Fitting models with JAGS

Statistics and Machine Learning II

Luis Da Silva

April 28, 2019

1 Sampling, JAGS and Seeds

As opposed to the frequentists framework in which model parameters are assumed to have only one "ground truth" and thus techniques such as the Maximum Likelihood Expectation (MLE) are used for estimating them, the Bayesian framework treats all unknown quantities as random variables described by a distribution. The computing of these distributions (i.e. posterior) requires a prior (i.e. set of believes about the phenomenon) and a likelihood (i.e. data). To do this, most non-trivial models requires the computation of integrals which are intractable (Yildirim, 2012), and that's when Monte Carlo Markov Chain (MCMC) techniques offer a way to do inference about those models.

Based on the idea that, by sampling, one can estimate any statistic of a posterior distribution, MCMC methods allows sampling from a probability distribution and then computing the statistics of interest from sample-based approximations (Rogers and Girolami, 2017). As an example, in a binomial distribution, the probability of obtaining a success may be approximated by equation 1 if $p(\mathbf{y}|\delta)$ is the sample-based density of estate $\mathbf{y}$ given δ features.

$$P(T = 1|\delta) = \int f(\mathbf{y})p(\mathbf{y}|\delta)d\mathbf{y} \tag{1}$$

Just another Gibbs sampler (JAGS), developed by Martin Plummer, is a program written in C++ that allows the implementation of MCMC methods

Table 1: Seeds dataset. r is the number of seeds that germinated out of n sown (BUGS, 2009).

seed O. aegyptiaco 75						*seed O. aegyptiaco 73*					
Bean			Cucumber			Bean			Cucumber		
r	n	r/n	r	n	r/n	r	n	r/n	r	n	r/n
10	39	0.26	5	6	0.83	8	16	0.50	3	12	0.25
23	62	0.37	53	74	0.72	10	30	0.33	22	41	0.54
23	81	0.28	55	72	0.76	8	28	0.29	15	30	0.50
26	51	0.51	32	51	0.63	23	45	0.51	32	51	0.63
17	39	0.44	46	79	0.58	0	4	0.00	3	7	0.43
			10	13	0.77						

across a wide range of platforms. In this report JAGS is used via its R integration for computing samples from the "Seeds" dataset (taken from Table 3 of Crowder, 1978), which concerns the proportion of seeds, categorized by type of seed and root extract, that germinated (BUGS, 2009). Table 1 shows the data for each of the 21 plates studied.

2 Model set up

Among many others, Seeds dataset presents the questions "which seeds are expected to germinate?" and "what is the expected probability of germination for a given seed?". One may give an answer to the first question without modelling the problem and just visualizing the data: with 20% jitter, data looks like it is shown in Figure 1. The size and color of the points show the proportion of germinated seeds (p) for each configuration. For the purpose of this report, a seed is expected to germinate if its probability of germination is ≥ 0.50. Thus, by approximating the germination probability by its sample germination proportion, one may see by eye that the only group that shows groups of seeds with germination probabilities consistently ≥ 0.50 is the one with O. aegyptiaco 75 seeds and Cucumber root extract.

Nevertheless, one is able to do this because there is little data. The second question may simply be answered by computing the mean proportion of each group, but again, as there is little data, this estimator is expected to have high variance. This situation may be improved by assuming:

$$logit(p_i) = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \beta_1 2 x_{1i} x_{2i} + b_i \tag{2}$$

Figure 1: Dataset with 20% jitter

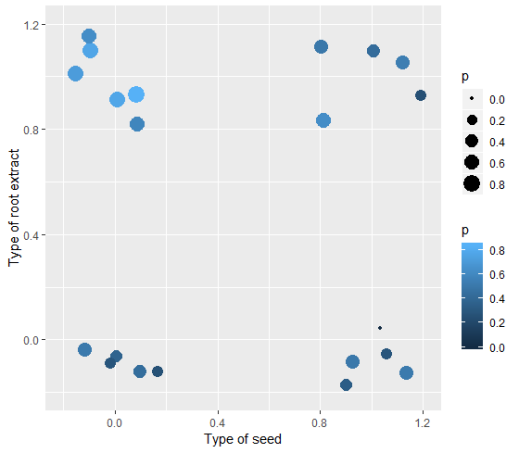


Table 2: Modelled probabilities for each group

75 Bean	75 Cucumber	73 Bean	73 Cucumber
37%	72%	33%	47%

where x_1 is a binary variable encoding the two seed types, x_2 encodes the kinds of root extract and b_i is the modelling error. The frequentist approach will get the MLE for each β by minimizing a loss function like $\sum_{i=1}^{N} b_i^2$ and will end up with the equation:

$$logit(p_i) = -0.53 - 0.20x_{1i} + 1.45x_{2i} - 0.85x_{1i}x_{2i} + b_i \tag{3}$$

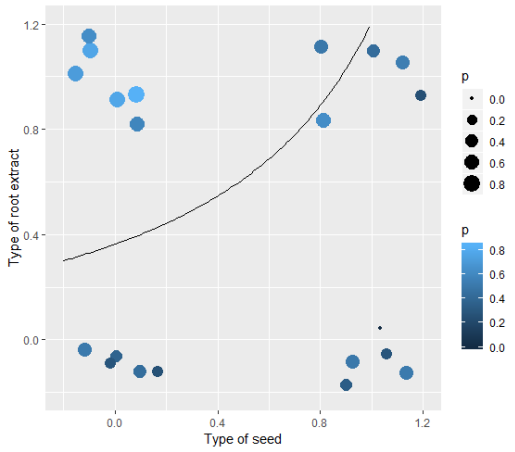
Table 2 presents the probabilities for each group given the above model and Figure 2 shows the decision boundary drawn by the model at 50% (the same conclusion remains). Remember that the data is jittered for visualization purposes, and thus the 50% actually isolates the upper left group perfectly.

On Bayesian statistics, all unknowns are treated as random variables, thus one must make additional assumptions on their distribution. First, as a seed can only germinate or not,

$$r_i \backsim Binomial(p_i, n_i)$$

with p_i being the probability of germination on the i plate. Then, from

Figure 2: Data with 50% MLE decision boundary



equation 2, b_i is assumed to be distributed as

$$b_i \backsim Normal(0, \tau)$$

In addition, as no knowledge is available on parameters $\beta_0, \beta_1, \beta_2, \beta_{12}$ and τ, they are given fairly non-informative priors:

$$\beta_0, \beta_1, \beta_2, \beta_{12} \backsim Normal(0, 1e-6)$$

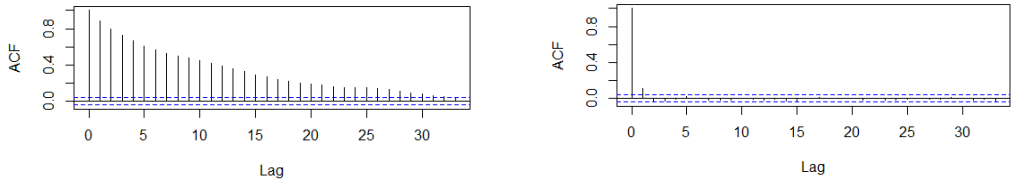
$$\tau \backsim Gamma(0.001, 0.001)$$

3 Results

Before sampling from JAGS, and as sample initializations is random, a burn-in of 2000 samples is made to avoid sampling from a non-representative distribution. Then, 200 sets of betas are sampled. Even then, because of how MCMC methods work, samples may be highly autocorrelated, making our sampling of poor quality. Figure 3a shows the autocorrelation issue for this first sampling. This problem is easily (although expensively) solved by a procedure called thinning, in which only one every k samples is selected.

As autocorrelation in Figure 3a seems to be high until about lag 30, a new set of 200 samples are calculated, but this time with a thinning of 30 samples (thus the procedure requires the computation of $200 * 30 = 6000$ samples). Figure 3b shows that the autocorrelation problem is mostly solved.

