BDA - Assignment 7

Anonymous

Contents


```
library('aaltobda')
data(drowning)
library (rstan)
rstan_options (auto_write = TRUE)
options(mc.cores = parallel ::detectCores())
```

Exercise 1. Linear model: drowning data with Stan

1.

There were two mistakes in the given Stan code. The errors are colored in red and the correction is colored in blue

Firstly, this:
parameters {
  real alpha;
  real beta;
  real<upper=0>sigma;
}

should be written as:
parameters {
  real alpha;
  real beta;
  real<lower=0>sigma;
}

Secondly, this:
generated quantities {
  real ypred;
  ypred = normal_rng(alpha + beta*xsigma);
}
should be written as:
generated quantities {
  real ypred;
  ypred = normal_rng(alpha + beta*xpredsigma);;
}

Below the full corrected Stan code

```
drowning = "
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;               // intercept
  real beta;                // coefficients for predictors
  real<lower=0> sigma;      // error scale
}

transformed parameters {
  vector[N] mu;
  mu = alpha + beta*x;
}

model {
  y ~ normal(mu, sigma);     // likelihood
}

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

2.

I determine tau for the weakly informative prior $N(0, \tau^2)$ by simulating 10000 rounds from normal distribution by changing the value of standard deviation so that 0.5% and 99.5% quantiles $\approx \pm$ 69, Hence $Pr(-69 \leqslant beta \leqslant 69) = 0.99$

```
sim = rnorm(1000000, 0, 26.7)
print(quantile(sim, c(0.005, 0.995)))
```

```
##      0.5%     99.5%
## -68.52450  68.61133
```

Now, weakly informative prior $N(0, \tau^2)$ is determined by $N(0, 26.7^2)$

3.

The weakly informative prior, $N(0, 26.7^2)$, is implemented in *model* block of the below Stan code

```
library('aaltobda')
data(drowning)

stan_code = "
```

```
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;               // intercept
  real beta;                // coefficients for predictors
  real<lower=0> sigma;      // error scale
}

transformed parameters {
  vector[N] mu;
  mu = alpha + beta*x;
}

model {
  beta ~ normal(0, 26.7);    // weakly informative prior
  y ~ normal(mu, sigma);     // likelihood
}

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

drowning_data =  list(N=nrow(drowning),
                      x=drowning$year,
                      y=drowning$drownings,
                      xpred=2019)

fit_drowning = stan(model_code=stan_code, data=drowning_data, refresh=0)
```

```
## Warning: There were 1135 transitions after warmup that exceeded the maximum treedepth. Increase max_
## http://mc-stan.org/misc/warnings.html#maximum-treedepth-exceeded
```

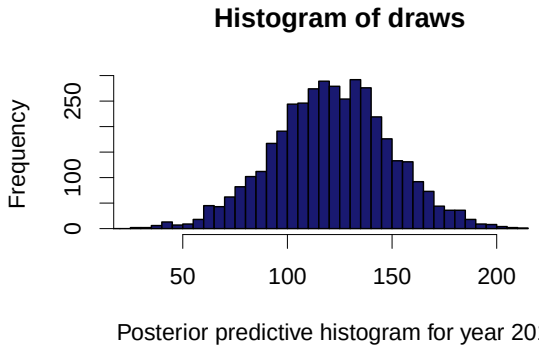
```
## Warning: Examine the pairs() plot to diagnose sampling problems
```

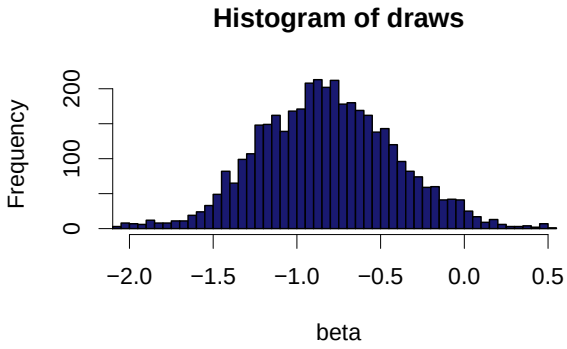
```
monitor(fit_drowning)
```

```
## Inference for the input samples (4 chains: each with iter = 2000; warmup = 0):
##
##            Q5    Q50    Q95   Mean    SD  Rhat Bulk_ESS Tail_ESS
## alpha   414.8 1791.2 3035.9 1777.2 809.1     1     1067     1126
## beta     -1.4   -0.8   -0.1   -0.8   0.4     1     1067     1126
## sigma    21.4   25.7   32.2   26.1   3.3     1     1500     1124
## mu[1]   138.9  152.8  166.5  152.8   8.5     1     1337     1342
## mu[2]   138.7  152.0  165.1  152.0   8.1     1     1366     1342
## mu[3]   138.4  151.2  163.8  151.2   7.8     1     1399     1373
```

