BDA - Assignment 9

Anonymous

23 11 2019

Contents

Used libraries:

```
library(dplyr)
library(rstan)
library(bayesplot)
library(aaltobda)
library(foreach)
library(knitr)
data("factory")
```

```
options(mc.cores = parallel::detectCores())
rstan_options(auto_write = TRUE)
```

Preface

I am not numbering the tasks in 1-5 order, but rather in the order it makes most sense, and corresponds to the rubric. I hope this makes it easier to read and grade.

Hierarchical model, factory data with Stan: Model run and check.

NOTE:I will write the model in a stan file, run it, and save the run in a single RDS file. That way I won't have to run the model each time I need to knit/fix/knit/etc.

CONCERNING THE STAN CODE - Idea for the model is as follows:

Hierarchical: Groups have their own mu_H which are pulled from N(mu_0_H, sigma_0_H) with uniform hyperpriors mu_0_H and sigma_0_H. In addition, all groups share sigma_H. Posterior predictions are drawn as per usual.

The model: (For testing, I used some weak priors as well; they are commented out)

```
writeLines(readLines('ass8_hierarch.stan'))
```

```
##
##
```

```
##       data {
##         int<lower=0> N;
##         int<lower=0> K;
##         int<lower=0, upper=K> x[N];
##         vector[N] y;
##       }
##       parameters {
##         real mu_0_H;
##         real<lower=0> sigma_0_H;
##         vector[K] mu_H;
##         real<lower=0> sigma_H;
##       }
##       transformed parameters {
##
##       }
##       model {
##         //NOTE: I removed the hyperpriors for easier comparison with reference results.
##         //mu_0_H ~ normal(90,50); // weak hyperprior (approx. the pooled distribution)
##         //sigma_0_H ~ scaled_inv_chi_square(K-1,8); // weak hyperprior
##         //sigma_H ~ cauchy(0,8); // weak prior
##         mu_H ~ normal(mu_0_H,sigma_0_H); // population prior
##         y ~ normal(mu_H[x], sigma_H);
##       }
##       generated quantities {
##         real ypred_H[K];
##         real mu_X_p;
##         real ypred_X;
##
##         //PREDICTION FOR 6 HIERACHICAL
##         ypred_H = normal_rng(mu_H,sigma_H);
##         //PREDICTION FOR 7th, i.e., Xth (drawing mu_0_H, sigma_0_H from common h.prior)
##         mu_X_p = normal_rng(mu_0_H, sigma_0_H);
##         ypred_X = normal_rng(mu_X_p, sigma_H);//sigma_H from common prior
##
##
##       }
##
```

Then the data. Input array x has identifiers for different groups and y has all quality measure data in order; 1,2,3,4,5,6,1,2,3,...

```
all_data <-list(N = 6*nrow(factory),
             K = 6,
             x = rep(1:6, nrow(factory)),
             y = c(t(factory[,1:6])))

stan_model_hierarch <- "ass8_hierarch.stan"
```

And the run. Four chains did suffice. Overall 8000 draws are produced for each posterior mu and prediction. Adapt_delta is high for better convergence.

NOTE: The result seemed decent. I saved it to file after test runs, and am using it from now on. So, the fitting is commented below, and I'm only reading in the run.

```
#fit_hierarch <- stan(file = stan_model_hierarch, data = all_data,
#                 warmup = 1000, iter = 3000, control=list(adapt_delta=0.98))
```

