032887485	0.006197581	0.0035250847	0.0048152746
Lag 10	-0.006106281	-0.0002380991	0.010823704	0.0078117050	0.0084014726
Lag 50	-0.020887207	0.0017020857	0.004608517	-0.0002187422	0.0008408908

	mu[2,3]	sig
Lag 0	1.000000000	1.000000000
Lag 1	-0.002391964	0.116438973
Lag 5	0.010960418	-0.011330403
Lag 10	0.003751526	-0.003565712
Lag 50	-0.002339933	0.007433827

mu[1,1]	mu[2,1]	mu[1,2]	mu[2,2]	mu[1,3]	mu[2,3]	sig
15286.93	15434.24	16185.77	14170.73	15000.00	15000.00	12183.27

```
[[1]]


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

	Burn-in (M)	Total (N)	Lower bound (Nmin)	Dependence factor (I)
mu[1,1]	2	3930	3746	1.050
mu[2,1]	2	3741	3746	0.999
mu[1,2]	2	3536	3746	0.944
mu[2,2]	2	3866	3746	1.030
mu[1,3]	3	4062	3746	1.080
mu[2,3]	2	3653	3746	0.975

```
 sig     2         3995  3746          1.070


[[2]]

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

          Burn-in  Total Lower bound  Dependence
          (M)      (N)   (Nmin)       factor (I)
 mu[1,1] 2         3774  3746         1.010
 mu[2,1] 2         3680  3746         0.982
 mu[1,2] 2         3866  3746         1.030
 mu[2,2] 2         3995  3746         1.070
 mu[1,3] 2         3803  3746         1.020
 mu[2,3] 2         3741  3746         0.999
 sig     2         3930  3746         1.050


[[3]]

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

          Burn-in  Total Lower bound  Dependence
          (M)      (N)   (Nmin)       factor (I)
 mu[1,1] 2         3930  3746         1.050
 mu[2,1] 2         3620  3746         0.966
```

```
mu[1,2] 2        3741  3746        0.999
mu[2,2] 2        3866  3746        1.030
mu[1,3] 2        3930  3746        1.050
mu[2,3] 2        3620  3746        0.966
sig     2        3803  3746        1.020
```

Let's compute the DIC and compare with our previous models.

```
Mean deviance:  52.07
penalty 7.25
Penalized deviance: 59.32
```

```
Mean deviance:  55.66
penalty 5.204
Penalized deviance: 60.86
```

```
Mean deviance:  66.69
penalty 4.102
Penalized deviance: 70.79
```

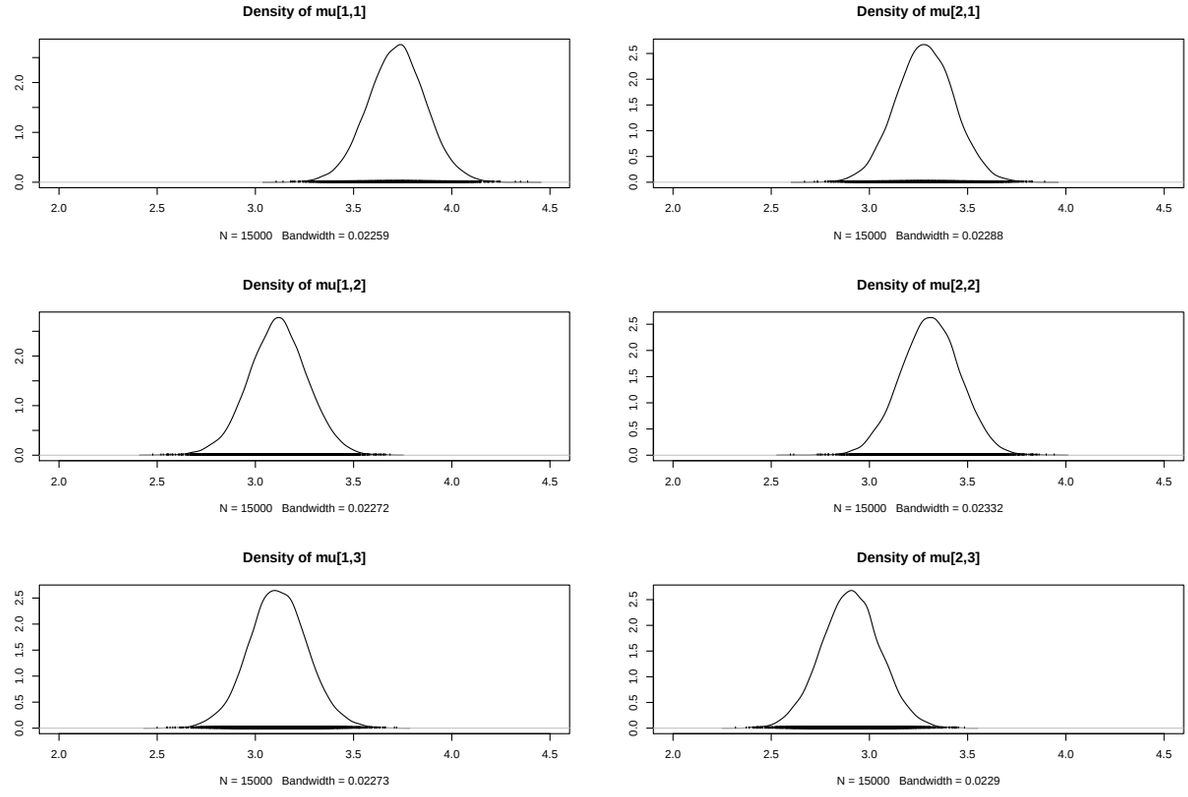
This suggests that the full model with interaction between wool and tension is the best for explaining/predicting warp breaks.

8.2.4 Results

```
Iterations = 1001:6000
Thinning interval = 1
Number of chains = 3
Sample size per chain = 5000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:
```

```
          Mean     SD  Naive SE Time-series SE
mu[1,1] 3.720 0.1473 0.0012028       0.001192
mu[2,1] 3.283 0.1479 0.0012075       0.001191
mu[1,2] 3.115 0.1493 0.0012189       0.001187
mu[2,2] 3.310 0.1507 0.0012301       0.001266
mu[1,3] 3.116 0.1490 0.0012164       0.001216
mu[2,3] 2.906 0.1498 0.0012229       0.001223
sig     0.443 0.0453 0.0003698       0.000411

2. Quantiles for each variable:

          2.5%   25%    50%    75%  97.5%
mu[1,1] 3.4284 3.622 3.7218 3.8178 4.0118
mu[2,1] 2.9963 3.185 3.2830 3.3829 3.5747
mu[1,2] 2.8150 3.017 3.1158 3.2136 3.4093
mu[2,2] 3.0150 3.210 3.3110 3.4116 3.6026
mu[1,3] 2.8226 3.018 3.1145 3.2142 3.4172
mu[2,3] 2.6114 2.805 2.9053 3.0034 3.2021
sig     0.3662 0.411 0.4392 0.4706 0.5428

            lower     upper
mu[1,1] 3.4230720 4.0050889
mu[2,1] 3.0006375 3.5782387
mu[1,2] 2.8374100 3.4279386
mu[2,2] 3.0088161 3.5958475
mu[1,3] 2.8291586 3.4224025
mu[2,3] 2.6093353 3.1986654
sig     0.3629701 0.5382795
attr(,"Probability")
```

```
[1] 0.95
```



It might be tempting to look at comparisons between each combination of treatments, but we warn that this could yield spurious results. When we discussed the statistical modeling cycle, we said it is best not to search your results for interesting hypotheses, because if there are many hypotheses, some will appear to show "effects" or "associations" simply due to chance. Results are most reliable when we determine a relatively small number of hypotheses we are interested in beforehand, collect the data, and statistically evaluate the evidence for them.

One question we might be interested in with these data is finding the treatment combination that produces the fewest breaks. To calculate this, we can go through our posterior samples and for each sample, find out which group has the smallest mean. These counts help us determine the posterior probability that each of the treatment groups has the smallest mean.

```
          2           3           4           5           6
0.01433333 0.11740000 0.01106667 0.12040000 0.73680000
```

The evidence supports wool B with high tension as the treatment that produces the fewest breaks.

8.3 Linear Regression

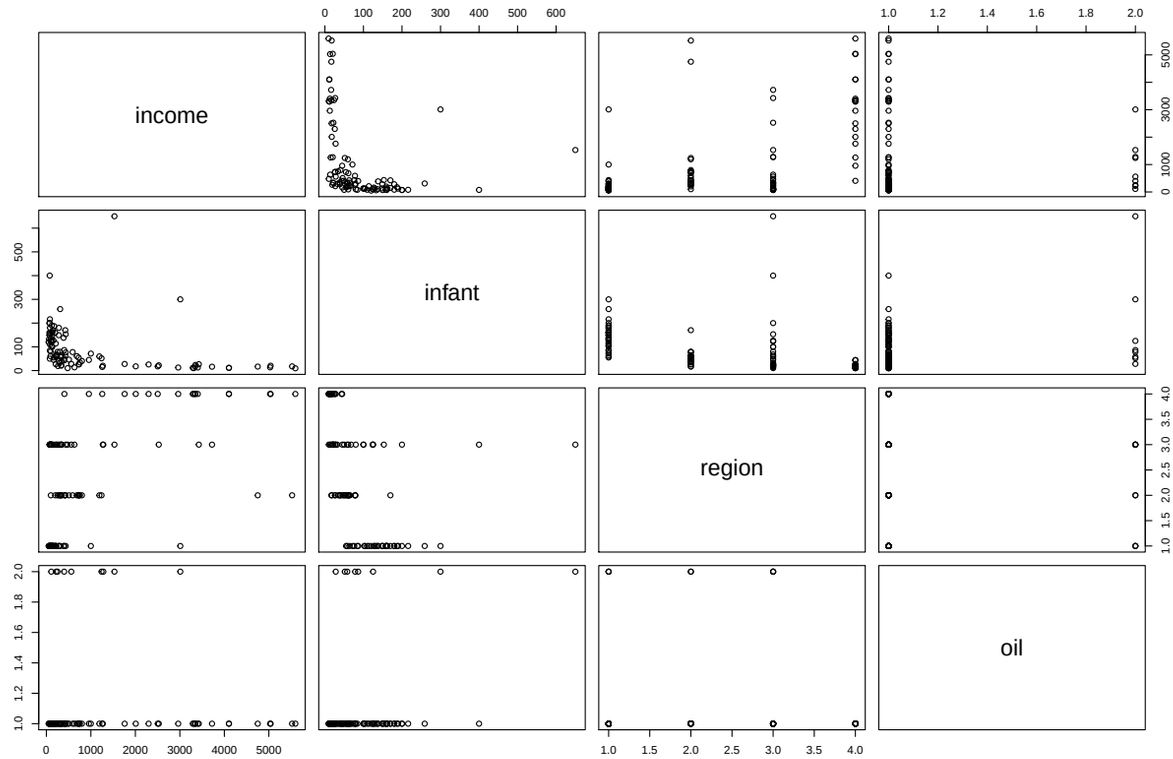
As an example of linear regression, we'll look at the Leinhardt data from the car package in R.

```
          income infant   region oil
Australia   3426   26.7     Asia  no
Austria     3350   23.7   Europe  no
Belgium     3346   17.0   Europe  no
Canada      4751   16.8 Americas  no
Denmark     5029   13.5   Europe  no
Finland     3312   10.1   Europe  no
```

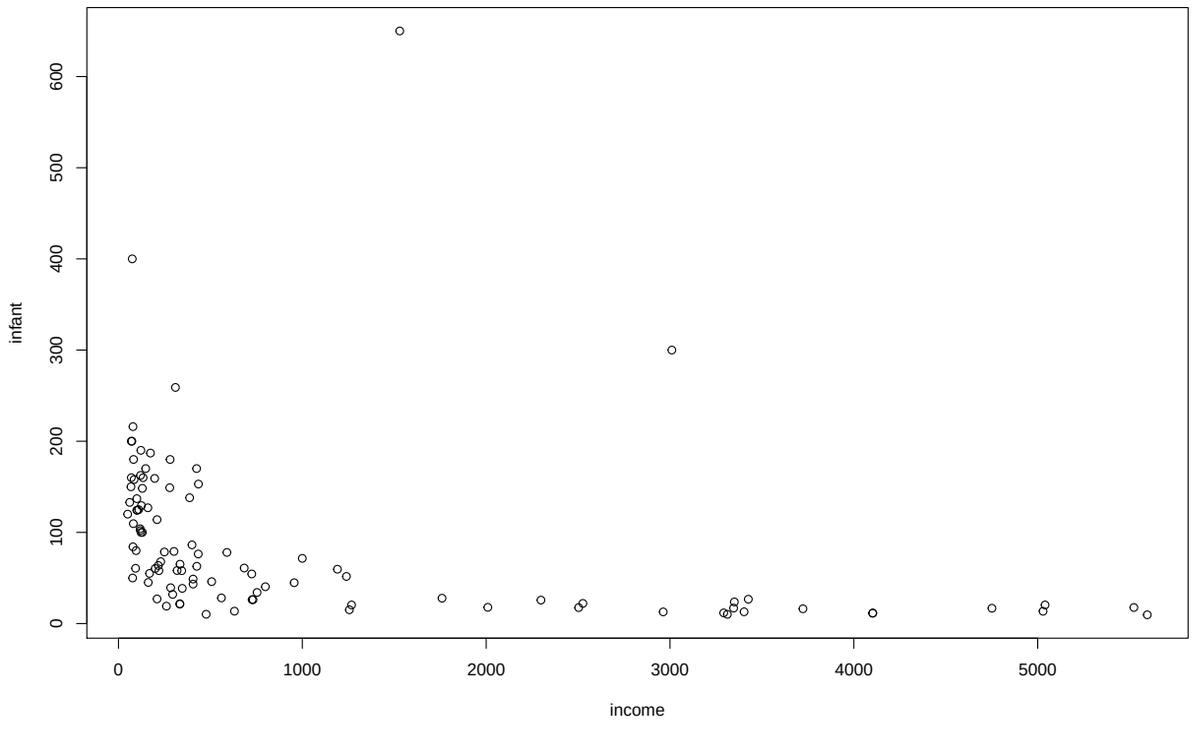
So the Leinhardt dataset has 105 observations and 4 variables: income, infant, region, oil.

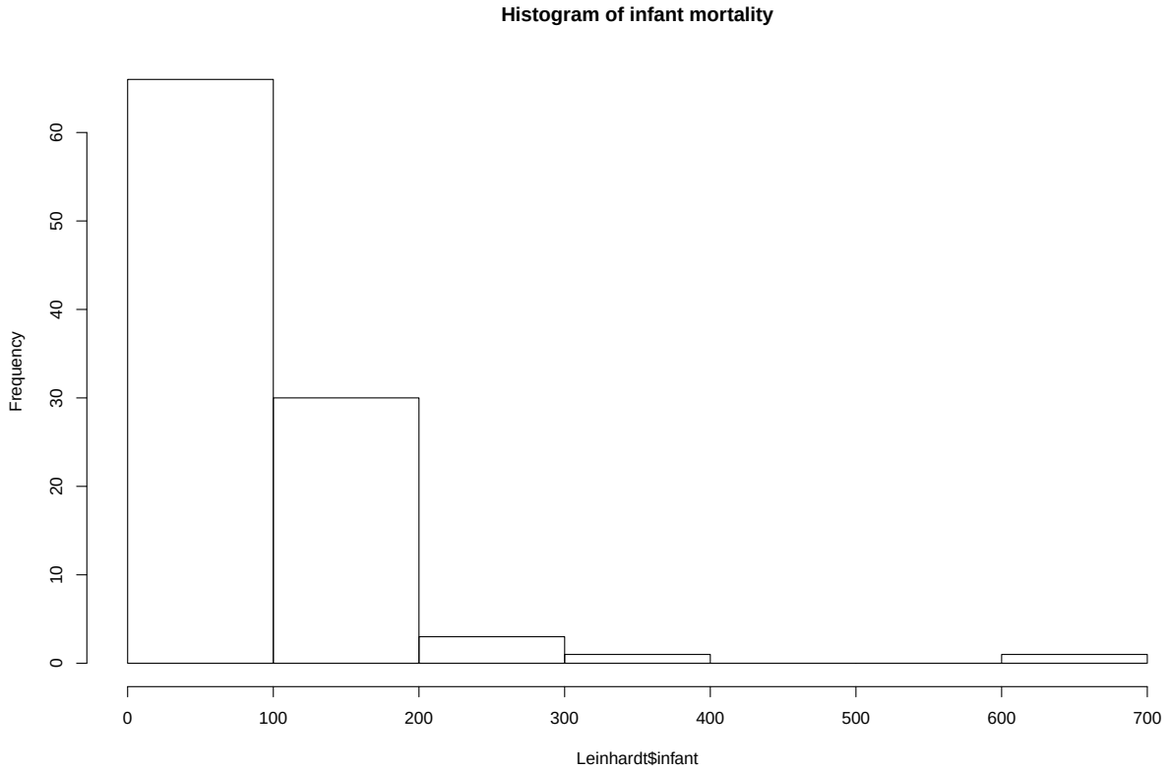
```
'data.frame':   105 obs. of  4 variables:
 $ income: int  3426 3350 3346 4751 5029 3312 3403 5040 2009 2298 ...
 $ infant: num  26.7 23.7 17 16.8 13.5 10.1 12.9 20.4 17.8 25.7 ...
 $ region: Factor w/ 4 levels "Africa","Americas",..: 3 4 4 2 4 4 4 4 4 4 .
 $ oil   : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
```

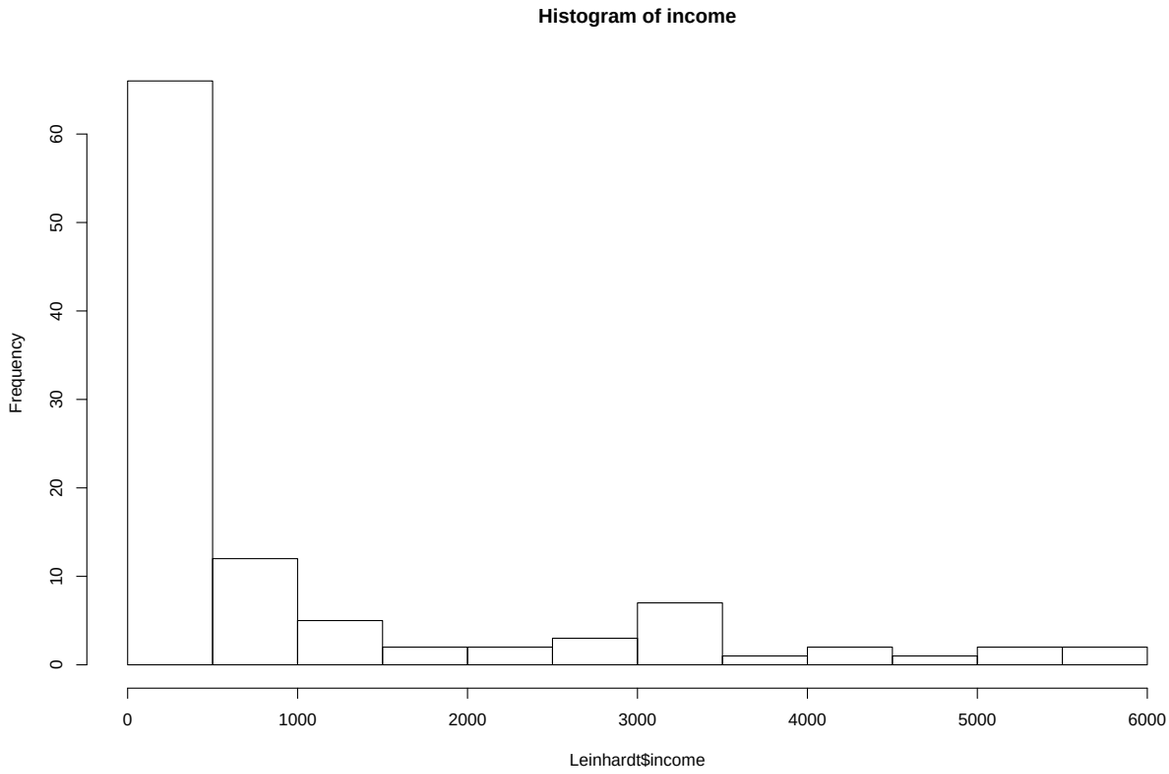
So the correlation between the variables is shown by the following paired scatterplot.



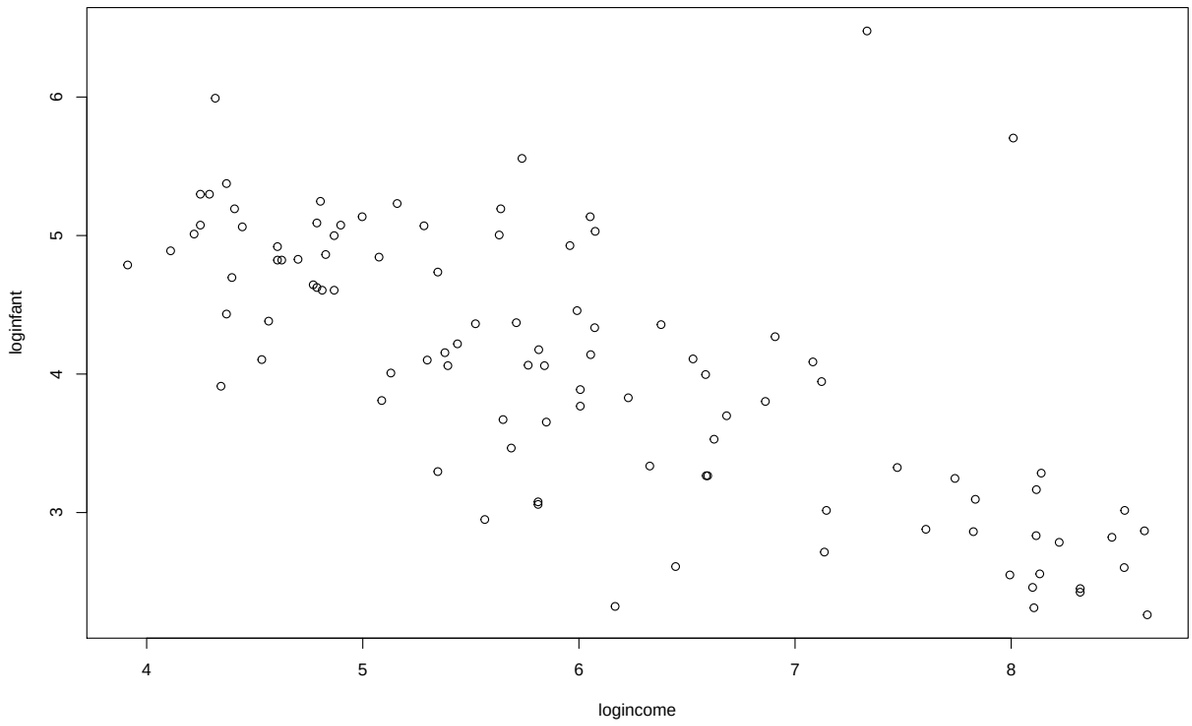
We'll start with a simple linear regression model that relates infant mortality to per capita income, but first let's take a look if there is a linear relation between the two variables. This can be easily tested by a scatterplot.


infant
0
100
200
300
400
500
600
0
1000
2000
3000
4000
5000
income


Histogram of infant mortality
Frequency
0
10
20
30
40
50
60
0
100
200
300
400
500
600
700
Leinhardt$infant



As we can easily observe our setting is not suitable for a linear regression model, because the variables are heavily skewed, plus the scatterplot shows no linearity. This can be corrected in some cases when we use the log transformation :

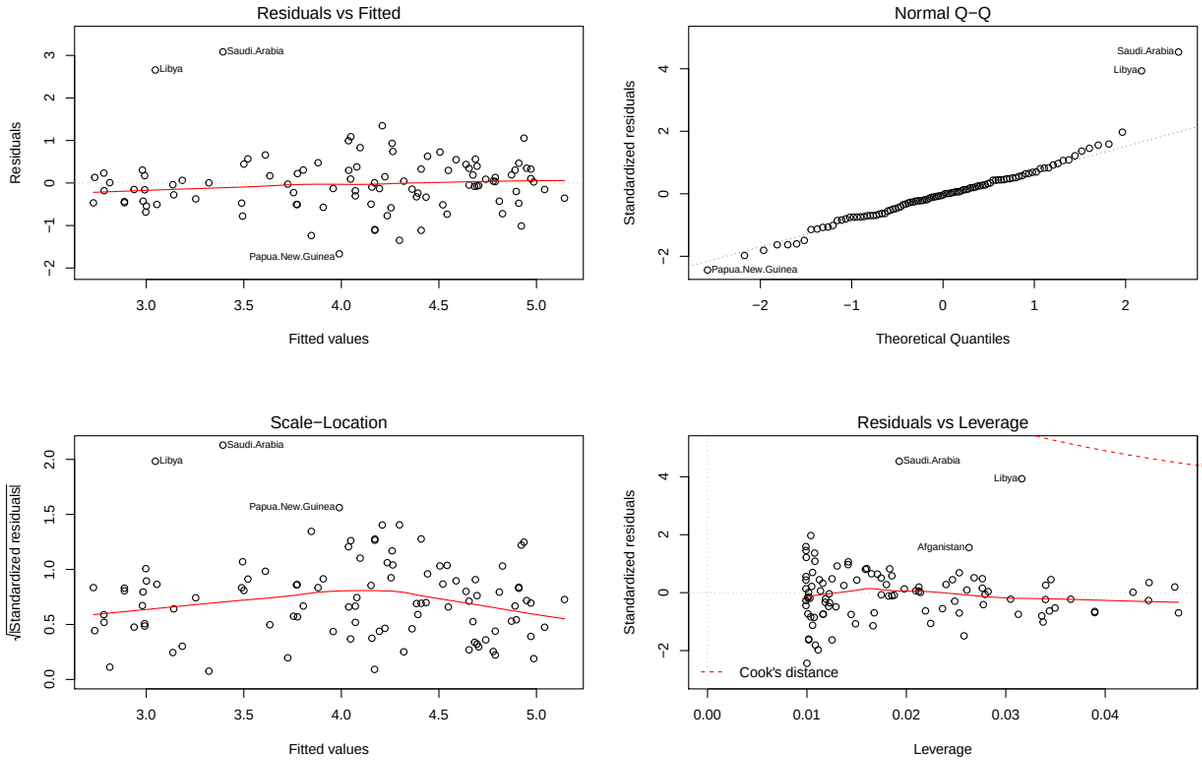


A linear model appears much more appropriate on this (log) scale. The first model we may apply is the frequentist (non informative Bayesian) linear model.

```
Call:
lm(formula = loginfant ~ logincome, data = Leinhardt)

Residuals:
     Min       1Q   Median       3Q      Max
-1.66694 -0.42779 -0.02649  0.30441  3.08415

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  7.14582    0.31654  22.575   <2e-16 ***
logincome   -0.51179    0.05122  -9.992   <2e-16 ***
```

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.6867 on 99 degrees of freedom
  (4 observations deleted due to missingness)
Multiple R-squared:  0.5021,    Adjusted R-squared:  0.4971
F-statistic: 99.84 on 1 and 99 DF,  p-value: < 2.2e-16
```

Let's check also the model fit.



