BDA - Assignment 7

Anonymous

```
library(tidyr)
library(rstan)
```

```
## Loading required package: StanHeaders
```

```
## Loading required package: ggplot2
```

```
## rstan (Version 2.19.2, GitRev: 2e1f913d3ca3)
```

```
## For execution on a local, multicore CPU with excess RAM we recommend calling
## options(mc.cores = parallel::detectCores()).
## To avoid recompilation of unchanged Stan programs, we recommend calling
## rstan_options(auto_write = TRUE)
```

```
## For improved execution time, we recommend calling
## Sys.setenv(LOCAL_CPPFLAGS = '-march=native')
## although this causes Stan to throw an error on a few processors.
```

```
##
## Attaching package: 'rstan'
```

```
## The following object is masked from 'package:tidyr':
##
##     extract
```

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

```
## This is loo version 2.1.0.
## **NOTE: As of version 2.0.0 loo defaults to 1 core but we recommend using as many as possible. Use tl
```

```
## **NOTE for Windows 10 users: loo may be very slow if 'mc.cores' is set in your .Rprofile file (see ht
```

```
##
## Attaching package: 'loo'
```

```
## The following object is masked from 'package:rstan':
##
##     loo
```

```
library(ggplot2)
library(gridExtra)
library(bayesplot)
```

```
## This is bayesplot version 1.7.0

## - Online documentation and vignettes at mc-stan.org/bayesplot

## - bayesplot theme set to bayesplot::theme_default()

##    * Does _not_ affect other ggplot2 plots

##    * See ?bayesplot_theme_set for details on theme setting
```

```
theme_set(bayesplot::theme_default(base_family = "sans"))
library(shinystan)
```

```
## Loading required package: shiny

## Registered S3 method overwritten by 'xts':
##   method     from
##   as.zoo.xts zoo

##
## This is shinystan version 2.5.0
```

```
source('stan_utility.R')
library(aaltobda)
SEED <- 48927 # set random seed for reproducability
```

Problem 1: Linear model: drowning data with Stan

Fixing errors

The first crucial mistake is in line 10:

```
real<upper=0> sigma;
```

The variance cannot be negative, therefor this line should be:

```
real<lower=0> sigma;
```

Anoher mistake is in generated quantities. mu is a vector parameter, therefore defining ypred as a real value will make a syntax error.

```
generated quantities {
  real ypred;
  ypred = normal_rng(mu , sigma);
}
```

The correct lines are as follows:

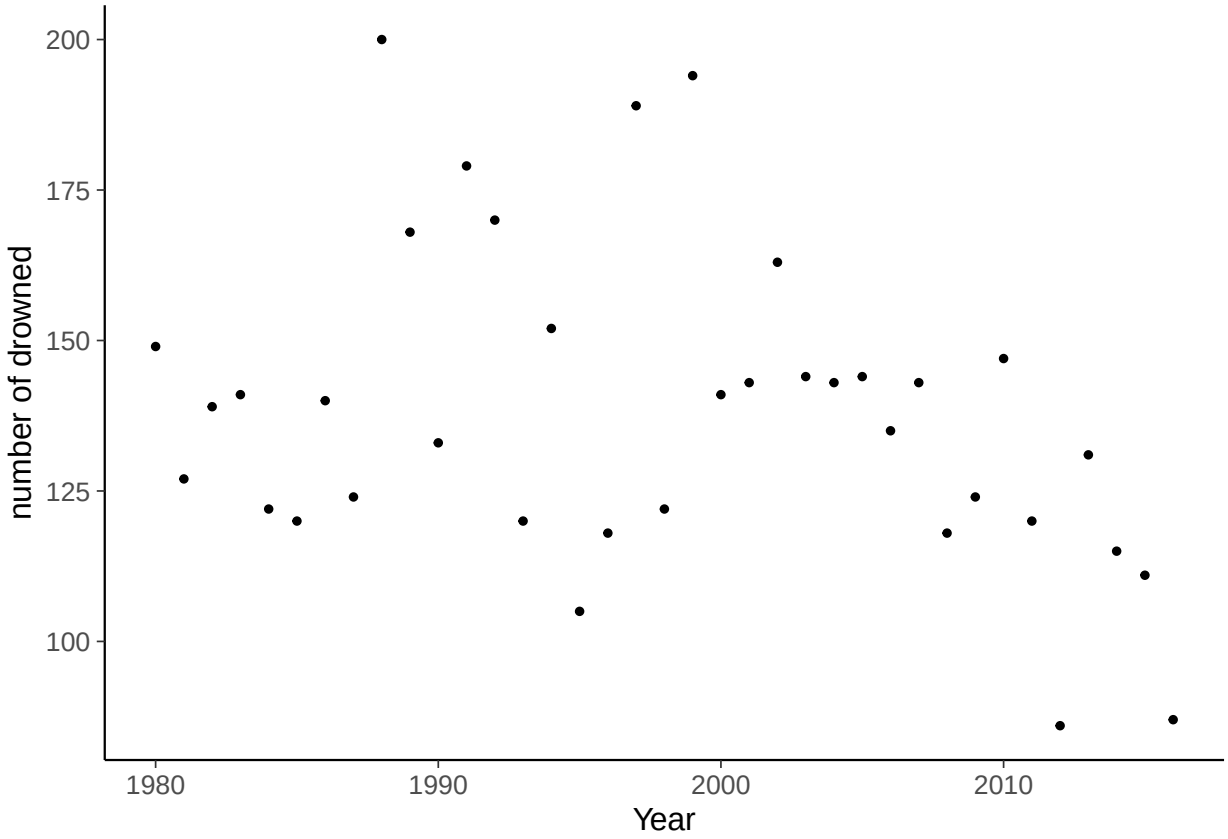
```
generated quantities {
  real ypred;
  ypred = normal_rng(alpha + beta*xpred , sigma);
}
```

