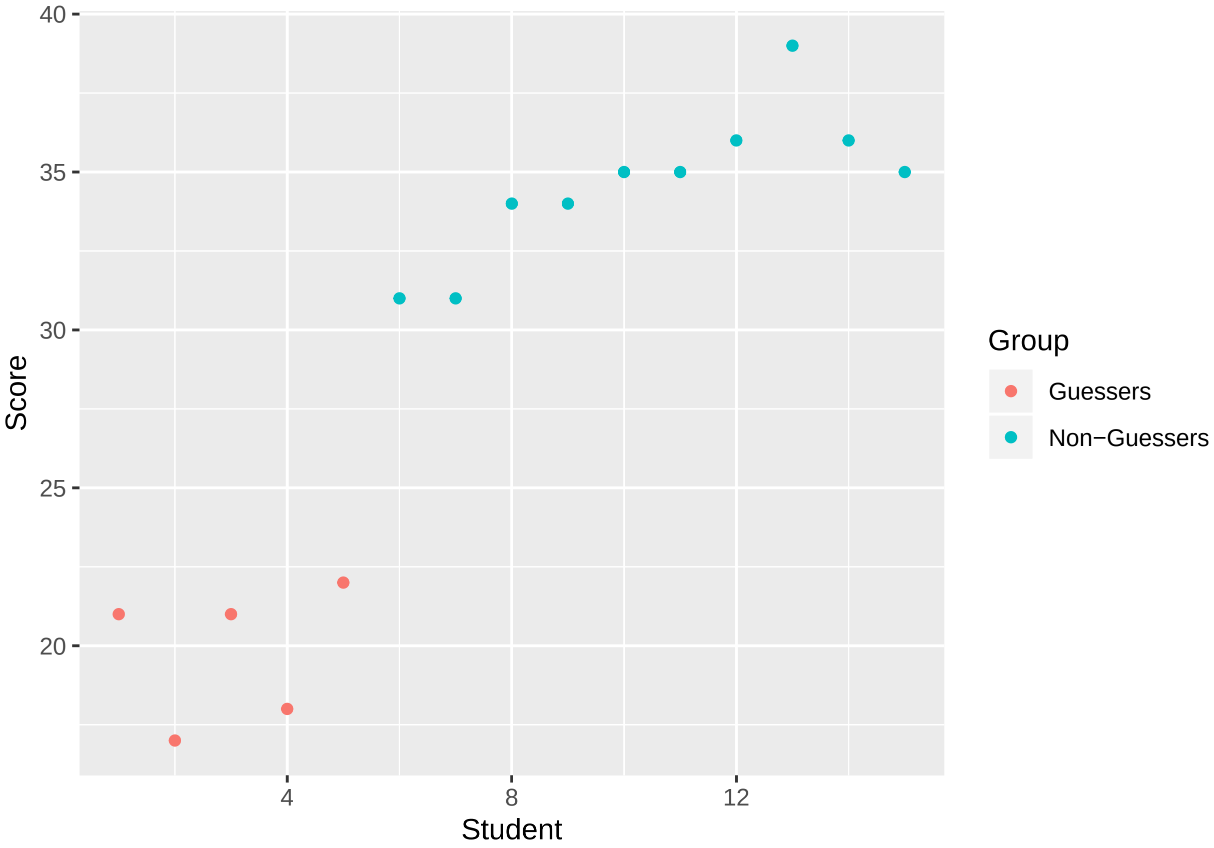
Case Study 2

Caroline Beech and Shashank Sule

25/11/2019

The Data

We suspect the first 5 test-takers just guessed their questions. So let's label them according to our suspicion.