Figure 3: Sampling autocorrelation



(a) No thinning (b) Thinning every 30 samples

Table 3: Summary statistics from sampled βs

	Mean	SD
beta0	-0.52491	0.1942
beta1	0.08097	0.3206
beta12	-0.81370	0.4600
beta2	1.32796	0.2614

By comparing equation 3 with table 3 one is able to spot some variations between frequentist and Bayesian approach to modelling. Figure 4 gives more insight into these by presenting the density and trace for every β in equation 2. These distributions are quite wide, with some βs varying between positive and negative, and thus meaning that there are several possible solutions. This is expected because the actual values of x_1, x_2 in groups may only lay in $\{0, 1\}$, therefore there are many ways of splitting them in a two dimensional continuous space. As an illustration, Figure 5 shows 5 sampled 50% logistic thresholds by assigning 5 sampled sets of betas to equation 2. It is specially interesting to notice that 2 out of these 5 thresholds also include the 73 Cucumber seeds as the ones that are expected to germinate. After all, 60% of these seeds indeed germinated according to the original data.

A final issue one might want to assess is whether the MCMC procedure has converged to sample from the objective distribution. Figure 6 shows the Gelman-Rubin convergence diagnostic plot for every β. The shrinking factor is very close to 1 in every β (the highest final value is around 1.005), thus convergence seems to have been achieved.