Determination of τ

The following lines will read the data. Data is summarized into a vectors of the years and the number of drownings in each year.

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

ggplot() +
  geom_point(aes(x, y), data = data.frame(d_lin), size = 1) +
  labs(y = 'number of drowned', x= "Year") +
  guides(linetype = F)
```



To analyse whether the number of drowned people is rising, we use a linear model with Gaussian model for the unexplained variation. The linear regression model for the drowningn data in Stan syntax can be read from the Assignment7a.stan file which is as follows:

```
writeLines(readLines("Assignment7a1.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);
## }
```

```
fit_lin <- stan(file="Assignment7a1.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 β, s.t. $p(-69 < \beta < 69) = 0.99$. Due to the symmetry of the normal distribution, this is equivalent to $p(\beta \leq -69) = 0.005$. From this we get that:

$$p(\beta \leq -69) = p(\tau Z \leq -69) = p(Z \leq -69/\tau) = 0.005$$

where $Z \sim \mathcal{N}(0, 1)$. This is true when

$$\tau = -\frac{69}{F^{-1}(0.005)}$$

where $F^{-1}(p)$ is the inverse CDF (quantile function) of $Z \sim \mathcal{N}(0, 1)$. It can be calculated as :

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

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

Implementation of the prior

In order to include the prior into the model we add the line $beta \sim normal(0, tau)$; into the model{} block and "real tau;" in data{} block.

Create another list with data and prior:

```
d_lin_prior <- c(list(
    tau = 26.7),
  d_lin)
```