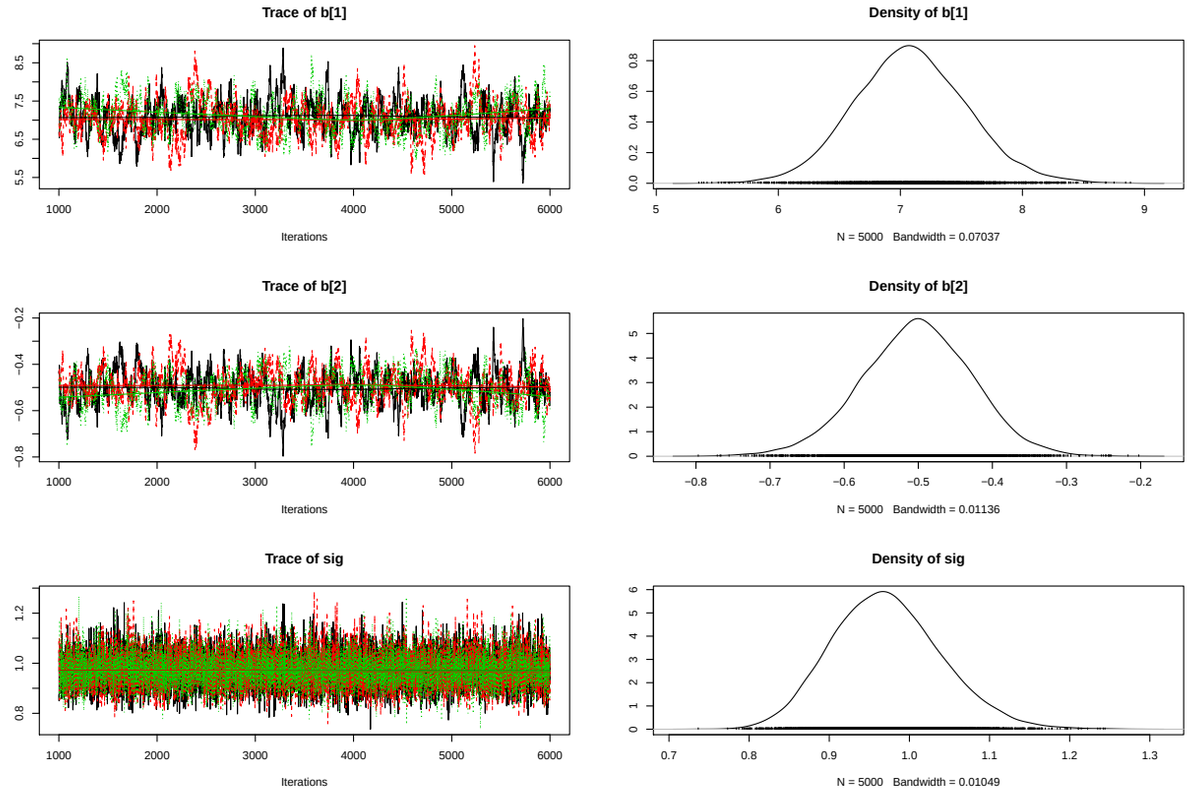

```
null device
          1
```

Now let's fit the same model using JAGS. We can also ommit the some countries with missing values for easier calculations.

```
Compiling model graph
   Resolving undeclared variables
```

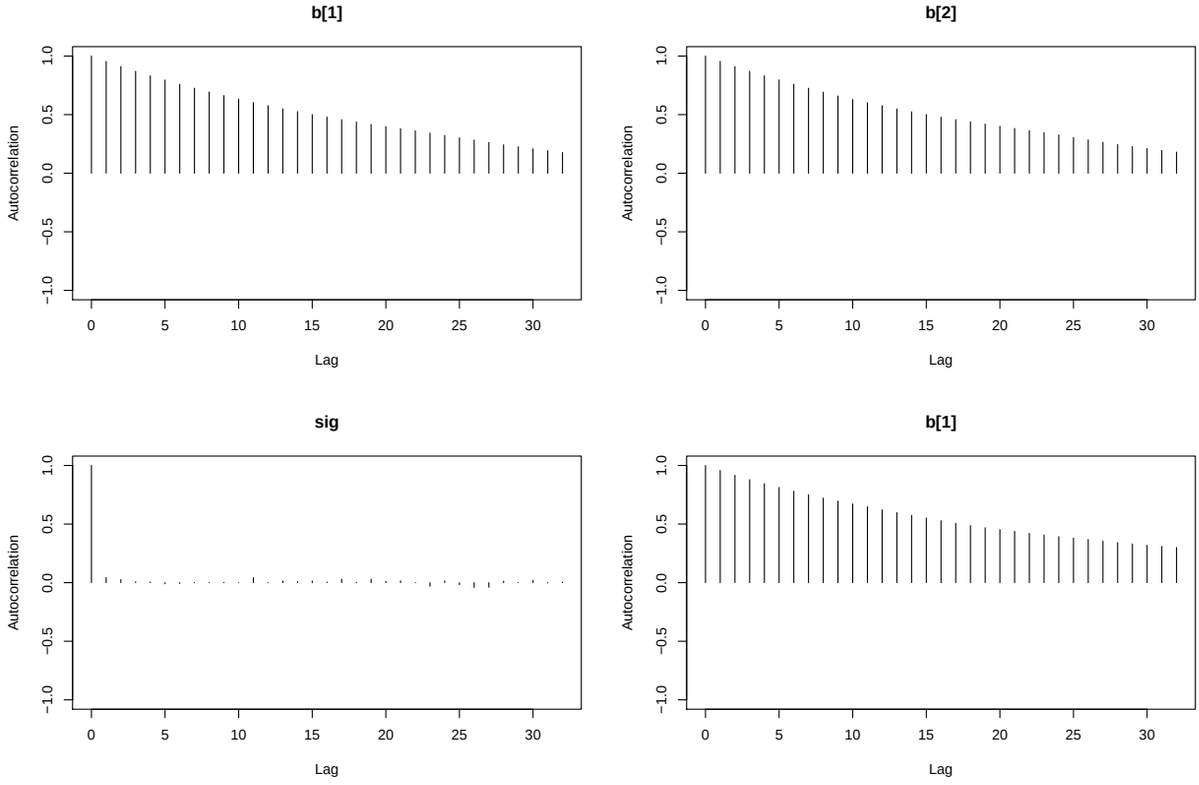
```
   Allocating nodes
Graph information:
   Observed stochastic nodes: 101
   Unobserved stochastic nodes: 3
   Total graph size: 411

Initializing model
```

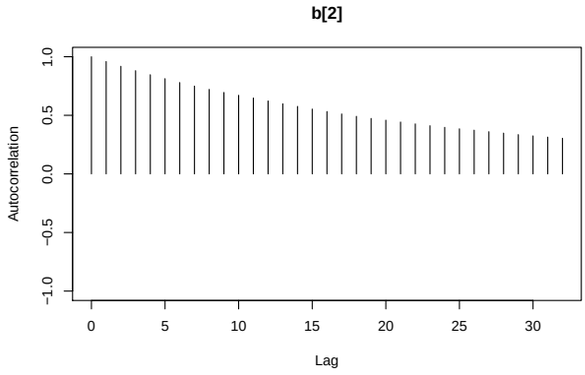
Before we check the inferences from the model, we should perform convergence diagnostics for our Markov chains.

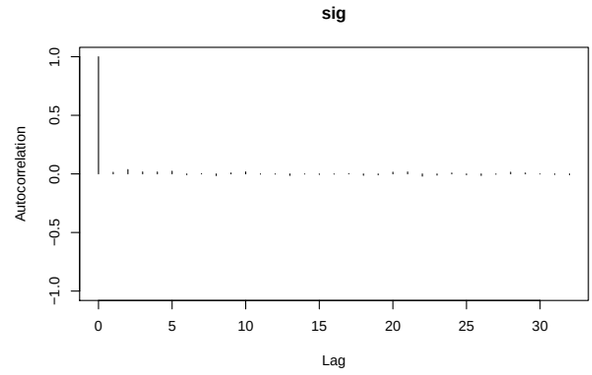


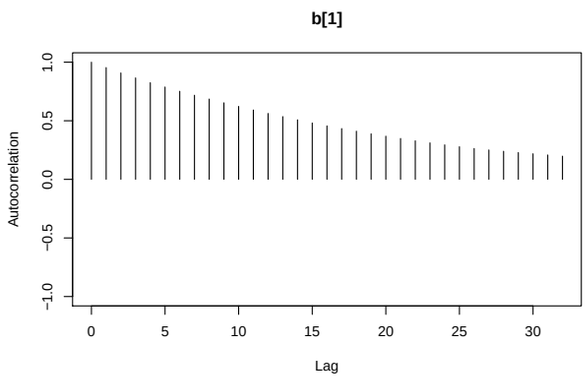

```
Potential scale reduction factors:

     Point est. Upper C.I.
b[1]       1.01       1.04
b[2]       1.01       1.04
```

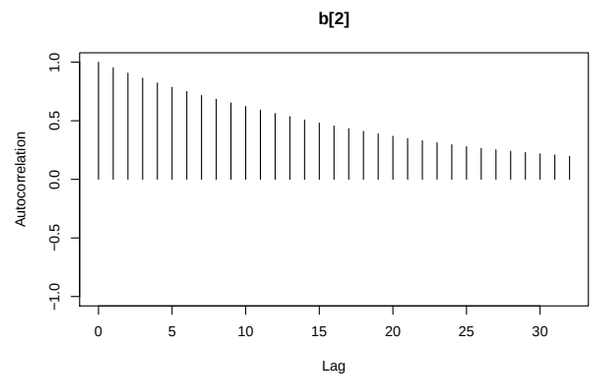
```
sig        1.00       1.00

Multivariate psrf

1.01

              b[1]      b[2]           sig
Lag 0   1.00000000 1.0000000  1.0000000000
Lag 1   0.95501553 0.9555351  0.0254982058
Lag 5   0.79850615 0.7982450 -0.0009937883
Lag 10  0.64202395 0.6405183  0.0039159611
Lag 50  0.09919276 0.1015041 -0.0042298830
```

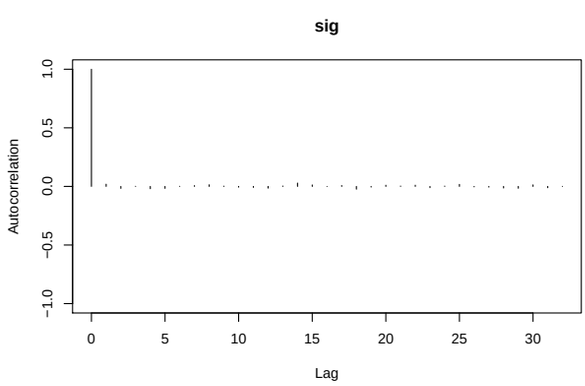



b[2]
Autocorrelation
1.0
0.5
0.0
−0.5
−1.0
0
5
10
15
20
25
30
Lag


sig
Autocorrelation
1.0
0.5
0.0
−0.5
−1.0
0
5
10
15
20
25
30
Lag


b[1]
Autocorrelation
1.0
0.5
0.0
−0.5
−1.0
0
5
10
15
20
25
30
Lag


b[2]
Autocorrelation
1.0
0.5
0.0
−0.5
−1.0
0
5
10
15
20
25
30
Lag

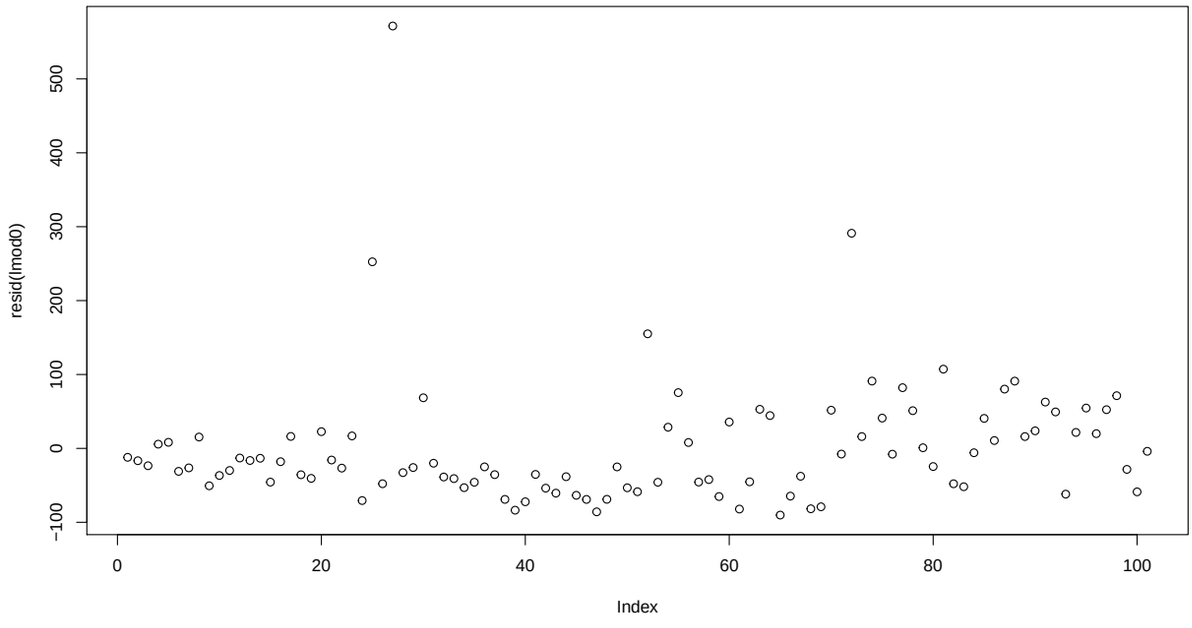

sig
Autocorrelation
1.0
0.5
0.0
−0.5
−1.0
0
5
10
15
20
25
30
Lag

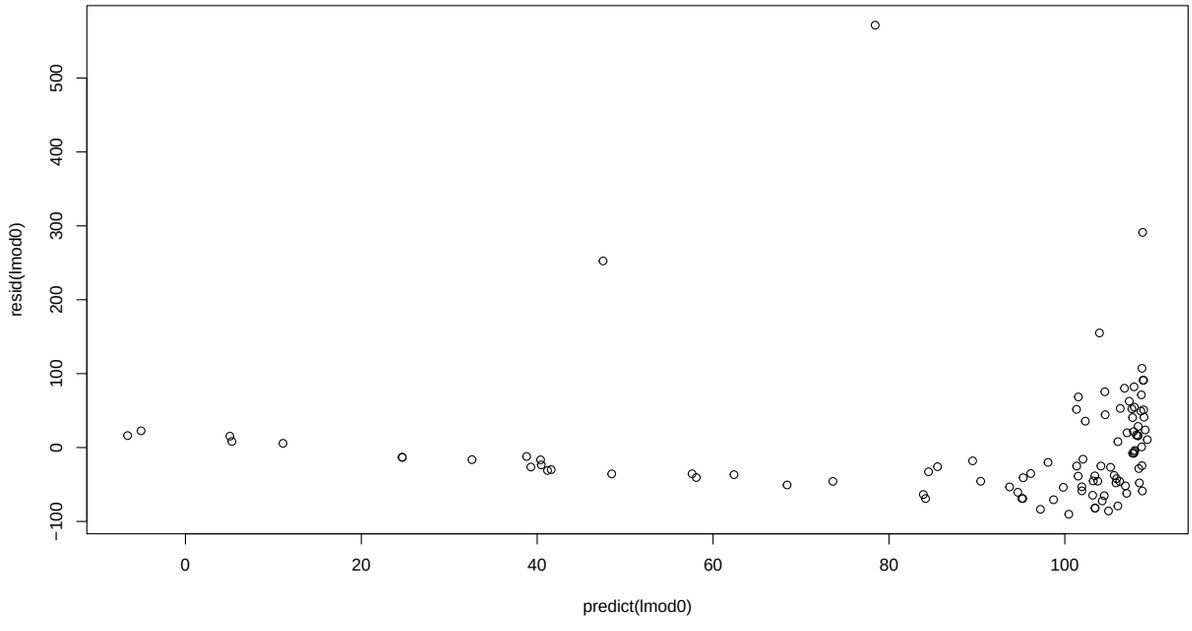
```
      b[1]       b[2]        sig
  350.0683   340.9388 13877.3995


Iterations = 1001:6000
Thinning interval = 1
Number of chains = 3
Sample size per chain = 5000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

        Mean      SD  Naive SE Time-series SE
b[1]  7.0853 0.46055 0.0037604      0.0245334
b[2] -0.5021 0.07442 0.0006077      0.0040219
sig   0.9718 0.06839 0.0005584      0.0005818

2. Quantiles for each variable:

        2.5%     25%     50%     75%   97.5%
b[1]  6.2003  6.7789  7.0774  7.3876  8.0196
b[2] -0.6532 -0.5506 -0.5008 -0.4523 -0.3587
sig   0.8482  0.9243  0.9684  1.0151  1.1150
```

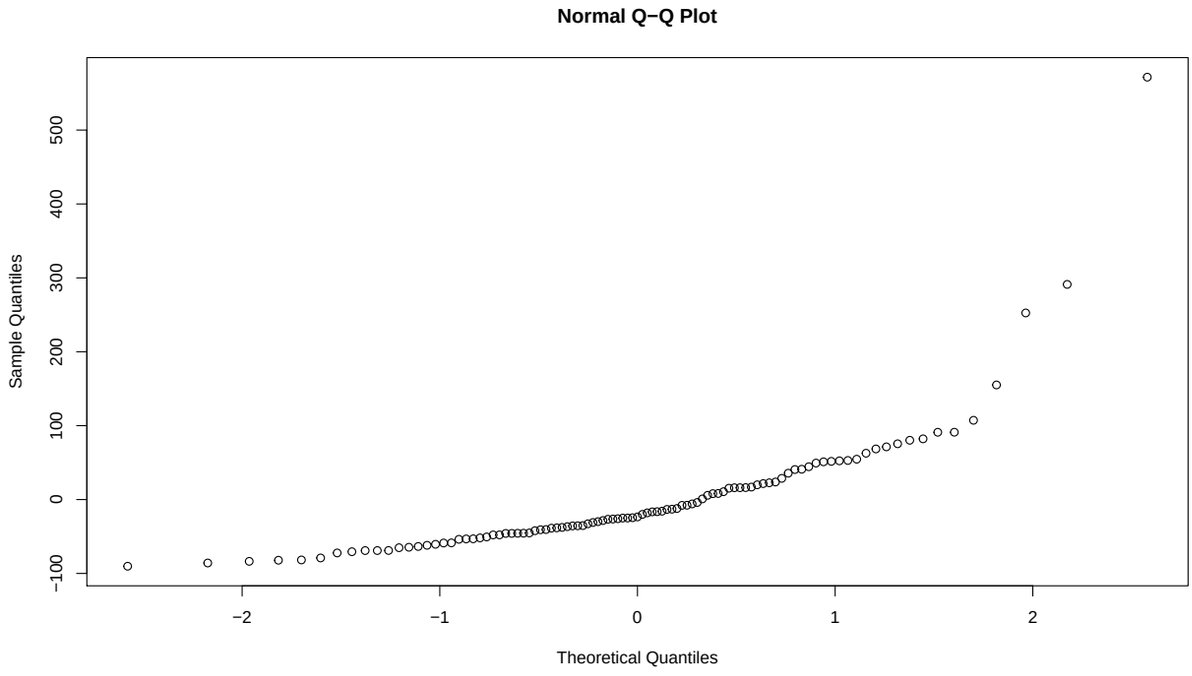
8.3.1 Residual checking

Checking residuals (the difference between the response and the model's prediction for that value) is important with linear models since residuals can reveal violations of the assumptions we made to specify the model. In particular, we are looking for any sign that the model is not linear, normally distributed, or that the observations are not independent (conditional on covariates).

First, let's look at what would have happened if we fit the reference linear model to the un-transformed variables.



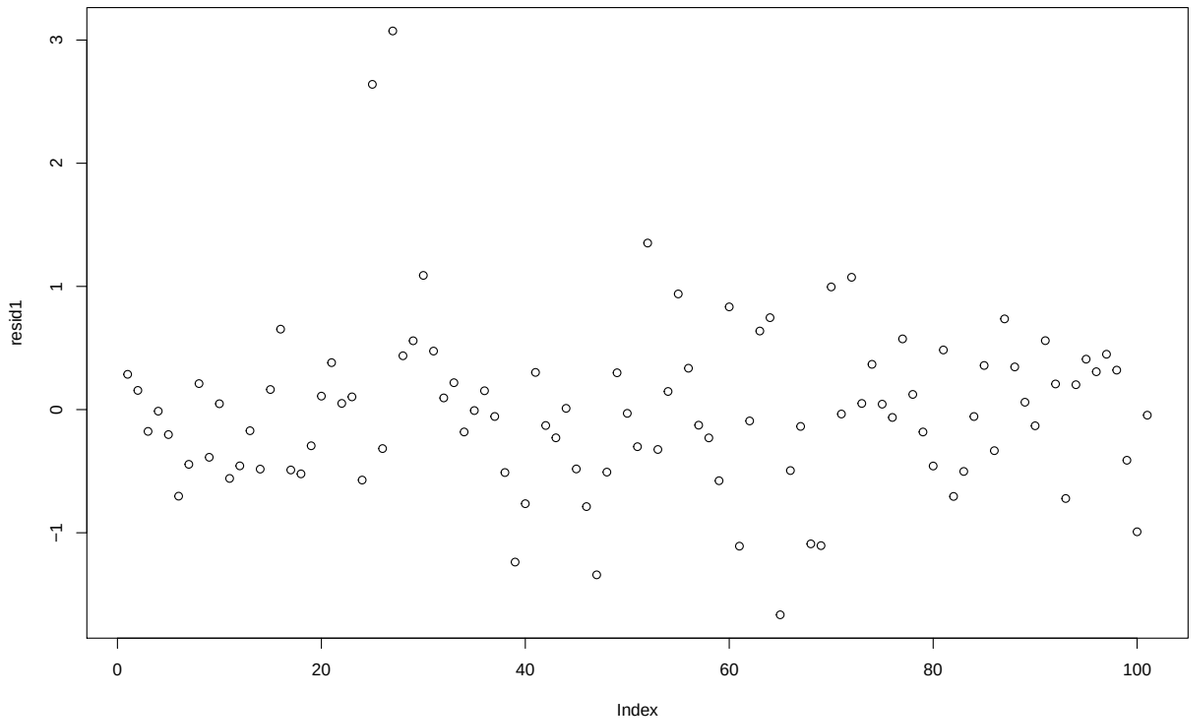


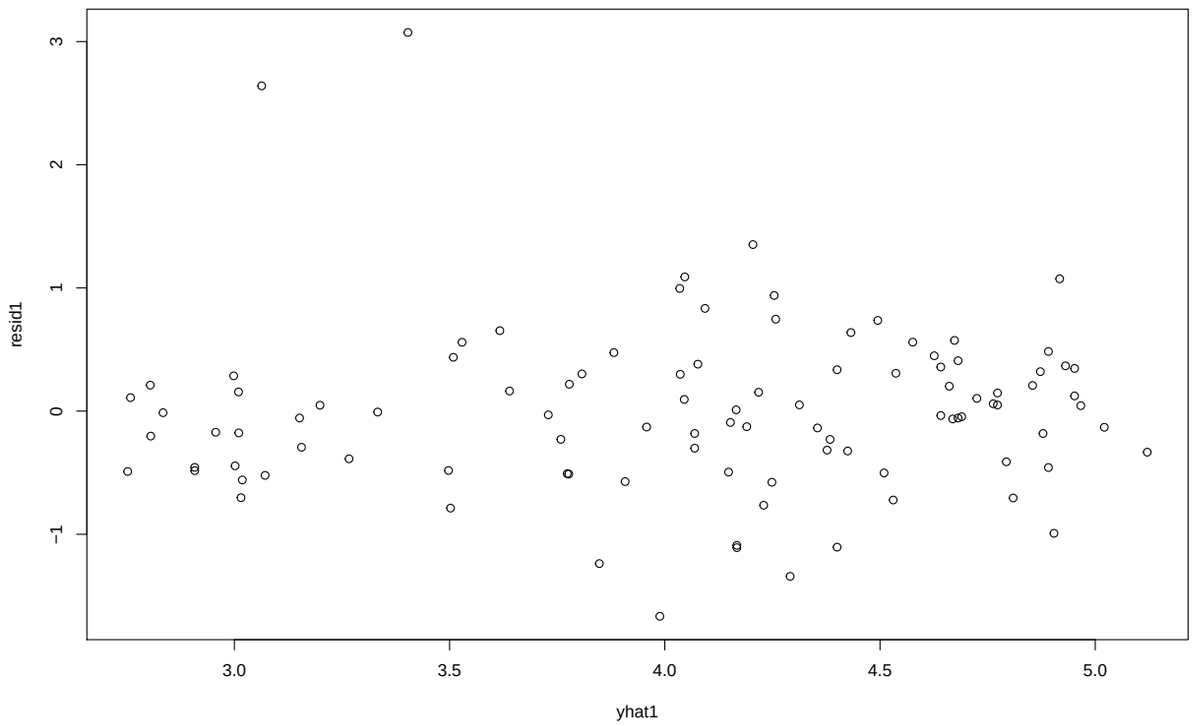


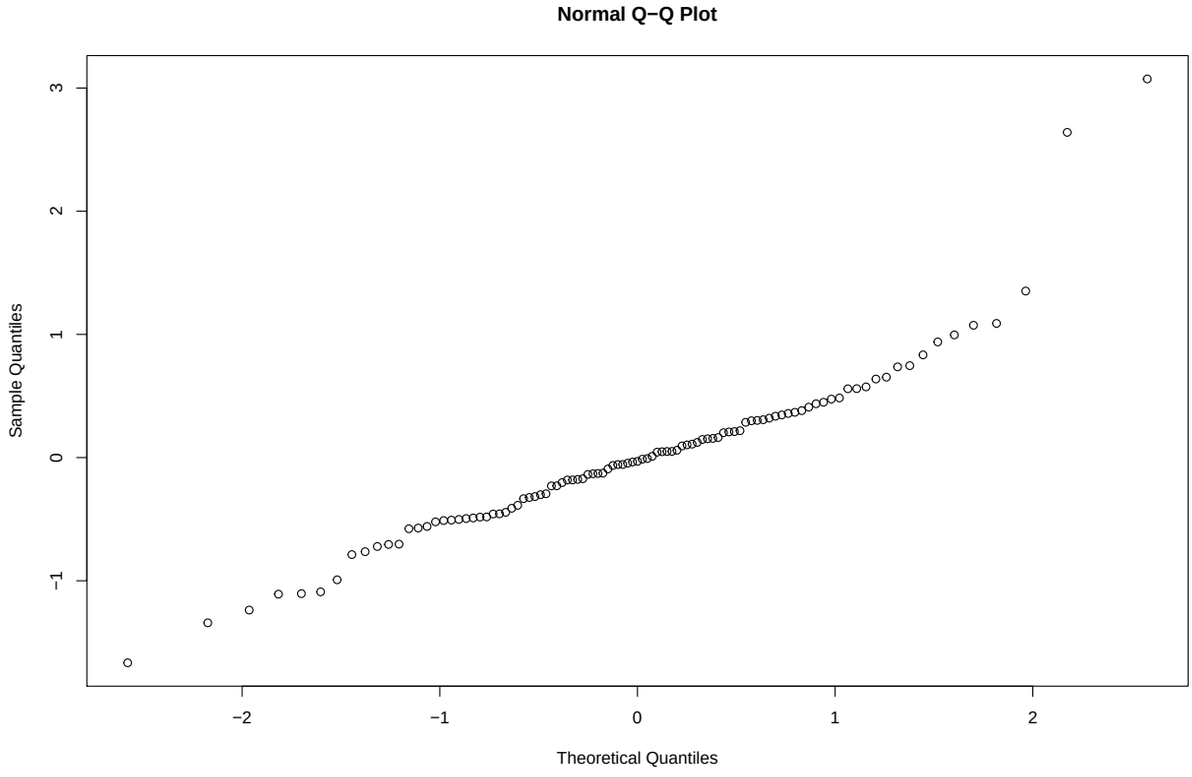
Now let's return to our model fit to the log-transformed variables. In a Bayesian model, we have distributions for residuals, but we'll simplify and look only at the residuals evaluated at the posterior mean of the parameters.

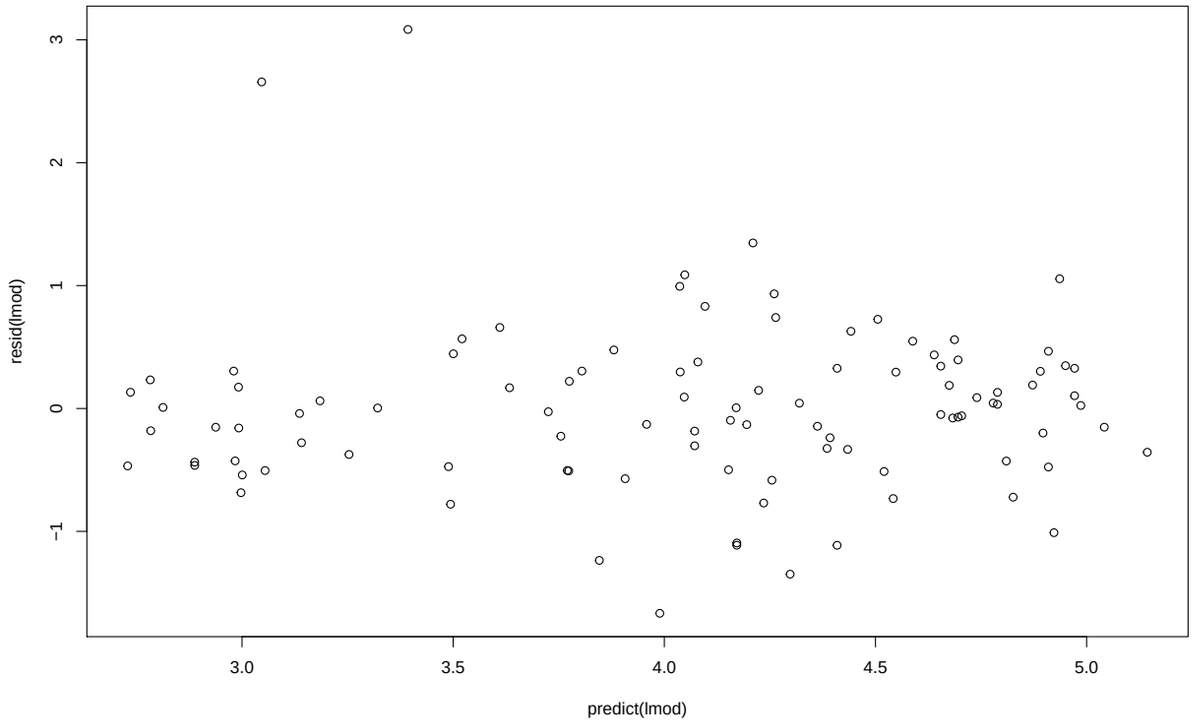
```
     [,1]     [,2]
[1,]    1 8.139149
[2,]    1 8.116716
[3,]    1 8.115521
[4,]    1 8.466110
[5,]    1 8.522976
[6,]    1 8.105308
```

```
      b[1]       b[2]        sig
 7.0852714 -0.5021251  0.9717984
```


resid1
3
2
1
0
−1
0
20
40
60
80
100
Index


resid1
3
2
1
0
−1
3.0
3.5
4.0
4.5
5.0
yhat1


Normal Q−Q Plot
Sample Quantiles
Theoretical Quantiles
−2
−1
0
1
2
3


resid(lmod)
predict(lmod)
3.0
3.5
4.0
4.5
5.0
−1
0
1
2
3

```
[1] "Saudi.Arabia" "Libya"        "Zambia"       "Brazil"
[5] "Afganistan"
```

The residuals look pretty good here (no patterns, shapes) except for two strong outliers, Saudi Arabia and Libya. When outliers appear, it is a good idea to double check that they are not just errors in data entry. If the values are correct, you may reconsider whether these data points really are representative of the data you are trying to model. If you conclude that they are not (for example, they were recorded on different years), you may be able to justify dropping these data points from the data set.