```
## mu[4]   138.0  150.4  162.4  150.4   7.5     1     1437     1465
## mu[5]   137.8  149.6  161.0  149.5   7.2     1     1482     1543
## mu[6]   137.5  148.8  159.7  148.7   6.8     1     1535     1494
## mu[7]   137.2  147.9  158.4  147.9   6.5     1     1600     1739
## mu[8]   137.0  147.1  157.1  147.1   6.2     1     1679     1770
## mu[9]   136.6  146.3  155.9  146.3   5.9     1     1774     1933
## mu[10]  136.1  145.5  154.7  145.4   5.7     1     1891     2149
## mu[11]  135.8  144.6  153.4  144.6   5.4     1     2034     2540
## mu[12]  135.3  143.8  152.2  143.8   5.2     1     2221     2551
## mu[13]  134.7  143.0  151.1  143.0   5.0     1     2458     2563
## mu[14]  134.2  142.2  150.0  142.2   4.8     1     2743     2537
## mu[15]  133.7  141.4  148.9  141.3   4.7     1     3078     2416
## mu[16]  133.1  140.6  147.9  140.5   4.5     1     3427     2465
## mu[17]  132.5  139.7  146.9  139.7   4.4     1     3684     2446
## mu[18]  131.6  138.9  146.1  138.9   4.4     1     3865     2544
## mu[19]  130.9  138.1  145.2  138.1   4.4     1     3927     2502
## mu[20]  130.1  137.2  144.5  137.2   4.4     1     3843     2528
## mu[21]  129.2  136.4  143.8  136.4   4.4     1     3645     2643
## mu[22]  128.1  135.5  143.1  135.6   4.5     1     3242     2535
## mu[23]  127.1  134.8  142.6  134.8   4.7     1     2826     2478
## mu[24]  126.1  134.0  142.0  134.0   4.8     1     2499     2511
## mu[25]  124.9  133.2  141.4  133.1   5.0     1     2261     2543
## mu[26]  123.7  132.4  140.9  132.3   5.2     1     2064     2445
## mu[27]  122.5  131.6  140.5  131.5   5.4     1     1904     2389
## mu[28]  121.3  130.8  140.0  130.7   5.7     1     1778     2287
## mu[29]  120.1  130.0  139.7  129.9   6.0     1     1677     2207
## mu[30]  118.8  129.1  139.2  129.0   6.2     1     1591     2009
## mu[31]  117.5  128.3  138.8  128.2   6.5     1     1521     1898
## mu[32]  116.2  127.5  138.5  127.4   6.8     1     1464     1775
## mu[33]  114.8  126.6  138.2  126.6   7.2     1     1415     1710
## mu[34]  113.5  125.7  138.0  125.8   7.5     1     1375     1665
## mu[35]  112.0  124.9  137.7  124.9   7.8     1     1334     1616
## mu[36]  110.7  124.0  137.5  124.1   8.1     1     1304     1622
## mu[37]  109.3  123.2  137.2  123.3   8.5     1     1279     1537
## ypred    74.5  121.5  166.8  121.3  28.1     1     3380     3482
## lp__   -137.8 -134.8 -133.7 -135.1   1.3     1      978     1060
##
## 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).
```

```
par(mfrow=c(1,2))

draws = as.data.frame(fit_drowning)[,41]
hist(draws, breaks=40,  pch = 19, lwd=0.1, cex=0.4, xlim=c(25,225),
     col='midnightblue', xlab='Posterior predictive histogram for year 2019')