Then, in order to fit the model on the data, we use the following command:

```
writeLines(readLines("Assignment7a2.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);
## }
```

```
fit_lin <- stan(file="Assignment7a2.stan", data = d_lin_prior, seed = SEED,control = list(max_treedepth
```

```
monitor(fit_lin, probs = c(0.1, 0.5, 0.9))
```

```
## Inference for the input samples (4 chains: each with iter = 2000; warmup = 0):
##
##            Q5    Q50    Q95   Mean    SD  Rhat Bulk_ESS Tail_ESS
## alpha   428.6 1782.8 3135.1 1777.3 822.4     1     1150     1368
## beta     -1.5   -0.8   -0.1   -0.8   0.4     1     1150     1373
## sigma    21.5   25.7   31.7   26.1   3.2     1     1417     1584
## mu[1]   138.6  152.6  166.7  152.7   8.6     1     1391     1863
## mu[2]   138.2  151.8  165.4  151.9   8.2     1     1416     1899
## mu[3]   138.0  151.0  164.0  151.0   7.9     1     1446     1960
## mu[4]   137.8  150.1  162.6  150.2   7.5     1     1484     1950
## mu[5]   137.5  149.3  161.2  149.4   7.2     1     1527     2023
## mu[6]   137.3  148.5  159.9  148.6   6.9     1     1579     2045
## mu[7]   137.0  147.7  158.5  147.8   6.5     1     1641     2044
## mu[8]   136.8  146.8  157.2  146.9   6.2     1     1716     2101
```

```
## mu[9]   136.4  146.0  155.9  146.1   5.9     1     1808     2174
## mu[10]  136.1  145.2  154.6  145.3   5.7     1     1919     2226
## mu[11]  135.7  144.4  153.4  144.5   5.4     1     2058     2252
## mu[12]  135.1  143.6  152.3  143.7   5.2     1     2231     2280
## mu[13]  134.6  142.8  151.2  142.8   5.0     1     2444     2336
## mu[14]  134.2  142.0  150.0  142.0   4.8     1     2697     2317
## mu[15]  133.6  141.2  149.0  141.2   4.6     1     3086     2363
## mu[16]  133.0  140.3  147.8  140.4   4.5     1     3517     2474
## mu[17]  132.4  139.5  146.8  139.6   4.4     1     3713     2488
## mu[18]  131.7  138.7  145.8  138.7   4.3     1     3827     2621
## mu[19]  130.9  137.9  145.0  137.9   4.3     1     3878     2626
## mu[20]  130.0  137.1  144.3  137.1   4.3     1     3850     2507
## mu[21]  129.1  136.3  143.4  136.3   4.4     1     3760     2595
## mu[22]  128.2  135.5  142.7  135.5   4.5     1     3593     2774
## mu[23]  127.2  134.7  142.1  134.6   4.6     1     3173     2875
## mu[24]  126.1  133.9  141.6  133.8   4.8     1     2766     2797
## mu[25]  125.0  133.1  140.9  133.0   4.9     1     2507     2675
## mu[26]  123.7  132.3  140.5  132.2   5.2     1     2292     2421
## mu[27]  122.5  131.4  140.2  131.3   5.4     1     2115     2247
## mu[28]  121.2  130.6  139.8  130.5   5.7     1     1970     2215
## mu[29]  119.9  129.8  139.3  129.7   5.9     1     1853     2199
## mu[30]  118.6  129.0  139.0  128.9   6.2     1     1759     2140
## mu[31]  117.3  128.1  138.8  128.1   6.5     1     1681     2133
## mu[32]  115.9  127.3  138.5  127.2   6.9     1     1615     2120
## mu[33]  114.5  126.4  138.1  126.4   7.2     1     1559     2031
## mu[34]  113.1  125.6  137.8  125.6   7.5     1     1513     1865
## mu[35]  111.7  124.8  137.7  124.8   7.9     1     1474     1812
## mu[36]  110.4  123.9  137.4  124.0   8.2     1     1439     1778
## mu[37]  108.9  123.1  137.2  123.1   8.6     1     1412     1791
## ypred    75.6  120.9  167.0  121.2  27.7     1     3219     3775
## lp__   -137.6 -134.8 -133.7 -135.1   1.2     1     1242     1704
##
## For each parameter, Bulk_ESS and Tail_ESS are crude measures of
## effective sample size for bulk and tail quantities respectively (an ESS > 100
## per chain is considered good), and Rhat is the potential scale reduction
## factor on rank normalized split chains (at convergence, Rhat <= 1.05).
```

```
print(fit_lin)
```

```
