BDA - Assignment 7

Anonymous

29/3/2020

```
# Loading packages
library(tidyverse)
library(aaltobda) # contains data "drowning" & "factory"
library(rstan)
# stan settings:
source('stan_utility.R') # diagnosis of rhats
options(mc.cores = parallel::detectCores()) #when using locally computer
rstan_options(auto_write = TRUE) # autosave compiled Stan program
# bay settings:
library(loo) #estimate predictive error of MCMC item-level log likelihood output
library(gridExtra)
library(bayesplot) #plots of posterior draws
library(shinystan) # model paramteres & MCMC simulations
#theme_set(bayesplot::theme_default(base_family = "sans"))
bayesplot_theme_set() #default
SEED <- 48927 # set random seed for reproducability
```

Q1 - Linear model: drowning data with Stan

The drowning data contains the number of people drowned in Finland 1980-2016, which we want to investiagte using a linear model with Gaussian noise.

Q1.1 - fixing errors

A Stan code is given with errors. We want to find and fix these. The errors are marked in the following:

```
data {
    int<lower=0> N; // number of data points
    vector[N] x; // observation year
    vector[N] y; // observation number of drowned
    real xpred; // prediction year
}
parameters {
  real alpha;
  real beta;
  real <upper=0> sigma; #error: variance can't be NEG
}
transformed parameters {
  vector[N] mu;
  mu = alpha + beta*x;
```

```
}
model {
  y ~ normal(mu, sigma);
}
generated quantities {
  real ypred;
  ypred = normal_rng(mu, sigma); #error: mu is a vector parametre
}
```

For sigma error, we need to replace upper with lower.

As for "mu is a vector parameter": ypred has been defined as a real value, which will cause an error since mu is a vector parametre.

After correcting the errors we get the following stan model:

```
writeLines(readLines("ex7_1.stan"))
```

```
## data {
##   int<lower=0> N; // number of data points
##   vector[N] x; // observation year
##   vector[N] y; // observation number of drowned
##   real xpred; // prediction year
##   real pbm;
##   real pbs;
##  }
## parameters {
##   real alpha;
##   real beta;
##   real<lower=0> sigma;
## }
## transformed parameters {
##  vector[N] mu;
##    mu = alpha + beta *x;
## }
## model {
##   beta ~ normal(pbm, pbs);  // prior on the slope
##   y ~ normal(mu, sigma);
## }
## generated quantities {
##   real ypred;
##   ypred = normal_rng(alpha + beta*xpred, sigma);
## }
```

Q1.2 - Suitable values for τ

No prior for the parametres is defined in "ex7_1.stan", which corresponds to using a uniform prior.

```
writeLines(readLines("ex7_2.stan"))
```

```
## data {
##   int<lower=0> N; // number of data points
```

```
##   vector[N] x; // observation year
##   vector[N] y; // observation number of drowned
##   real xpred; // prediction year
##  }
## parameters {
##   real alpha;
##   real beta;
##   real<lower=0> sigma;
## }
## transformed parameters {
##  vector[N] mu;
##    mu = alpha + beta *x;
## }
## model {
##   y ~ normal(mu, sigma);
## }
## generated quantities {
##   real ypred;
##   ypred = normal_rng(alpha + beta*xpred, sigma);
## }
```

```
data("drowning")

d_lin <- list(N = nrow(drowning),
              x = drowning$year,
              y = drowning$drownings ,
              xpred = 2019)

fit_lin <- stan(file="ex7_2.stan", data = d_lin, seed = SEED,control = list(max_treedepth =15))
```

We would like to apply a weakly informative prior $\beta \sim \mathcal{N}(0, \tau^2)$ for the slope paramtere β. Given is: $Pr(-69 < \beta < 69) = 0.99$. Due to symmetry of the normal distribution, we get $p(\beta \leq -69) = 0.005$, so tau can be calculated by:

```
# tau calcuation
tau <- -69 / qnorm(0.005)
tau
```

```
## [1] 26.78749
```