draws = as.data.frame(fit_drowning)[,2]
hist(draws, breaks=40,  pch = 19, lwd=0.1, cex=0.4, xlim=c(-2,0.5),
     col='midnightblue', xlab='beta')
```





Here we can see a almost exact match to the histograms provided in the assignment instructions. What is worth mentioning is that the given linear model with gaussian noise is not probably the best model here as the model output negative number of drownings.

Exercise 2. Hierarchical model: factory data with Stan

Separate model

In the separate model, each machine j will have its own mean μ_j and standard deviation σ_j.

Although the assignment asked us to use uniform priors (i.e. the default in Stan), there was a separate message on behalf of the course staff that recommended using weakly informative priors instead. Therefore, I use $N \sim (100, 20^2)$ as the weakly informative prior for the means and $\tau \sim Cauchy(0, 10^2)$ as the weakly informative prior for the standard deviations.

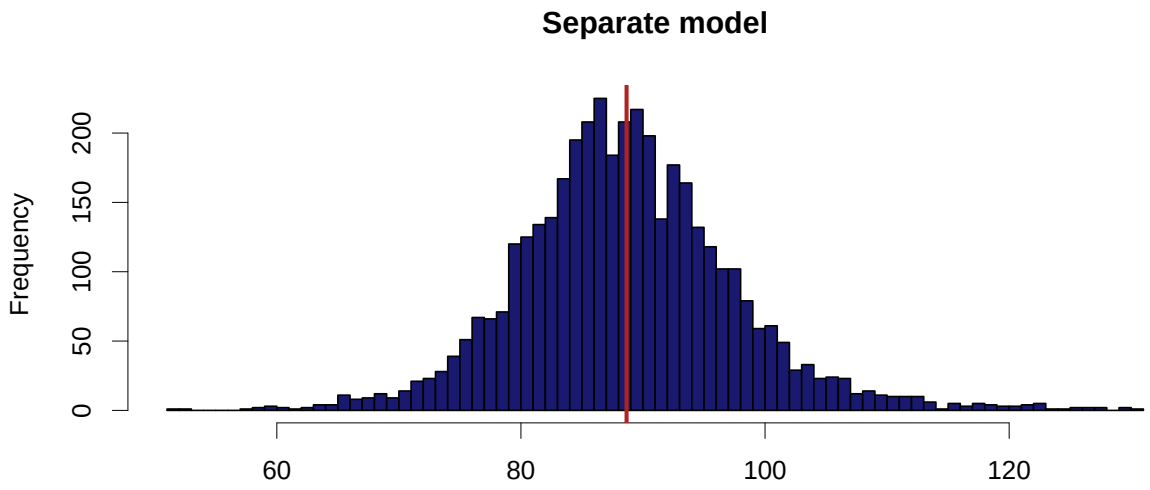
```
library('aaltobda')
data(factory)

factory_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 standard deviations
}
model {
  mu ~ normal(100, 20);             // weakly informative prior for mean
  sigma ~ cauchy(0, 10);            // weakly informative prior for st.deviation
  y ~ normal(mu[x], sigma[x]);      // ypred for each model with their own means and st.deviations
}
generated quantities {
  real ypred;
  ypred = normal_rng(mu[6], sigma[6]); // ypred for sixth model with its own mean and st.deviation
}
"
```

```
factory_pooled = list(N = nrow(factory)*ncol(factory),
                      K = ncol(factory),
                      x = rep(1:6, nrow(factory)),     # [1,6] as a bracket
                      y = c(t(factory[,1:6])))

fit_sep = stan(model_code=factory_stan, data=factory_pooled, refresh=0)
monitor(fit_sep)
```

```
## Inference for the input samples (4 chains: each with iter = 2000; warmup = 0):
##
##             Q5   Q50   Q95  Mean   SD  Rhat Bulk_ESS Tail_ESS
## mu[1]     66.0  80.0  97.2  80.7  9.7     1     3063     1854
## mu[2]     95.4 105.7 115.4 105.6  6.2     1     2321     1560
## mu[3]     79.7  88.8  99.5  89.1  6.1     1     3967     2504
## mu[4]    103.8 111.1 117.6 111.0  4.5     1     2687     1732
## mu[5]     82.2  90.6  99.8  90.8  5.6     1     3107     1920
## mu[6]     75.1  88.2 103.5  88.7  9.0     1     3491     1901
## sigma[1]  12.4  20.0  39.3  22.3  9.4     1     2802     2013
## sigma[2]   7.7  12.3  23.8  13.6  5.5     1     3196     2070
## sigma[3]   8.1  13.2  25.3  14.5  6.0     1     3787     2194
## sigma[4]   5.0   8.4  16.5   9.3  3.9     1     3280     2474
## sigma[5]   6.9  11.2  21.6  12.4  5.0     1     3720     2255
## sigma[6]  12.1  19.1  36.7  21.1  8.8     1     3052     1967
## ypred     50.6  87.7 128.0  88.2 24.6     1     3451     3100
## lp__     -92.7 -87.0 -83.5 -87.4  2.8     1     1442     2119
##
## 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).
```

Separate model i)

```
draws_sep_mu = as.data.frame(fit_sep)[,6]
hist(draws_sep_mu, breaks=100,  pch = 19, lwd=0.1, cex=0.4,
     main='Separate model', col='midnightblue',
     xlab='The posterior distribution of the mean of the quality measurements of the sixth machine')
abline(v=mean(draws_sep_mu), col='firebrick', lwd=2.5)
```



The posterior distribution of the mean of the quality measurements of the sixth machine