## Inference for Stan model: Assignment7a2.
## 4 chains, each with iter=2000; warmup=1000; thin=1;
## post-warmup draws per chain=1000, total post-warmup draws=4000.
##
##           mean se_mean     sd    2.5%     25%     50%     75%   97.5%
## alpha  1777.34   24.30 822.39   98.42 1247.80 1782.77 2318.67 3415.04
## beta     -0.82    0.01   0.41   -1.64   -1.09   -0.82   -0.56    0.02
## sigma    26.05    0.08   3.16   20.89   23.71   25.71   28.07   33.00
## mu[1]   152.68    0.23   8.56  135.92  147.08  152.65  158.41  170.02
## mu[2]   151.86    0.22   8.21  135.75  146.44  151.78  157.29  168.48
## mu[3]   151.04    0.21   7.86  135.68  145.88  150.98  156.28  166.94
## mu[4]   150.22    0.20   7.52  135.57  145.34  150.15  155.20  165.55
## mu[5]   149.40    0.18   7.19  135.38  144.67  149.34  154.16  164.07
## mu[6]   148.58    0.17   6.86  135.20  144.02  148.52  153.11  162.70
## mu[7]   147.76    0.16   6.55  134.77  143.47  147.67  152.10  161.19
```

```
## mu[8]   146.94    0.15   6.24  134.55  142.83  146.85  151.07  159.82
## mu[9]   146.12    0.14   5.95  134.35  142.23  146.02  150.08  158.42
## mu[10]  145.30    0.13   5.67  134.14  141.63  145.18  149.01  156.96
## mu[11]  144.48    0.12   5.41  134.02  141.02  144.42  148.02  155.67
## mu[12]  143.66    0.11   5.17  133.64  140.35  143.58  147.07  154.27
## mu[13]  142.84    0.10   4.95  133.20  139.64  142.78  146.12  152.89
## mu[14]  142.02    0.09   4.76  132.86  138.94  141.95  145.16  151.66
## mu[15]  141.19    0.08   4.60  132.37  138.21  141.16  144.18  150.56
## mu[16]  140.37    0.08   4.46  131.83  137.47  140.34  143.26  149.50
## mu[17]  139.55    0.07   4.37  131.03  136.71  139.51  142.39  148.34
## mu[18]  138.73    0.07   4.31  130.42  135.95  138.69  141.57  147.37
## mu[19]  137.91    0.07   4.29  129.63  135.05  137.90  140.73  146.39
## mu[20]  137.09    0.07   4.31  128.74  134.22  137.12  139.96  145.61
## mu[21]  136.27    0.07   4.37  127.85  133.32  136.29  139.25  144.83
## mu[22]  135.45    0.07   4.46  126.83  132.40  135.51  138.44  144.14
## mu[23]  134.63    0.08   4.59  125.64  131.51  134.71  137.72  143.65
## mu[24]  133.81    0.09   4.76  124.48  130.55  133.90  136.95  143.23
## mu[25]  132.99    0.10   4.95  123.41  129.67  133.09  136.24  142.85
## mu[26]  132.17    0.11   5.16  122.24  128.76  132.29  135.53  142.35
## mu[27]  131.35    0.12   5.41  120.87  127.76  131.43  134.90  142.07
## mu[28]  130.53    0.13   5.66  119.33  126.82  130.61  134.24  141.85
## mu[29]  129.71    0.14   5.94  117.87  125.89  129.79  133.60  141.50
## mu[30]  128.89    0.15   6.23  116.55  124.87  128.97  132.89  141.25
## mu[31]  128.07    0.16   6.54  115.18  123.92  128.13  132.28  141.11
## mu[32]  127.25    0.17   6.85  113.72  122.93  127.30  131.70  141.03
## mu[33]  126.43    0.18   7.18  112.32  121.86  126.44  131.05  141.05
## mu[34]  125.60    0.19   7.51  110.79  120.83  125.64  130.42  140.81
## mu[35]  124.78    0.21   7.86  109.29  119.81  124.76  129.84  140.60
## mu[36]  123.96    0.22   8.20  107.76  118.76  123.95  129.21  140.37
## mu[37]  123.14    0.23   8.56  106.20  117.73  123.12  128.60  140.15
## ypred   121.17    0.49  27.71   66.13  103.03  120.90  138.97  176.09
## lp__   -135.10    0.04   1.23 -138.33 -135.71 -134.80 -134.15 -133.67
##        n_eff Rhat
## alpha   1145    1
## beta    1145    1
## sigma   1405    1
## mu[1]   1383    1
## mu[2]   1409    1
## mu[3]   1440    1
## mu[4]   1476    1
## mu[5]   1518    1
## mu[6]   1569    1
## mu[7]   1630    1
## mu[8]   1703    1
## mu[9]   1792    1
## mu[10]  1902    1
## mu[11]  2037    1
## mu[12]  2203    1
## mu[13]  2407    1
## mu[14]  2653    1
## mu[15]  3017    1
## mu[16]  3525    1
## mu[17]  3705    1
## mu[18]  3819    1
```

```
## mu[19]  3869    1
## mu[20]  3844    1
## mu[21]  3752    1
## mu[22]  3609    1
## mu[23]  3109    1
## mu[24]  2715    1
## mu[25]  2463    1
## mu[26]  2253    1
## mu[27]  2081    1
## mu[28]  1941    1
## mu[29]  1827    1
## mu[30]  1734    1
## mu[31]  1657    1
## mu[32]  1594    1
## mu[33]  1541    1
## mu[34]  1497    1
## mu[35]  1459    1
## mu[36]  1427    1
## mu[37]  1400    1
## ypred   3198    1
## lp__    1158    1
##
## Samples were drawn using NUTS(diag_e) at Thu Jan 30 12:11:53 2020.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