If you conclude that the outliers are part of data and should not be removed, we have several modeling options to accommodate them. We will address these in the next segment.

8.3.2 Adding covariates

The first approach is to look for additional covariates that may be able to explain the outliers. For example, there could be a number of variables that provide information about infant mortality above and beyond what income provides.

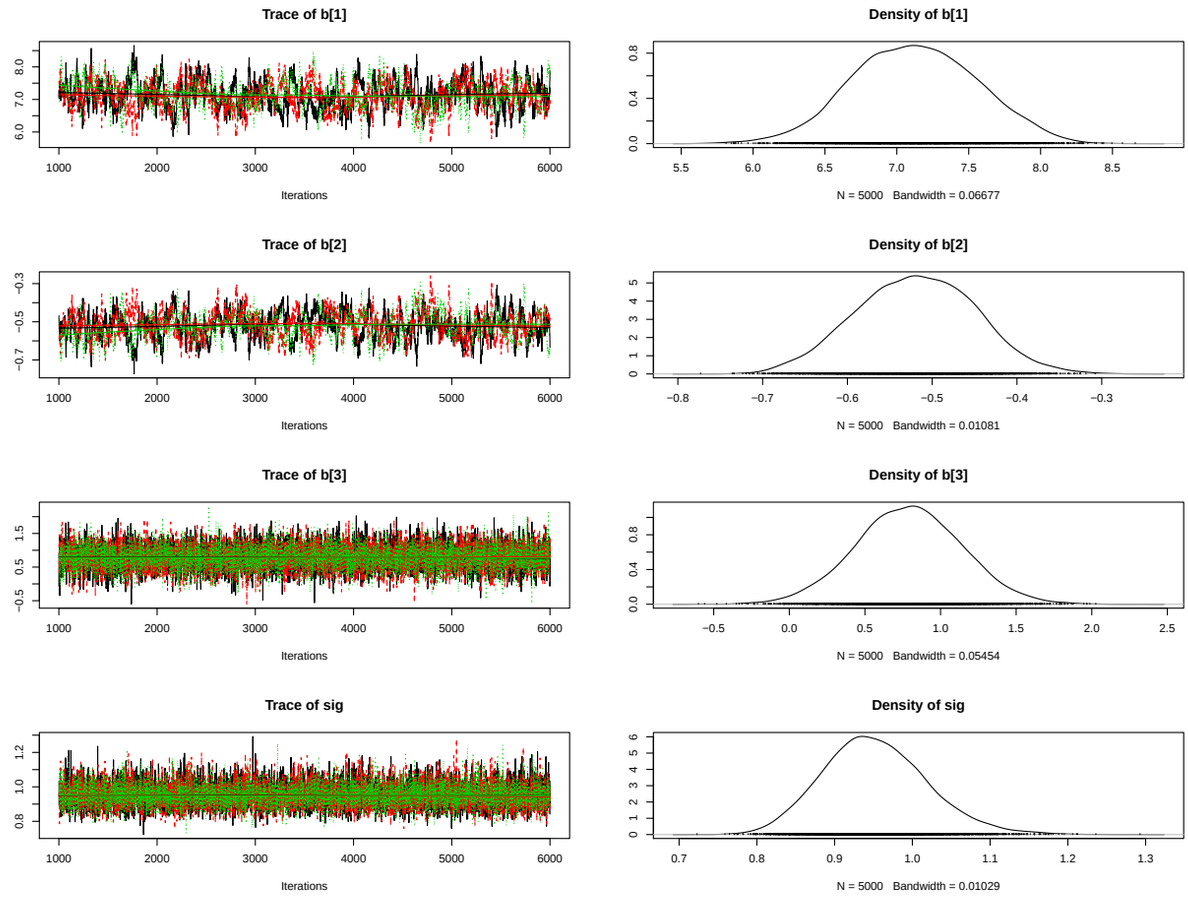
Looking back at our data, there are two variables we haven't used yet: region and oil. The oil variable indicates oil-exporting countries. Both Saudi Arabia and Libya are oil-exporting countries, so perhaps this might explain part of the anomaly.

```
  [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0
 [36] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 [71] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
```

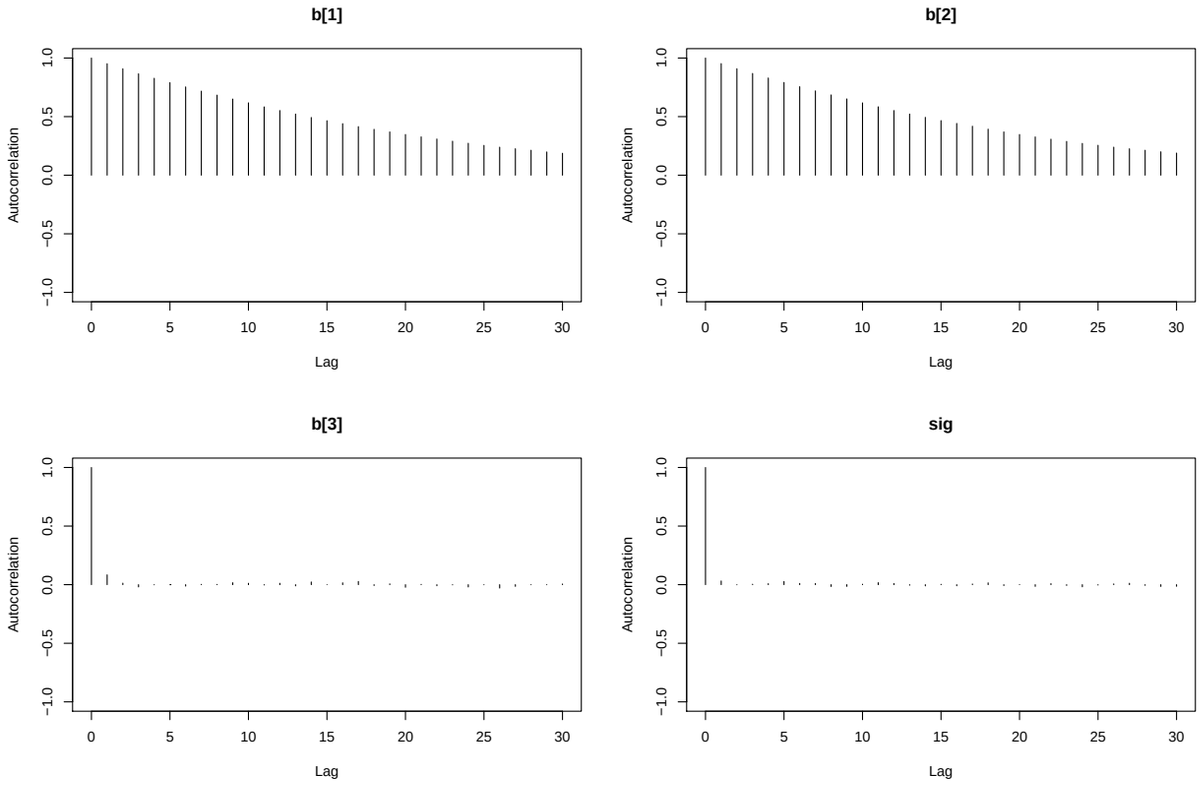
```
Observed stochastic nodes: 101
Unobserved stochastic nodes: 4
Total graph size: 517

Initializing model
```

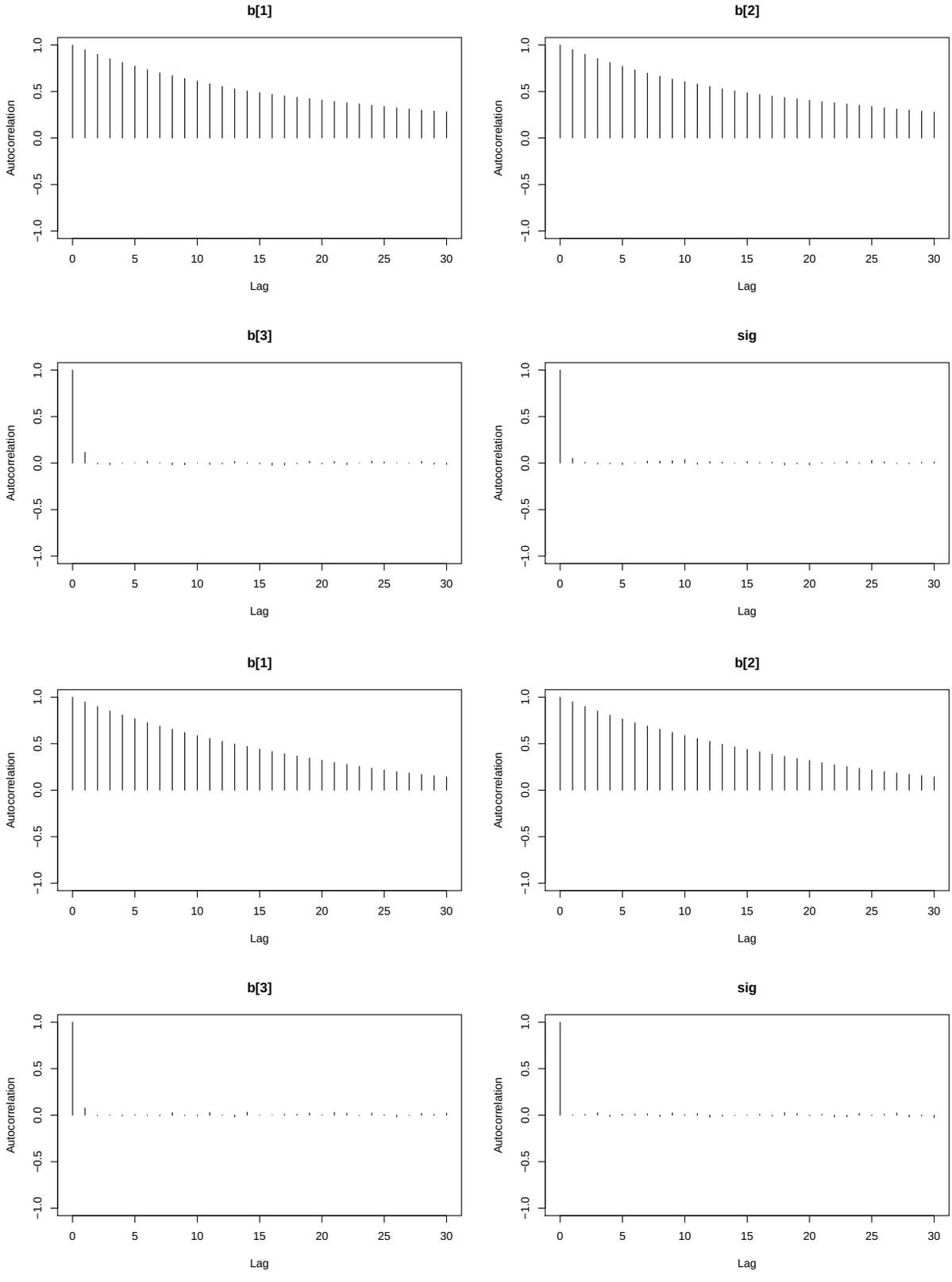
As usual, check the convergence diagnostics.




```
Potential scale reduction factors:

     Point est. Upper C.I.
b[1]          1          1
b[2]          1          1
b[3]          1          1
```

```
sig           1           1

Multivariate psrf

1

             b[1]      b[2]         b[3]         sig
Lag 0   1.0000000 1.0000000  1.000000000  1.000000000
Lag 1   0.9496535 0.9508186  0.092109639  0.029141671
Lag 5   0.7771381 0.7761333  0.001369690  0.006708442
Lag 10  0.6056598 0.6034888  0.002132794  0.016336731
Lag 50  0.1164832 0.1177909 -0.015289957 -0.003043256
```




b[1]
b[2]
b[3]
sig
b[1]
b[2]
b[3]
sig
Autocorrelation
Lag
1.0
0.5
0.0
−0.5
−1.0
0
5
10
15
20
25
30

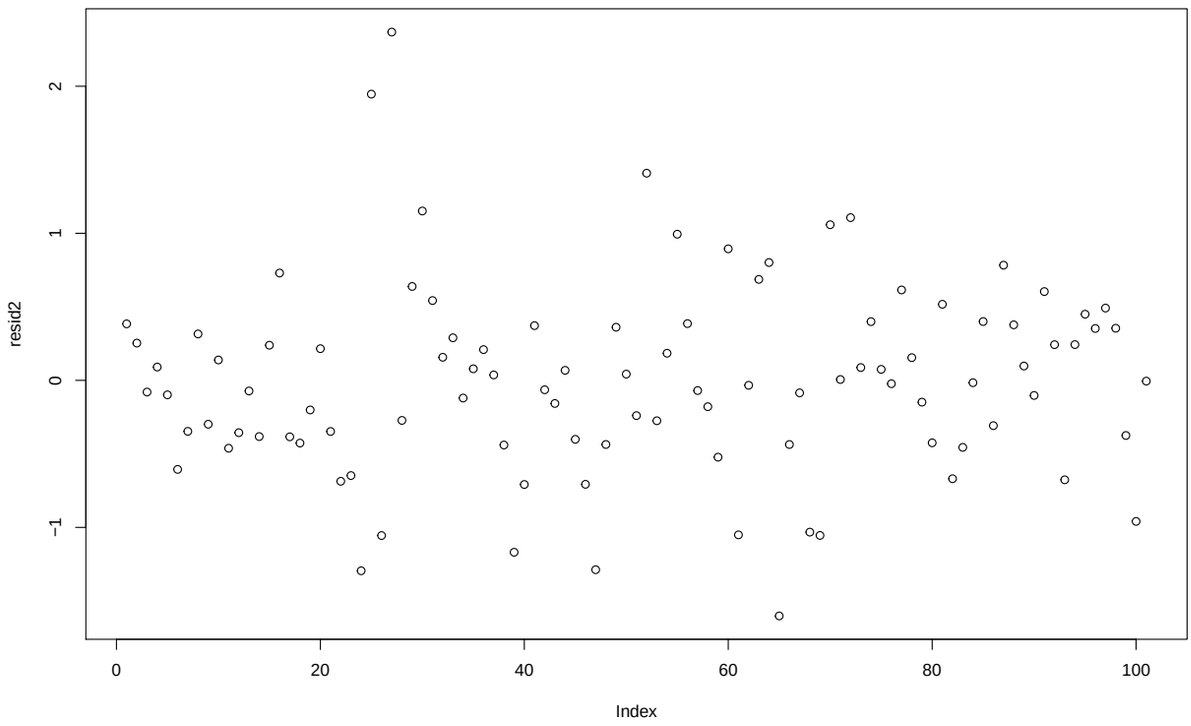
```
      b[1]       b[2]       b[3]        sig
  378.4028   385.5440 12643.0096 14195.7763


Iterations = 1001:6000
Thinning interval = 1
Number of chains = 3
Sample size per chain = 5000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

        Mean      SD  Naive SE Time-series SE
b[1]  7.1270 0.43102 0.0035192      0.0221922
b[2] -0.5193 0.06978 0.0005698      0.0035524
b[3]  0.7896 0.35206 0.0028746      0.0031312
sig   0.9522 0.06715 0.0005483      0.0005641

2. Quantiles for each variable:

         2.5%     25%     50%     75%   97.5%
b[1]  6.29616  6.8244  7.1224  7.4298  7.9613
b[2] -0.65550 -0.5679 -0.5188 -0.4702 -0.3845
b[3]  0.09928  0.5536  0.7911  1.0262  1.4841
sig   0.83213  0.9059  0.9485  0.9949  1.0962
```

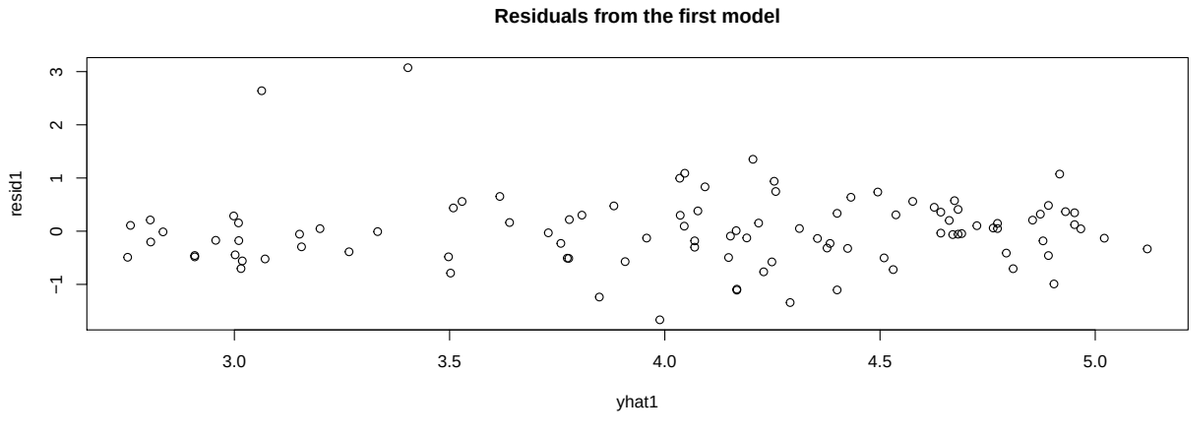
It looks like there is a positive relationship between oil-production and log-infant mortality. Because these data are merely observational, we cannot say that oil-production causes an increase in infant mortality (indeed that most certainly isn't the case), but we can say that they are positively correlated.

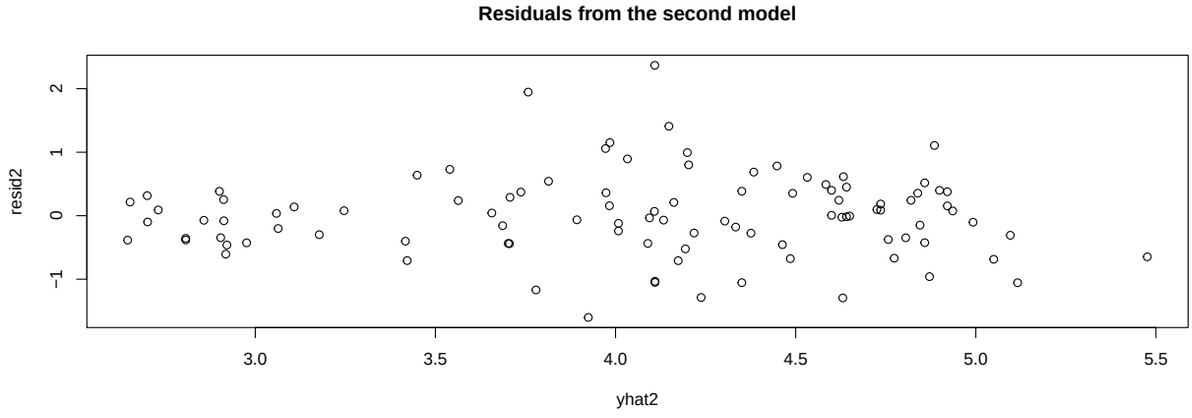
Now let's check the residuals.

```
     [,1]     [,2] [,3]
[1,]    1 8.139149    0
[2,]    1 8.116716    0
[3,]    1 8.115521    0
[4,]    1 8.466110    0
[5,]    1 8.522976    0
[6,]    1 8.105308    0
```

```
      b[1]       b[2]       b[3]       sig
 7.1269622 -0.5192649  0.7896347  0.9522417
```








```
[1] 0.6488744
```

These look much better, although the residuals for Saudi Arabia and Libya are still more than three standard deviations away from the mean of the residuals. We might consider adding the other covariate region, but instead let's look at another option when we are faced with strong outliers.

t likelihood

Let's consider changing the likelihood. The normal likelihood has thin tails (almost all of the probability is concentrated within the first few standard deviations from the mean). This does not accommodate outliers well. Consequently, models with the normal likelihood might be overly-influenced by outliers. Recall that the tt distribution is similar to the normal distribution, but it has thicker tails which can accommodate outliers.

The t linear model might look something like this. Notice that the tt distribution has three parameters, including a positive "degrees of freedom" parameter. The smaller the degrees of freedom, the heavier the tails of the distribution. We might fix the degrees of freedom to some number, or we can assign it a prior distribution.

8.3.3 Compare models using Deviance Information Criterion

We have now proposed three different models. How do we compare their performance on our data? In the previous course, we discussed estimating parameters in models using the maximum likelihood method. Similarly, we can choose between competing models using the same idea.

We will use a quantity known as the deviance information criterion (DIC). It essentially calculates the posterior mean of the log-likelihood and adds a penalty for model complexity.

Let's calculate the DIC for our first two models:

the simple linear regression on log-income,

```
Mean deviance:  231.7
penalty 3.215
Penalized deviance: 234.9
```

```
Mean deviance:  225.2
penalty 3.875
Penalized deviance: 229.1
```

The first number is the Monte Carlo estimated posterior mean deviance, which equals -2 times the log-likelihood (plus a constant that will be irrelevant for comparing models). Because of that -2 factor, a smaller deviance means a higher likelihood.

Next, we are given a penalty for the complexity of our model. This penalty is necessary because we can always increase the likelihood of the model by making it more complex to fit the data exactly. We don't want to do this because over-fit models

generalize poorly. This penalty is roughly equal to the effective number of parameters in your model. You can see this here. With the first model, we had a variance parameter and two betas, for a total of three parameters. In the second model, we added one more beta for the oil effect.

We add these two quantities to get the DIC (the last number). The better-fitting model has a lower DIC value. In this case, the gains we receive in deviance by adding the is_oil covariate outweigh the penalty for adding an extra parameter. The final DIC for the second model is lower than for the first, so we would prefer using the second model.

We encourage you to explore different model specifications and compare their fit to the data using DIC. Wikipedia provides a good introduction to DIC and we can find more details about the JAGS implementation through the rjags package documentation by entering ?dic.samples in the R console.

9 Linear Regression 2nd example

New York City Crime Data

As an example of a Bayesian linear regression model, we look at New York City crime data from 1966 to 1967. The outcome variable (THEFT) is the increase or decrease in the seasonally adjusted rate of grand larcenies in 23 Manhattan police precincts from a 27-week pre-intervention period compared to a 58-week intervention period. The predictor variables are the percent increase or decrease in the number of police officers in a precinct (MAN), and whether the precinct was located uptown, midtown or downtown.

We specify the model as:

$$THEFT_i \sim N(\mu, \sigma^) \mu = \beta_0 + \beta_1 * MAN + district \frac{1}{\sigma^2} \sim \Gamma(0.001, 0.001) \beta_0 \sim N(0, 100000) \beta_1 \sim$$

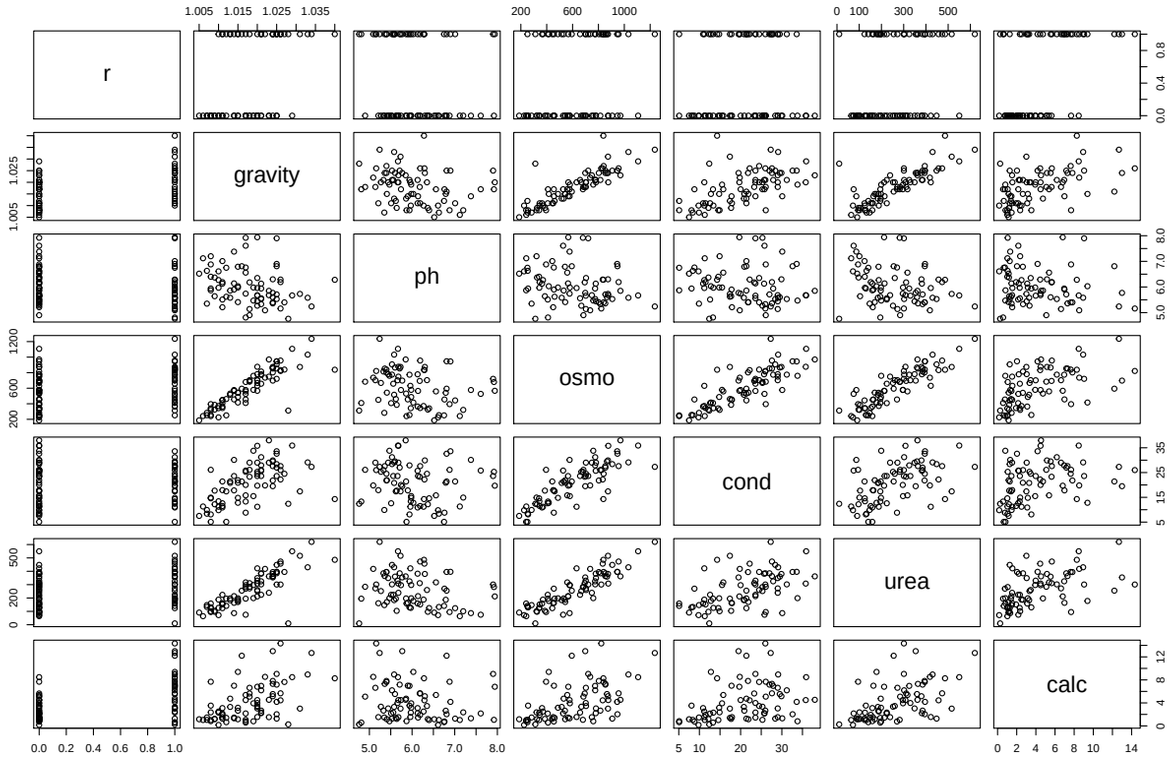
We will need a prior for the effect of geographic area (district), but will discuss that in a moment. First, we consider how best to code an indicator variable in BUGS. There are two possible approaches. We can create a "design matrix" of dummy variables. In this approach we create two variables named DIST1 and DIST2, for which downtown precincts are coded 0,0 , midtown precincts are coded 1,0 and uptown precincts are coded 0,1. The BUGS code for this model would then be:

10 Logistic regression

For an example of logistic regression, we'll use the urine data set from the boot package in R. The response variable is r, which takes on values of 0 or 1. We will remove some rows from the data set which contain missing values.

```
  r gravity   ph osmo cond urea calc
1 0   1.021 4.91  725   NA  443 2.45
2 0   1.017 5.74  577 20.0  296 4.49
3 0   1.008 7.20  321 14.9  101 2.36
4 0   1.011 5.51  408 12.6  224 2.15
5 0   1.005 6.52  187  7.5   91 1.16
6 0   1.020 5.27  668 25.3  252 3.34
```

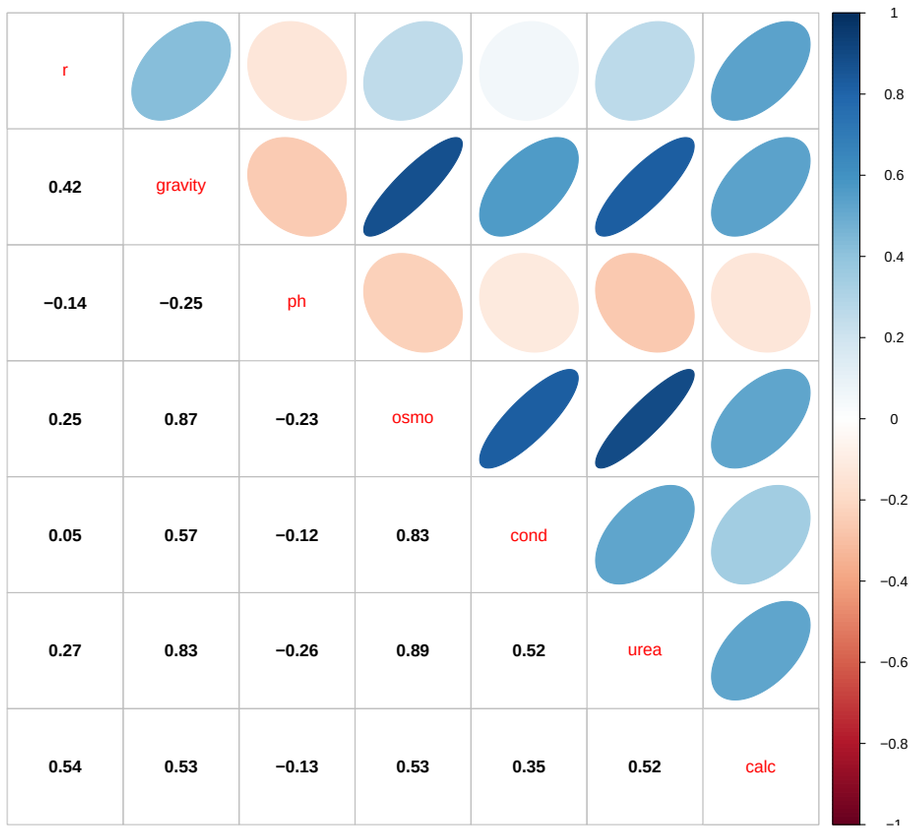
Let's look at pairwise scatter plots of the seven variables.



One thing that stands out is that several of these variables are strongly correlated

with one another. For example gravity and osmo appear to have a very close linear relationship. Collinearity between xx variables in linear regression models can cause trouble for statistical inference. Two correlated variables will compete for the ability to predict the response variable, leading to unstable estimates. This is not a problem for prediction of the response, if prediction is the end goal of the model. But if our objective is to discover how the variables relate to the response, we should avoid collinearity.

We can more formally estimate the correlation among these variables using the corrplot package.