So we find tau to 26.8

Q1.3 - Implementation of the prior

We add the calculated tau:

```
d_lin_prior <- c(list(
    tau = 26.8),
  d_lin)
```

Implementing in a new stan model:

```
writeLines(readLines("ex7_3.stan"))
```

```
## data {
##   int<lower=0> N; // number of data points
##   vector[N] x; // observation year
##   vector[N] y; // observation number of drowned
##   real xpred; // prediction year
##   real tau;  // prior sd for slope parameter (beta)
##  }
## parameters {
##   real alpha;
##   real beta;
##   real<lower=0> sigma;
## }
## transformed parameters {
##  vector[N] mu;
##    mu = alpha + beta *x;
## }
## model {
##   beta ~ normal(0, tau);  // prior on the slope
##   y ~ normal(mu, sigma);
## }
## generated quantities {
##   real ypred;
##   ypred = normal_rng(alpha + beta*xpred, sigma);
## }
```

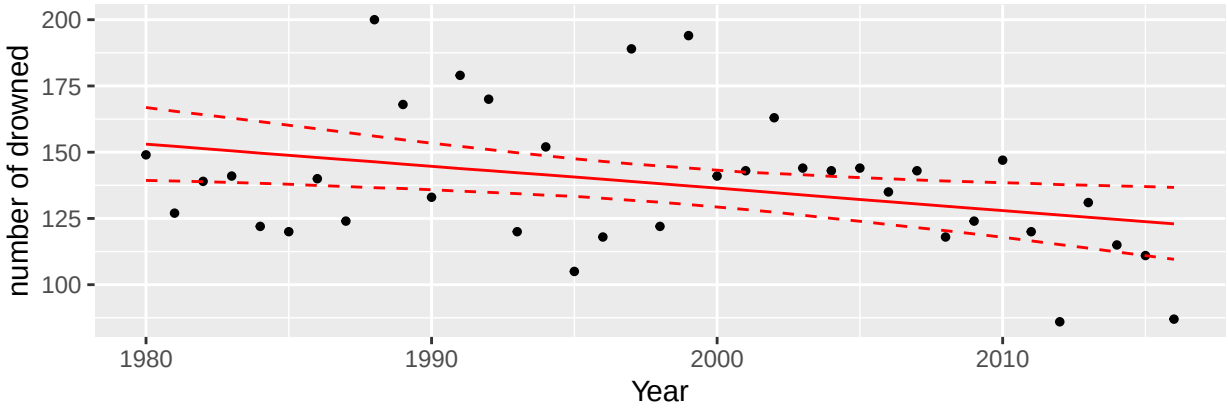
Fitting the model to the data

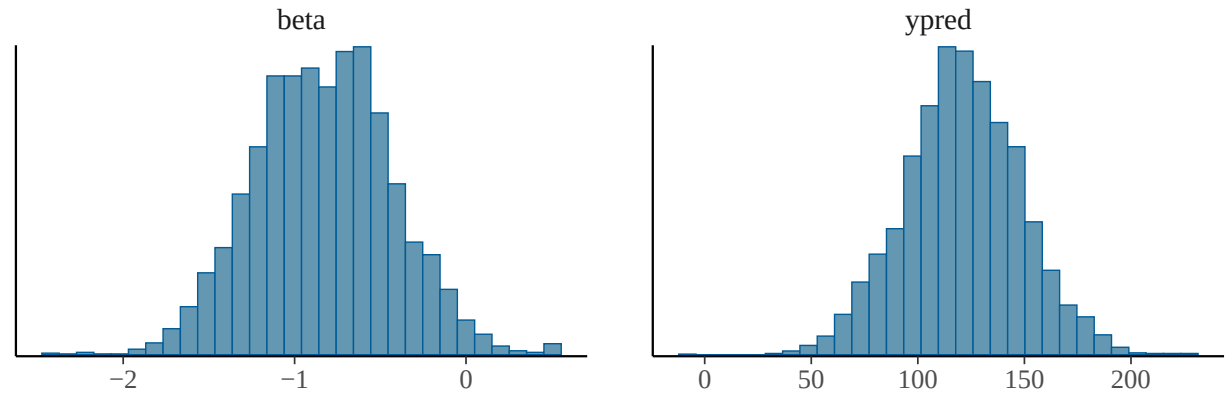
```
fit_lin <- stan(file="ex7_3.stan", data = d_lin_prior, seed=SEED, control = list(max_treedepth =15))
```