```
check_hmc_diagnostics(fit_lin)
```

```
##
## Divergences:

## 0 of 4000 iterations ended with a divergence.

##
## Tree depth:

## 0 of 4000 iterations saturated the maximum tree depth of 15.

##
## Energy:

## E-BFMI indicated no pathological behavior.
```

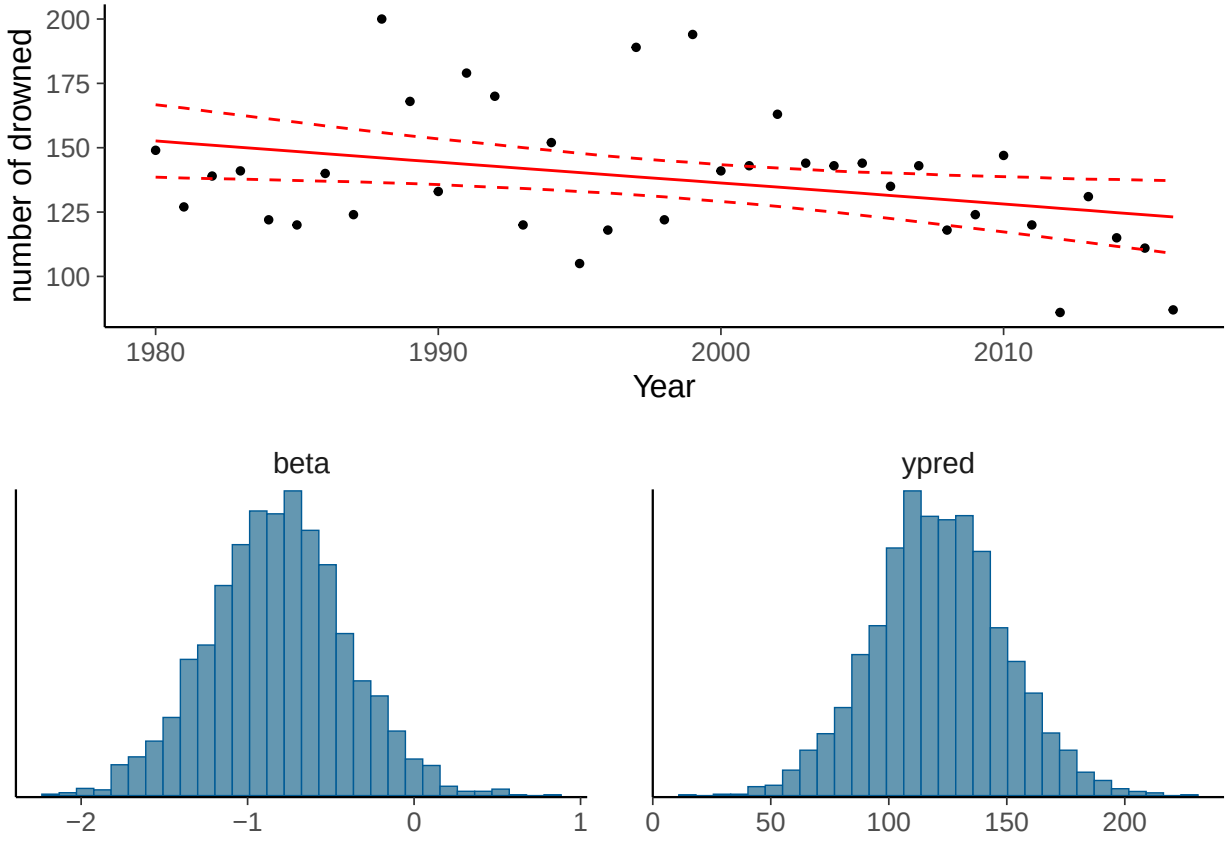
```
samples_lin <- rstan::extract(fit_lin, permuted = T)
# mean(samples_lin$beta>0) # probability that beta > 0

mu <- apply(samples_lin$mu, 2, quantile, c(0.05, 0.5, 0.95)) %>%
  t() %>% data.frame(x = d_lin$x, .)  %>% gather(pct, y, -x)

pfit <- 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(pfit, phist, nrow = 2)
```

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



As it can be seen the histagrams for posterior beta and posterior predictive for year 2019 match the plots provided in the exercise.

Problem 2: Hierarchical model: factory data with Stan

Seperated Gaussian Model

First we consider the separated model. For this model, each machine j is assumed to have unrelated means μ_j and σ_j and the posterior distribution is determined as follows: $y_j \sim \mathcal{N}(\mu_j, \sigma^2)$. The Stan implementation is as follows:

```
data("factory")
writeLines(readLines("Assignment7b_separat.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_separate <-list(N = 6*nrow(factory),
                     K = 6,
                     x = rep(1:6, nrow(factory)),
                     y = c(t(factory)))
```

We fit the separate model in stan as follow:

```
fit_sep <- stan(file="Assignment7b_separat.stan", data = data_separate, seed = SEED)
```

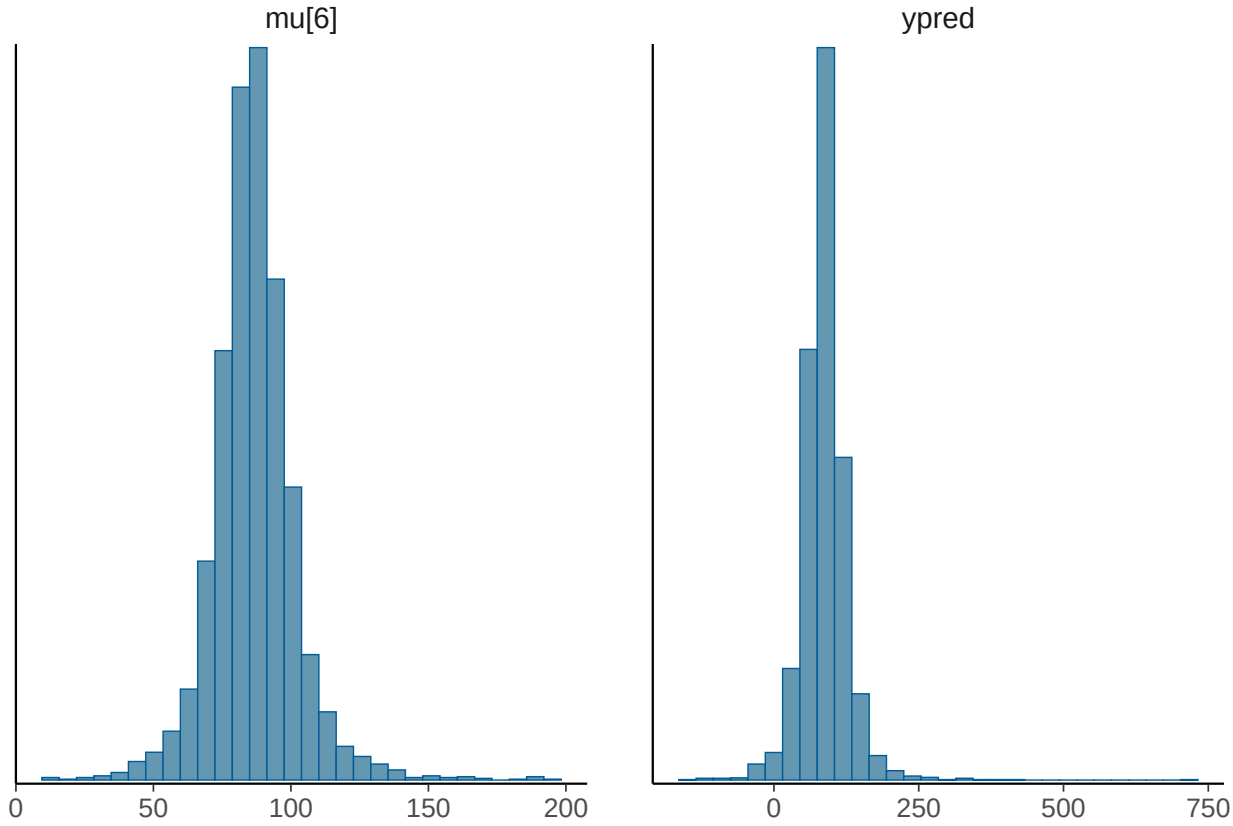
```
## hash mismatch so recompiling; make sure Stan code ends with a blank line
```

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

$$p(\mu_6|\sigma_6, y_6) \propto \mathcal{N}(\mu_6, \sigma_6^2)$$