Figure 4: Sampled betas' distribution

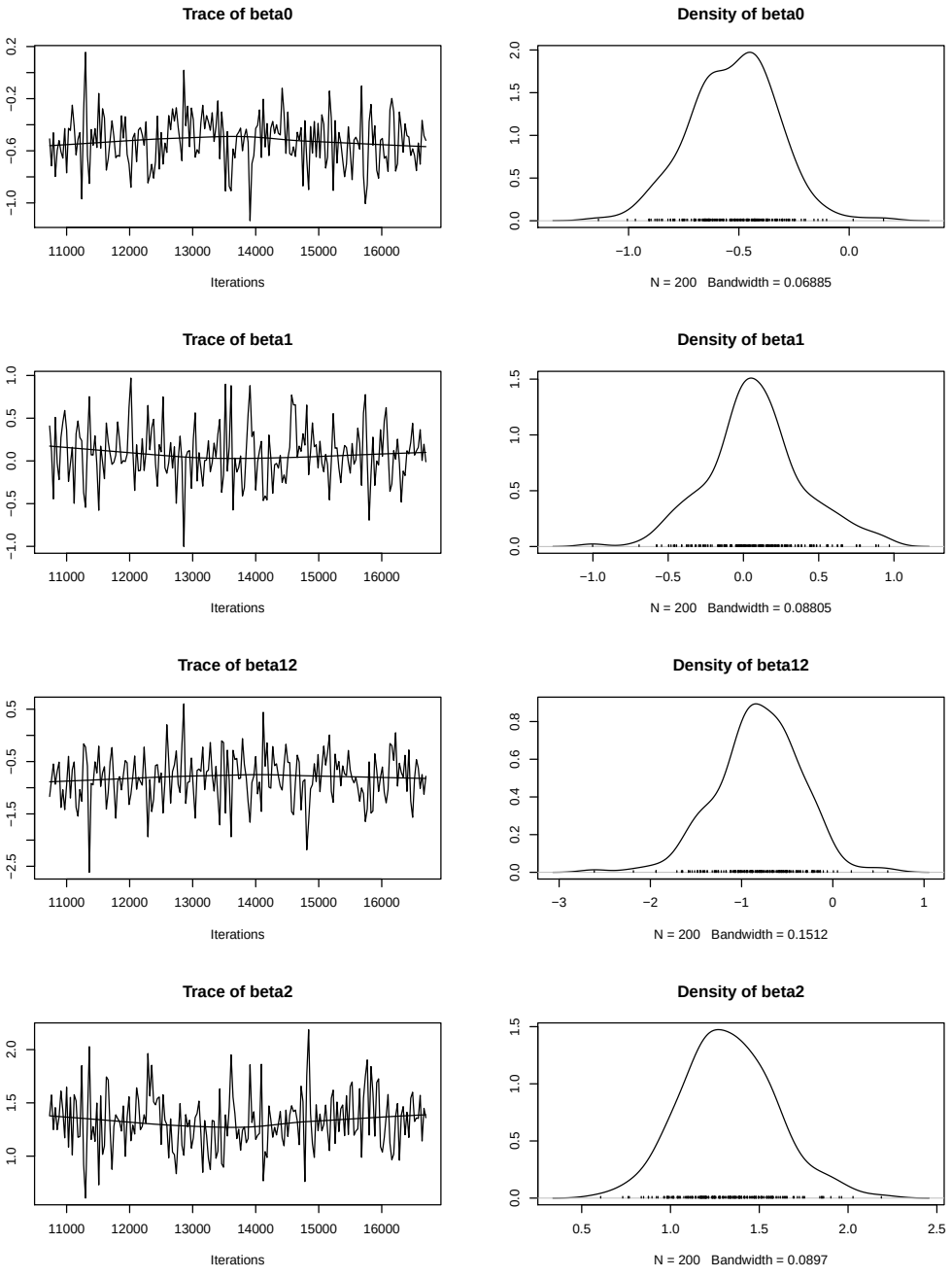


Figure 5: 5 sampled logistic 50% thresholds

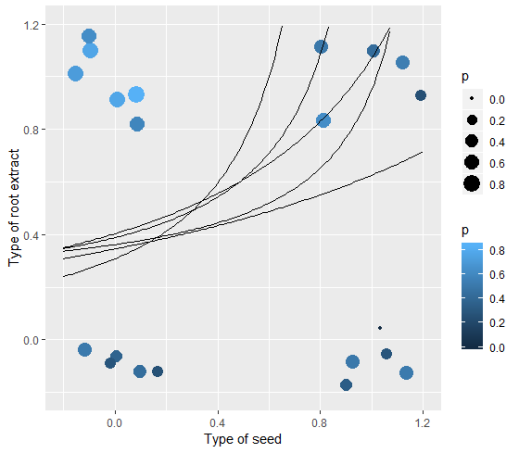
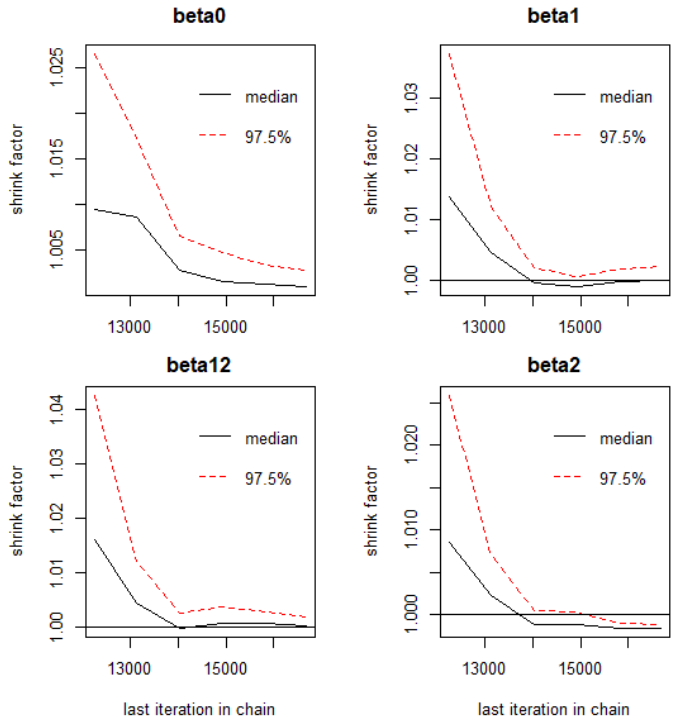


Figure 6: Gelman-Rubin convergence diagnostic plot



4 Conclusion

Although frequentist statistics are easier to code, compute and interpret, these adopt a deterministic approach to modelling that may be sometimes misleading as it does not allow the user to see the full picture. Bayesian statistics, on the other hand, allow to get more information from data (although maybe biasing it a little bit by choosing a wrong prior) and thus produces a more useful model.