10.0.1 Variable selection

One primary goal of this analysis is to find out which variables are related to the presence of calcium oxalate crystals. This objective is often called "variable selection." We have already seen one way to do this: fit several models that include different sets

of variables and see which one has the best DIC. Another way to do this is to use a linear model where the priors for the β coefficients favor values near 0 (indicating a weak relationship). This way, the burden of establishing association lies with the data. If there is not a strong signal, we assume it doesn't exist.

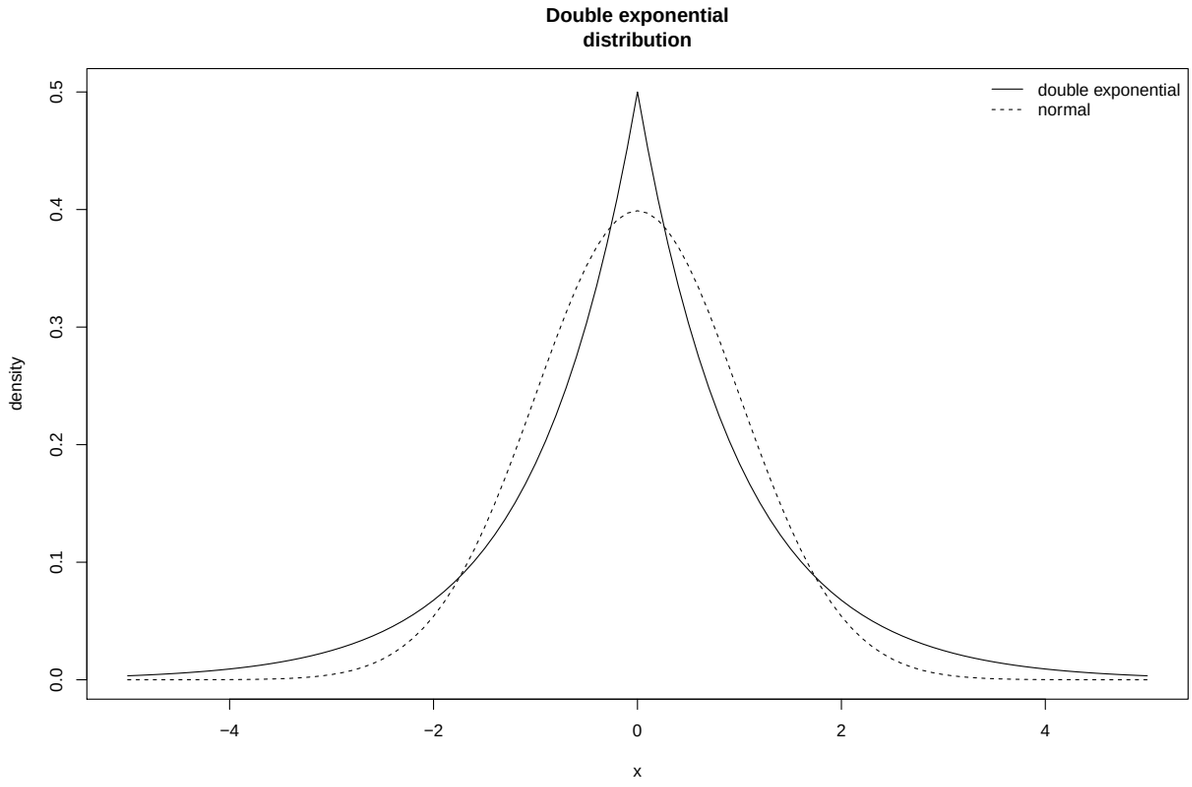
Rather than tailoring a prior for each individual β based on the scale its covariate takes values on, it is customary to subtract the mean and divide by the standard deviation for each variable.

```
         2          3          4          5          6          7
-0.1403037 -1.3710690 -0.9608139 -1.7813240  0.2699514 -0.8240622

      gravity           ph         osmo         cond         urea
-9.861143e-15  8.511409e-17  1.515743e-16 -1.829852e-16  7.335402e-17
         calc
-1.689666e-18
```

```
gravity      ph    osmo    cond    urea    calc
      1       1       1       1       1       1
```

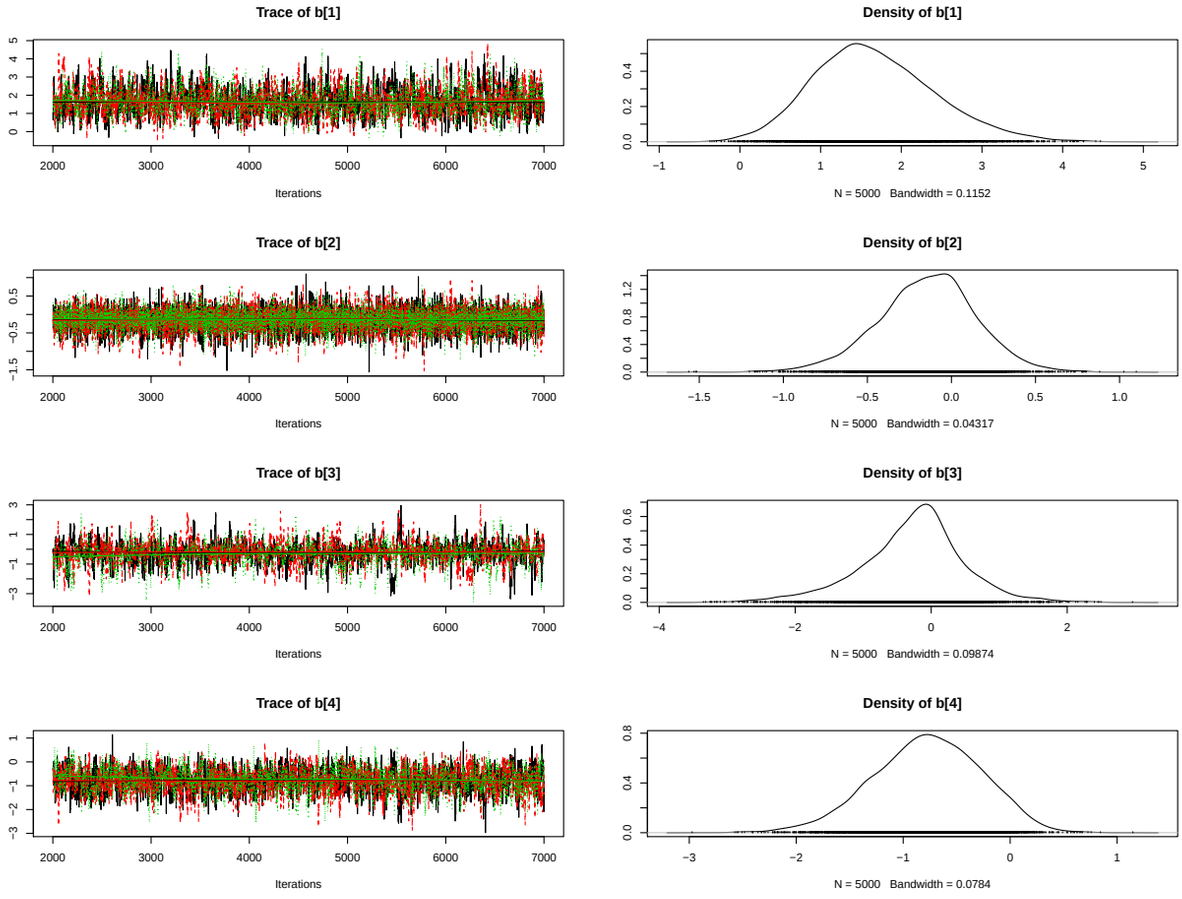
10.0.2 Model

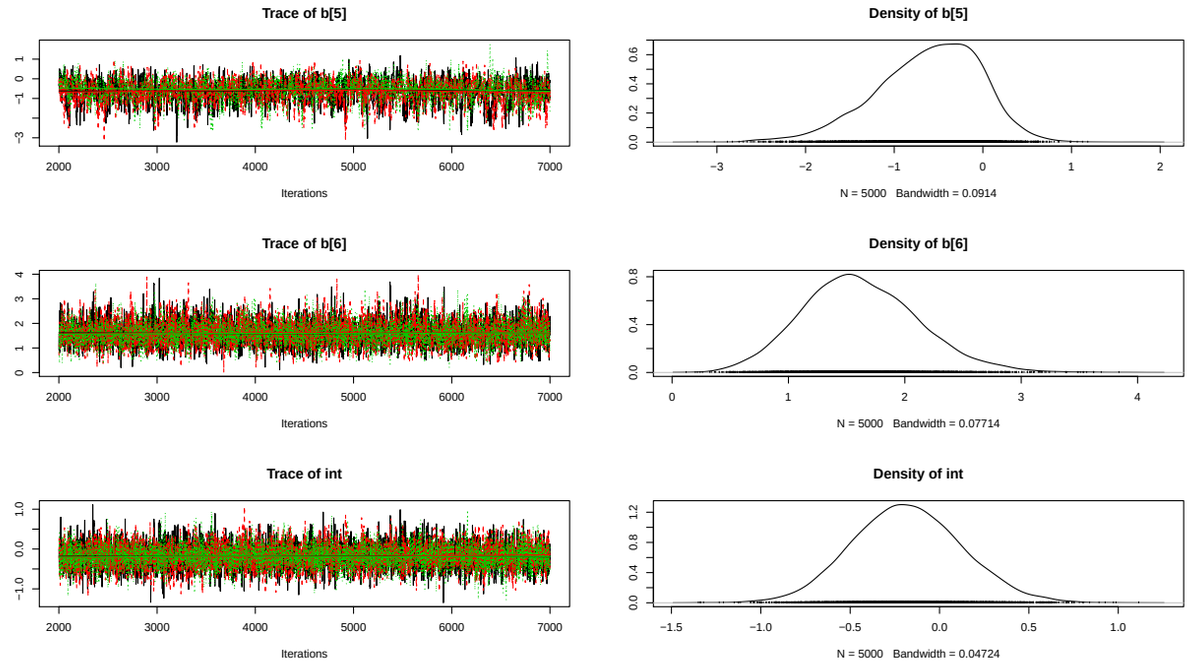
Our prior for the β (which we'll call bb in the model) coefficients will be the double exponential (or Laplace) distribution, which as the name implies, is the exponential distribution with tails extending in the positive direction as well as the negative direction, with a sharp peak at 0. We can read more about it in the JAGS manual. The distribution looks like:




```
     gravity         ph       osmo       cond        urea        calc
2 -0.1403037 -0.4163725 -0.1528785 -0.1130908  0.25747827  0.09997564
3 -1.3710690  1.6055972 -1.2218894 -0.7502609 -1.23693077 -0.54608444
4 -0.9608139 -0.7349020 -0.8585927 -1.0376121 -0.29430353 -0.60978050
5 -1.7813240  0.6638579 -1.7814497 -1.6747822 -1.31356713 -0.91006194
6  0.2699514 -1.0672806  0.2271214  0.5490664 -0.07972172 -0.24883614
7 -0.8240622 -0.5825618 -0.6372741 -0.4379226 -0.51654898 -0.83726644

Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 77
   Unobserved stochastic nodes: 7
   Total graph size: 1096
```

```
Initializing model
```



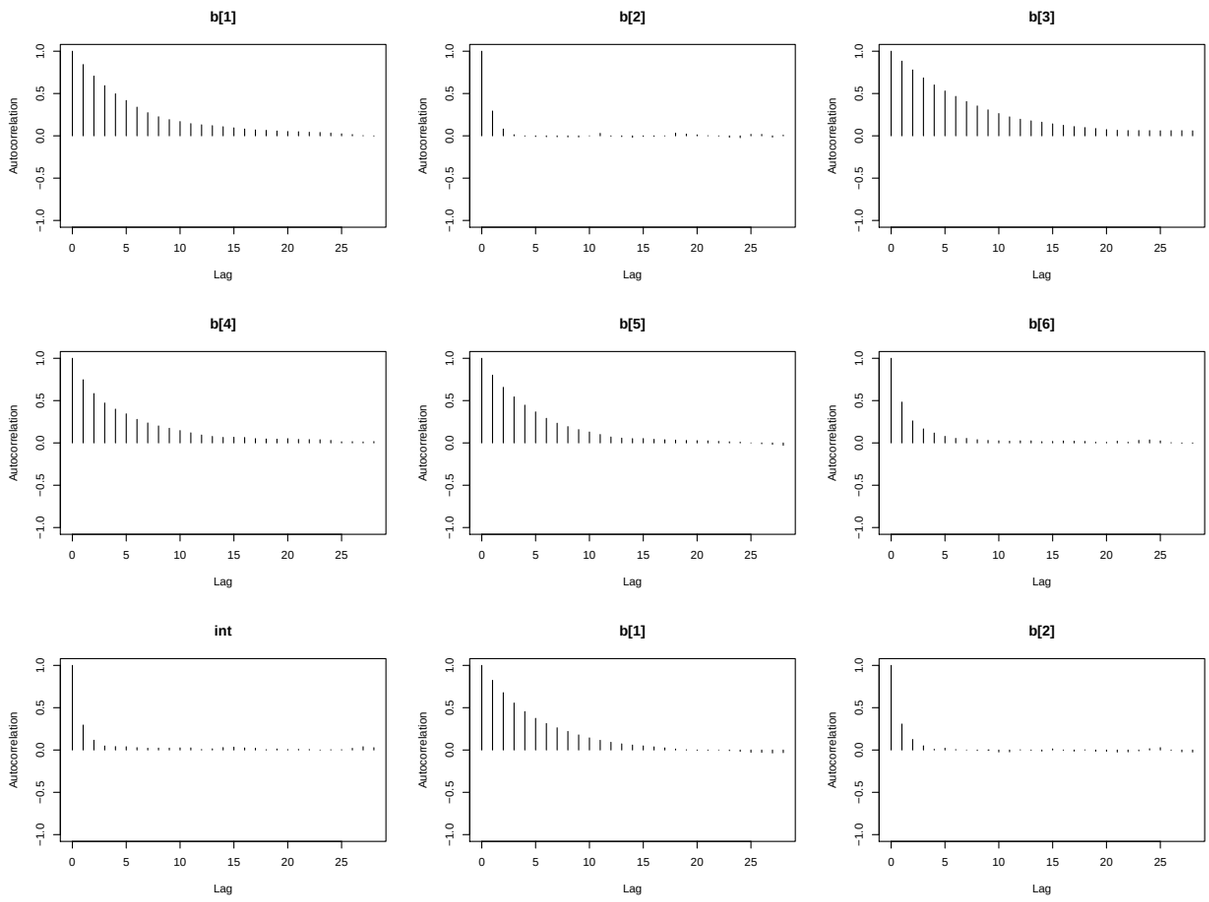


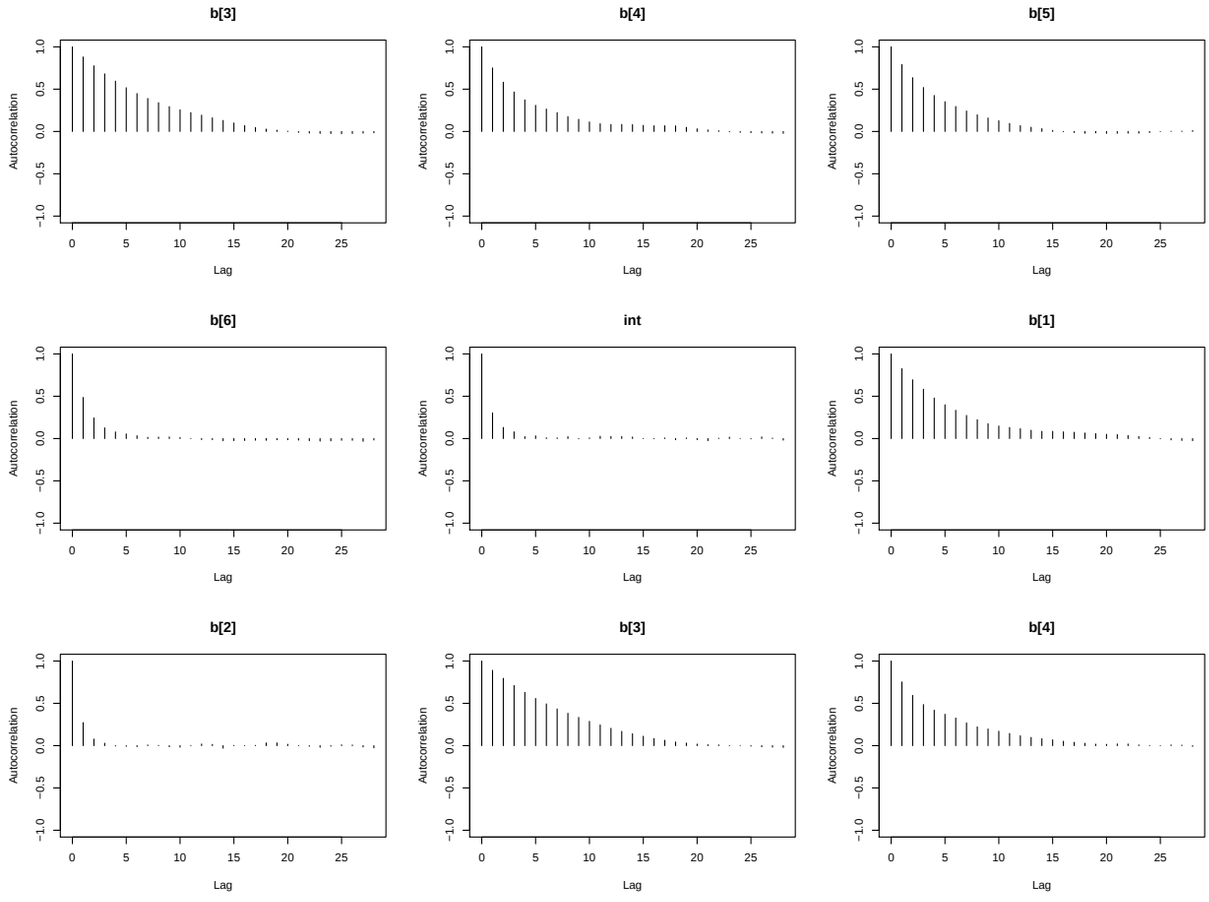

```
Potential scale reduction factors:

     Point est. Upper C.I.
b[1]          1       1.00
b[2]          1       1.00
b[3]          1       1.01
b[4]          1       1.00
b[5]          1       1.01
b[6]          1       1.00
int           1       1.00

Multivariate psrf
```

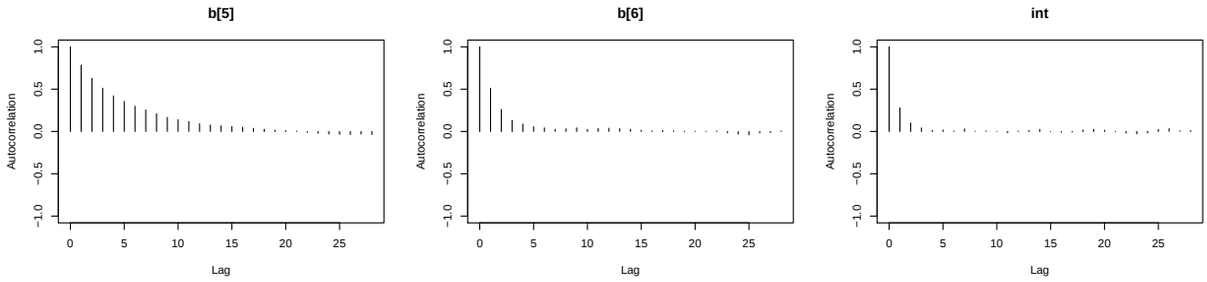
1

	b[1]	b[2]	b[3]	b[4]	b[5]
Lag 0	1.000000000	1.000000000	1.00000000	1.000000000	1.00000000
Lag 1	0.830663471	0.290961453	0.88433832	0.749981741	0.79155928
Lag 5	0.396923691	0.001814541	0.53506430	0.341079775	0.35823885
Lag 10	0.153395212	-0.013863391	0.26891655	0.142911892	0.13236002
Lag 50	0.004162061	-0.007245810	-0.02189131	0.006198761	0.03683274

	b[6]	int
Lag 0	1.000000000	1.000000000
Lag 1	0.493217504	0.292377945
Lag 5	0.062750118	0.029781480
Lag 10	0.020818304	0.011639405
Lag 50	-0.003655699	-0.007353806




b[3]
b[4]
b[5]
b[6]
int
b[1]
b[2]
b[3]
b[4]
Autocorrelation
Lag

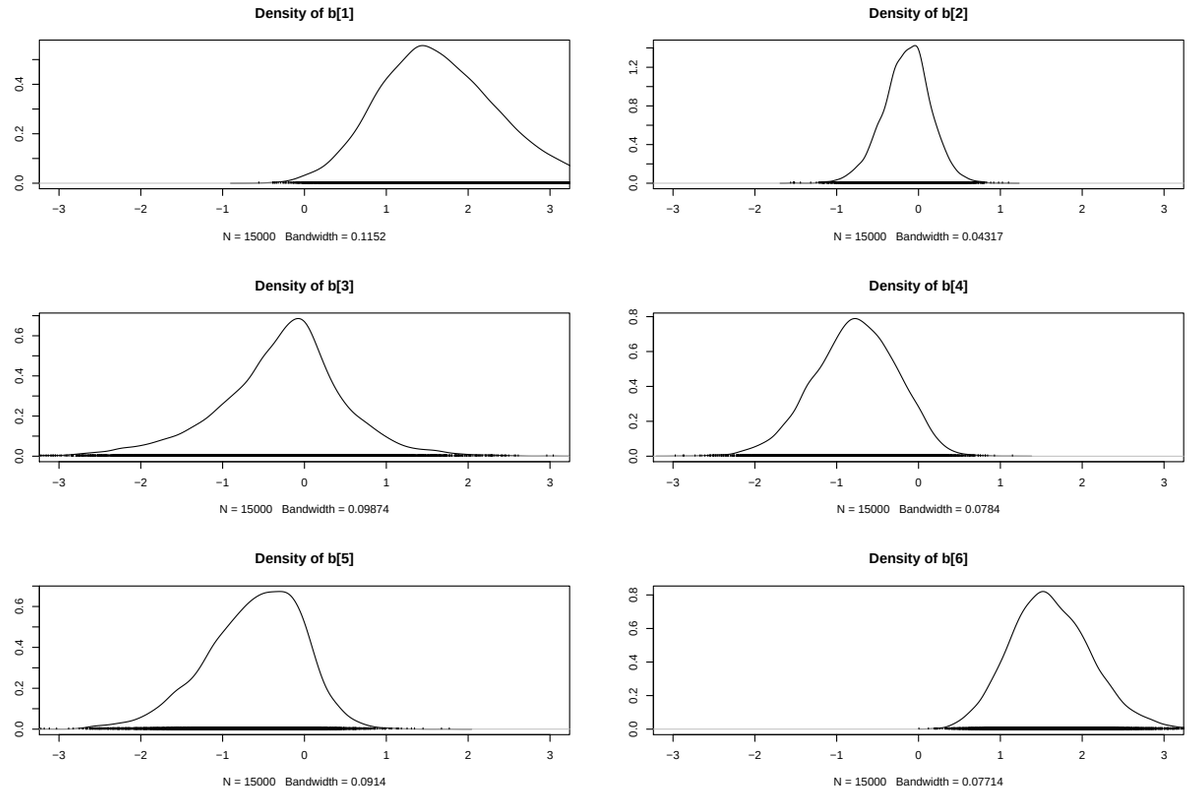



```
     b[1]       b[2]       b[3]       b[4]       b[5]       b[6]        int
1383.5098 8068.1358  920.6007 1523.7652 1529.3297 4790.0415 7493.4092
```

Let's look at the results.

```
Iterations = 2001:7000
Thinning interval = 1
Number of chains = 3
Sample size per chain = 5000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:
```

```
        Mean     SD Naive SE Time-series SE
b[1]  1.6661 0.7469 0.006098       0.020135
b[2] -0.1458 0.2940 0.002400       0.003282
b[3] -0.2800 0.7634 0.006233       0.025109
b[4] -0.7763 0.5061 0.004132       0.012976
b[5] -0.6158 0.5900 0.004818       0.015062
b[6]  1.6186 0.4981 0.004067       0.007207
int  -0.1810 0.3060 0.002498       0.003543

2. Quantiles for each variable:

        2.5%     25%     50%      75%  97.5%
b[1]  0.3610  1.1412  1.6039  2.13725 3.2824
b[2] -0.7548 -0.3286 -0.1328  0.04488 0.4158
b[3] -2.0083 -0.6944 -0.2131  0.15963 1.1849
b[4] -1.8185 -1.1080 -0.7629 -0.42363 0.1422
b[5] -1.9039 -0.9862 -0.5558 -0.18493 0.3831
b[6]  0.7109  1.2722  1.5871  1.93945 2.6770
int  -0.7626 -0.3885 -0.1868  0.02019 0.4294
```




```
[1] "gravity" "ph"      "osmo"    "cond"    "urea"    "calc"
```

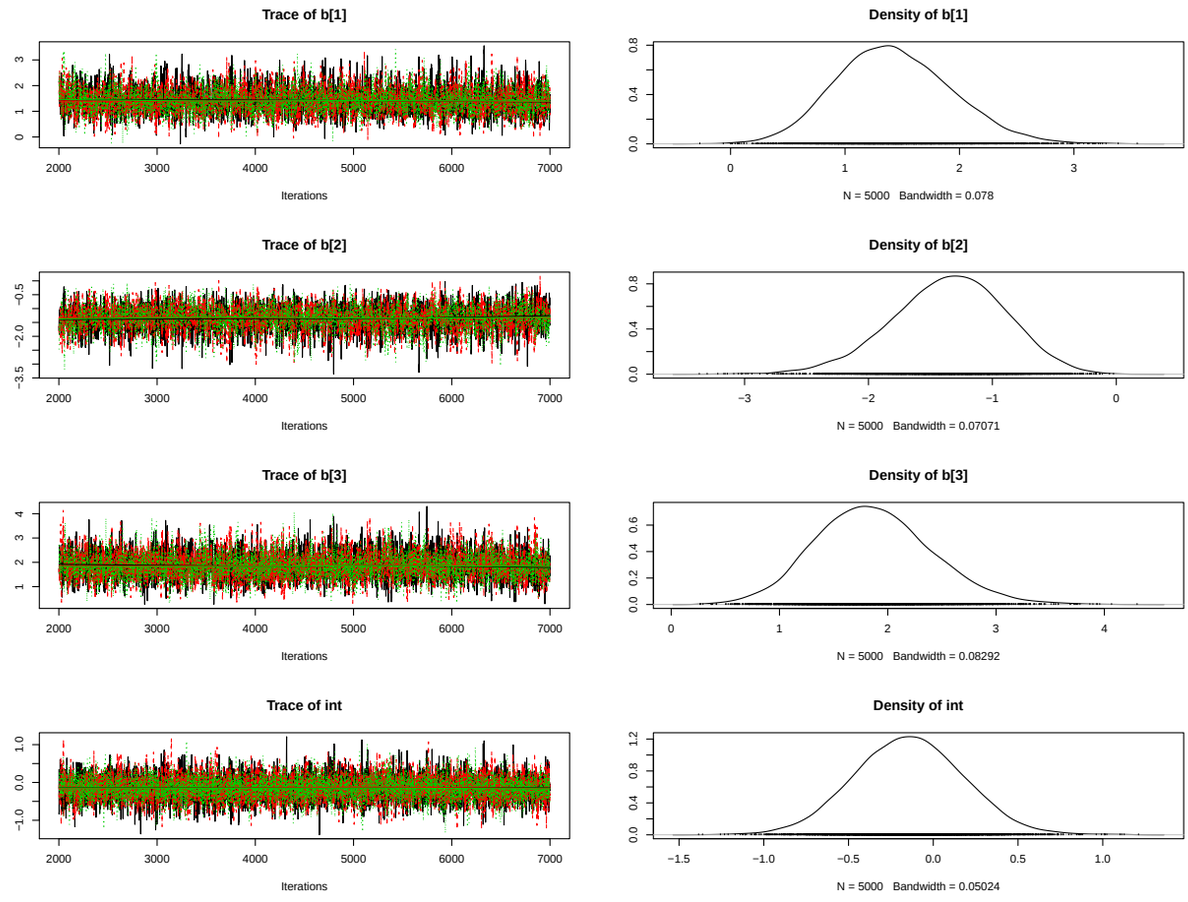
It is clear that the coefficients for variables gravity, cond (conductivity), and calc (calcium concentration) are not 0. The posterior distribution for the coefficient of osmo (osmolarity) looks like the prior, and is almost centered on 0 still, so we'll conclude that osmo is not a strong predictor of calcium oxalate crystals. The same goes for ph.

urea (urea concentration) appears to be a borderline case. However, if we refer back to our correlations among the variables, we see that urea is highly correlated with gravity, so we opt to remove it.