```
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`.
```



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

$$p(\hat{y_6}|\mu, \sigma) \propto \mathcal{N}(\mu_6, \sigma_6^2)$$

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

Since the machine is completely separate from the others, we cannot infer anything about the model from the posteriors of the other machines. Therefore the best prediction for μ_7 is the prior, which is in this case poorly defined, since we are using the uninformative uniform priors.

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("Assignment7b_pooled.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 = "Assignment7b_pooled.stan", data = data_pooled, seed = SEED)
```

```
## hash mismatch so recompiling; make sure Stan code ends with a blank line
```

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:

$$p(\hat{y}_6|\mu, \sigma) \propto \mathcal{N}(\mu, \sigma^2)$$

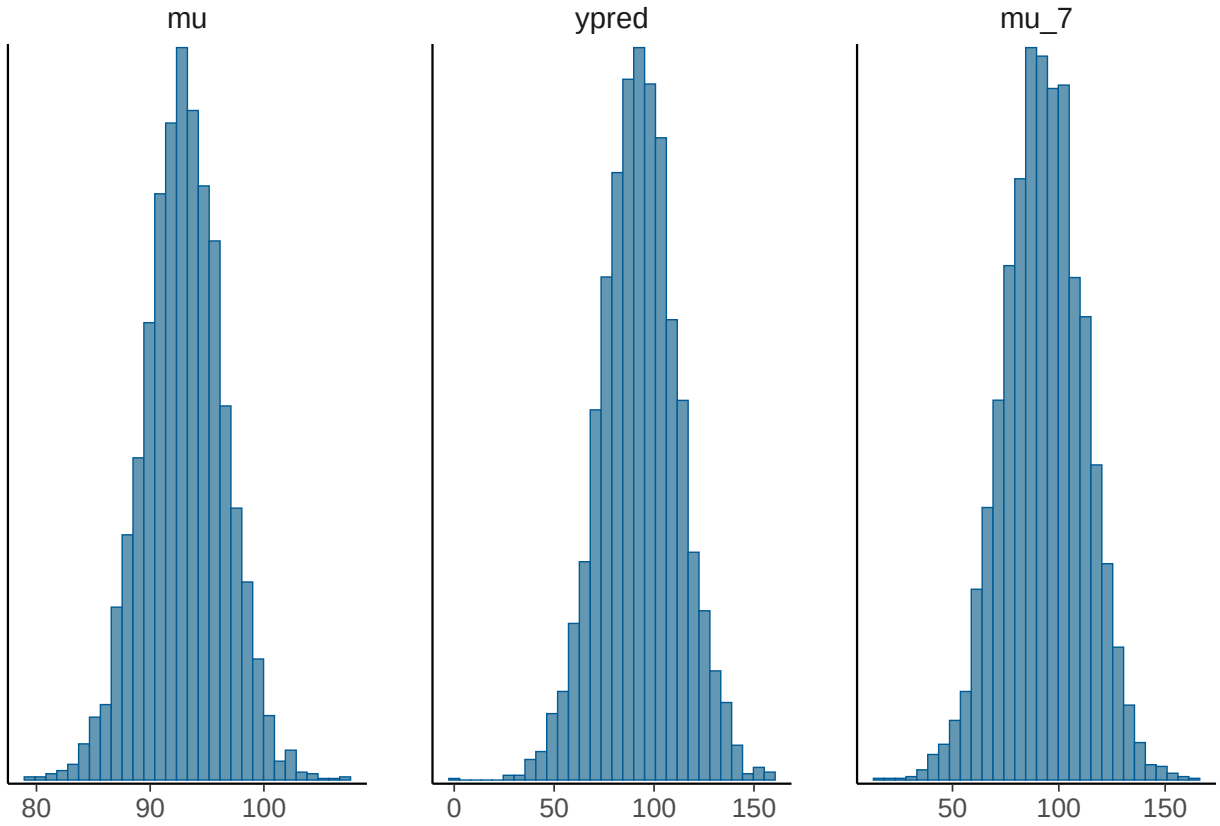
iii) The posterior distribution of the mean of the quality measurements of the seventh machine:

Since the machines are assumed to be identical, the posterior distribution of the mean of the seventh machine is given as:

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

```
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`.
```



Compared to the previous model, the poserior distribution is less narrow and smaller compard to the seperated gaussian model. The reason for this is that the standard deviation is larger in this model. The posterior distribution of the mean of the quality measurements of the seventh machine corresponds to the answer of the question i) for the pooled model.