```
print(mean(draws_sep_mu))
```

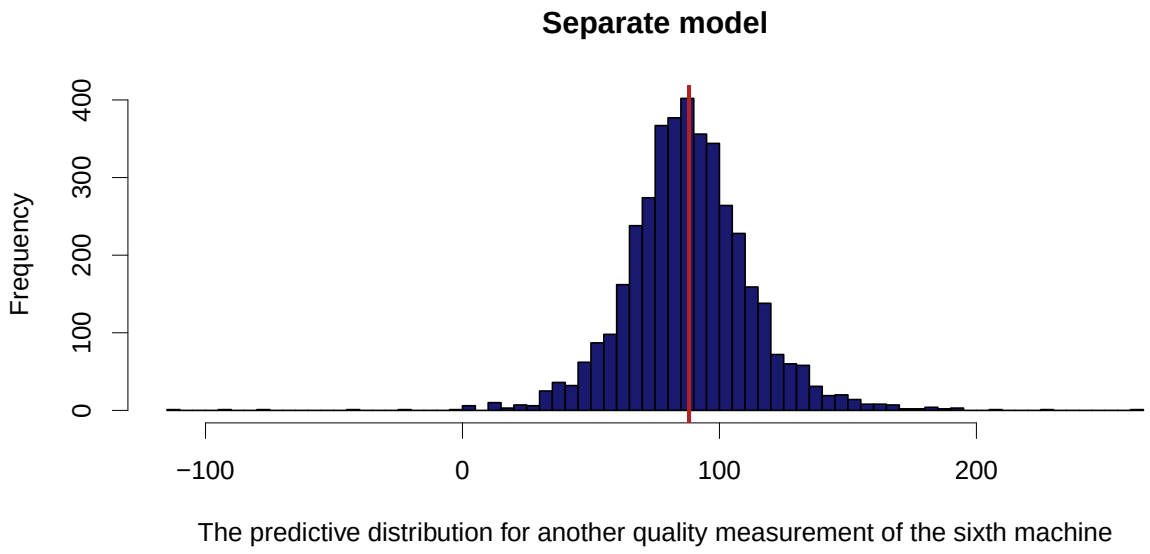
```
## [1] 88.65498
```

```
#axis(side=1, at=seq(0,200, 25))
```

We can notice that the R-hat value obtained for mu[6] with the stan model is close to 1 so we can assume that chains have converged and indeed use the obtained model results to plot the posterior distribution of mu[6]. By looking at the distribution, we can observe that it is quite narrow and. The median is approximately 89.

Separate model ii)

```
draws_sep_pred = as.data.frame(fit_sep)[,13]
hist(draws_sep_pred, breaks=100,  pch = 19, lwd=0.1, cex=0.4,
     main='Separate model', col='midnightblue',
     xlab='The predictive distribution for another quality measurement of the sixth machine')
abline(v=mean(draws_sep_pred), col='firebrick', lwd=2.5)
```




```
print(mean(draws_sep_pred))
```

```
## [1] 88.17024
```

```
#axis(side=1, at=seq(-200,2500,50))
```

The R-hat value of the generated quantitity "ypred" is also close to 1 so we can assume that chains have converged and indeed use the obtained model results to plot the posterior distribution of "ypred".Here we see that the predictive distribution for another quality measurement of the sixth machine is even narrower and has even higher kurtosis than above. The distribution of the prediction mass is quite concentrated around the range of 50-125. The median is approximately 89.

Separate model iii)

By using separate estimates for each of the quality measures, we assume that the experiments were performed independently and that each machine has its own model i.e. there is no link between the models. Without any quality measurement for the seventh machine, we cannot infer any posterior distribution of the mean of the quality measurements of this new machine.

Pooled model

For the pooled model, each machine j will have a common mean μ and standard deviation σ meaning that the parameter μ and σ are now single real valued parameters instead of vectors. However, I still use $N \sim (100, 20^2)$ as the weakly informative prior for the means and $\tau \sim Cauchy(0, 10^2)$ as the weakly informative prior for the standard deviations.

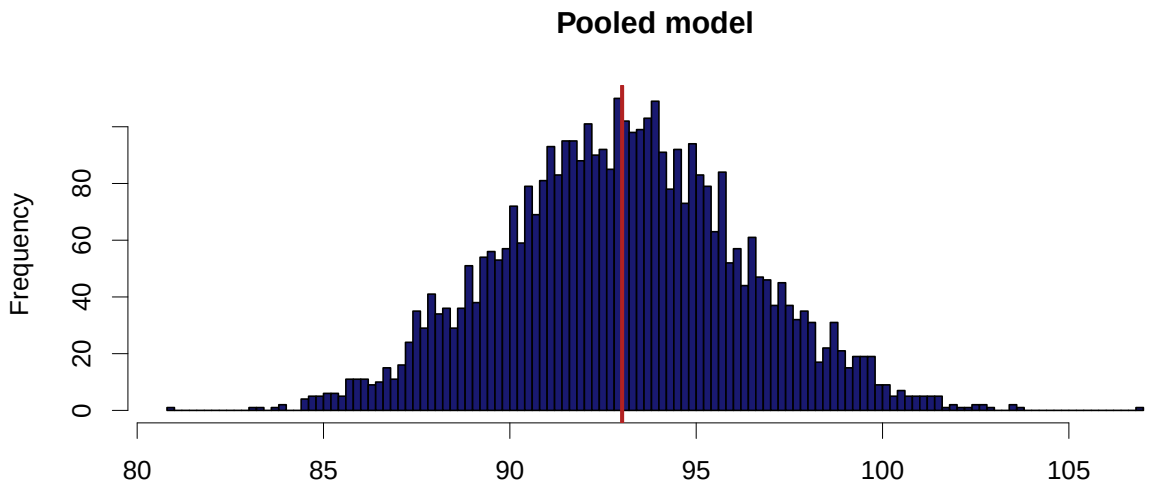
```
library('aaltobda')
data(factory)