Our second model looks like this:

```
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
```

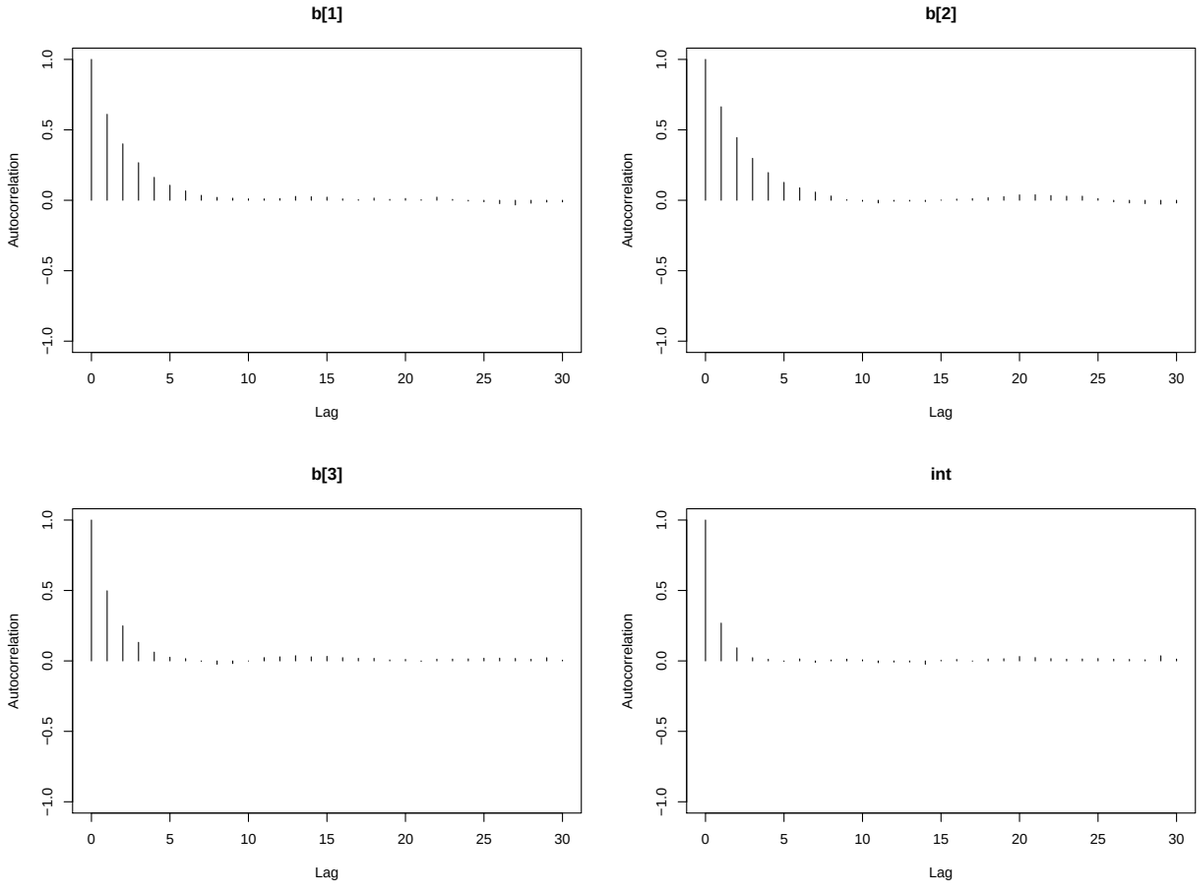
```
Graph information:
   Observed stochastic nodes: 77
   Unobserved stochastic nodes: 4
   Total graph size: 644

Initializing model
```

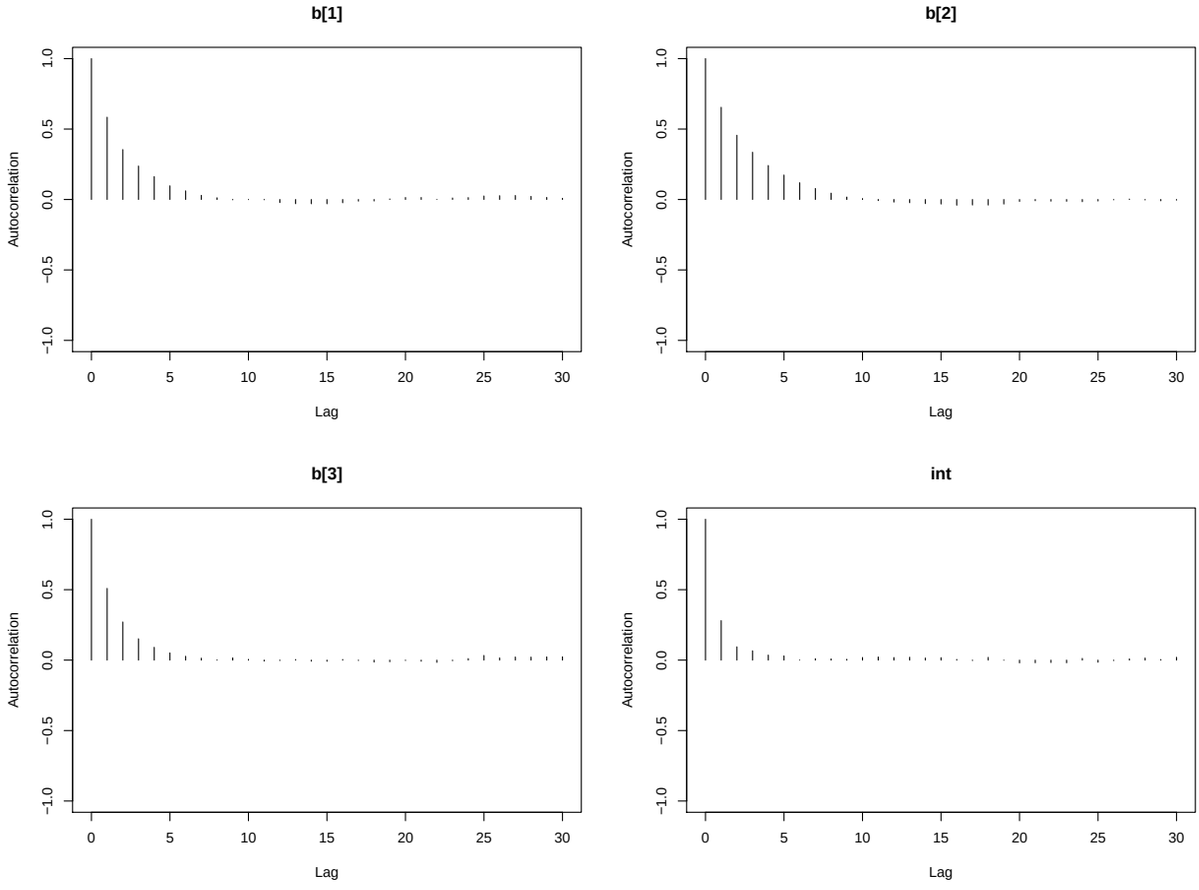


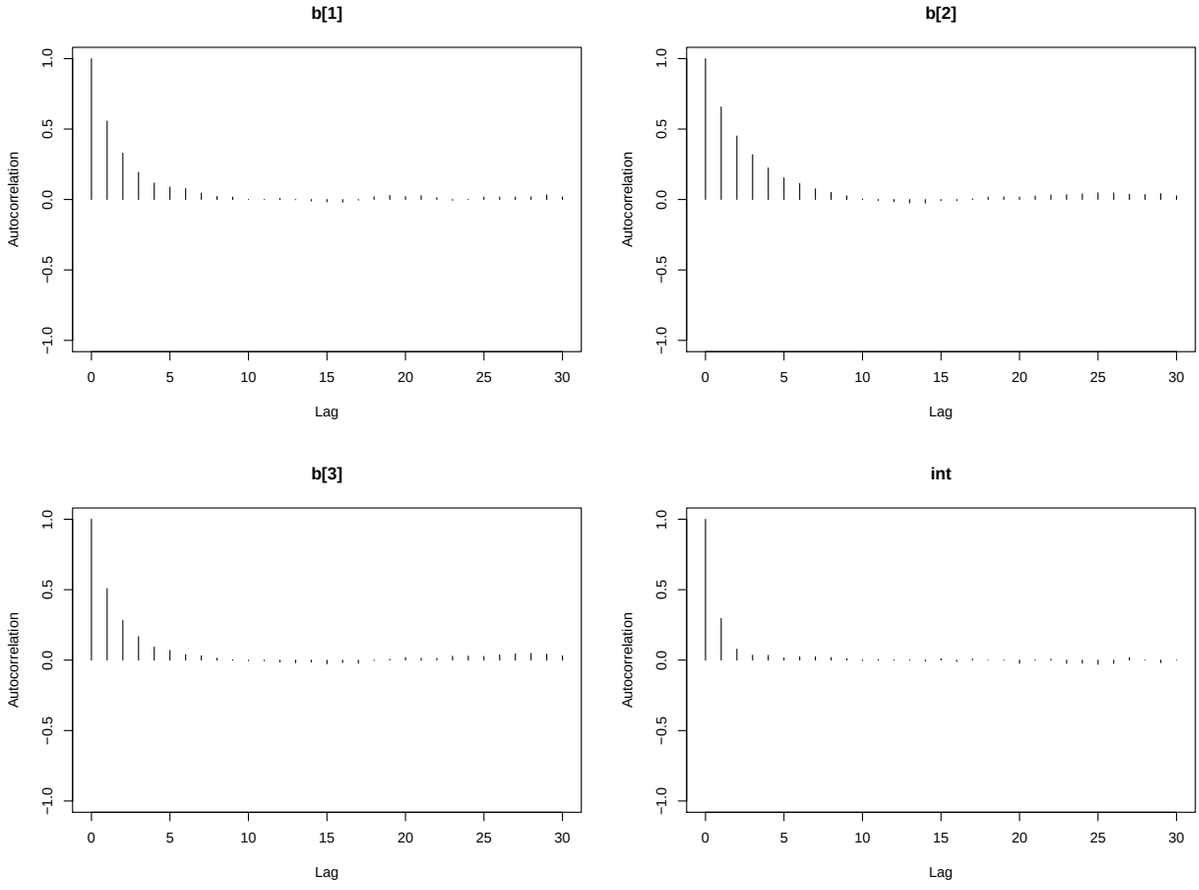

```
Potential scale reduction factors:

     Point est. Upper C.I.
b[1]          1          1
b[2]          1          1
b[3]          1          1
int           1          1
```

```
Multivariate psrf

1
```

	b[1]	b[2]	b[3]	int
Lag 0	1.000000000	1.000000000	1.0000000000	1.000000000
Lag 1	0.583629166	0.658218980	0.5049672064	0.281485847
Lag 5	0.097222129	0.151275621	0.0488905589	0.014657960
Lag 10	0.003341648	0.001116155	0.0000819758	0.007385288
Lag 50	0.001312952	0.004687680	0.0054379963	0.004808176




b[1]
b[2]
b[3]
int
Autocorrelation
Lag
1.0
0.5
0.0
−0.5
−1.0
0
5
10
15
20
25
30

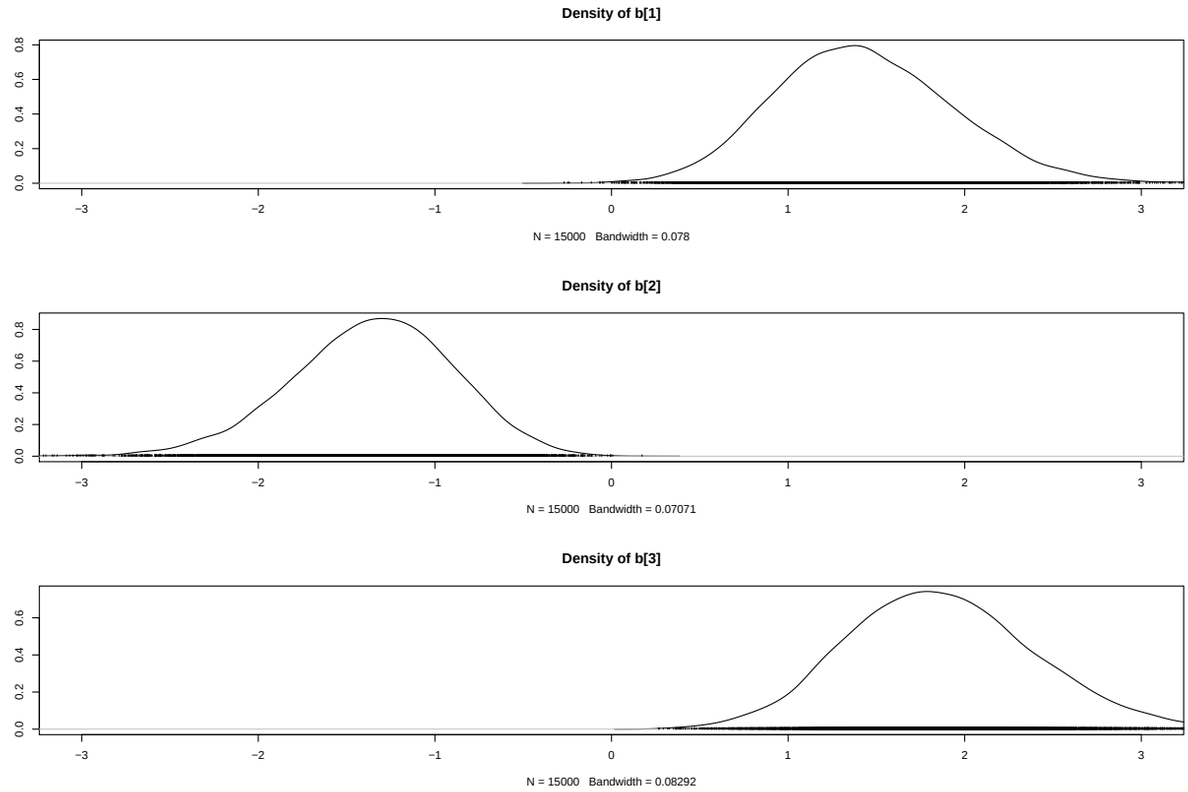



```
    b[1]      b[2]      b[3]       int
3630.462 2862.668 4832.014 7984.114
```

10.0.3 Results

```
Mean deviance:  68.37
penalty 5.6
Penalized deviance: 73.97

Mean deviance:  71.19
penalty 4.058
Penalized deviance: 75.25
```

```
Iterations = 2001:7000
Thinning interval = 1
Number of chains = 3
Sample size per chain = 5000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

        Mean     SD Naive SE Time-series SE
b[1]  1.4299 0.5035 0.004111       0.008398
b[2] -1.3535 0.4577 0.003737       0.008563
b[3]  1.8703 0.5390 0.004401       0.007760
int  -0.1443 0.3246 0.002650       0.003635

2. Quantiles for each variable:

        2.5%     25%     50%      75%   97.5%
b[1]  0.5119  1.0786  1.4037  1.75659  2.4781
b[2] -2.3148 -1.6477 -1.3331 -1.03611 -0.5077
b[3]  0.8833  1.4943  1.8422  2.21153  3.0087
int  -0.7720 -0.3636 -0.1459  0.07098  0.4942

          lower      upper
b[1]  0.4627420  2.4148939
b[2] -2.2710840 -0.4747041
b[3]  0.8319319  2.9463279
int  -0.7888715  0.4753254
attr(,"Probability")
[1] 0.95
```




```
[1] "gravity" "cond"    "calc"
```

The DIC is actually better for the first model. Note that we did change the prior between models, and generally we should not use the DIC to choose between priors. Hence comparing DIC between these two models may not be a fair comparison. Nevertheless, they both yield essentially the same conclusions. Higher values of gravity and calc (calcium concentration) are associated with higher probabilities of calcium oxalate crystals, while higher values of cond (conductivity) are associated with lower probabilities of calcium oxalate crystals. There are more modeling options in this scenario, perhaps including transformations of variables, different priors, and interactions between the predictors, but we'll leave it to you to see if you can improve the model.

10.0.4 Prediction from a logisic regression model

How do we turn model parameter estimates into model predictions? The key is the form of the model. Remember that the likelihood is Bernoulli, which is 1 with probability pp. We modeled the logit of pp as a linear model, which we showed in the first segment of this lesson leads to an exponential form for E(y)=p(y)=p.

Take the output from our model in the last segment. We will use the posterior means as point estimates of the parameters.

```
      b[1]        b[2]        b[3]         int
 1.4299043 -1.3535443  1.8703223 -0.1442578
```

The posterior mean of the intercept was about -0.15. Since we centered and scaled all of the covariates, values of 0 for each xx correspond to the average values. Therefore, if we use our last model, then our point estimate for the probability of calcium oxalate crystals when gravity, cond, and calc are at their average values is 1/(1+e^{-(-0.15)})== 0.4625702.

Now suppose we want to make a prediction for a new specimen whose value of gravity is average, whose value of cond is one standard deviation below the mean, and whose value of calc is one standard deviation above the mean. Our point estimate for the probability of calcium oxalate crystals is 1/(1+e^{-(-0.15 + 1.4*0.0* - *1.3*(-1.0) + 1.9*(1.0))})= 0.9547825.

If we want to make predictions in terms of the original xx variable values, we have two options:

1. For each x variable, subtract the mean and divide by the standard deviation for that variable in the original data set used to fit the model.
2. Re-fit the model without centering and scaling the covariates.

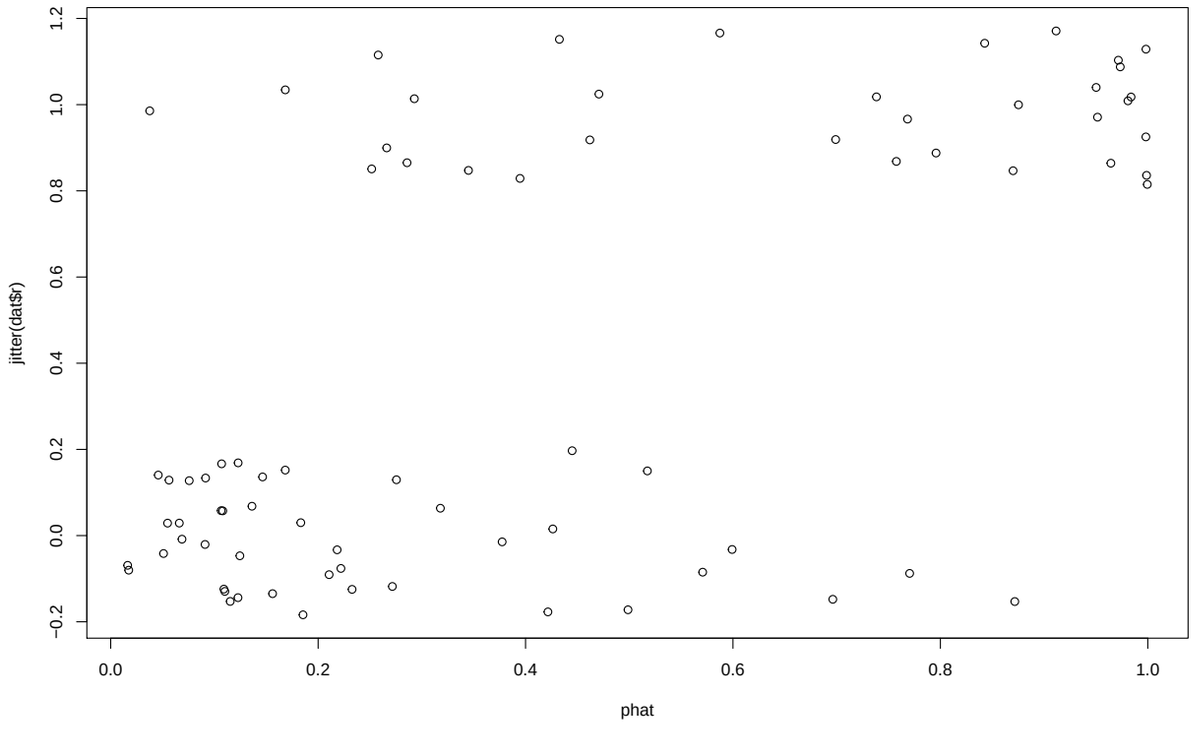
10.0.4.1 Predictive checks

We can use the same ideas to make predictions for each of the original data points. This is similar to what we did to calculate residuals with earlier models.

First we take the X matrix and matrix multiply it with the posterior means of the coefficients. Then we need to pass these linear values through the inverse of the link function as we did above.

```
        [,1]
2 0.49879536
3 0.10807104
4 0.22198056
5 0.10653840
6 0.27551029
7 0.09147155
```

These phat values are the model's predicted probability of calcium oxalate crystals for each data point. We can get a rough idea of how successful the model is by plotting these predicted values against the actual outcome.



Suppose we choose a cutoff for these predicted probabilities. If the model tells us the probability is higher than 0.5, we will classify the observation as a 1 and if it is less than 0.5, we will classify it as a 0. That way the model classifies each data point. Now we can tabulate these classifications against the truth to see how well the model predicts the original data.

```
        0  1
FALSE 38 12
TRUE   6 21
```

```
[1] 0.7662338
```

The correct classification rate is about 76%, not too bad, but not great.

Now suppose that it is considered really bad to predict no calcium oxalate crystal when there in fact is one. We might then choose to lower our threshold for classifying

data points as 1s. Say we change it to 0.3. That is, if the model says the probability is greater than 0.3, we will classify it as having a calcium oxalate crystal.

```
        0  1
  FALSE 32  7
  TRUE  12 26
```

```
[1] 0.7532468
```

It looks like we gave up a little classification accuracy, but we did indeed increase our chances of detecting a true positive.

We could repeat this exercise for many thresholds between 0 and 1, and each time calculate our error rates. This is equivalent to calculating what is called the ROC (receiver-operating characteristic) curve, which is often used to evaluate classification techniques.

These classification tables we have calculated were all in-sample. They were predicting for the same data used to fit the model. We could get a less biased assessment of how well our model performs if we calculated these tables for data that were not used to fit the model. For example, before fitting the model, you could withhold a set of randomly selected "test" data points, and use the model fit to the rest of the "training" data to make predictions on your "test" set.

10.1 Poisson regression

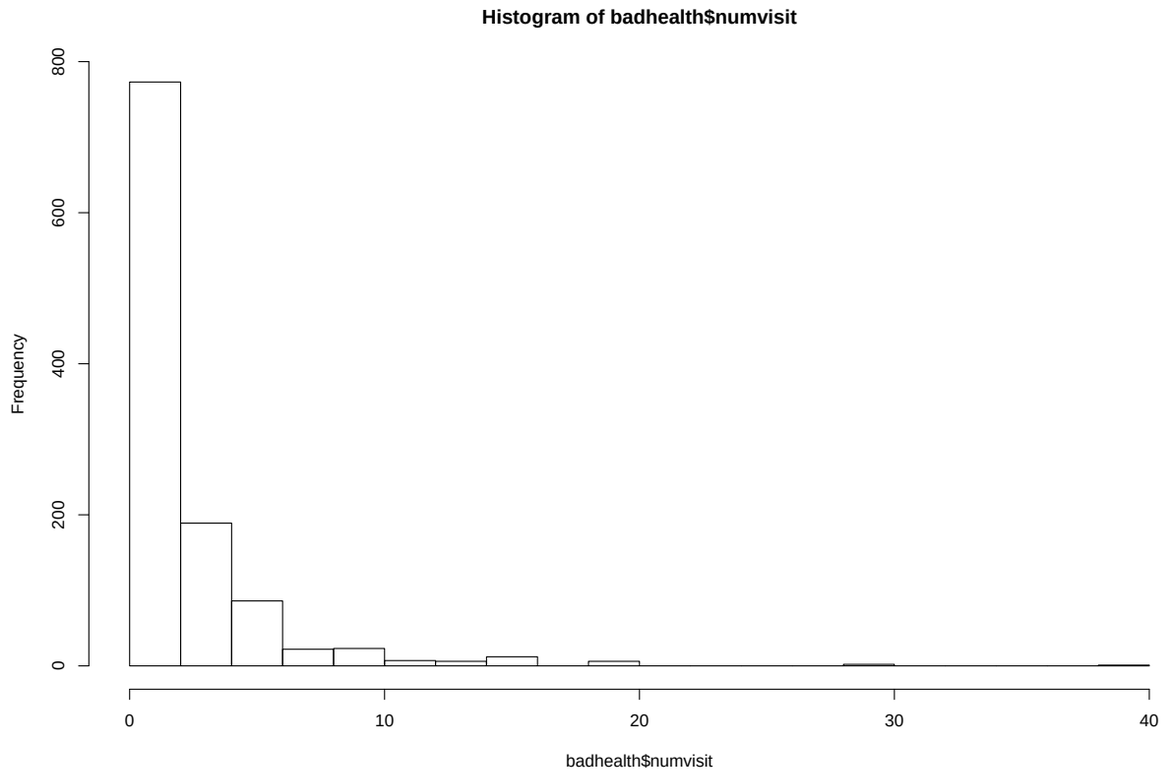
Data

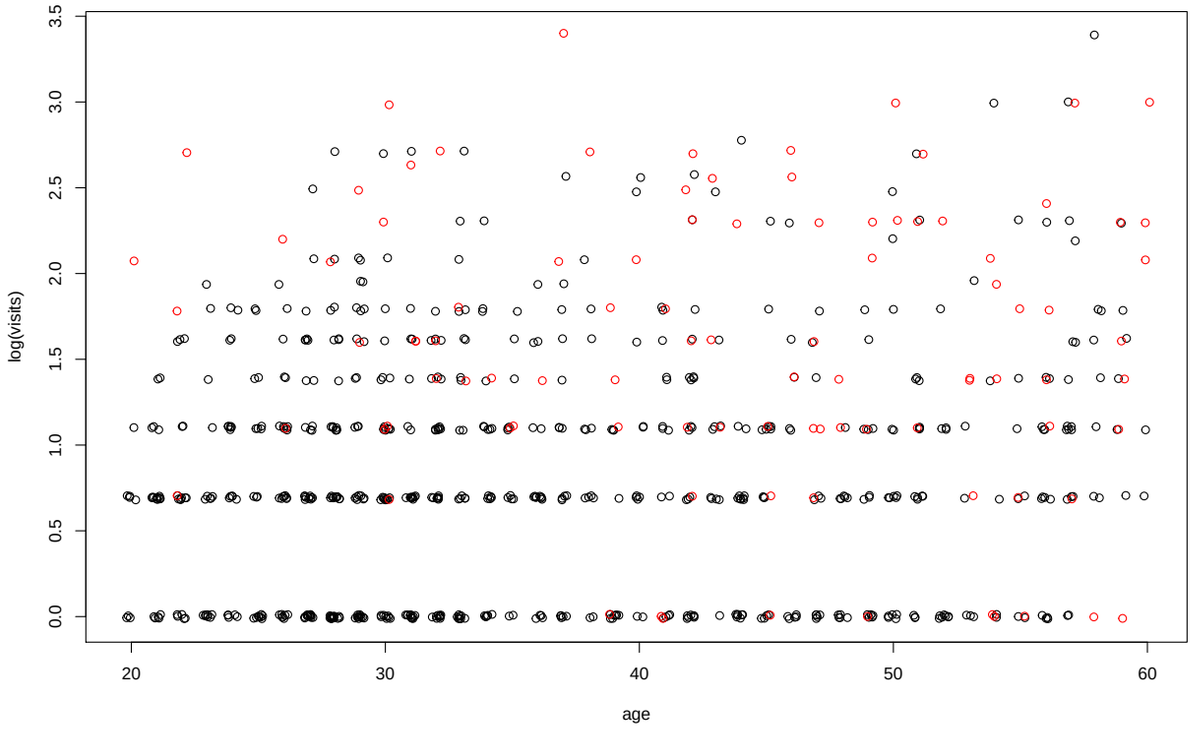
For an example of Poisson regression, we'll use the badhealth data set from the COUNT package in R.

```
  numvisit badh age
1       30    0  58
2       20    0  54
3       16    0  44
4       20    0  57
5       15    0  33
6       15    0  28
```

```
[1] FALSE
```

As usual, let's visualize these data.



