Introductory Bayesian Course

Dr. Joseph L. Thorley

October 20th, 2014

Contents

1 Background

The purpose of these course notes is to introduce participants to Bayesian analysis with R, RStudio and JAGS. It is assumed that participants are familiar with R and RStudio as covered in the Introductory R Course notes at http://www.poissonconsulting.ca/course/2014/09/12/an-introduction-to-r-course.html.

1.1 Licence

The notes, which are released under a CC BY 4.0 license, are a draft of the material to be presented at the Introductory Bayesian Course in Kelowna on November 20th-21st, 2014. They were written by Dr. Joseph Thorley R.P.Bio..

1.2 Installation

If you haven't already done so, download the the most recent version of the R base distribution binary for your platform from http://cran.r-project.org/ and install using the default options. Next download and install RStudio from http://www.rstudio.com/products/rstudio/download/ using the default options. Then, download JAGS from http://sourceforge.net/projects/mcmc-jags/files/JAGS/ and install with the default options.

To make sure you have all the required packages installed on your hard drive execute the following code at the command line

```
install.packages("devtools", quiet = TRUE)
library(devtools)

install.packages("dplyr", quiet = TRUE)

install.packages("ggplot2", quiet = TRUE)
install.packages("scales", quiet = TRUE)

install_github("poissonconsulting/tulip@v0.0.11")
install_github("poissonconsulting/datalist@v0.4")
install_github("poissonconsulting/juggler@v0.1.3")
install_github("poissonconsulting/jaggernaut@v2.0.0")
```

Then start any scripts with

```
library(dplyr)
library(ggplot2)
library(scales)
library(jaggernaut)
```

1.3 Bayesian and Frequentist Statistical Analysis

Statistical analysis uses probability models to provide bounded estimates of parameter values (θ) from the data (y).

There are two primary approaches to statistical analysis: Bayesian and frequentist. As far as a frequentist is concerned the best estimates of θ are those values that maximise the *likelihood* which is the probability of the data given the estimates, i.e., $p(y|\theta)$. A Bayesian on the other hand chooses the values with the highest *posterior* probability - that is to say the probability of the estimates given the data , i.e., $p(\theta|y)$.

1.3.1 Coin Flips

Consider the case where $n = 10$ flips of a coin produce $y = 3$ tails. We can model this using a binomial distribution

$$y \sim dbin(\theta, n)$$

where θ is the probability of throwing a head.

1.3.1.1 Maximum Likelihood

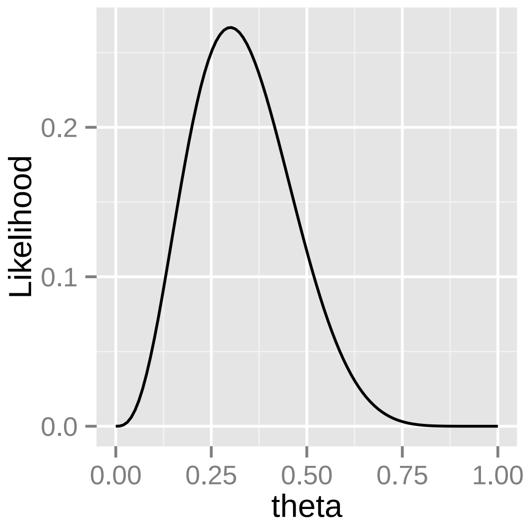
The likelihood for the binomial model is given by the following equation

$$p(y|\theta) = \binom{n}{y}\theta^y(1-\theta)^{n-y}$$

.

The likelihood values for different values of θ are therefore as follows

```
likelihood <- function(theta, n = 10, y = 3) {
    choose(n, y) * theta^y * (1 - theta)^(n - y)
}
theta <- seq(from = 0, to = 1, length.out = 100)

qplot(theta, likelihood(theta), geom = "line", ylab = "Likelihood", xlab = "theta")
```



The estimate $(\hat{\theta})$ is the value of θ with the maximum likelihood (ML) value, which in this case is 0.3.