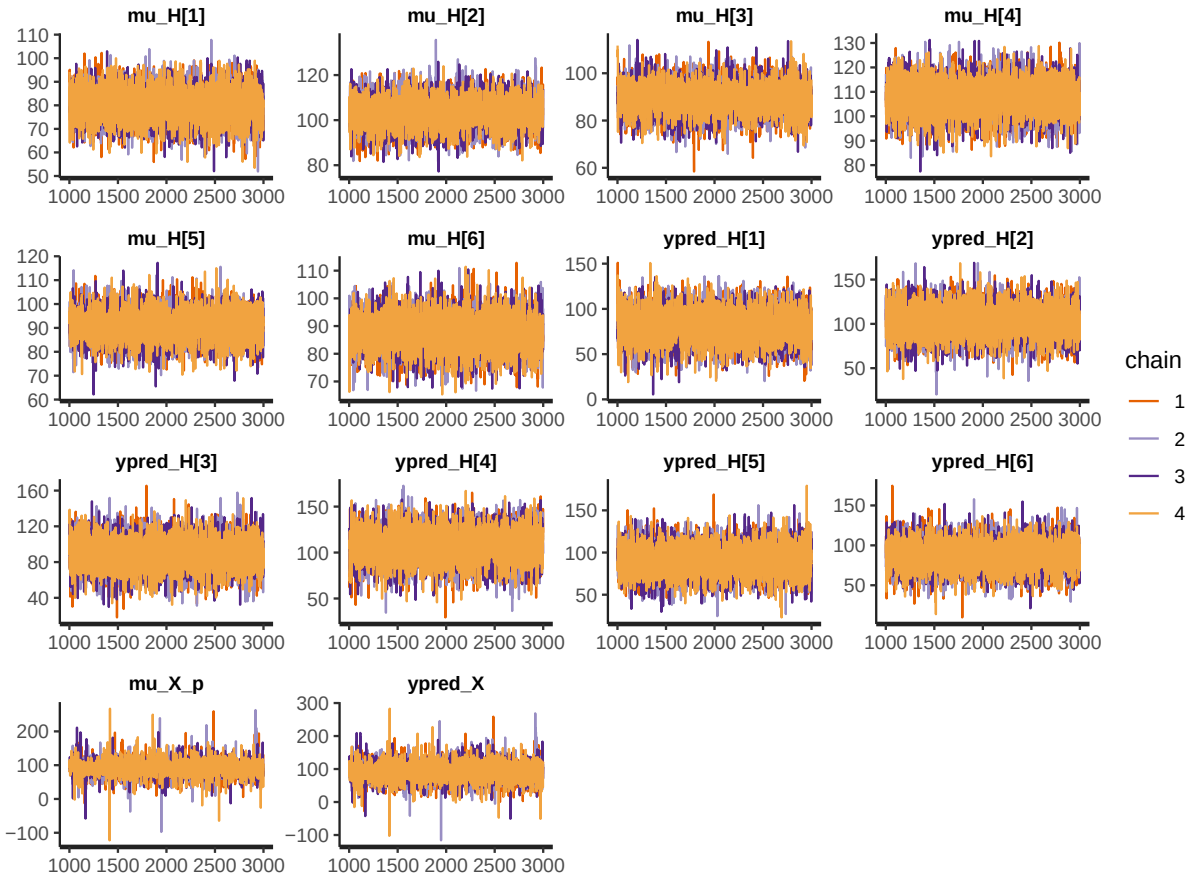
```
fit_hierarch <- readRDS('fit_hierarch.rds')
posterior_hierarch <- extract(fit_hierarch)
m_hierarch <- monitor(fit_hierarch)
```

Let's check the convergence first (some trace's and split Rhat):

```
posterior_hierarch <- extract(fit_hierarch)
rhat(fit_hierarch)
```