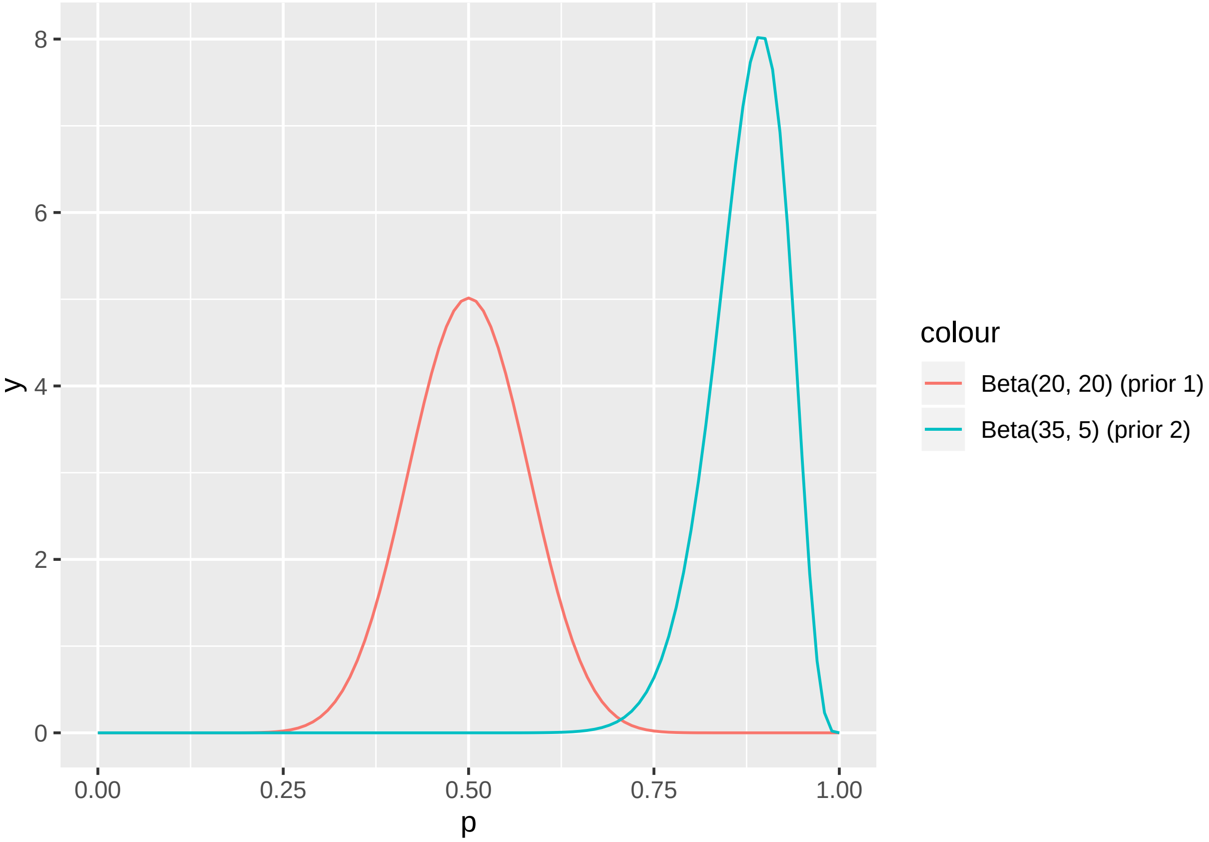
Hmm. Perhaps it is prudent to make two groups of students, suspected guessers and non-guessers with relatively sharp Beta priors and a binomial likelihood. Furthermore, the hyperpriors come from a poisson distribution (because np represents the rate at which the questions are answered out of n and it scales like a Poisson process).

Model

From the previous section, we have the following model:

$$
\begin{aligned}
Y_{ij} &\sim Binom(40, p_j) \\
p_1 &\sim \text{Beta}(n, n) \\
n &\sim \text{Poisson}(20) \\
p_2 &\sim \text{Beta}(m, 40 - m) \\
m &\sim \text{Poisson}(35)
\end{aligned}
$$

The expected priors:



```
## $y
## [1] "Density"
##
## attr(,"class")
## [1] "labels"
```

Now our tests will lend credence to the belief of whether or not the first group was guessing.

Simulation

```
modelString <-"
model {
## likelihood
for (i in 1:N){
y[i] ~ dbinom(p_j[x[i]], 40)
}

## priors
p_j[1] ~ dbeta(n,n)
p_j[2] ~ dbeta(m, 40-m)

## hyperpriors
n ~ dpois(n0)
m ~ dpois(m0)
}
"
```

```
y <- TrueFalseScores$Score
x <- TrueFalseScores$Suspect_Group
J <- length(unique(x))
N <- length(y)

initsfunction <- function(chain){
  .RNG.seed <- c(1,2)[chain]
  .RNG.name <- c("base::Super-Duper",
                 "base::Wichmann-Hill")[chain]
  return(list(.RNG.seed=.RNG.seed,
              .RNG.name=.RNG.name))
}

the_data <- list("y" = y, "x" = x, "N" = N, n0 = 20, m0 = 35)
```

```
posterior <- run.jags(modelString,
                      n.chains = 1,
                      data = the_data,
                      monitor = c("p_j", "n", "m"),
                      adapt = 1000,
                      burnin = 5000,
                      sample = 5000,
                      thin = 1,
                      inits = initsfunction)
```