A 95% confidence interval (CI) can then be calculated using the asymptotic normal approximation

$$\hat{\theta} \pm 1.96 \frac{1}{\sqrt{I(\hat{\theta})}}$$

where $I(\hat{\theta})$ is the expected second derivative of the log-likelihood at the estimate. This calculation is based on the assumption that the sample size is of sufficient size that the likelihood is normally distributed.

In the current case,

$$I(\hat{\theta}) = \frac{n}{\hat{\theta}(1-\hat{\theta})}$$

which gives a point estimate of 0.3 and lower and upper 95% confidence intervals of 0.02 and 0.58 respectively.

1.3.1.2 Posterior Probability

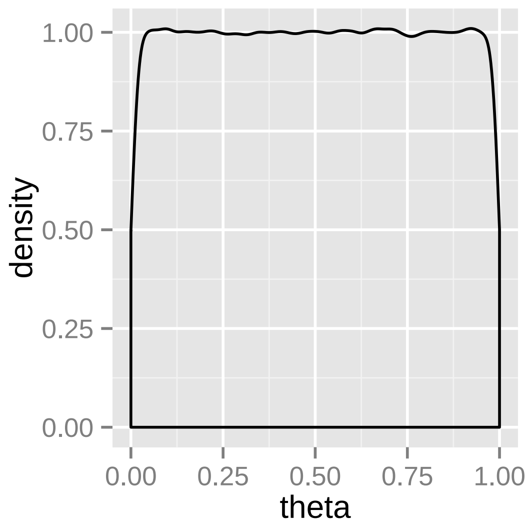
The posterior probability on the other hand is given by Bayes rule which states that

$$p(\theta|y) \propto p(y|\theta)p(\theta)$$

where $p(\theta)$ is the prior probability.

Bayesians consider this an advantage because prior information can be incorporated into an analysis while frequentists consider it subjective. In most cases Bayesians use *low-information* priors which have very small effects on the posteriors. For example a uniform distribution with a lower limit of 0 and an upper limit of 1 $(dunif(0,1))$ is commonly used for probabilities.

```
qplot(runif(10^6, 0, 1), geom = "density", xlab = "theta")
```



As it is generally not possible to calculate the posterior probability, the posterior probability distribution is sampled using Markov Chain Monte Carlo (MCMC) algorithms such as Gibbs Sampling.

1.3.1.2.1 Gibbs Sampling Consider the case where the parameters $\theta = (\theta_1, \theta_2, \ldots, \theta_k)$ then Gibbs Sampling proceed as follows

Step 1 Choose starting *initial* values for $\theta_1^{(0)}$ and $\theta_2^{(0)}$

Step 2 Sample $\theta_1^{(1)}$ from $p(\theta_1|\theta_2^{(0)}, y)$
Sample $\theta_2^{(1)}$ from $p(\theta_2|\theta_1^{(1)}, y)$

Step 3 *Iterate* step 2 thousands (or millions) of times to obtain a sample from $p(\theta|y)$.

1.4 JAGS and BUGS

Programming an efficient MCMC algorithm for a particular model is outside the scope of most research projects. Fortunately, JAGS (which stands for Just Another Gibbs Sampler) can take a dataset and a model specified in the simple but flexible BUGS language (which stands for Bayesian Analysis Using Gibbs Sampling) and perform MCMC sampling for us.

In order to do this we will use the `jaggernaut` package to talk to the standalone JAGS program via the `rjags` package.