factory_stan = "
```

```
data {
  int<lower=0> N;                   // numebr of data points
  vector[N] y;
}
parameters {
  real mu;                          // Common mean
  real<lower=0> sigma;              // Common standard deviation
}
model {
  mu ~ normal(100, 20);             // weakly informative prior for mean
  sigma ~ cauchy(0, 10);            // weakly informative prior for st.deviation
  y ~ normal(mu, sigma);            // ypred for each model with their common means and st.deviations
}
generated quantities {
  real ypred;
  ypred = normal_rng(mu, sigma);   // ypred for sixth model with common mean and st.deviation
}
"

length = length(c(factory$V1, factory$V2, factory$V3, factory$V4, factory$V5, factory$V6))
factory_pooled = list(N=length,
                      y = c(factory$V1, factory$V2, factory$V3, factory$V4, factory$V5, factory$V6))

fit_pool = stan(model_code=factory_stan, data=factory_pooled, refresh=0)
monitor(fit_pool)
```

```
## Inference for the input samples (4 chains: each with iter = 2000; warmup = 0):
##
##           Q5    Q50   Q95   Mean   SD  Rhat Bulk_ESS Tail_ESS
## mu      87.6   93.0  98.5   93.0  3.2     1     2755     2561
## sigma   14.8   17.9  22.4   18.2  2.4     1     2899     2235
## ypred   62.5   93.1 123.4   93.3 18.4     1     3690     3891
## lp__  -102.7 -100.5 -99.9 -100.8  1.0     1     1835     2413
##
## 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).
```

Pooled model i)

```
draws_pool_mu = as.data.frame(fit_pool)[,1]
hist(draws_pool_mu, breaks=100,  pch = 19, lwd=0.1, cex=0.4,
     main='Pooled model', col='midnightblue',
     xlab='The posterior distribution of the mean of the quality measurements of the sixth machine')
abline(v=mean(draws_pool_mu), col='firebrick', lwd=2.5)
```



The posterior distribution of the mean of the quality measurements of the sixth machine

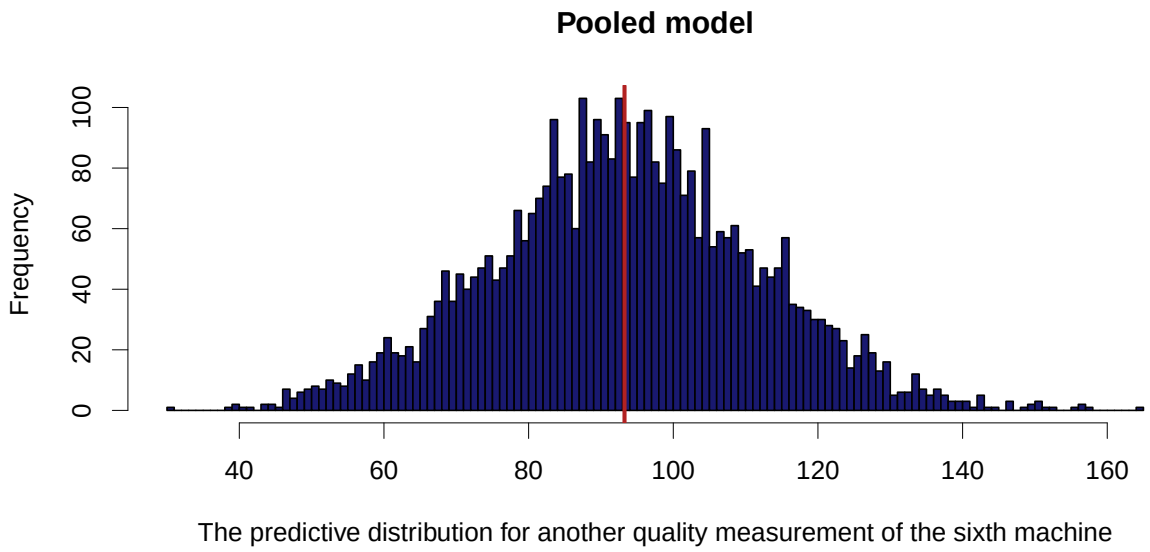
```
print(mean(draws_pool_mu))
```