Visualizing to compare with plot form ex7:

```
p <- ggplot() +
  geom_point(aes(x, y), data = data.frame(d_lin), size = 1) +
  geom_line(aes(x, y, linetype = pct), data = mu, color = 'red') +
  scale_linetype_manual(values = c(2,1,2)) +
  labs(y = 'number of drowned', x= "Year") +
  guides(linetype = F)

pars <- intersect(names(samples_lin), c('beta','ypred'))
draws <- as.data.frame(fit_lin)
phist <- mcmc_hist(draws, pars = pars)
grid.arrange(p, phist, nrow = 2)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```





Comparing with the plot given in the exercise: Looks satisfying

Q2 - Hierarchical model: factory data with Stan

Seperated Gaussian Model

Implementing the seperated model:

```
writeLines(readLines("ex7_4.stan"))
```

```
##
## data {
##   int<lower=0> N; // number of data points
##   int<lower=0> K; // number of groups
##   int<lower=1,upper=K> x[N]; // group indicator
##   vector[N] y; //
## }
## parameters {
##   vector[K] mu; // group means
##   vector<lower=0>[K] sigma; // group stds
## }
## model {
##   y ~ normal(mu[x], sigma[x]);
## }
## generated quantities {
```

```
##   real ypred;
##   ypred = normal_rng(mu[6], sigma[6]);
## }
```

The data related to this model is :

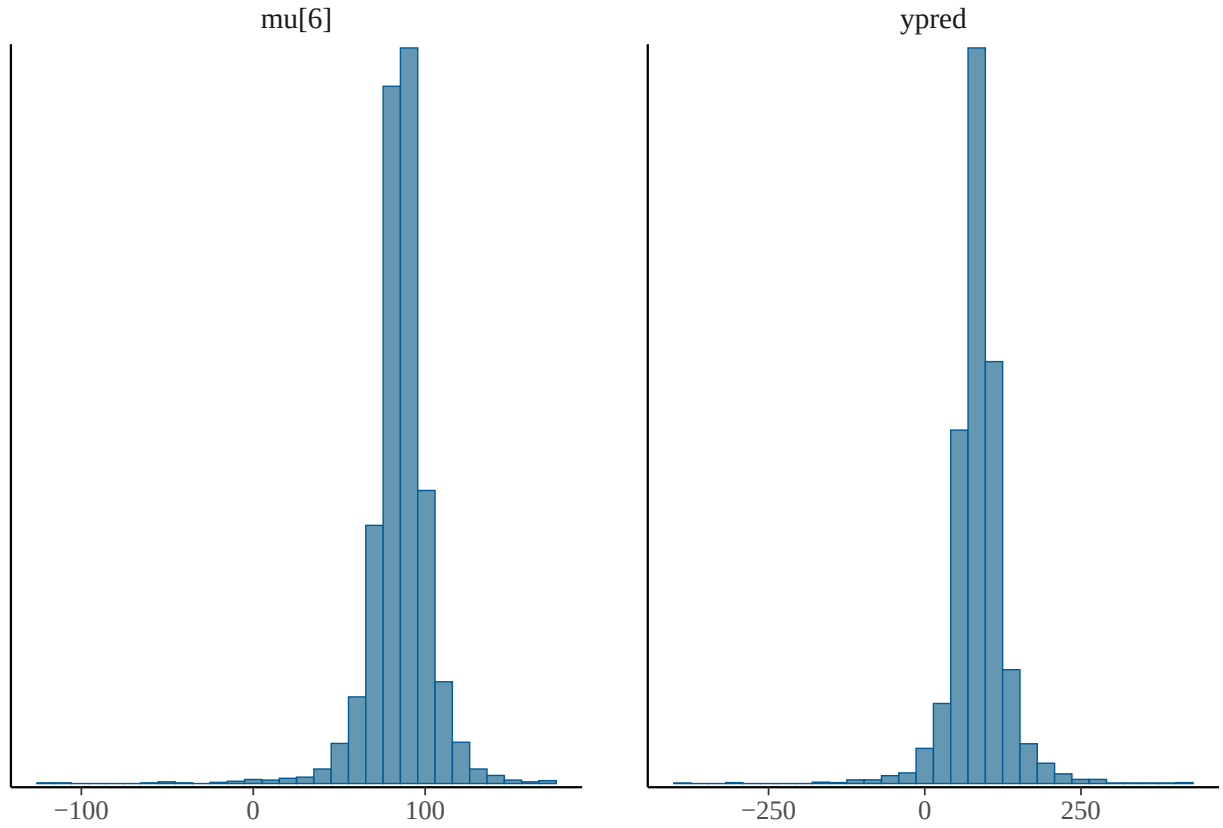
```
data("factory")
data_separate <-list(N = 6*nrow(factory),
                     K = 6,
                     x = rep(1:6, nrow(factory)),
                     y = c(t(factory)))