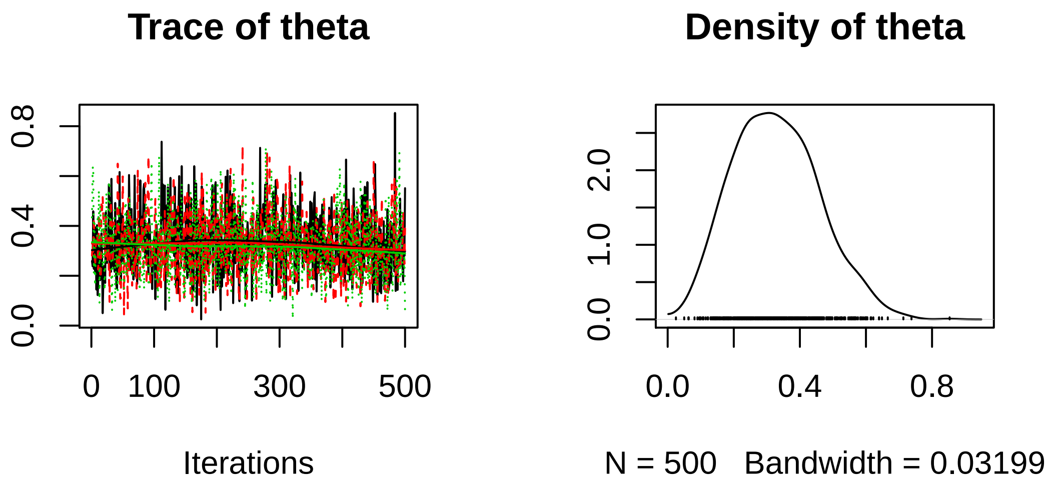
First we need to specify the underlying probability model in the BUGS language and save it as an object of class `jags_model`.

```
model1 <- jags_model("model {
  theta ~ dunif(0, 1)
  y ~ dbin(theta, n)
}")
```

then we call JAGS using `jaggernaut` in the default report mode to generate samples from θ's posterior probability distribution.

```r
data <- data.frame(n = 10, y = 3)
analysis1 <- jags_analysis(model1, data = data)
```

```r
plot(analysis1)
```



```r
coef(analysis1)
```

```
##        estimate      lower     upper      sd error significance
## theta 0.3279413 0.09894562 0.5993426 0.13029    76            0
```

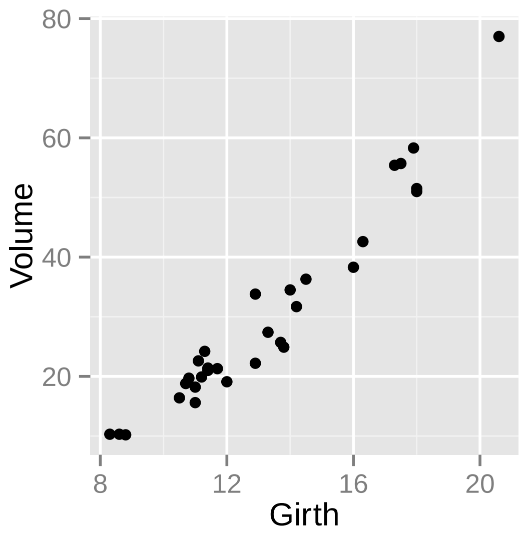
The model output indicates that the point estimate (in this case the mean of the samples) is 0.33 and the 95% credible interval (in this case the 2.25th and 97.75th percentiles) is 0.1 to 0.6.

Exercise 1 *Previous studies indicate that the coin was definitely biased towards tails. Modify the prior distribution accordingly and rerun the above model. How does the posterior distribution change?*

1.5 Black Cherry Trees

The `trees` data set in the dataset package provides information on the girth and volume of 31 black cherry trees.

```r
qplot(x = Girth, y = Volume, data = trees)
```



Algebraically, the linear regression of `Volume` against `Girth` can be defined as follows

$$Volume_i = \alpha + \beta * Girth_i + \epsilon_i$$

where α is the intercept and β is the slope and the error terms (ϵ_i) are drawn from a normal distribution with an standard deviation of σ.

The model can be defined as follows in the BUGS language where `<-` indicates a *deterministic* as opposed to *stochastic* node (which is indicated by `~`).