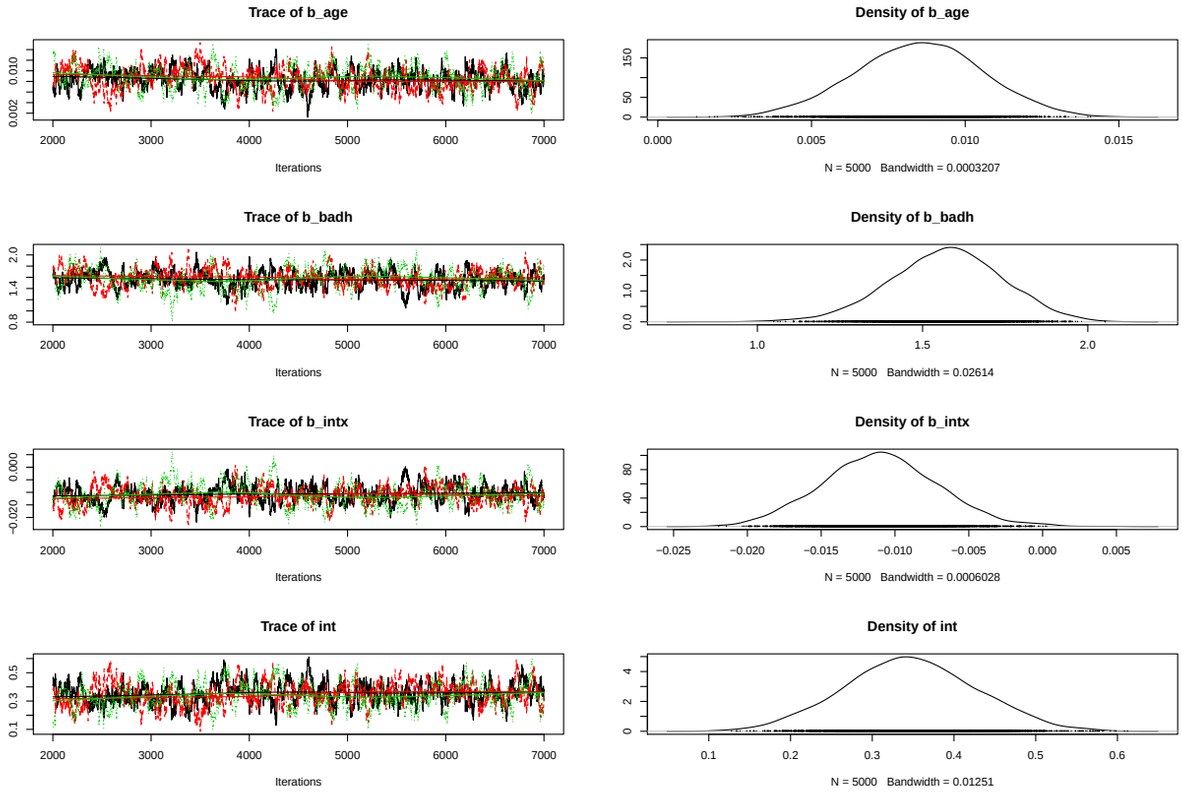


10.1.0.1 Model

It appears that both age and bad health are related to the number of doctor visits. We should include model terms for both variables. If we believe the age/visits relationship is different between healthy and non-healthy populations, we should also include an interaction term. We will fit the full model here and leave it to you to compare it with the simpler additive model.

```
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 1127
   Unobserved stochastic nodes: 4
   Total graph size: 3673
```

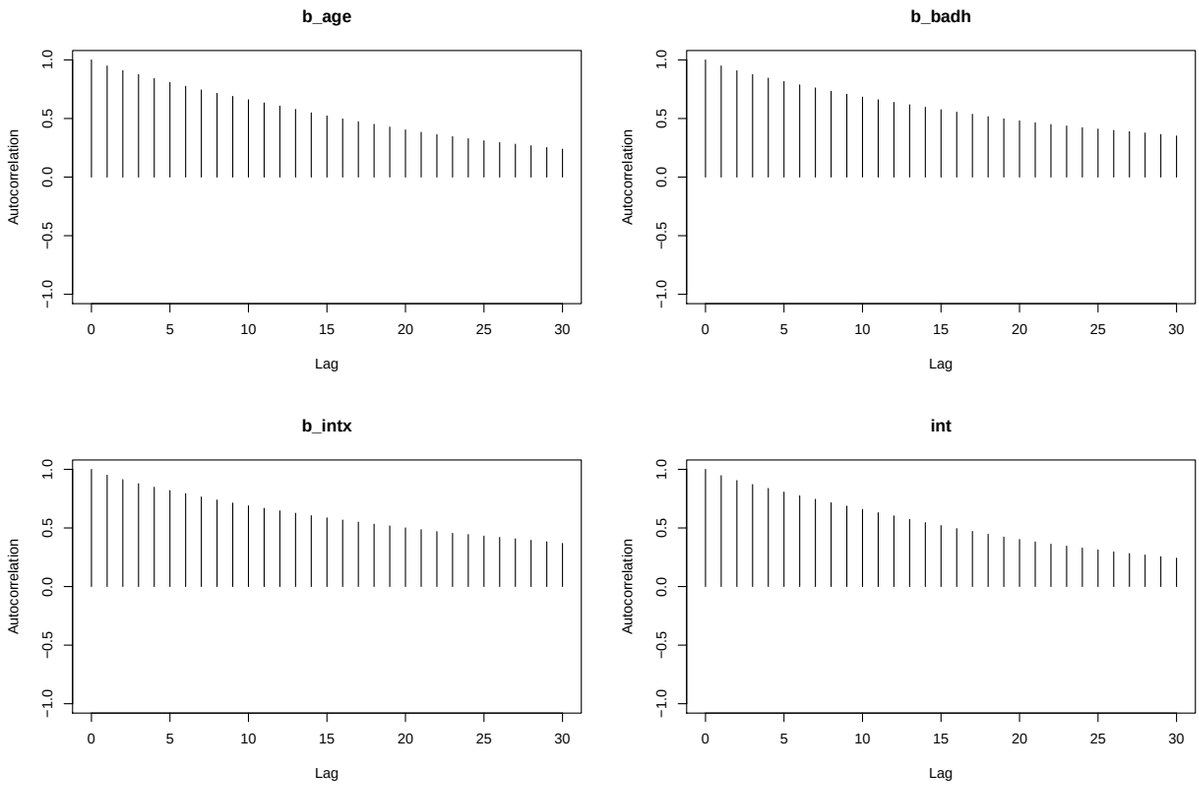
```
Initializing model
```

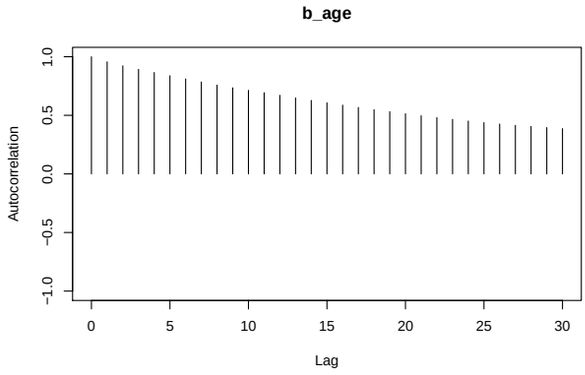


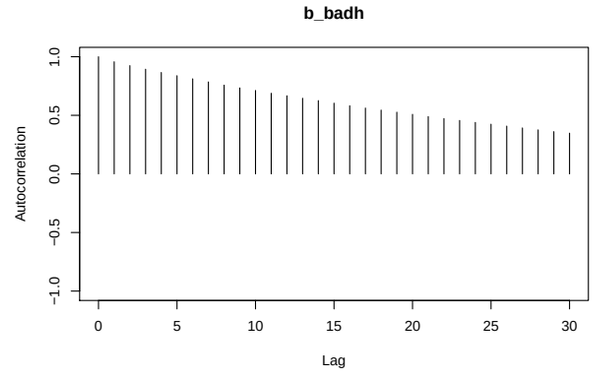

```
Potential scale reduction factors:

       Point est. Upper C.I.
b_age        1.02       1.05
b_badh       1.03       1.06
b_intx       1.03       1.06
int          1.02       1.05

Multivariate psrf

1.01
```

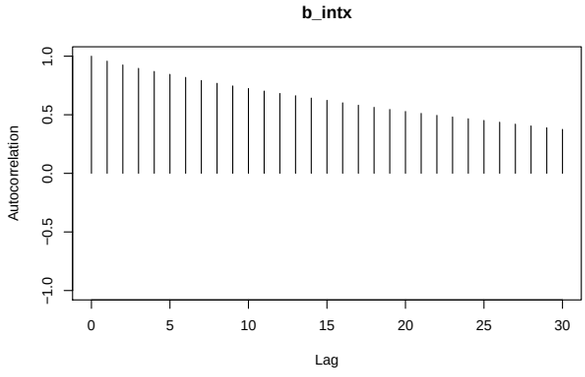
```
           b_age    b_badh    b_intx       int
```

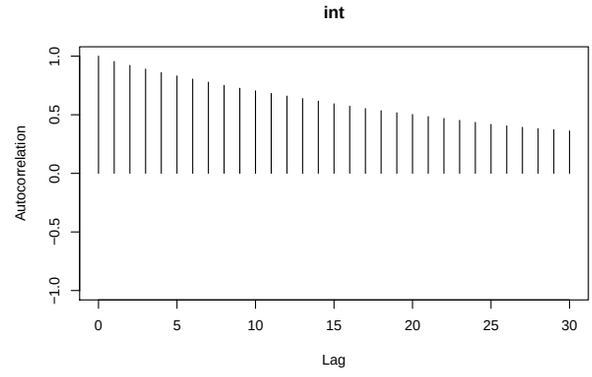
```
Lag 0  1.0000000 1.0000000 1.0000000 1.0000000
Lag 1  0.9570904 0.9587923 0.9602977 0.9545761
Lag 5  0.8362040 0.8446282 0.8512093 0.8320046
Lag 10 0.7060022 0.7246247 0.7355487 0.7015426
Lag 50 0.2060845 0.2076156 0.2261041 0.2035973
```

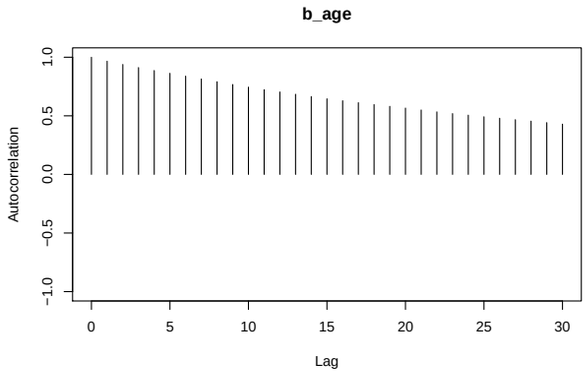


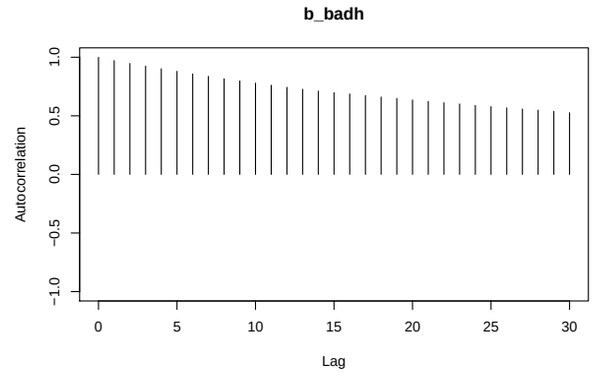

b_age
Autocorrelation
Lag

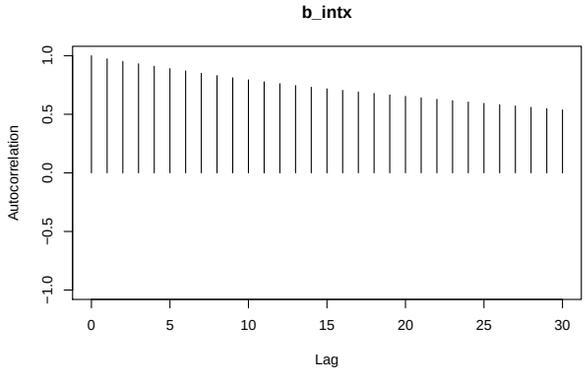

b_badh
Autocorrelation
Lag

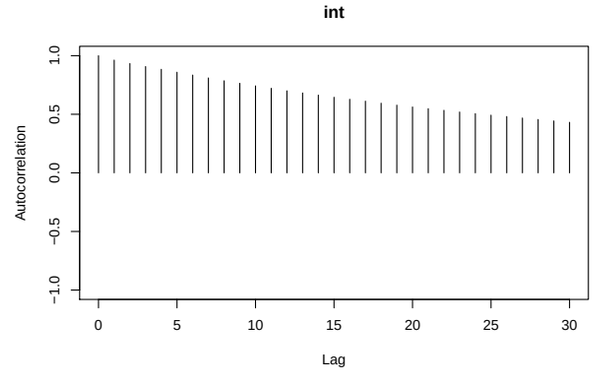

b_intx
Autocorrelation
Lag


int
Autocorrelation
Lag


b_age
Autocorrelation
Lag


b_badh
Autocorrelation
Lag


b_intx
Autocorrelation
Lag


int
Autocorrelation
Lag

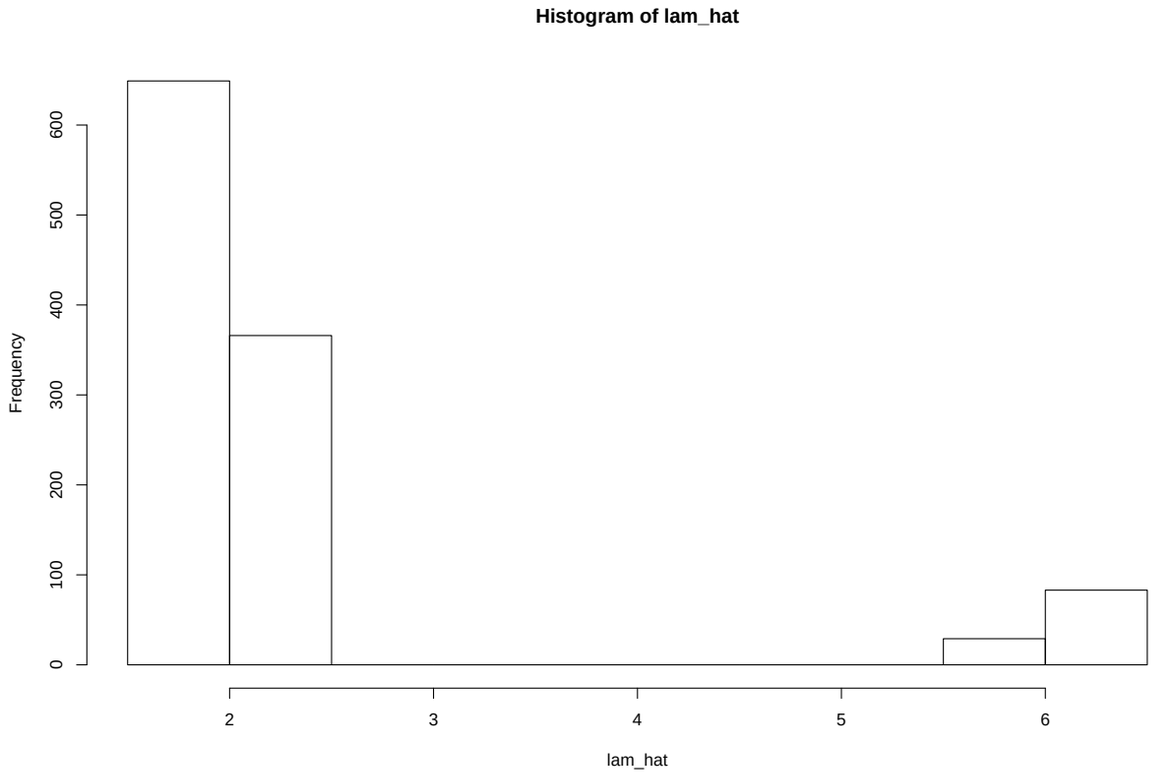
```
   b_age    b_badh    b_intx       int
253.8482 227.0535 221.1072 259.3206
```

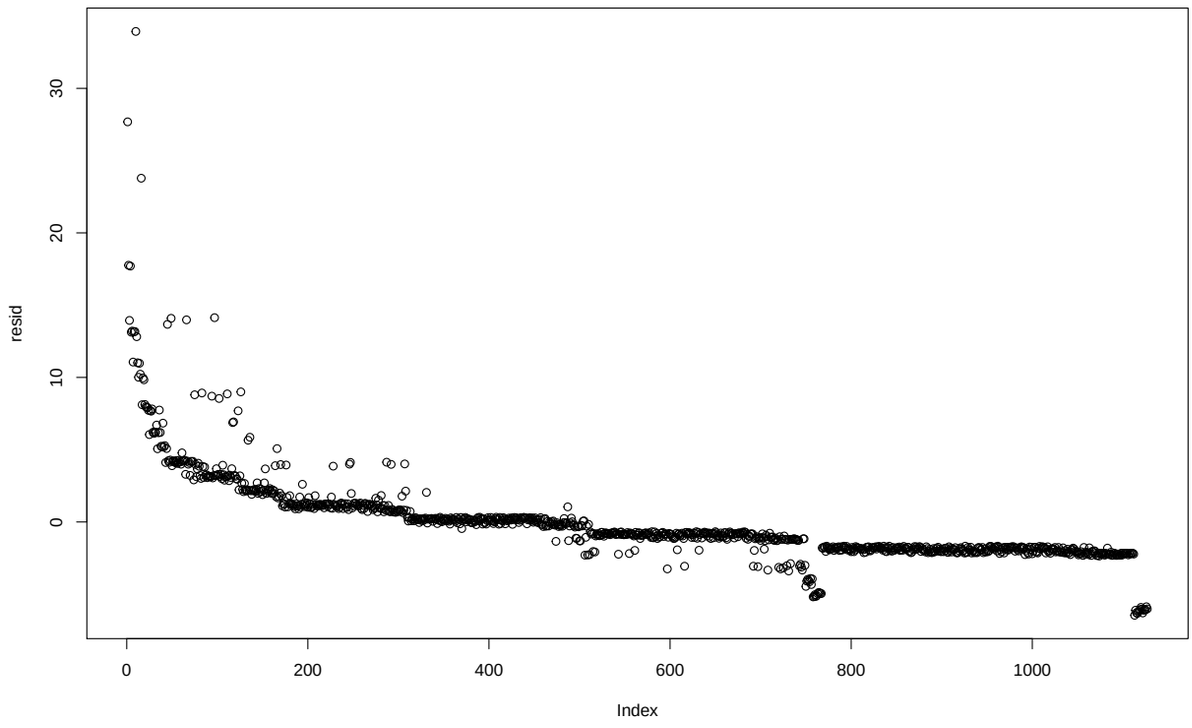
10.1.0.2 Model checking

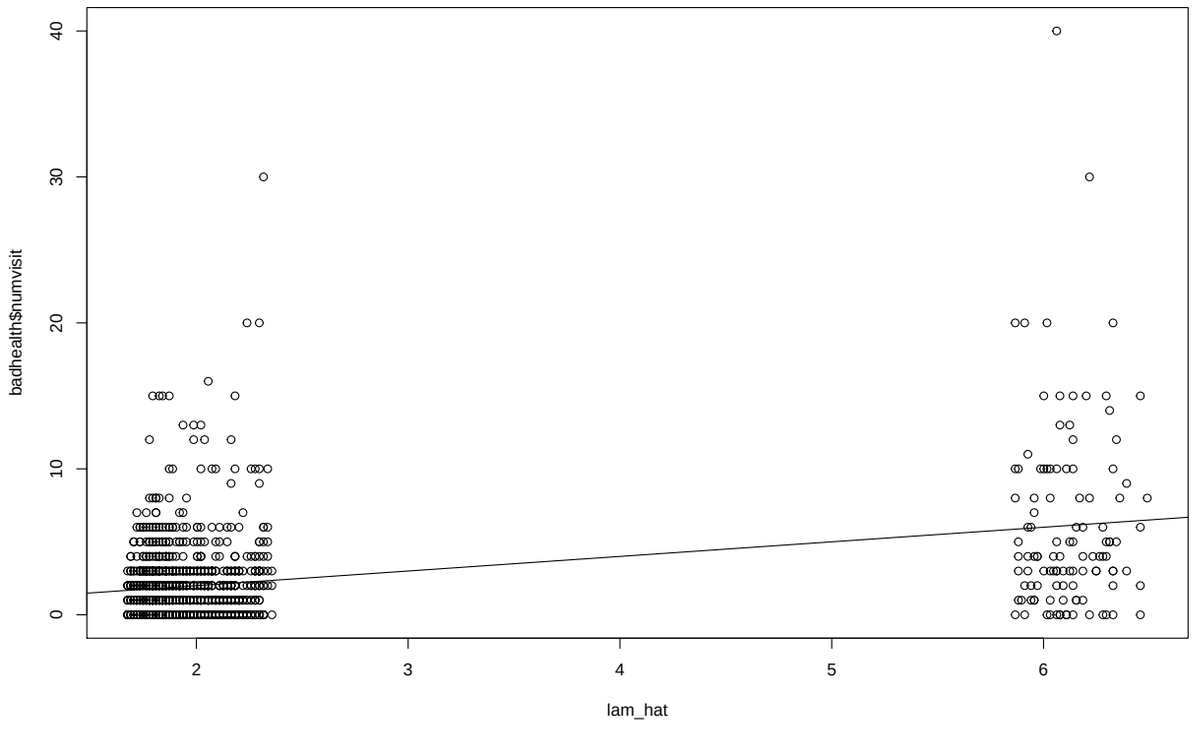
To get a general idea of the model's performance, we can look at predicted values and residuals as usual. Don't forget that we must apply the inverse of the link function to get predictions for λ.

```
     badh age
[1,]    0  58 0
[2,]    0  54 0
[3,]    0  44 0
[4,]    0  57 0
[5,]    0  33 0
[6,]    0  28 0
```

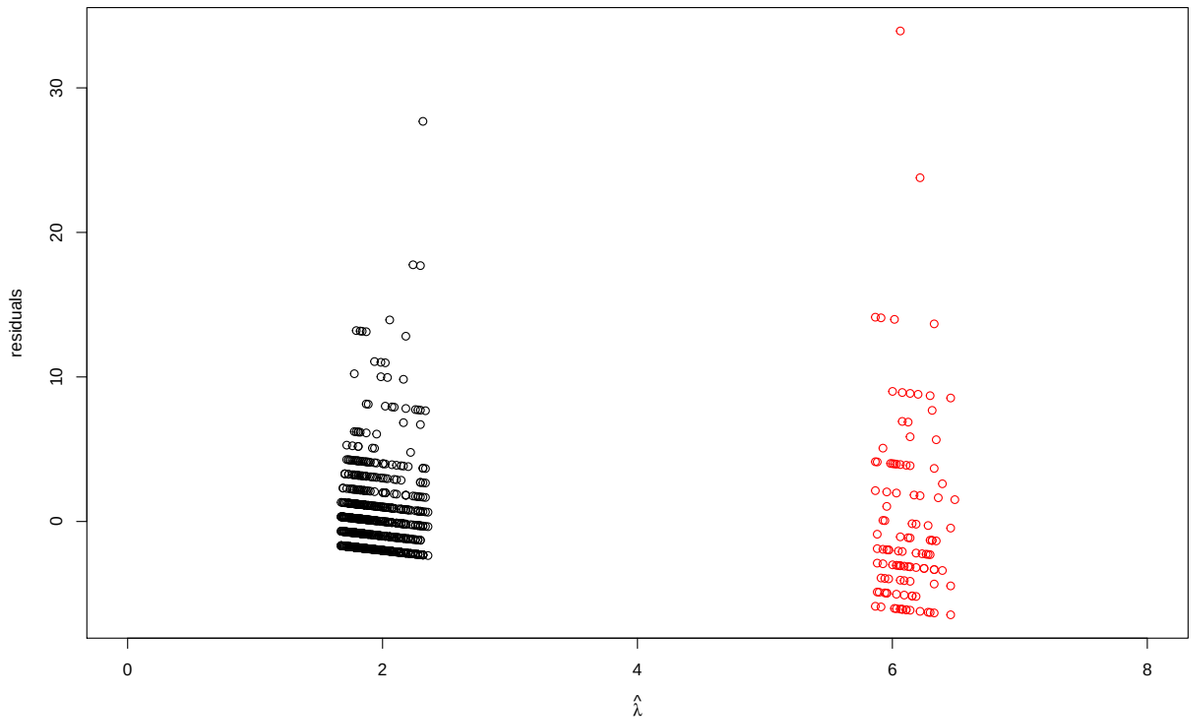
```
       b_age        b_badh       b_intx          int
 0.008529124  1.575239598 -0.011054619  0.345521806
```


Histogram of lam_hat
Frequency
0
100
200
300
400
500
600
2
3
4
5
6
lam_hat


resid
0
10
20
30
0
200
400
600
800
1000
Index



It is not surprising that the variability increases for values predicted at higher values since the mean is also the variance in the Poisson distribution. However, observations predicted to have about two visits should have variance about two, and observations predicted to have about six visits should have variance about six.




```
[1] 7.022515
```

```
[1] 41.19612
```

Clearly this is not the case with these data. This indicates that either the model fits poorly (meaning the covariates don't explain enough of the variability in the data), or the data are "overdispersed" for the Poisson likelihood we have chosen. This is a common issue with count data. If the data are more variable than the Poisson likelihood would suggest, a good alternative is the negative binomial distribution, which we will not pursue here.

10.1.0.3 Results

Assuming the model fit is adequate, we can interpret the results.

```
Iterations = 2001:7000
```

```
Thinning interval = 1
Number of chains = 3
Sample size per chain = 5000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

            Mean       SD  Naive SE Time-series SE
b_age    0.008496 0.002070 1.690e-05      0.0001317
b_badh   1.571334 0.171733 1.402e-03      0.0121322
b_intx -0.010999 0.003958 3.231e-05      0.0002788
int      0.346656 0.080726 6.591e-04      0.0051233

2. Quantiles for each variable:

            2.5%       25%       50%       75%     97.5%
b_age    0.00435  0.007097  0.008529  0.009912  0.012496
b_badh   1.22255  1.461476  1.575240  1.687556  1.896277
b_intx -0.01852 -0.013655 -0.011055 -0.008440 -0.003107
int      0.19082  0.292009  0.345522  0.400879  0.504955
```

The intercept is not necessarily interpretable here because it corresponds to a healthy 0-year-old, whereas the youngest person in the data set is 20 years old.

For healthy individuals, it appears that age has a positive association with number of doctor visits. Clearly, bad health is associated with an increase in expected number of visits. The interaction coefficient is interpreted as an adjustment to the age coefficient for people in bad health. Hence, for people with bad health, age is essentially unassociated with number of visits.

10.1.0.4 Predictive distributions

Let's say we have two people aged 35, one in good health and the other in poor health. What is the posterior probability that the individual with poor health will have more doctor visits? This goes beyond the posterior probabilities we have calculated comparing expected responses in previous lessons. Here we will create Monte Carlo samples for the responses themselves. This is done by taking the Monte Carlo samples of the model parameters, and for each of those, drawing a sample from the likelihood. Let's walk through this.

First, we need the xx values for each individual. We'll say the healthy one is Person 1 and the unhealthy one is Person 2. Their xx values are

The posterior samples of the model parameters are stored in mod_csim:

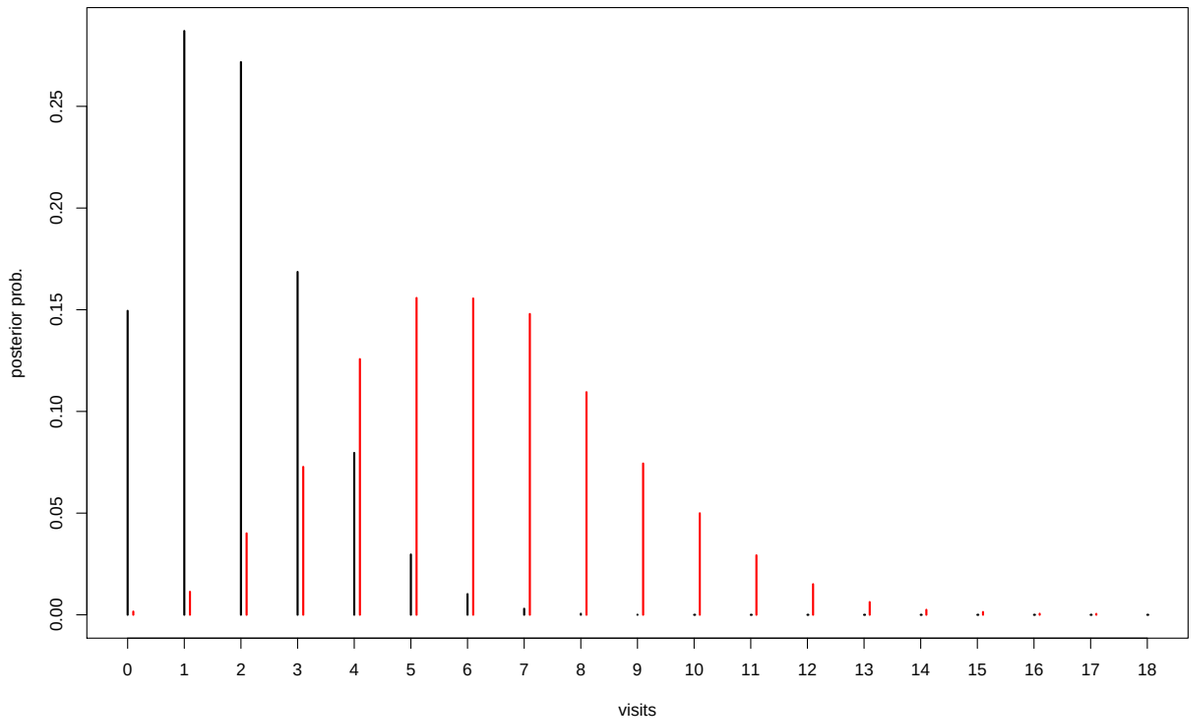
```
Markov Chain Monte Carlo (MCMC) output:
Start = 1
End = 7
Thinning interval = 1
           b_age   b_badh      b_intx       int
[1,] 0.007289075 1.645301 -0.01271235 0.4143942
[2,] 0.006277269 1.661431 -0.01305915 0.4092395
[3,] 0.006947994 1.605846 -0.01258092 0.4684650
[4,] 0.006185381 1.649199 -0.01157492 0.4440343
[5,] 0.005819361 1.655503 -0.01155981 0.4302537
[6,] 0.006127379 1.650063 -0.01130789 0.4381056
[7,] 0.006223390 1.667560 -0.01374777 0.4084692
```

First, we'll compute the linear part of the predictor:

Next we'll apply the inverse link:

The final step is to use these samples for the λ parameter for each individual and simulate actual number of doctor visits using the likelihood:

```
[1] 15000
```



Finally, we can answer the original question: What is the probability that the person with poor health will have more doctor visits than the person with good health?

```
[1] 0.9212
```

Because we used our posterior samples for the model parameters in our simulation, this posterior predictive distribution on the number of visits for these two new individuals naturally takes into account our uncertainty in the model estimates. This is a more honest/realistic distribution than we would get if we had fixed the model parameters at their MLE or posterior means and simulated data for the new individuals.