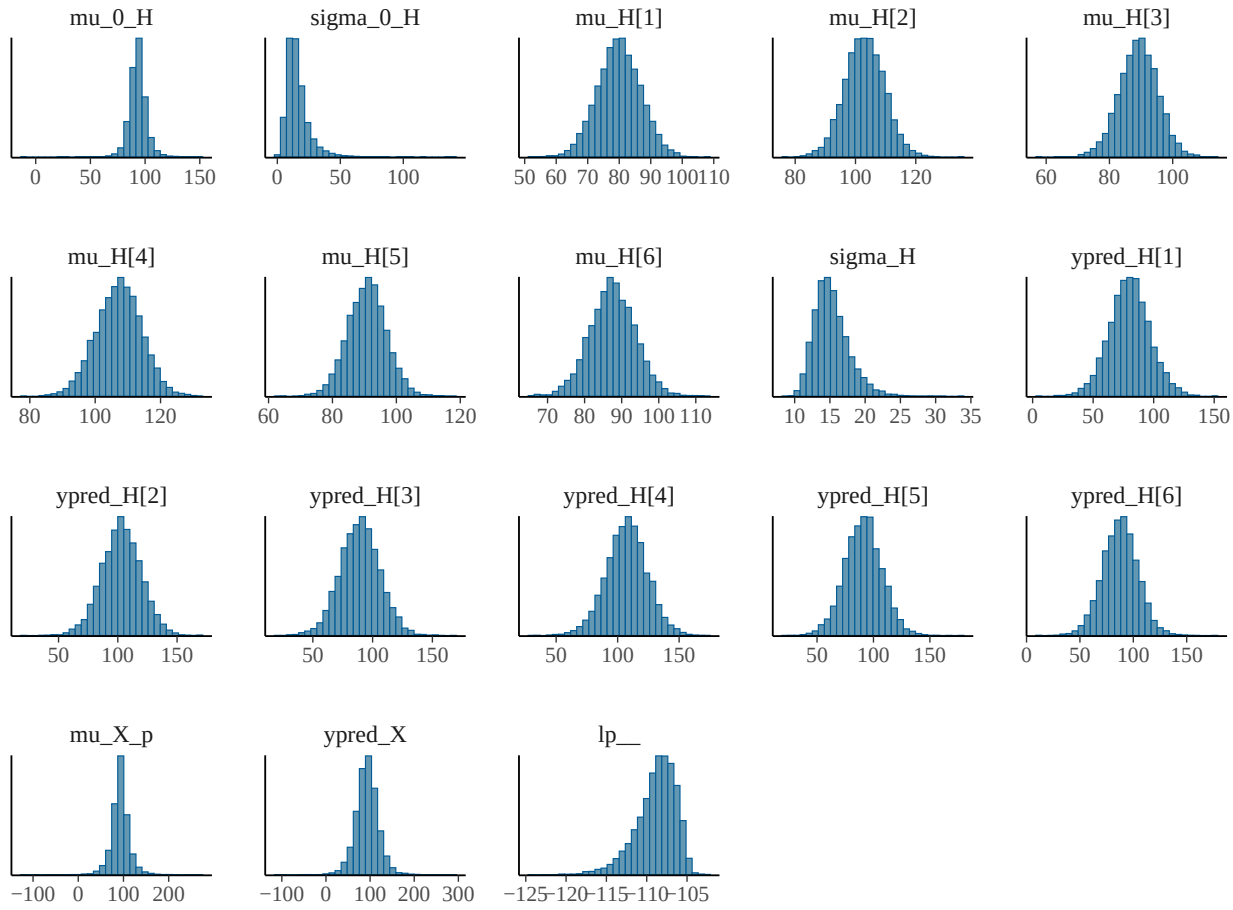
```
##      mu_0_H   sigma_0_H     mu_H[1]     mu_H[2]     mu_H[3]     mu_H[4]
##   1.0013656   1.0001677   1.0001754   1.0005130   1.0006597   0.9999626
##     mu_H[5]     mu_H[6]     sigma_H ypred_H[1] ypred_H[2] ypred_H[3]
##   1.0000120   1.0004901   1.0007038   0.9999411   0.9999853   0.9998558
## ypred_H[4] ypred_H[5] ypred_H[6]      mu_X_p     ypred_X          lp__
##   1.0000594   1.0000363   1.0000898   1.0003678   1.0000562   1.0005365
```

```
traceplot(fit_hierarch, pars = c('mu_H','ypred_H', 'mu_X_p','ypred_X'))
```



Some simple histograms for the run:

```
mcmc_hist(fit_hierarch)
```



Here's how you can save the results:

```
#saveRDS(fit_hierarch, file = 'fit_hierarch.rds')
```

So, everything should be ok. From here on out, these are the exact draws I'm using.

Utility function

UTILITIES: I will calculate all the utilities in R. The calculation is simple:

$$\frac{sold * 200 - all * 106}{all}$$

, where *all* means all predictive draws (per machine). Practically, this means giving a utility of 94 for the "positive" instance (MQM >= 85), and a utility of -106 for the "negative" (MQM < 85). And instead of probabilities, we use draws from the predictive distribution directly (in combination with Kronecker delta type trigger).

NOTE:I'm unsure if we were to have an option of not using the machines 1-6 to produce the next product, but in any case that would have the utility of ZERO, and therefore it would not influence the overall expected utility.

```
utility <- function(draws) {
  n <- length(draws)
  sold <- sum(draws >= 85)
```

```
  sold*200/n - 106
}
#simple TEST
utility(c(123.8, 85.23, 70.16,80.57,84.91))
```

```
## [1] -26
```

I will calculate and rank all the utilities below.