5 R code

```
# Fitting models with JAGS
# Dataset Seeds is available at
# http://www.openbugs.net/Examples/Seeds.html
# Classic-bugs tools at
# https://sourceforge.net/projects/mcmc-jags/files/Examples/2.x/

library("tidyverse")
library("rjags")
source("classic-bugs/R/Rcheck.R")

# Load data
data <- read.jagsdata("classic-bugs/vol1/seeds/seeds-data.R")
df <- tibble(r=data$r, n=data$n, x1=data$x1, x2=data$x2)
df$p <- df$r / df$n
df

# Plot data
main_plot <- ggplot() +
  geom_jitter(aes(x=x1, y=x2, color=p, size=p), data = df,
  width = 0.2, height = 0.2) +
  xlim(-0.2,1.2) + ylim(-0.2,1.2) + xlab("Type␣of␣seed") +
  ylab("Type␣of␣root␣extract")
main_plot

# Build standard logistic regression
logic1 <- glm(p ~ x1*x2, data=df, family="quasibinomial")
summary(logic1)
```

```
# Plot logistic regression
add.thresh.line <- function(betas, thrs=0.5) {
  x2 <- 0
  x1 <- seq(-0.2, 1.2, 0.01)
  thrs <- -log(1/thrs - 1)
  for (i in 1:length(x1)) {
    x2[i] <- (thrs-betas[1]-betas[2]*x1[i]) / (betas[4]+
    betas[3]*x1[i])
  }
  return (geom_line(aes(x=x1, y=x2)))
}

coef <- c(logic1$coefficients[1], logic1$coefficients[2],
 logic1$coefficients[4], logic1$coefficients[3])
main_plot + add.thresh.line(coef)

# Use JAGS to sample and fit the same model
# Adapted from Matin Plummer's Test1
jags_str <- "model␣{
␣␣␣#␣Set␣prior␣distribution␣of␣coefficients
␣␣␣beta0␣␣~␣dnorm(0.0,1.0E-6);
␣␣␣beta1␣␣~␣dnorm(0.0,1.0E-6);
␣␣␣beta2␣␣~␣dnorm(0.0,1.0E-6);
␣␣␣beta12␣~␣dnorm(0.0,1.0E-6);

␣␣␣#␣Error␣variance
␣␣␣tau␣␣␣␣~␣dgamma(1.0E-3,1.0E-3);␣␣␣␣#␣1/sigma^2
␣␣␣sigma␣␣<-␣1.0/sqrt(tau);

␣␣␣#␣Get␣N␣fits
␣␣␣for␣(i␣in␣1:N)␣{
␣␣␣␣␣␣b[i]␣␣␣␣␣␣␣␣␣␣~␣dnorm(0.0,tau);
␣␣␣␣␣␣logit(p[i])␣<-␣beta0␣+␣beta1*x1[i]␣+beta2*x2[i]␣+
␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣beta12*x1[i]*x2[i]␣+␣b[i];
␣␣␣␣␣␣r[i]␣␣␣␣␣␣␣␣␣␣~␣dbin(p[i],n[i]);
␣␣␣}
}"

inits <- list("tau"=1, "beta0"=0, "beta1"=0, "beta2"=0,
```

```
"beta12"=0)

jags.obj <- jags.model(textConnection(jags_str),
        data=data, inits = inits, n.chains = 10,
        n.adapt = 2500)

# Do a burn-in and sample
update(jags.obj, 2000)
samples <- coda.samples(jags.obj, c('beta0','beta1',
        'beta2', 'beta12'), 200, 1)

# Autocorr plot
samples_matrix <- as.matrix(samples)
acf(samples_matrix[,1])

# Sample again but with an appropriate thining
samples <- coda.samples(jags.obj, c('beta0','beta1',
        'beta2', 'beta12'), 200*30, 30)
samples_matrix <- as.matrix(samples)
acf(samples_matrix[,1])
summary(samples[1])
plot(samples[1])
gelman.plot(samples)

# Plot several betas
several_plot <- main_plot
idx <- sample( 1:nrow(samples_matrix), 5)
for(i in idx) {
  several_plot <- several_plot +
  add.thresh.line(samples_matrix[i,] )
}
several_plot
```

References

BUGS (2009), "Seeds." URL `http://www.openbugs.net/Examples/Seeds.html`.

Rogers, Simon and Mark Girolami (2017), *A first course in Machine Learning*. CRC press.

Yildirim, Ilker (2012), "Bayesian inference: Gibbs sampling."