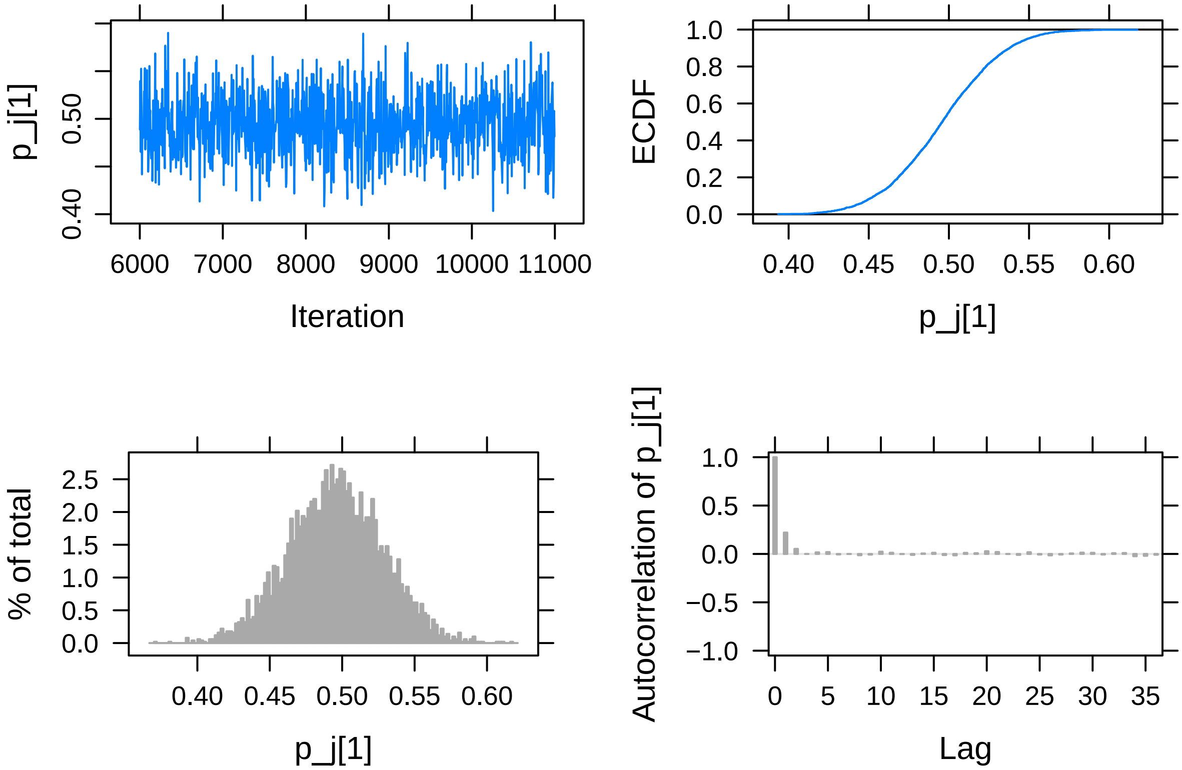
```
## Calling the simulation...
## Welcome to JAGS 4.3.0 on Mon Nov 25 14:59:59 2019
## JAGS is free software and comes with ABSOLUTELY NO WARRANTY
## Loading module: basemod: ok
## Loading module: bugs: ok
## . . Reading data file data.txt
## . Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 15
##    Unobserved stochastic nodes: 4
##    Total graph size: 39
## . Reading parameter file inits1.txt
## . Initializing model
## . Adapting 1000
## -------------------------------------------------| 1000
## ++++++++++++++++++++++++++++++++++++++++++++++++++ 100%
## Adaptation successful
## . Updating 5000
## -------------------------------------------------| 5000
## ************************************************** 100%
## . . . . Updating 5000
## -------------------------------------------------| 5000
## ************************************************** 100%
## . . . . Updating 0
## . Deleting model
## .
## Simulation complete.  Reading coda files...
## Coda files loaded successfully
## Calculating summary statistics...
## Finished running the simulation
```