```
## [1] 93.01198
```

For the pooled model, the obtained R-hat value for "mu" is also close to 1 and therefore the results obtained should be reliable. The posterior distribution of the mean of the quality measurements of the sixth machine resembles highly of normal distribution. The median is approximately 93.

Pooled ii)

```
draws_pool_pred = as.data.frame(fit_pool)[,3]
hist(draws_pool_pred, breaks=100,  pch = 19, lwd=0.1, cex=0.4,
     main='Pooled model', col='midnightblue',
     xlab='The predictive distribution for another quality measurement of the sixth machine')
abline(v=mean(draws_pool_pred), col='firebrick', lwd=2.5)
```




```
print(mean(draws_pool_pred))
```

```
## [1] 93.25814
```

```
#axis(side=1, at=seq(0,200, 25))
```

Again, for the pooled model, the obtained R-hat value for "ypred" is also close to 1 and therefore ther results obtained should be reliable.

For the predictive distribution for another quality measurement of the sixth machine with the pooled model, we see again that the distribution is really normally distributed with some outliers. However, as we want to spesifically identify the predictive distribution for another quality measurement of the sixth machine, this model seems rather useless as it makes no disparity between different machines and measurements. The median is approximately 93.

Pooled iii)

By using the combined estimate averaging data from all quality measurement, we assume that the machines and measurements are quite similar and that the machines have no influence on quality measurements. Therefore, the prediction for the seventh machine simply returns global mean value, as the model again assumes the mean is shared among all machines. Subsuquently, the distribution of the mean of the quality measurements of the seventh machine is the same as the one obtained in question i) for the sixth machine.

Hierarchical model

As informed by the course staff, applying hyperprios is preferred over uniform prior. Hence, I will again use $N \sim (100, 20^2)$ as the weakly informative prior for the means and $\tau \sim Cauchy(0, 10^2)$ as the weakly informative prior for the standard deviations.

```
library('aaltobda')
data(factory)

factory_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;              // common st.deviation
}
model {
  mu0 ~ normal(100,20);             // weakly informative prior for hierarchical mean (hyperprior)
  sigma0 ~ cauchy(0, 10);           // weakly informative prior for hierarchical sigma (hyperprior)
  mu ~ normal(mu0, sigma0);         // weakly informative prior for mean
  sigma ~ cauchy(0,10);             // weakly informative prior for st.deviation
  y ~ normal(mu[x], sigma);
}
generated quantities {
  real ypred;
  real mu7;
  // ypred for sixth model with individual means, common st.deviation and hyperprios
  ypred = normal_rng(mu[6], sigma);
  // posterior for the new seventh machine
  mu7 = normal_rng(mu0, sigma0);
}
"

factory_pooled = list(N = nrow(factory)*ncol(factory),
                      K = ncol(factory),
                      x = rep(1:6, nrow(factory)),     # [1,6] as a bracket
                      y = c(t(factory[,1:6])))

fit_hier = stan(model_code=factory_stan, data=factory_pooled, refresh=0)
```

```
## Warning: There were 8 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
```

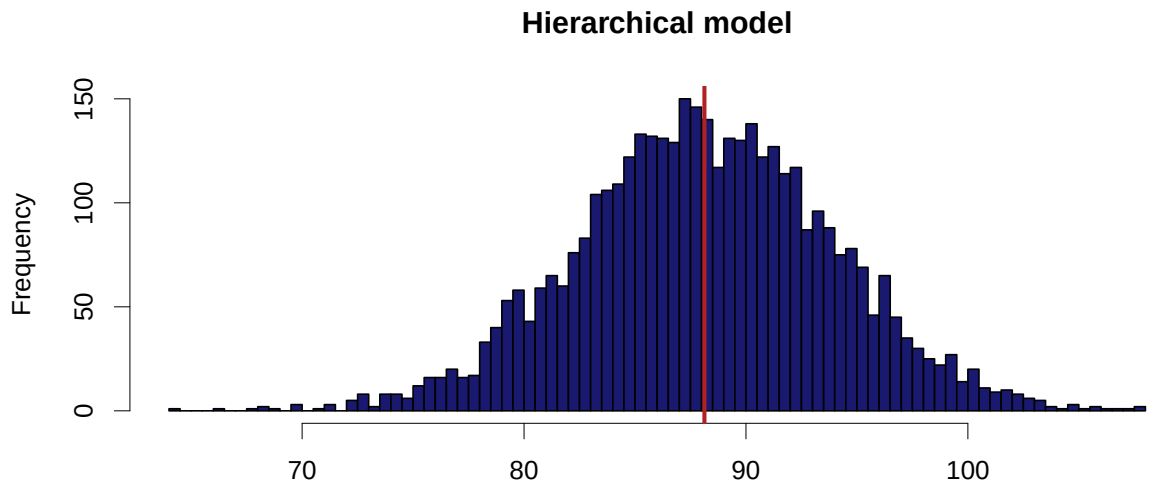
```
monitor(fit_hier)
```