Hierarchical model

In this model the means of the different machines are assumed to have a common standard deviation σ and means μ that are drawn from a normal distribution with μ_0 and σ_0. The Stan implementation of the model is as follows:

```
writeLines(readLines("Assignment7b_hierarchical.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 {
##   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 as follow:

```
fit_hierarchical <- stan(file="Assignment7b_hierarchical.stan", data = data_separate, seed = SEED)
```

```
## Warning: There were 21 divergent transitions after warmup. Increasing adapt_delta above 0.8 may help
## http://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
```

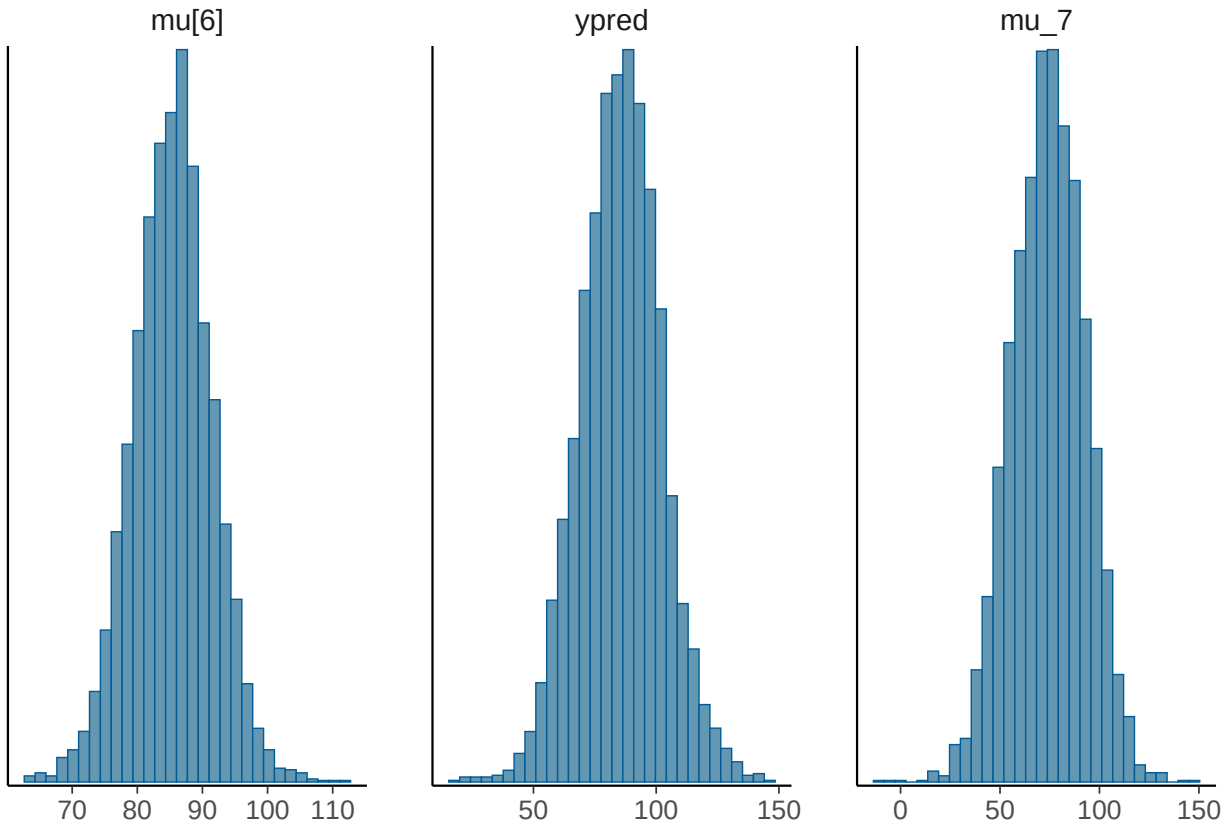
```
## Warning: Examine the pairs() plot to diagnose sampling problems
```

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

$$p(\hat{y_6}|\mu, \sigma) \propto \mathcal{N}(\mu_6, \sigma_6^2)$$

```
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`.
```


mu[6]
70
80
90
100
110
ypred
50
100
150
mu_7
0
50
100
150