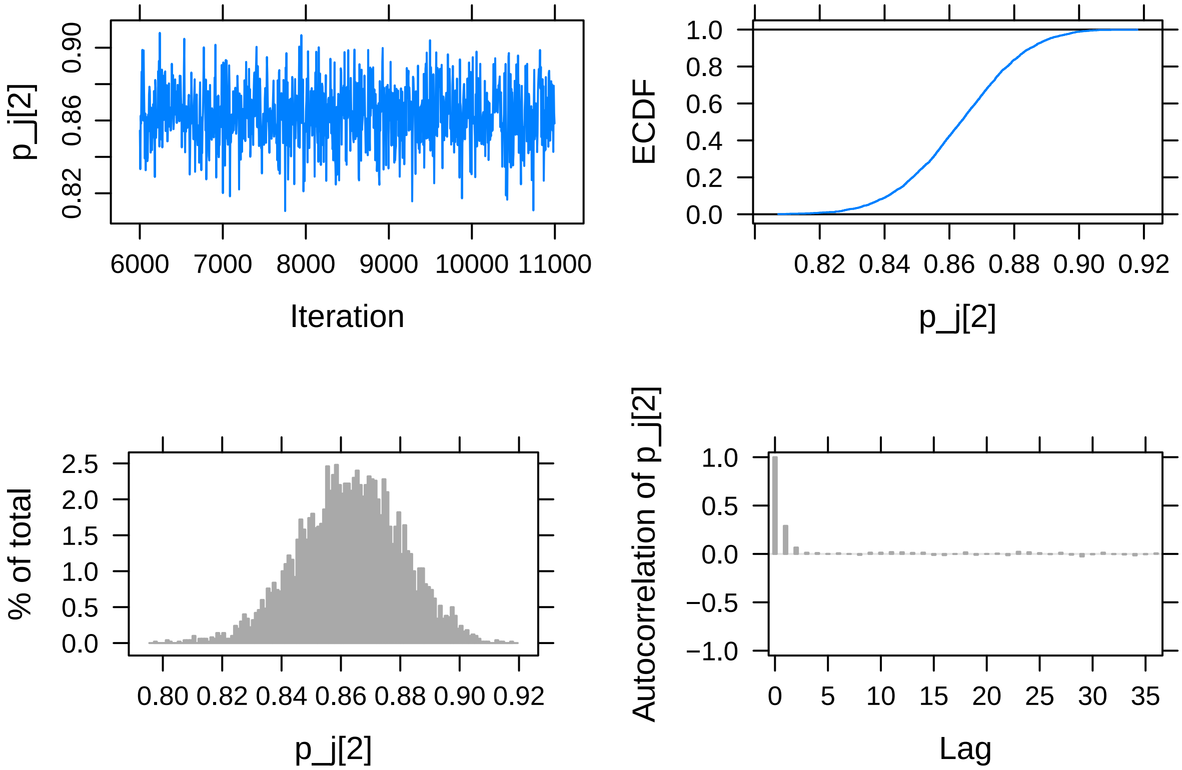
```
summary(posterior)
```

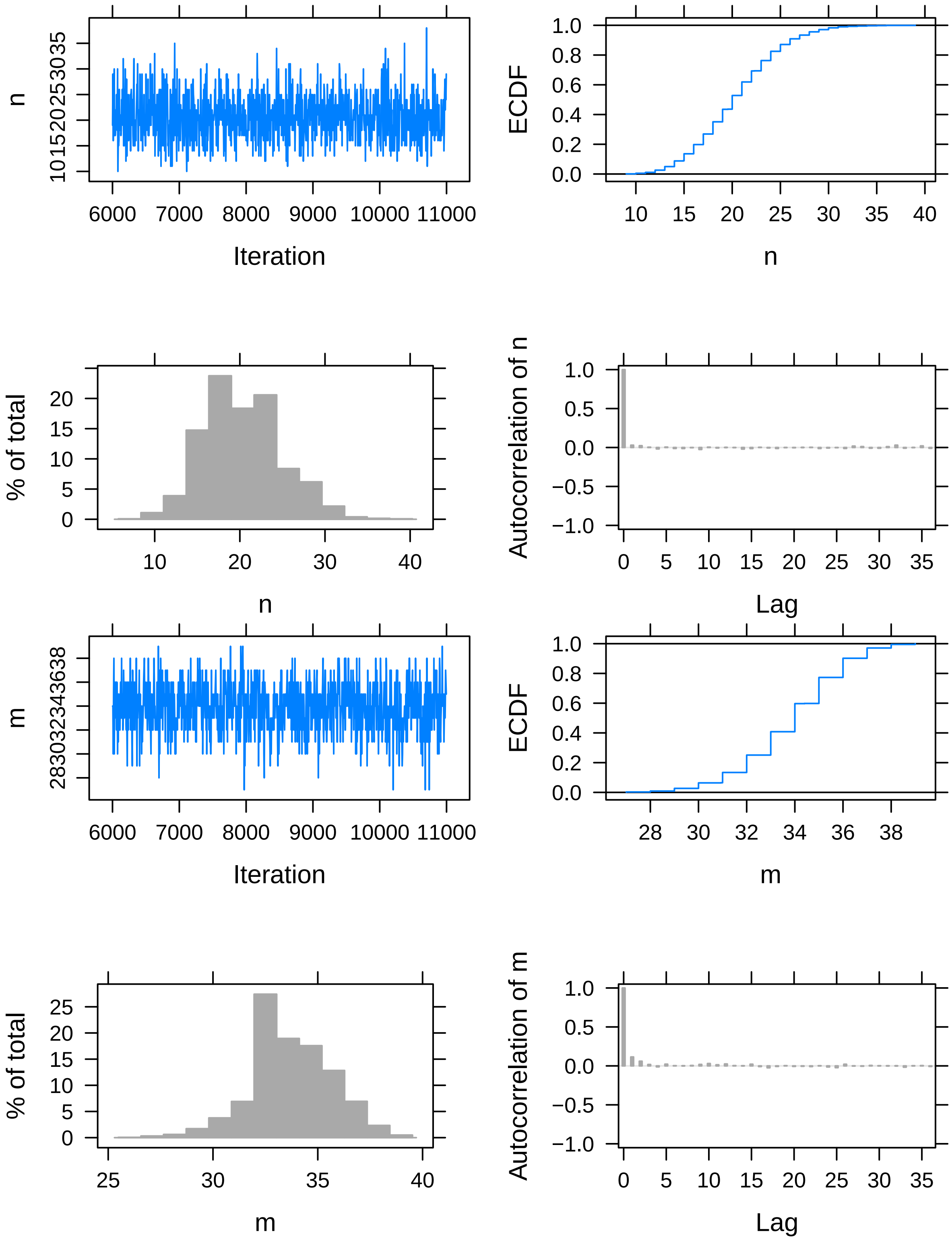
```
##          Lower95     Median   Upper95       Mean         SD Mode
## p_j[1]  0.432899  0.4958565  0.559048  0.4957228 0.03253322   NA
## p_j[2]  0.830661  0.8634900  0.897505  0.8631088 0.01716415   NA
## n      11.000000 20.0000000 28.000000 20.3876000 4.46700762   20
## m      29.000000 34.0000000 37.000000 33.8586000 2.12054365   34
##               MCerr MC%ofSD SSeff       AC.10 psrf
## p_j[1] 0.0005761358     1.8  3189 0.023454132   NA
## p_j[2] 0.0003273584     1.9  2749 0.013517244   NA
## n      0.0666393106     1.5  4493 0.004865179   NA
## m      0.0353153143     1.7  3606 0.030522814   NA
```

All 4 parameters converge well in MCMC (the highest SSeff is 4493 and all the parameters have rapidly decaying autocorrelation):

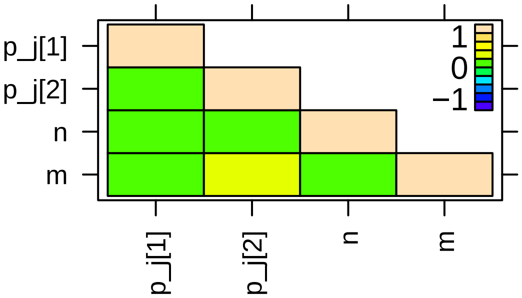
```
## Generating plots...
```







n
10
15
20
25
30
35
6000
7000
8000
9000
10000
11000
Iteration
ECDF
1.0
0.8
0.6
0.4
0.2
0.0
10
15
20
25
30
35
40
n
% of total
20
15
10
5
0
10
20
30
40
n
Autocorrelation of n
1.0
0.5
0.0
−0.5
−1.0
0
5
10
15
20
25
30
35
Lag
m
28
30
32
34
36
38
6000
7000
8000
9000
10000
11000
Iteration
ECDF
1.0
0.8
0.6
0.4
0.2
0.0
28
30
32
34
36
38
m
% of total
25
20
15
10
5
0
25
30
35
40
m
Autocorrelation of m
1.0
0.5
0.0
−0.5
−1.0
0
5
10
15
20
25
30
35
Lag



Analysis

Gathering the data from the posterior, we get the following 90% credible intervals for p_1 and p_2:

```
##         Lower95    Median  Upper95
## p_j[1] 0.432899 0.4958565 0.559048
## p_j[2] 0.830661 0.8634900 0.897505
```

Thus, 90% of the time the first group behaves like it is almost guessing while the second group behaves like it has some knowledge of the problems.