```
## Inference for the input samples (4 chains: each with iter = 2000; warmup = 0):
##
##            Q5    Q50    Q95   Mean   SD  Rhat Bulk_ESS Tail_ESS
## mu0      84.1   93.2  102.5   93.3  5.6  1.00     2437     2545
## sigma0    4.7   11.0   22.3   12.0  5.7  1.01     1077      845
```

```
## mu[1]    70.6   81.3   92.5   81.3  6.6  1.00     2010     2359
## mu[2]    92.1  102.1  112.7  102.2  6.4  1.00     1935     2447
## mu[3]    79.9   89.5   99.0   89.5  5.8  1.00     3407     2590
## mu[4]    94.6  106.2  117.0  106.0  6.7  1.00     1764     1555
## mu[5]    81.7   91.0  100.6   91.1  5.9  1.00     3294     2657
## mu[6]    78.8   88.1   97.5   88.1  5.8  1.00     3078     2580
## sigma    11.8   14.6   19.2   15.0  2.3  1.00     2357     2396
## ypred    61.7   88.0  114.0   88.0 16.1  1.00     3579     3930
## mu7      70.0   93.2  116.4   93.2 14.3  1.00     3610     3378
## lp__   -114.8 -109.9 -107.1 -110.3  2.4  1.00     1200     1850
##
## 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).
```

Hierarchical i)

```
draws_hier_mu = as.data.frame(fit_hier)[,8]
hist(draws_hier_mu, breaks=100,  pch = 19, lwd=0.1, cex=0.4,
     main='Hierarchical model',col='midnightblue',
     xlab='The posterior distribution of the mean of the quality measurements of the sixth machine')
abline(v=mean(draws_hier_mu), col='firebrick', lwd=2.5)
```




```
print(mean(draws_hier_mu ))
```

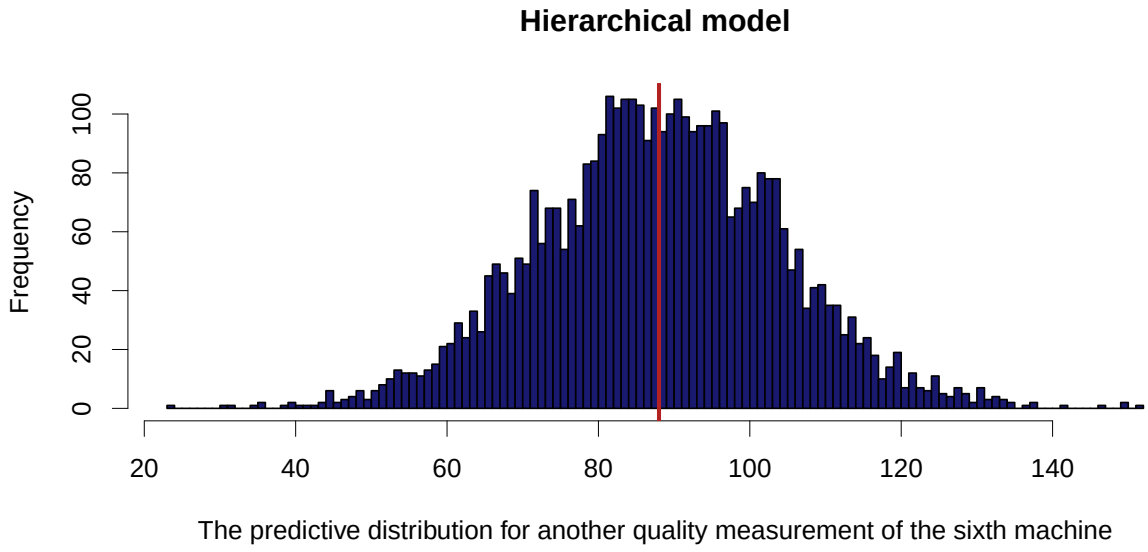
```
## [1] 88.12872
```

For the hierarchical model, the obtained R-hat value for "mu[6]" is also close to 1 and therefore ther results obtained should be reliable.

For the posterior distribution of the mean of the quality measurements of the sixth machine with the hierarchical model we see that the distribution resembles a lot like the one with the pooled model. The difference is that the hierarchical model has fewer outliers and that the median is approximately 88, which is quite a bit less than in the pooled model and a lot closer to the non-pooled model. This can be explained by the fact that this model and the separate model take into account the specificities of the measurements of the sixth machine, that is, they account for machine spesific means.

Hierarchical ii)