#fitting in stan:
fit_sep <- stan(file="ex7_4.stan", data = data_separate, seed = SEED)
```

```
## Warning: Tail Effective Samples Size (ESS) is too low, indicating posterior variances and tail quanti
## Running the chains for more iterations may help. See
## http://mc-stan.org/misc/warnings.html#tail-ess
```

The posterior distribution of the mean of the sixth machine:

```
draws_separate <- as.data.frame(fit_sep)
mcmc_hist(draws_separate, pars = c("mu[6]", "ypred"))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



The posterior distribution of the mean of the quality measurements of the seventh machine:

Pooled model

For the pooled model we assume that μ and σ are the same for all machines. The Stan implementation for pooled model is as:

```
writeLines(readLines("ex7_5.stan"))
```

```
##
## data {
##   int<lower=0> N; // number of data points
##   vector[N] y; //
## }
## parameters {
##   real mu; // group means
##   real<lower=0> sigma; // common std
## }
## model {
##   y ~ normal(mu, sigma);
## }
## generated quantities {
##   real ypred;
##   real mu_7;
##   ypred = normal_rng(mu, sigma);
```

```
##   mu_7 = normal_rng(mu, sigma);
## }
```

The data related to this model is :

```
data_pooled <- list(N = 6*nrow(factory),
                    y = c(t(factory)))
```

We fit the pooled model in stan as follows:

```
fit_pooled <- stan(file = "ex7_5.stan", data = data_pooled, seed = SEED)
```

i) The posterior of the mean of the sixth machine:

$$p(\mu_6|\sigma, y) \propto \mathcal{N}(\mu, \sigma^2)$$

ii) The predictive distribution for another quality measurement from the sixth machine:

$$p(\hat{y_6}|\mu, \sigma) \propto \mathcal{N}(\mu, \sigma^2)$$

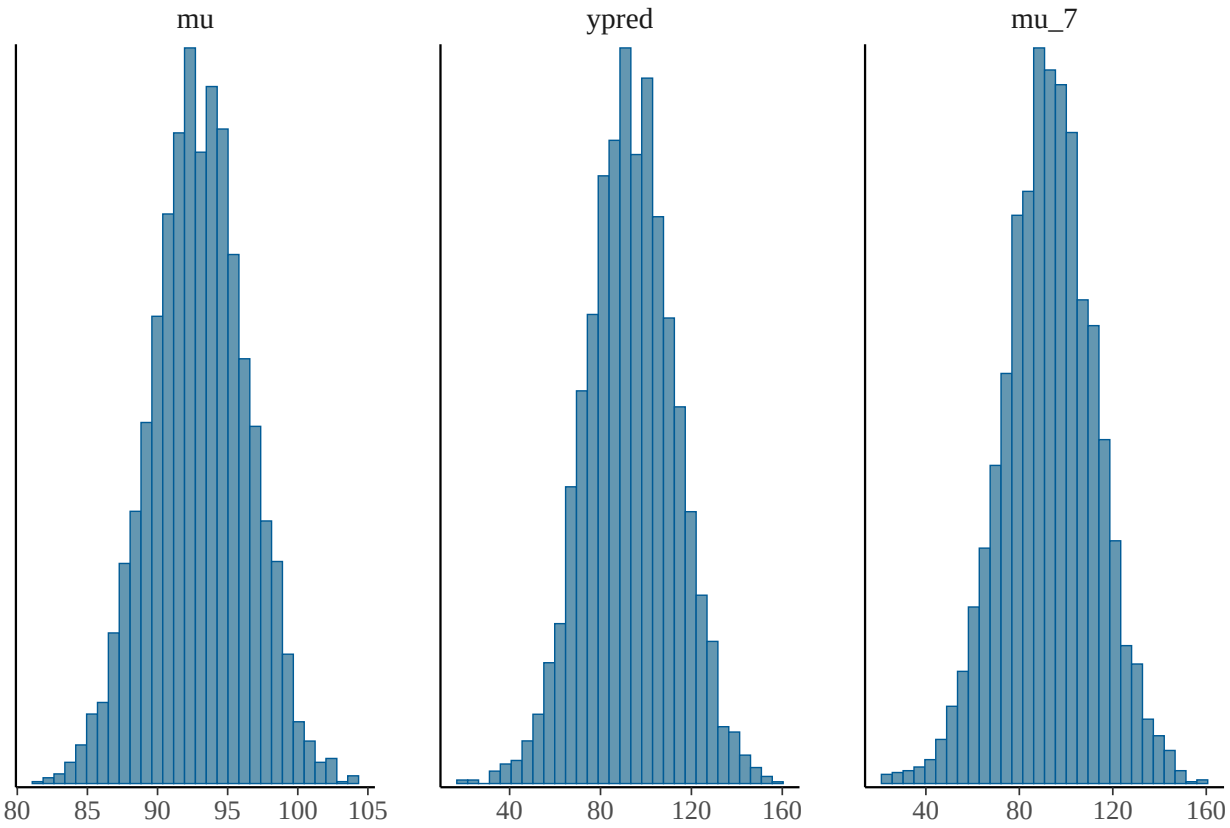
iii) The posterior distribution of the mean of the quality measurements of the seventh machine:

assuming the machines are identical, gives us:

$$p(\mu_7|\mu, \sigma) \propto \mathcal{N}(\mu, \sigma^2)$$

```
draws_pooled <- as.data.frame(fit_pooled)
mcmc_hist(draws_pooled, pars = c("mu", "ypred", "mu_7"))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Hierarchical model

The Stan implementation of the model:

```
writeLines(readLines("ex7_6.stan"))
```

```
## data {
##   int<lower=0> N; // number of data points
##   int<lower=0> K; // number of groups
##   int<lower=1,upper=K> x[N]; // group indicator
##   vector[N] y; // target
## }
## parameters {
##   real mu0; // prior mean
##   real<lower=0> sigma0; // prior std
##   vector[K] mu; // group means
##   real<lower=0> sigma; // group stds
## }
## model {
##   mu0 ~ normal(50, 10); // weakly informative prior
##   sigma0 ~ cauchy(0,4); // weakly informative prior
##   mu ~ normal(mu0, sigma0); // population prior with unknown parameters
##   sigma ~ cauchy(0,4); // weakly informative prior
##   y ~ normal(mu[x] , sigma);
```

```
## }
## generated quantities {
##   real ypred;
##   real mu_7;
##   ypred = normal_rng(mu[6], sigma);
##   mu_7 = normal_rng(mu0, sigma);
## }
```

We fit the separate model in stan:

```
fit_hierarchical <- stan(file="ex7_6.stan", data = data_separate, seed = SEED)
```

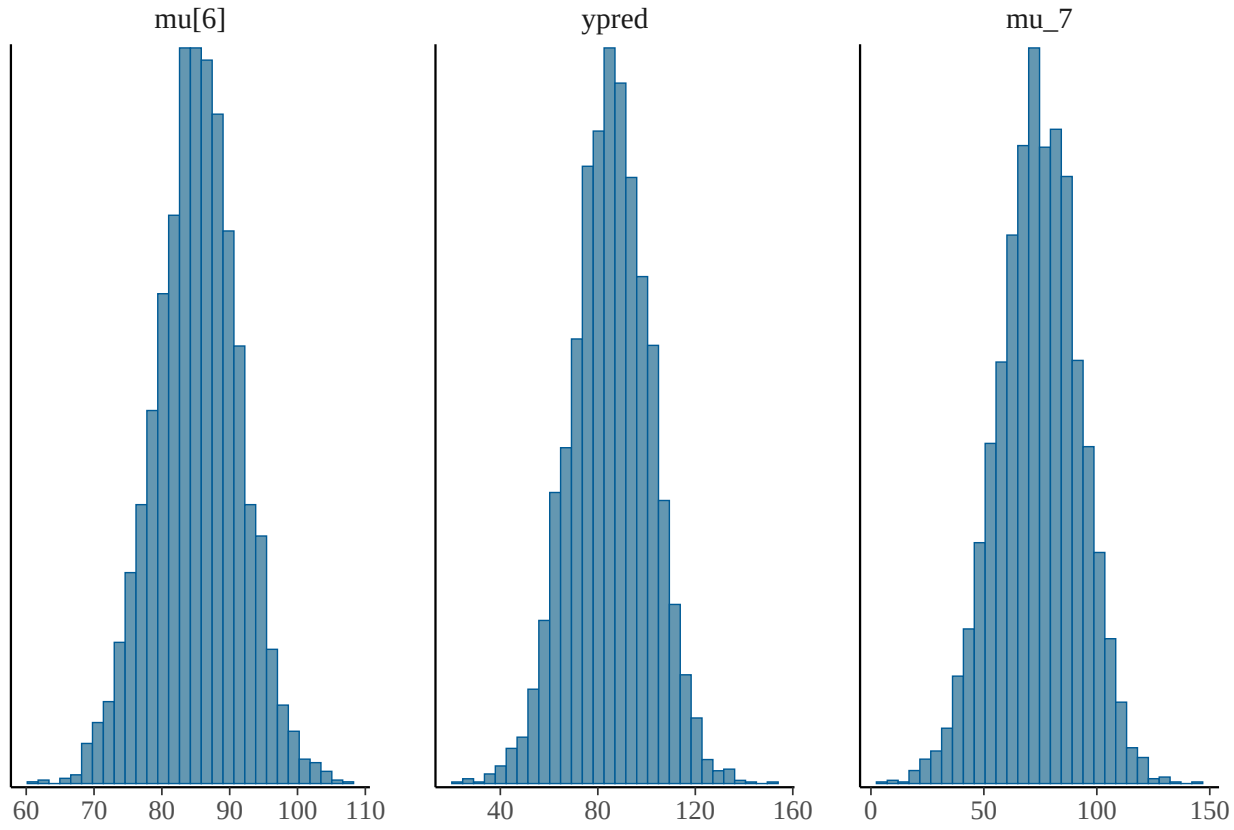
```
## Warning: There were 11 divergent transitions after warmup. Increasing adapt_delta above 0.8 may help
## http://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
```

```
## Warning: Examine the pairs() plot to diagnose sampling problems
```

The predictive distribution for another quality measurement from the sixth machine:

```
draws_hierarchical <- as.data.frame(fit_hierarchical)
mcmc_hist(draws_hierarchical, c("mu[6]", "ypred", "mu_7"))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



References:

Code examples are found at Github:
https://github.com/avehtari/BDA_R_demos