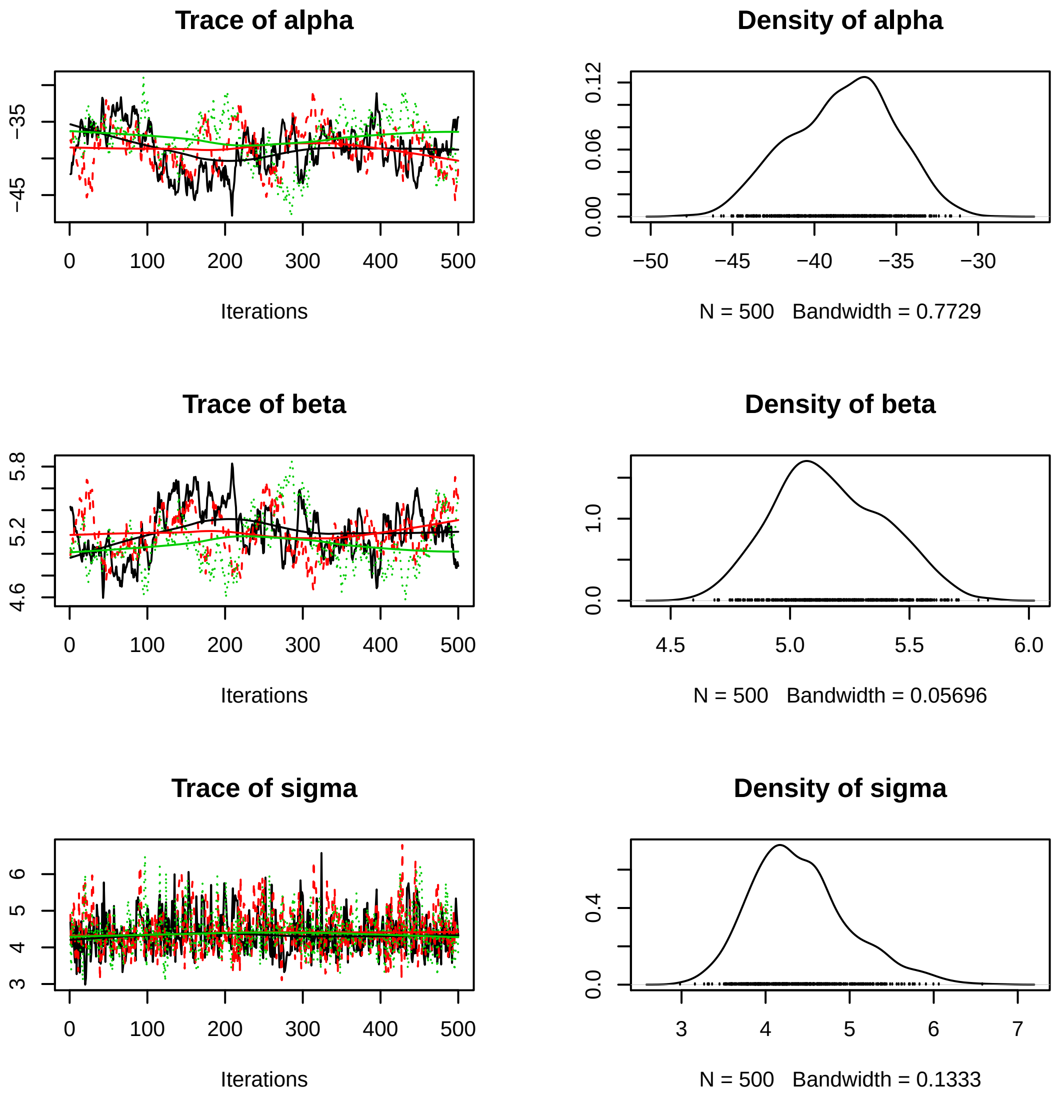
```
model1 <- jags_model("model {
  alpha ~ dnorm(0, 50^-2)
  beta ~ dnorm(0, 10^-2)
  sigma ~ dunif(0, 10)

  for(i in 1:length(Volume)) {
    eMu[i] <- alpha + beta * Girth[i]
    Volume[i] ~ dnorm(eMu[i], sigma^-2)
  }
}")
```

The standard deviations of the normal distributions are raised to the power of `-2` because (for historical reasons) Bayesians quantify variation in terms of the *precision* (τ) as opposed to the variance (σ^2) or standard deviation (σ) where $\tau = 1/\sigma^2$.