Utilities calculations and ranking

EXPECTED UTILITIES:

Here are the expected utilities for all 6 known machines, and the "7th" predicted (unknown) machine.

```
# UTILITIES FOR 6 MACHINES
utilities <- foreach(i=1:6, .combine = 'rbind') %do% {
  utility(posterior_hierarch$ypred_H[,i])
}
# ADD 7th MACHINE
utilities <- rbind(utilities, utility(posterior_hierarch$ypred_X))
# NAMING
rownames(utilities) <- c('1st machine',
                         '2nd machine',
                         '3rd machine',
                         '4th machine',
                         '5th machine',
                         '6th machine',
                         'predicted (7th)')

# ORDER IN DECREASING ORDER,  AND MAKE A TABLE (with a... kable)
order(utilities, decreasing = T) %>% utilities[., 1] %>%
  cbind(.,c('Best',2:6,'worst')) %>% kable(col.names = c('utility', 'order'))
```

	utility	order
4th machine	75.95	Best
2nd machine	66.9	2
predicted (7th)	22.9	3
5th machine	22.45	4
3rd machine	15.05	5
6th machine	8.05	6
1st machine	-29.875	worst

Summary of the results, discussion

DISCUSSION: Based on these results (the table above), 4th machine is expected to be most profitable (well performing wrt. mean quality measurement). *Excluding the 1st machine*, all known machines are expected to make a winning (for the next product). Also for the unknown future machine the expectation is good, i.e., it is reasonable to assume it will make a profit.

Of course there are other considerations outside our model and chosen utility (and those could be possibly

included in a further simulations), but, with these given data and parameters for the task, **the result for "7th" machine seem to be in favour of buying the new machine**. However, it is also good to remember that our result is only with regards to the next product ,and makes no promises about the continued operation of the machine(s).

Utility for not buying the new machine (and also for not producing the next product) is ZERO (no cost no profit). Utility for buying the new machine is ~ 22.9 . Therefore the expected utility is ~ 22.9, and so, the company owner should buy the new machine.

For the 1st machine, if one assigns a utility of ZERO for not producing the next product, it would be better to not produce the next product (the assignment was perhaps a bit unclear if we were to consider not producing a product as a viable option for machines 1-6).

EXTRA: Just for comparison, here are the summary statistics for the draws. Utilities over the predictive distributions seem to have the same *ordering* as one would have gotten from the means directly. This is due to highly symmetric distributions and symmetric utilities (94 ~ |-106|), with given "trigger" value of 85 being close to the means of most of the posteriors. But the means them selves can not tell the actual utility and disregard the spread entirely, as well as the weight each case should be given (over/under 85). So even if using posterior or predictive means alone for the descision would have backed the same conclusions, it would not have been a principalled Bayesian result.

```
m_hierarch %>% as.data.frame() %>% select(mean, sd, `2.5%`, `97.5%`)
```

```
##                 mean        sd        2.5%      97.5%
## mu_0_H       93.18784  8.218581   77.784120  109.78316
## sigma_0_H    16.23159 10.053889    4.268366   41.30911
## mu_H[1]      79.96594  6.883561   66.553599   93.43778
## mu_H[2]     103.06343  6.563984   90.288036  115.94096
## mu_H[3]      89.20841  6.173745   76.756640  101.17973
## mu_H[4]     107.25363  6.979655   93.511461  120.55401
## mu_H[5]      90.83929  6.105524   78.836949  102.88015
## mu_H[6]      87.55510  6.181163   75.286251   99.60716
## sigma_H      15.25858  2.476817   11.331222   20.99162
## ypred_H[1]   80.08671 16.882826   46.362821  114.02875
## ypred_H[2]  102.97076 16.657978   69.424997  135.16458
## ypred_H[3]   89.31874 16.703799   56.866481  122.21501
## ypred_H[4]  107.30327 16.850247   73.197214  140.25624
## ypred_H[5]   90.96547 16.555165   57.936980  123.11405
## ypred_H[6]   87.70292 16.613330   54.894864  120.22458
## mu_X_p       93.07202 20.705752   52.041343  134.49087
## ypred_X      93.04081 25.963958   41.120666  143.96390
## lp__       -108.85373  2.511792 -114.815020 -105.17450
```