11 Multi-level models

11.1 Hierarchical models

11.1.1 Data

Let's fit our hierarhical model for counts of chocolate chips. The data can be found in

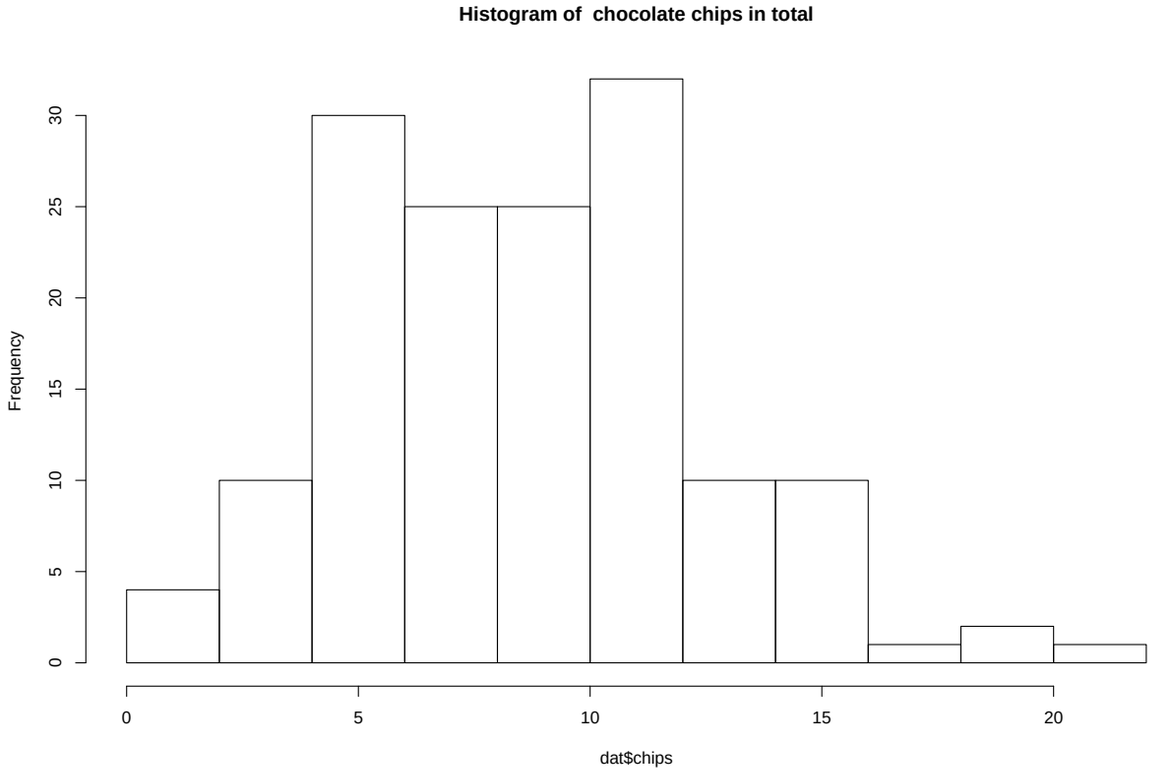
Table 1: First 10 values

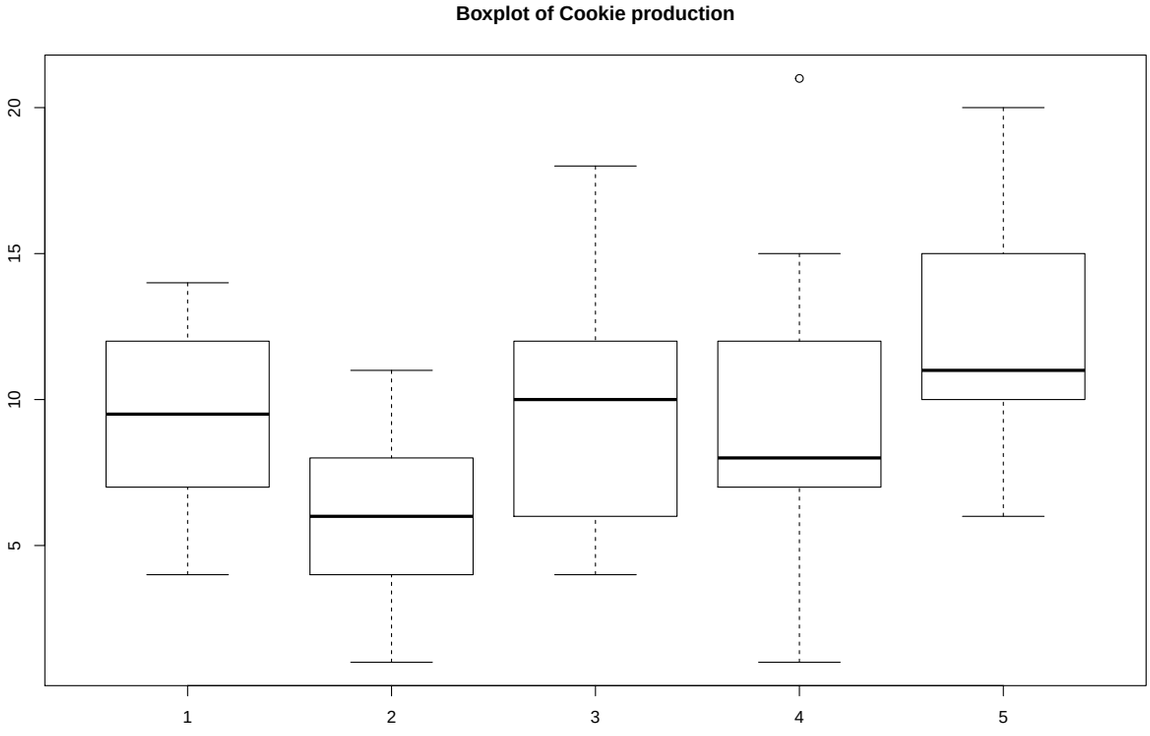
chips	location
12	1
12	1
6	1
13	1
12	1
12	1
9	1
10	1
7	1
14	1

Table 2: number of cookies per location

1	2	3	4	5
30	30	30	30	30

We can also visualize the distribution of chips by location.


Histogram of chocolate chips in total
Frequency
0
5
10
15
20
25
30
0
5
10
15
20
dat$chips


Boxplot of Cookie production
5
10
15
20
1
2
3
4
5

11.1.2 Prior predictive checks

Before implementing the model, we need to select prior distributions for α and β, the hyperparameters governing the gamma distribution for the λ parameters. First, think about what the λ's represent. For location j, λ_j is the expected number of chocolate chips per cookie. Hence, α and β control the distribution of these means between locations. The mean of this gamma distribution will represent the overall mean of number of chips for all cookies. The variance of this gamma distribution controls the variability between locations. If this is high, the mean number of chips will vary widely from location to location. If it is small, the mean number of chips will be nearly the same from location to location.

To see the effects of different priors on the distribution of λ's, we can simulate. Suppose we try independent exponential priors for α and β.

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  0.021   2.983   9.852  61.127  29.980 4858.786
```

```
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
  0.1834   3.3663   8.5488  41.8137  22.2219 2865.6461
```

After simulating from the priors for α and β, we can use those samples to simulate further down the hierarchy:

```
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
   0.000    1.171    7.667   83.062   28.621 11005.331
```

Or for a prior predictive reconstruction of the original data set:

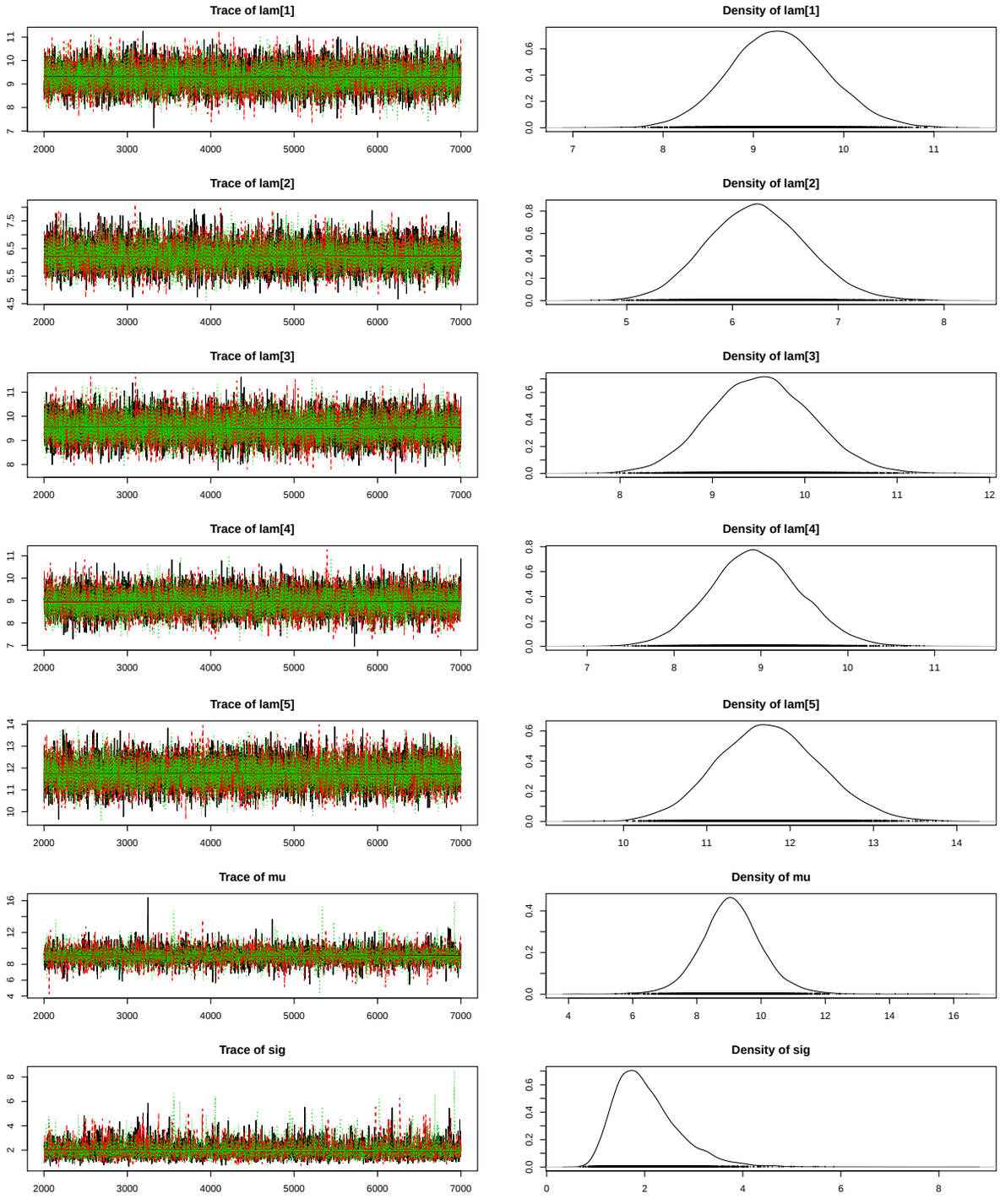
```
[1] 66.444084  9.946688  6.028319 15.922568 47.978587
```

```
 [1] 63 58 64 63 70 62 61 48 71 73 70 77 66 60 72 77 69 62 66 71 49 80 66
[24] 75 74 55 62 90 65 57 12  9  7 10 12 10 11  7 14 13  9  6  6 13  7 10
[47] 12  9  9 10  7  8  6  9  7 10 13 13  8 12  6 10  3  6  7  4  6  7  5
[70]  5  4  3  6  2  8  4  8  4  5  7  1  4  5  3  8  8  3  1  7  3 16 14
```

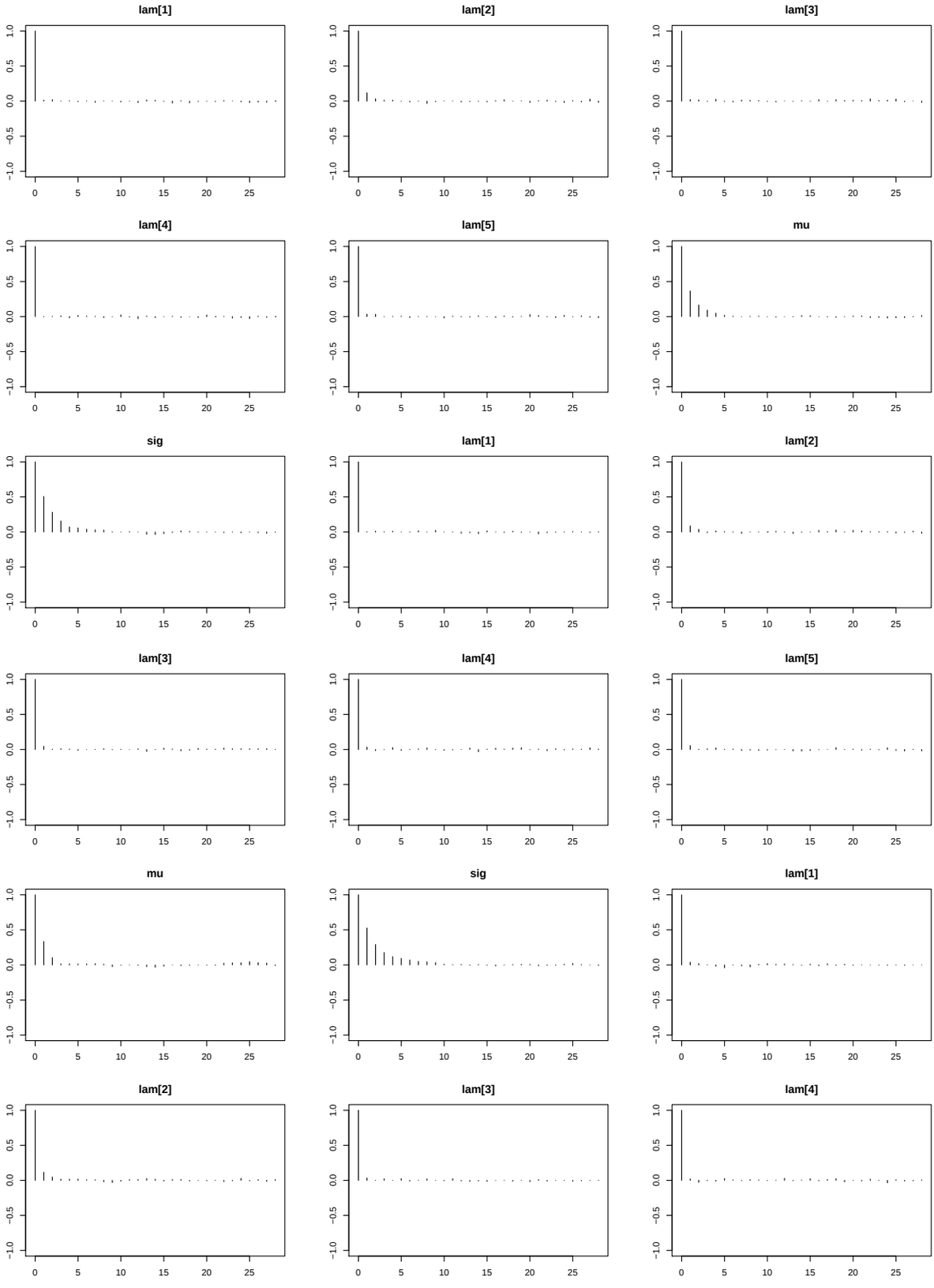
```
 [93] 13 17 17 12 13 13 16 16 15 14 11 10 13 17 16 19 16 17 15 16  7 17 21
[116] 16 12 15 14 13 52 44 51 46 39 40 40 44 46 59 45 49 58 42 31 52 43 47
[139] 53 41 48 57 35 60 51 58 36 34 41 59
```

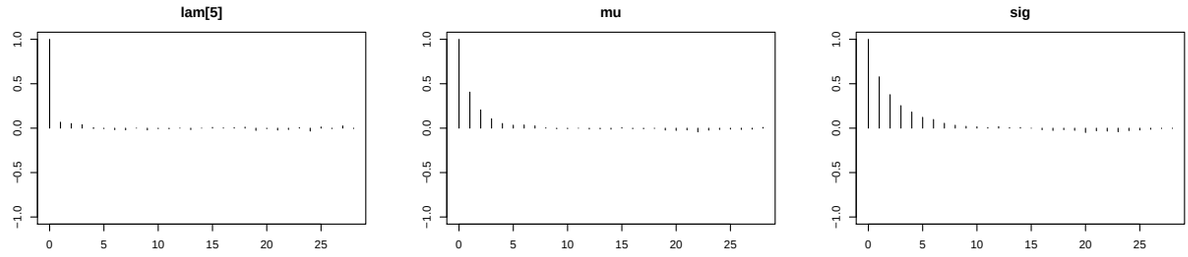
Because these priors have high variance and are somewhat noninformative, they produce unrealistic predictive distributions. Still, enough data would overwhelm the prior, resulting in useful posterior distributions. Alternatively, we could tweak and simulate from these prior distributions until they adequately represent our prior beliefs. Yet another approach would be to re-parameterize the gamma prior, which we'll demonstrate as we fit the model.

```
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 150
   Unobserved stochastic nodes: 7
   Total graph size: 322

Initializing model
```


Trace of lam[1]
Density of lam[1]
Trace of lam[2]
Density of lam[2]
Trace of lam[3]
Density of lam[3]
Trace of lam[4]
Density of lam[4]
Trace of lam[5]
Density of lam[5]
Trace of mu
Density of mu
Trace of sig
Density of sig

```
Potential scale reduction factors:

       Point est. Upper C.I.
lam[1]       1.00       1.00
lam[2]       1.00       1.00
lam[3]       1.00       1.00
lam[4]       1.00       1.00
lam[5]       1.00       1.00
mu           1.00       1.01
sig          1.02       1.03

Multivariate psrf

1
             lam[1]       lam[2]       lam[3]       lam[4]       lam[5]
Lag 0   1.000000000  1.000000000  1.000000000  1.000000000  1.000000000
Lag 1   0.018178127  0.107547058  0.033609022  0.016447685  0.051954601
Lag 5  -0.016375682  0.007917758  0.002429397  0.009918063  0.001782960
Lag 10  0.001860342 -0.004128700 -0.002509929  0.002949134 -0.012104619
Lag 50 -0.011908450 -0.013250725 -0.001946227 -0.009560602  0.003400869
                mu           sig
Lag 0   1.000000000  1.000000000
Lag 1   0.368725629  0.536664005
Lag 5   0.022403557  0.091404071
Lag 10 -0.003570828  0.008506027
Lag 50  0.008855208 -0.020855933
```


lam[1]
lam[2]
lam[3]
lam[4]
lam[5]
mu
sig
lam[1]
lam[2]
lam[3]
lam[4]
lam[5]
mu
sig
lam[1]
lam[2]
lam[3]
lam[4]



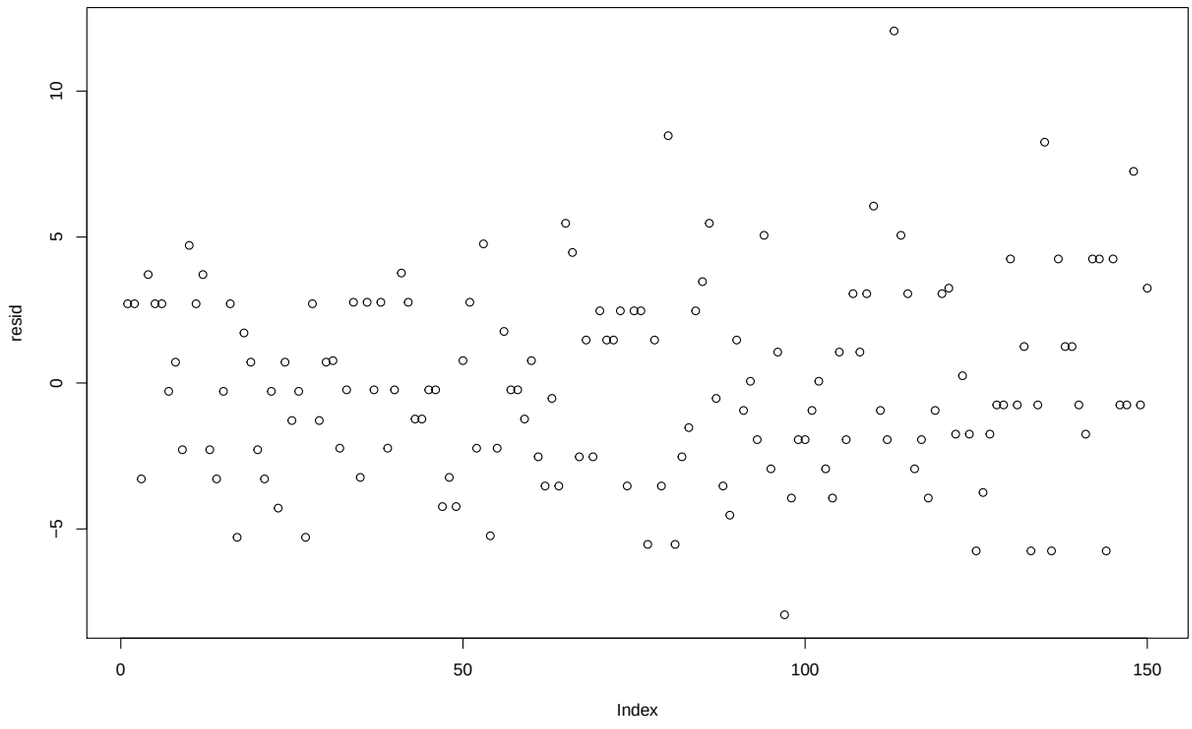

```
   lam[1]    lam[2]    lam[3]    lam[4]    lam[5]        mu       sig
15044.570 11409.473 14024.582 14771.578 12587.898  6461.266  3964.569
```

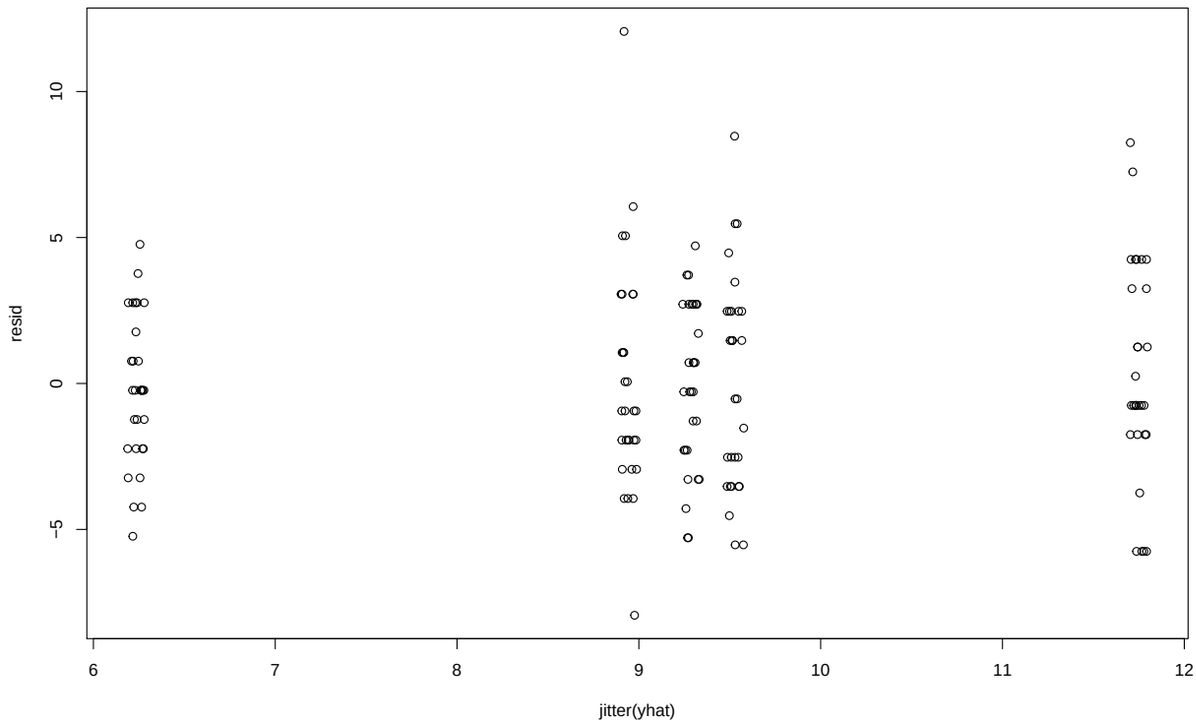
11.1.3 Model checking

After assessing convergence, we can check the fit via residuals. With a hierarhcical model, there are now two levels of residuals: the observation level and the location mean level. To simplify, we'll look at the residuals associated with the posterior means of the parameters.

First, we have observation residuals, based on the estimates of location means.

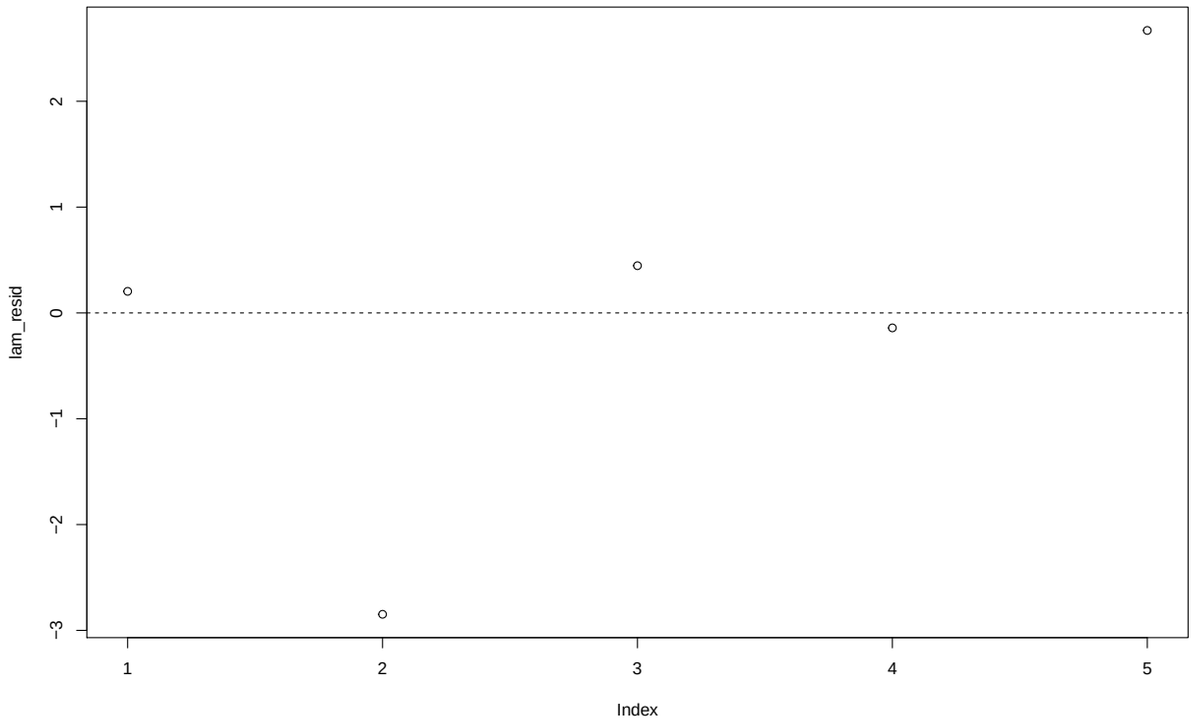
```
   lam[1]    lam[2]    lam[3]    lam[4]    lam[5]        mu       sig
 9.286068  6.234426  9.528687  8.941003 11.751594  9.082348  2.049746
```


resid
10
5
0
−5
0
50
100
150
Index


resid
10
5
0
−5
6
7
8
9
10
11
12
jitter(yhat)

```
[1] 6.447126
```

```
[1] 13.72414
```



We don't see any obvious violations of our model assumptions.

11.1.4 Results

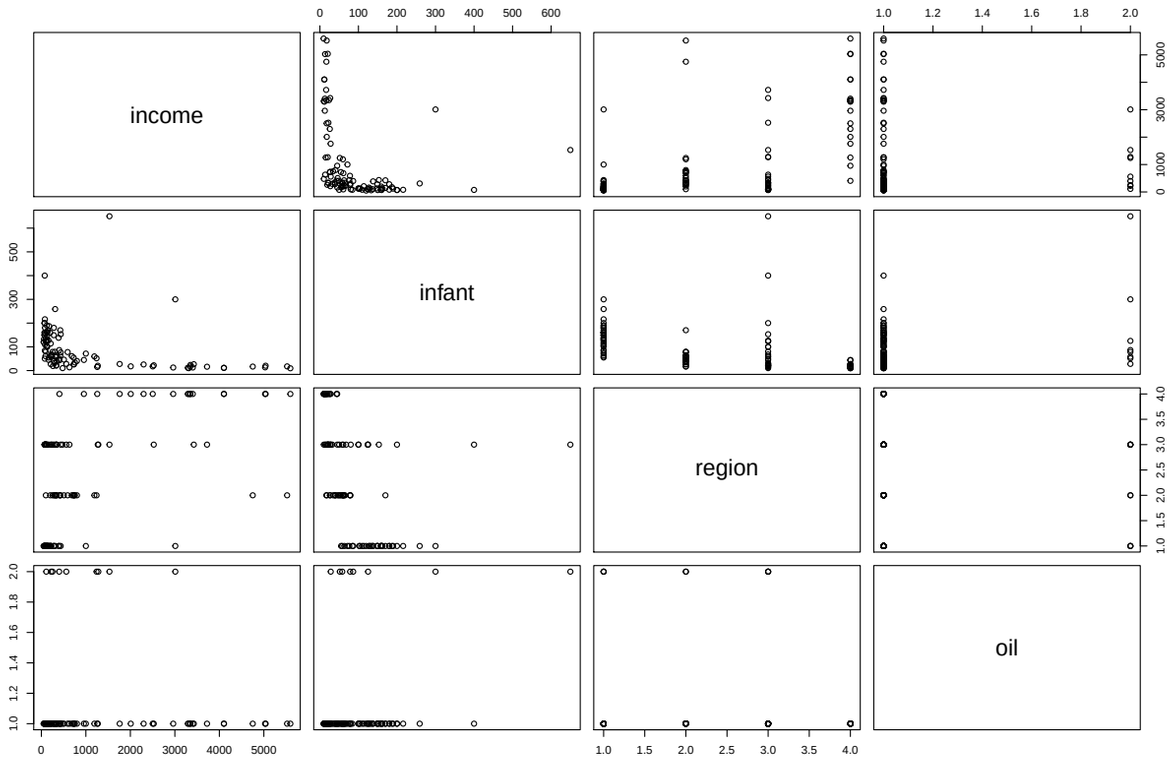
```
Iterations = 2001:7000
Thinning interval = 1
Number of chains = 3
Sample size per chain = 5000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:
```

```
         Mean     SD Naive SE Time-series SE
lam[1]  9.286 0.5392 0.004402       0.004395
lam[2]  6.234 0.4664 0.003808       0.004370
lam[3]  9.529 0.5459 0.004457       0.004610
lam[4]  8.941 0.5227 0.004268       0.004304
lam[5] 11.752 0.6170 0.005038       0.005518
mu      9.082 0.9682 0.007905       0.012134
sig     2.050 0.6793 0.005546       0.010900

2. Quantiles for each variable:

         2.5%    25%    50%    75%  97.5%
lam[1]  8.256  8.917  9.275  9.641 10.377
lam[2]  5.352  5.914  6.226  6.543  7.184
lam[3]  8.489  9.153  9.520  9.896 10.618
lam[4]  7.936  8.589  8.930  9.281  9.985
lam[5] 10.570 11.323 11.737 12.159 13.000
mu      7.207  8.486  9.063  9.656 11.056
sig     1.088  1.569  1.926  2.395  3.664
```

11.1.5 Random intercept linear model

We can extend the linear model for the Leinhardt data on infant mortality by incorporating the region variable. We'll do this with a hierarhcical model, where each region has its own intercept.