```
draws_hier_pred = as.data.frame(fit_hier)[,10]
hist(draws_hier_pred, breaks=100,  pch = 19, lwd=0.1, cex=0.4,
     main='Hierarchical model',col='midnightblue',
     xlab='The predictive distribution for another quality measurement of the sixth machine')
abline(v=mean(draws_hier_pred), col='firebrick', lwd=2.5)
```




```
print(mean(draws_hier_pred))
```

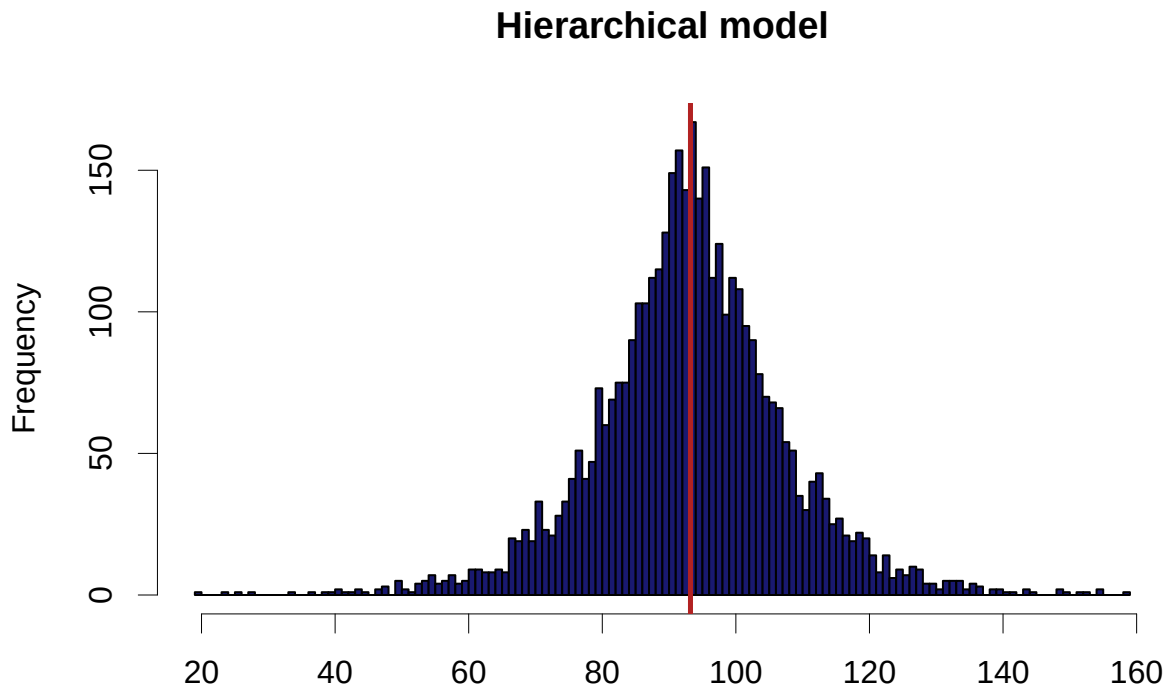
```
## [1] 88.00372
```

Again, for the hierarchical model, the obtained R-hat value for "ypred" is also close to 1 and therefore ther results obtained should be reliable.

For the predictive distribution for another quality measurement of the sixth machine with the hierarchical model we see that the distribution resembles a lot like the one with the pooled model. The difference is that the hierarchical model has fewer outliers and that is is more evenly distributed. The median is approximately 89, which again is less than in the pooled model and a lot closer to the non-pooled model.

Hierarchical iii)

```
draws_hier_7 = as.data.frame(fit_hier)[,11]
hist(draws_hier_7, breaks=100,  pch = 19, lwd=0.1, cex=0.4,
     main='Hierarchical model',col='midnightblue',
     xlab='The posterior distribution of the mean of the quality measurements of the seventh machine.')
abline(v=mean(draws_hier_7), col='firebrick', lwd=2.5)
```




```
print(mean(draws_hier_7))
```

```
## [1] 93.20223
```

Again, for the hierarchical model, the obtained R-hat value for "mu7" is also close to 1 and therefore ther results obtained should be reliable.

The mean is approximately 94, which is almost equal to the one in the pooled model. However, the spread is a lot narrower in comparison to the above distribution. This is because here we sample from the posterior of the hierarchical prior.