The resultant trace plots and coefficients are as follows

```
data(trees)
analysis1 <- jags_analysis(model1, data = trees)
```

```
plot(analysis1)
```



```
coef(analysis1)
```

```
##          estimate      lower      upper      sd error significance
## alpha -38.292346 -44.499492 -32.697578 3.14810    15            0
## beta    5.165349   4.754296   5.633272 0.23201     9            0
## sigma   4.413185   3.428491   5.773842 0.58716    27            0
```

1.5.1 Predictions and Residuals

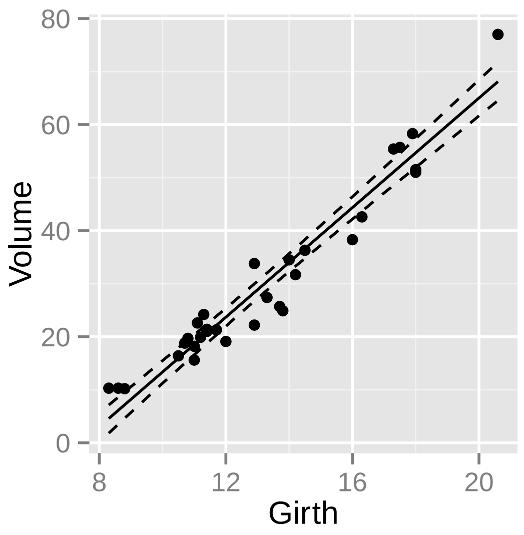
Many researchers estimate fitted values, predictions and residuals by monitoring additional nodes in their model code. The disadvantages of this approach are that:

- the model code becomes more complicated.
- the MCMC sampling takes longer.
- adding derived parameters requires a model rerun.
- the table of parameter estimates becomes unwiedly.

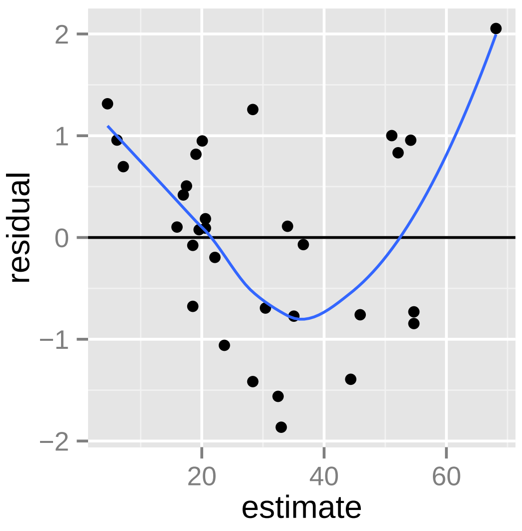
jaggernaut overcomes these problems by allowing derived parameters to be defined in a separate chunk of BUGS code as demonstrated below.

```
dcode <- "data {
  for(i in 1:length(Volume)) {
    prediction[i] <- alpha + beta * Girth[i]
  }
  residual <- (Volume - prediction) / sigma
}"

prediction <- predict(analysis1, newdata = "Girth", derived_code = dcode)
```

```
gp <- ggplot(data = prediction, aes(x = Girth, y = estimate))
gp <- gp + geom_point(data = dataset(analysis1), aes(y = Volume))
gp <- gp + geom_line()
gp <- gp + geom_line(aes(y = lower), linetype = "dashed")
gp <- gp + geom_line(aes(y = upper), linetype = "dashed")
gp <- gp + scale_y_continuous(name = "Volume")

print(gp)
```



```
fitted <- fitted(analysis1, derived_code = dcode)
fitted$residual <- residuals(analysis1, derived_code = dcode)$estimate

qplot(estimate, residual, data = fitted) + geom_hline(yintercept = 0) +
    geom_smooth(se = FALSE)
```



As discussed in the R course notes the relationship between Volume and Girth is expected to be allometric because the cross-sectional area at an given point scales to the square of the girth (circumference).

Expressed as an allometric relationship the model becomes

$$Volume_i = \alpha * Girth_i^{\beta} * \epsilon_i$$

which can be reparameterised as a linear regression by log transforming

$$log(Volume_i) = \alpha + \beta * log(Girth_i) + \epsilon_i$$