```
'data.frame':   105 obs. of  4 variables:
 $ income: int  3426 3350 3346 4751 5029 3312 3403 5040 2009 2298 ...
 $ infant: num  26.7 23.7 17 16.8 13.5 10.1 12.9 20.4 17.8 25.7 ...
 $ region: Factor w/ 4 levels "Africa","Americas",..: 3 4 4 2 4 4 4 4 4 4 .
 $ oil   : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
```



	income	infant	region	oil
Australia	3426	26.7	Asia	no
Austria	3350	23.7	Europe	no
Belgium	3346	17.0	Europe	no
Canada	4751	16.8	Americas	no
Denmark	5029	13.5	Europe	no
Finland	3312	10.1	Europe	no

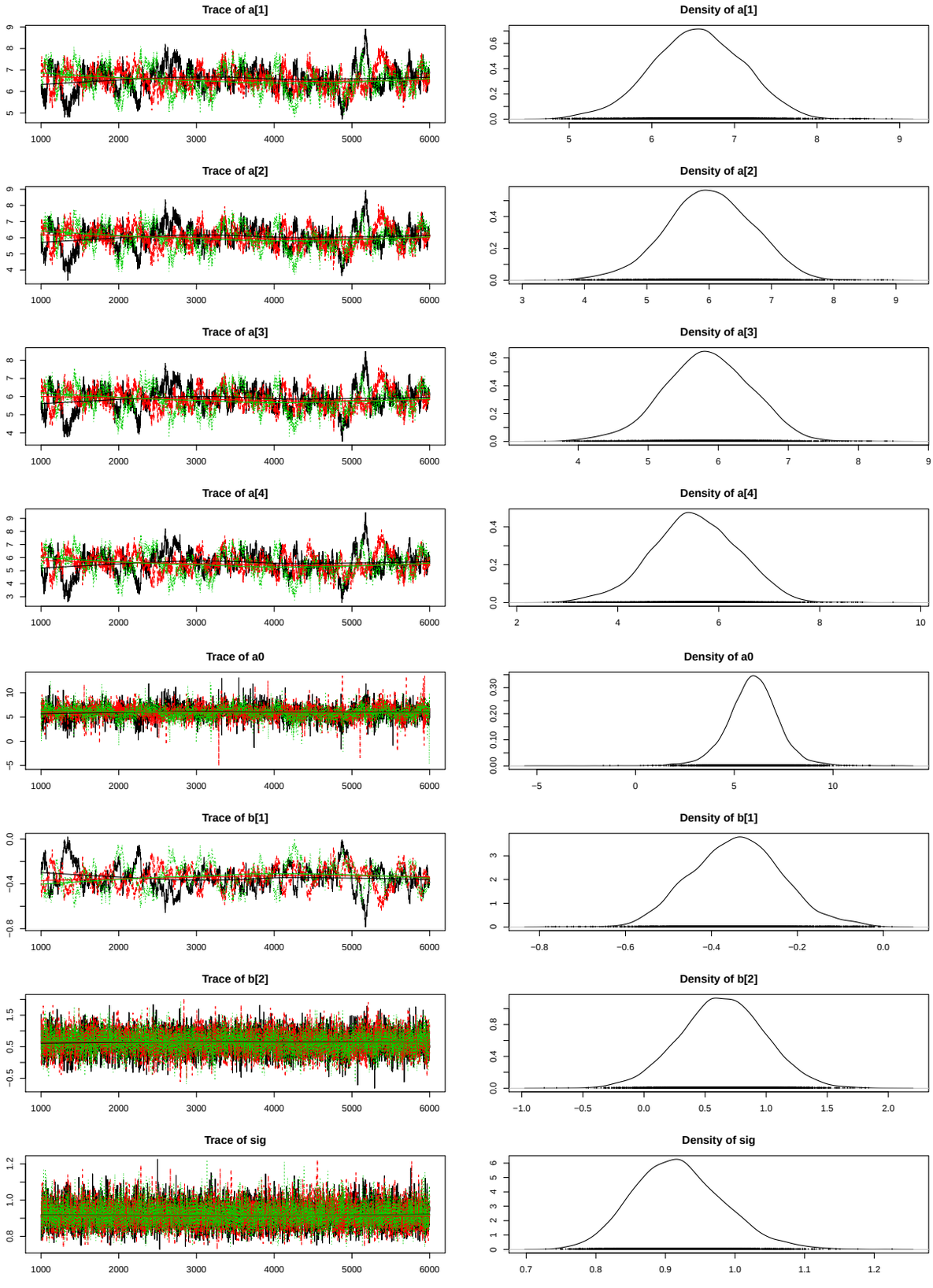
Previously, we worked with infant mortality and income on the logarithmic scale. Recall also that we had to remove some missing data.

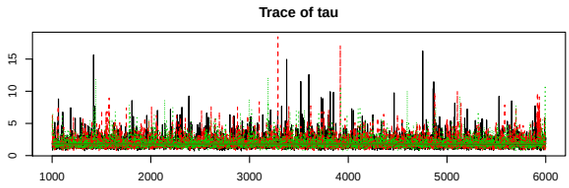
```
'data.frame':   101 obs. of  6 variables:
 $ income   : int  3426 3350 3346 4751 5029 3312 3403 5040 2009 2298 ...
 $ infant   : num  26.7 23.7 17 16.8 13.5 10.1 12.9 20.4 17.8 25.7 ...
 $ region   : Factor w/ 4 levels "Africa","Americas",..: 3 4 4 2 4 4 4 4 4 4
 $ oil      : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
```

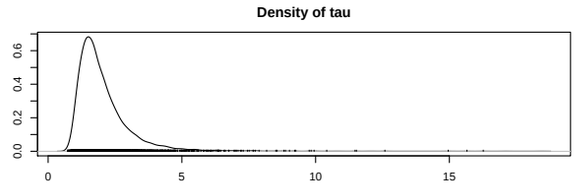
```
 $ logincome: num  8.14 8.12 8.12 8.47 8.52 ...
 $ loginfant: num  3.28 3.17 2.83 2.82 2.6 ...
 - attr(*, "na.action")=Class 'omit'  Named int [1:4] 24 83 86 91
  .. ..- attr(*, "names")= chr [1:4] "Iran" "Haiti" "Laos" "Nepal"
```

Now we can fit the proposed model:

```
  [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0
 [36] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 [71] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0


    1  2  3  4
 0 31 20 24 18
 1  3  2  3  0

Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 101
   Unobserved stochastic nodes: 9
   Total graph size: 639

Initializing model
```


Trace of a[1]
Density of a[1]
Trace of a[2]
Density of a[2]
Trace of a[3]
Density of a[3]
Trace of a[4]
Density of a[4]
Trace of a0
Density of a0
Trace of b[1]
Density of b[1]
Trace of b[2]
Density of b[2]
Trace of sig
Density of sig



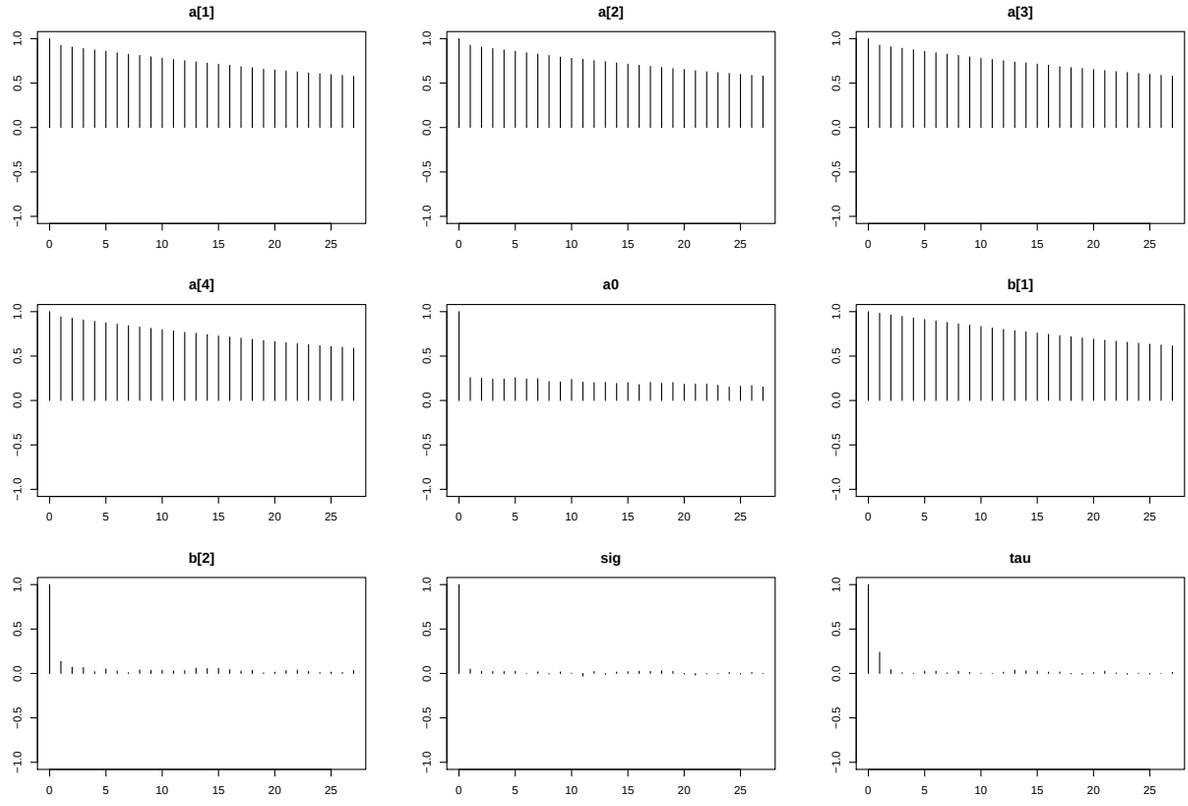


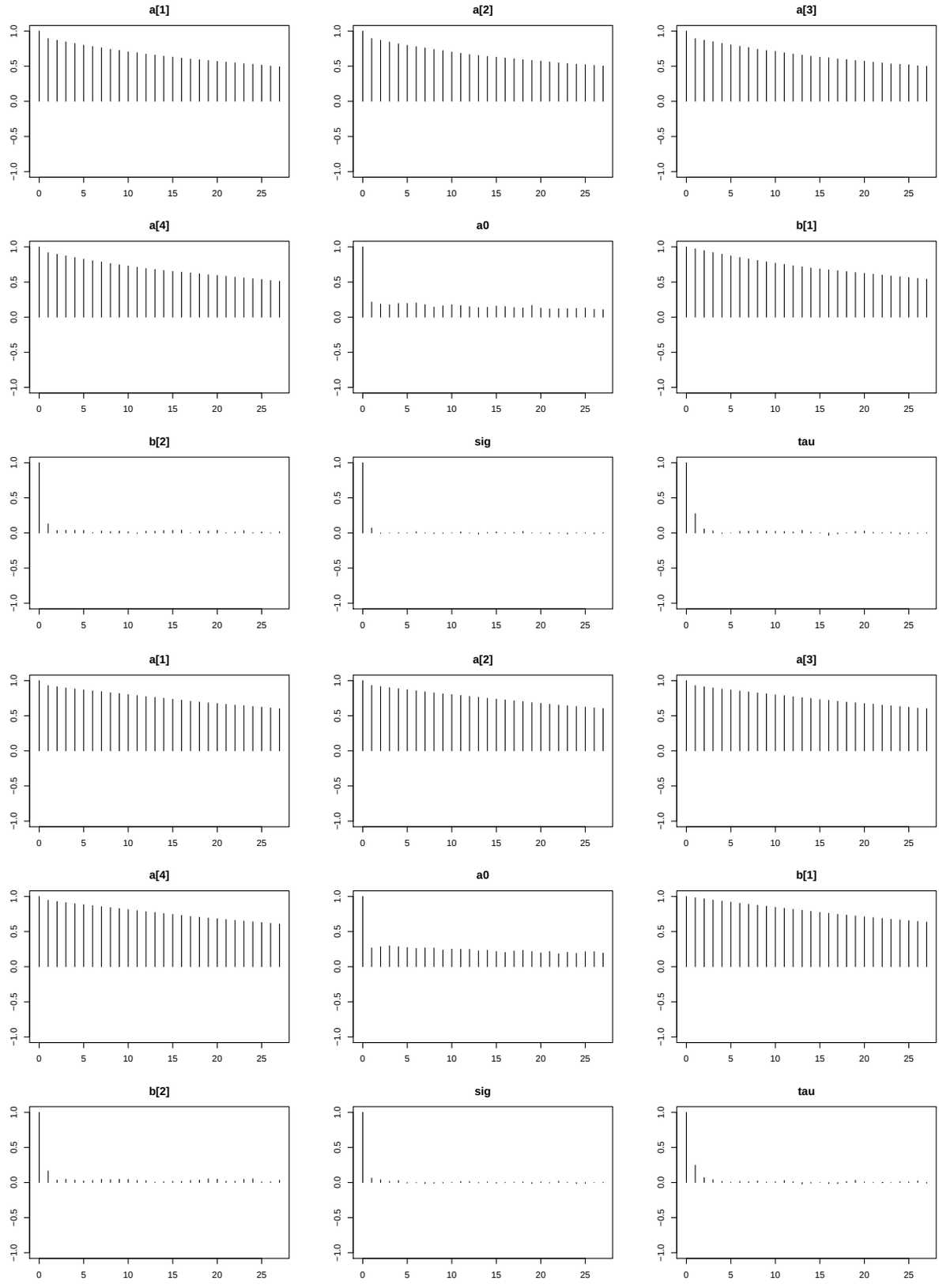

```
Potential scale reduction factors:

     Point est. Upper C.I.
a[1]       1.02       1.05
a[2]       1.02       1.05
a[3]       1.02       1.06
a[4]       1.02       1.06
a0         1.00       1.01
b[1]       1.02       1.06
b[2]       1.00       1.01
sig        1.00       1.00
tau        1.00       1.00

Multivariate psrf
```

```
1.01
```

	a[1]	a[2]	a[3]	a[4]	a0	b[1]
Lag 0	1.0000000	1.0000000	1.0000000	1.0000000	1.0000000	1.0000000
Lag 1	0.9163680	0.9178566	0.9167263	0.9353727	0.2464476	0.9792751
Lag 5	0.8416684	0.8422693	0.8435939	0.8613111	0.2417908	0.9009854
Lag 10	0.7626344	0.7611779	0.7629623	0.7806742	0.2218850	0.8159371
Lag 50	0.3807805	0.3896008	0.3785199	0.3936167	0.1090260	0.4092661

	b[2]	sig	tau
Lag 0	1.000000000	1.000000000	1.000000000
Lag 1	0.143758607	0.061033125	0.254784171
Lag 5	0.037273836	0.004500278	0.010074385
Lag 10	0.033336961	0.001864659	0.013391169
Lag 50	0.005743313	-0.003394234	-0.008654215




a[1]
a[2]
a[3]
a[4]
a0
b[1]
b[2]
sig
tau
a[1]
a[2]
a[3]
a[4]
a0
b[1]
b[2]
sig
tau

a[1]	a[2]	a[3]	a[4]	a0	b[1]
164.8460	175.9739	168.8203	164.2439	699.0692	157.1086

b[2]	sig	tau
4912.4640	12273.5245	9033.3812

11.2 Meta analysis

12 Prior Sensitivity Analysis

When communicating results from any analysis, a responsible statistician will report and justify modeling decisions, especially assumptions. In a Bayesian analysis, there is another assumption that is open to scrutiny: the choices of prior distributions. In the models considered so far in this course, there are an infinite number of prior distributions we could have chosen from.

How do you justify the priors you choose? If they truly represent your beliefs about the parameters before analysis and the model is appropriate, then the posterior distribution truly represents your updated beliefs. If you don't have any strong beliefs beforehand, there are often default, reference, or noninformative prior options, and you will have to select one. However, a collaborator or a boss (indeed, somebody somewhere) may not agree with your choice of prior. One way to increase the credibility of your results is to repeat the analysis under a variety of priors, and report how the results differ as a result. This process is called prior sensitivity analyis.

At a minimum you should always report your choice of model and prior. If you include a sensitivity analysis, select one or more alternative priors and describe how the results of the analysis change. If they are sensitive to the choice of prior, you will likely have to explain both sets of results, or at least explain why you favor one prior over another. If the results are not sensitive to the choice of prior, this is evidence that the data are strongly driving the results. It suggests that different investigators coming from different backgrounds should come to the same conclusions.

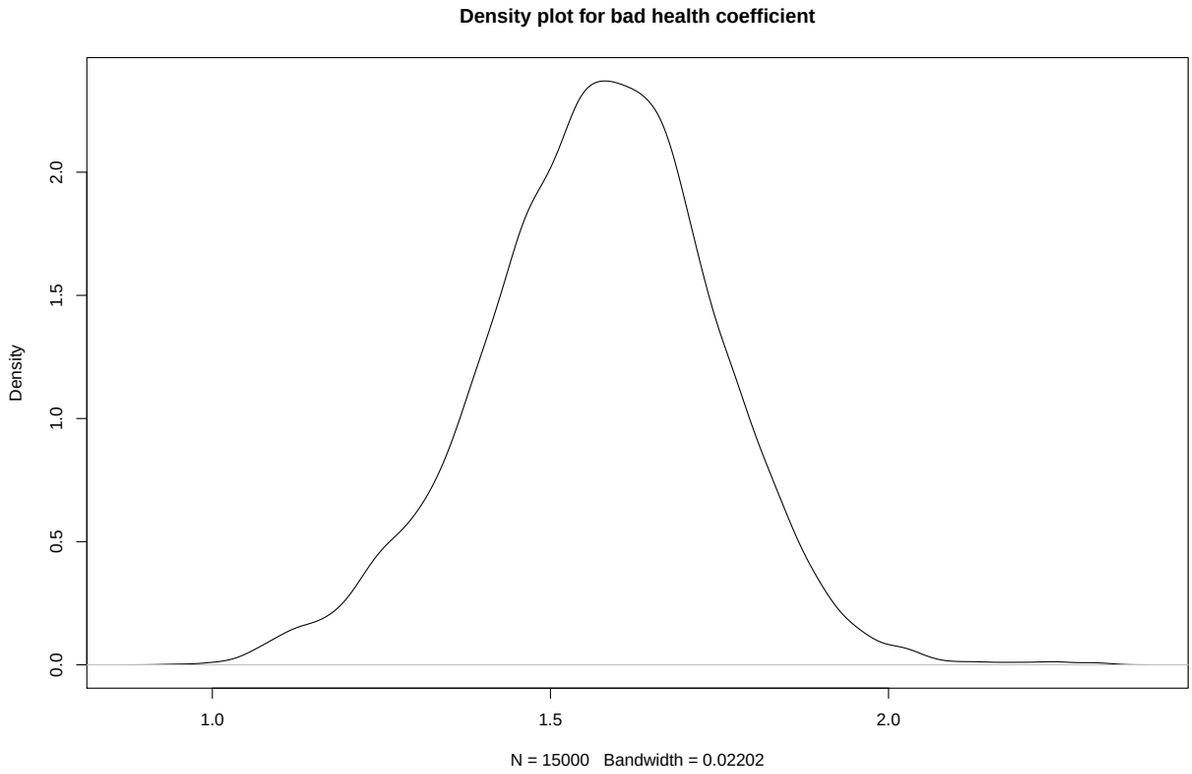
If the purpose of your analysis is to establish a hypothesis, it is often prudent to include a "skeptical" prior which does not favor the hypothesis. Then, if the posterior distribution still favors the hypothesis despite the unfavorable prior, you will be able to say that the data substantially favor the hypothesis. This is the approach we will take in the following example, continued from the previous lesson.

12.1 Example

Let's return to the example of number of doctor visits (Poisson regression). We concluded from our previous analysis of these data that both bad health and increased age are associated with more visits. Suppose the burden of proof that bad health is actually associated with more visits rests with us, and we need to convince a skeptic.

First, let's re-run the original analysis and remind ourselves of the posterior distribution for the badh (bad health) indicator.

```
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 1127
   Unobserved stochastic nodes: 4
   Total graph size: 3673

Initializing model
```

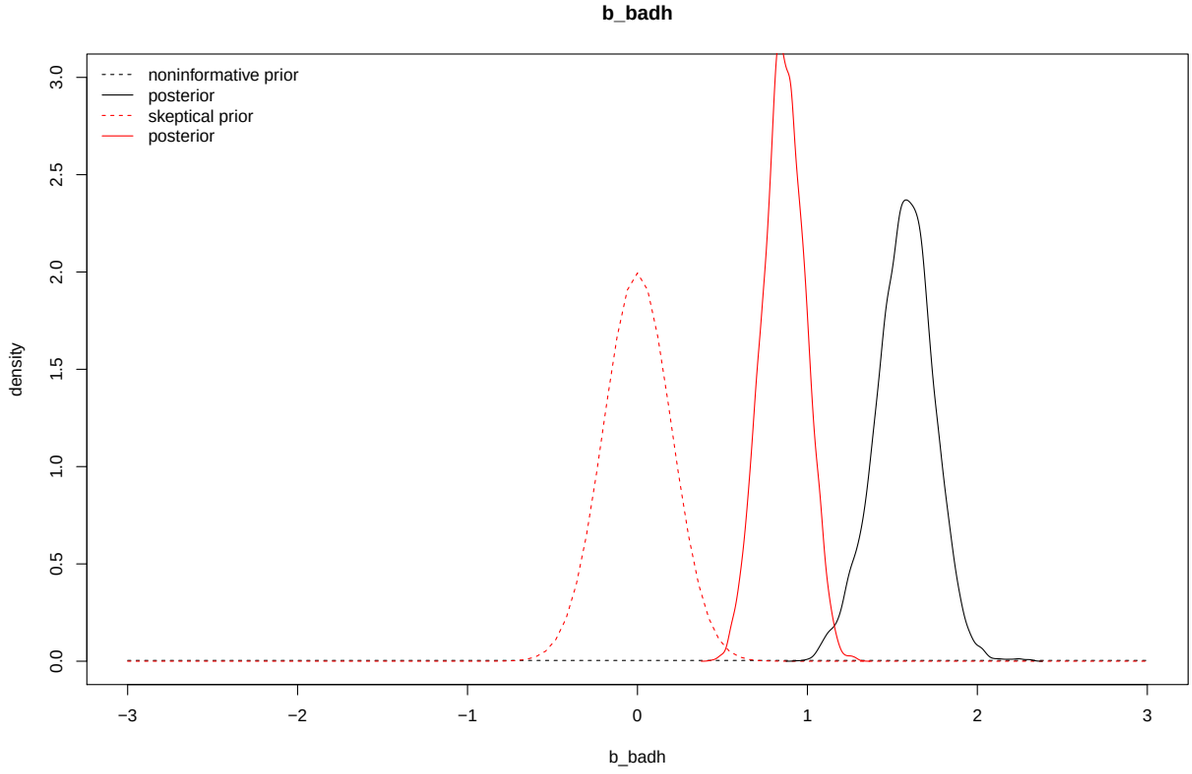


Essentially all of the posterior probability mass is above 0, suggesting that this coefficient is positive (and consequently that bad health is associated with more visits). We obtained this result using a relatively noninformative prior. What if we use a prior that strongly favors values near 0? Let's repeat the analysis with a normal prior on the badh coefficient that has mean 0 and standard deviation 0.2, so that the prior probability that the coefficient is less than 0.6 is >0.998>0.998. We'll also use a small variance on the prior for the interaction term involving badh (standard deviation 0.01 because this coefficient is on a much smaller scale).

```
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 1127
   Unobserved stochastic nodes: 4
```

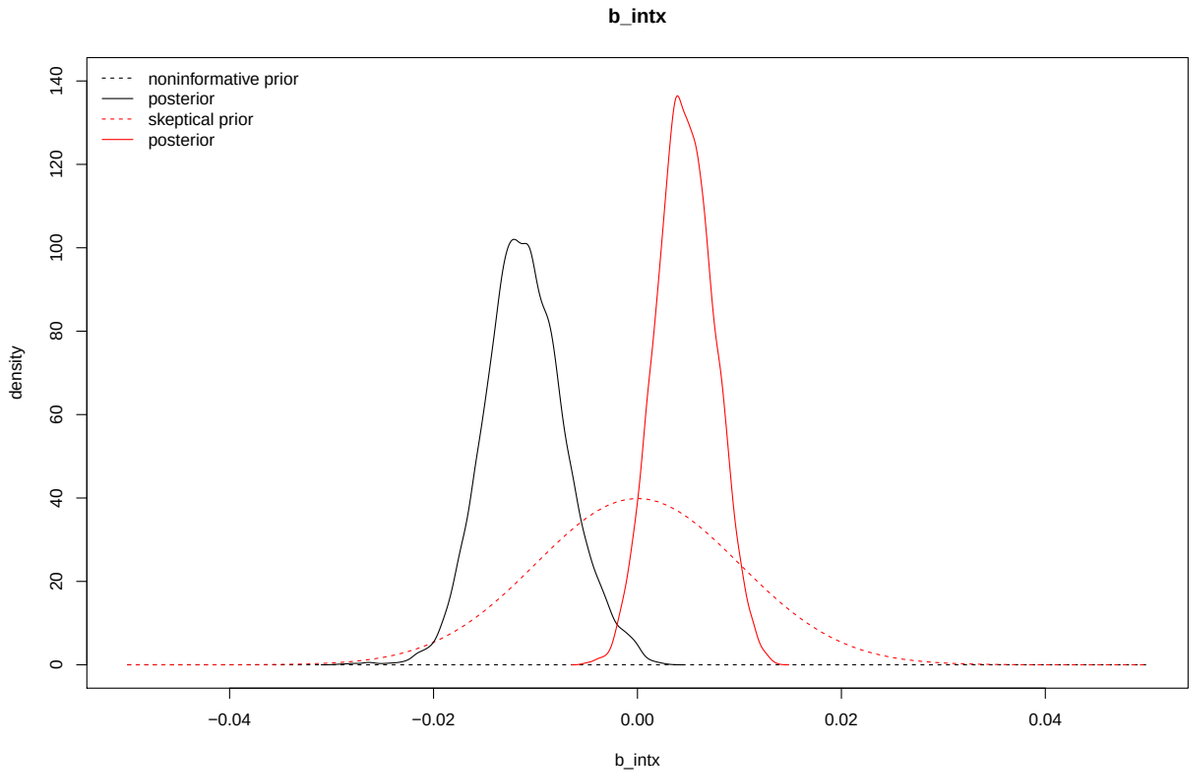
```
Total graph size: 3679

Initializing model
```

How did the posterior distribution for the coefficient of badh change?



Under the skeptical prior, our posterior distribution for b_badh has significantly dropped to between about 0.6 and 1.1. Although the strong prior influenced our inference on the magnitude of the bad health effect on visits, it did not change the fact that the coefficient is significantly above 0. In other words: even under the skeptical prior, bad health is associated with more visits, with posterior probability near 1.

We should also check the effect of our skeptical prior on the interaction term involving both age and health.




```
[1] 0.9490667
```

The result here is interesting. Our estimate for the interaction coefficient has gone from negative under the noninformative prior to positive under the skeptical prior, so the result is sensitive. In this case, because the skeptical prior shrinks away much of the bad health main effect, it is likely that this interaction effect attempts to restore some of the positive effect of bad health on visits. Thus, despite some observed prior sensitivity, our conclusion that bad health positively associates with more visits remains unchanged.

Bibliography

Efron, B., 2013. Bayes theorem in the 21st century. Science 340, 1177–1178.