BDA - Assignment 7

Contents

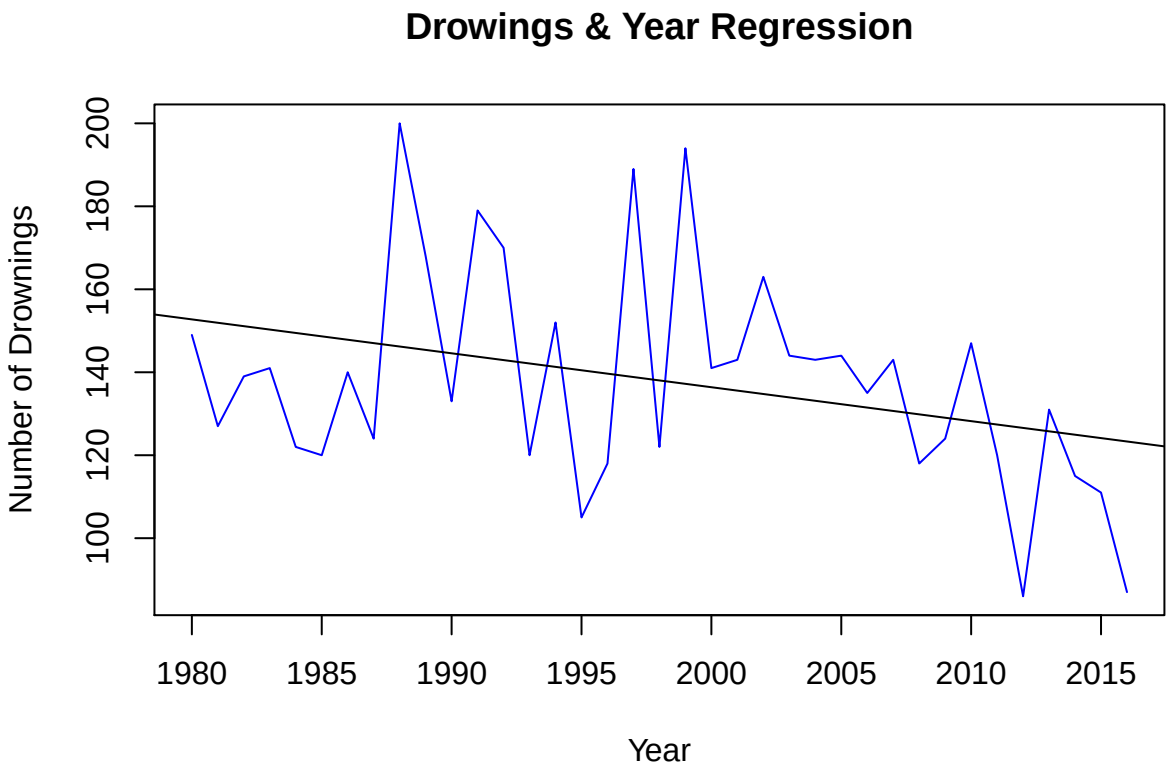
Exercise 1.

1)

```
library(aaltobda)
data("drowning")
```

We can first plot the histogram of the slope of the linear model to see the trend of the number of people drown per year.

```
year<-drowning$year
drownings<-drowning$drownings
relation <- lm(drownings~year)
plot(year,drownings,type="l",col = "blue",main = "Drowings & Year Regression",
     cex = 1.3,pch = 16,xlab = "Year",ylab = "Number of Drownings")
abline(relation)
```



2)

Then we can correct the provided broken stan code.
The stan code provided had 2 main issues:
The bound for sigma should be a lower bound not a upper bound.
The correct declaration is:

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

The second issue was in generated quantities:

```
ypred=normal_rng(mu, sigma)
```

That is incorrect because we have to evaluate the prediction on years, which is xpred. The correct declaration is:

```
ypred = normal_rng(alpha + beta * xpred, sigma);
```

3)

The numerical value of τ is: $\tau = 26.788$.
It was calculate by putting various value into sd and check the value of pnorm(-69,mean=0,sd=26.78888). The value of dist.cdf(-69) should be approximately 0.1/2 for the correct τ.

```
cat(pnorm(-69,mean=0,sd=26.788))
```

```
## 0.00500071
```

The initial value of both α and β in stan model were taken from uniform prior. But according to the requirement we changed the prior for β to weekly-informative prior:

```
beta ~ normal(0, tau);
```

This line is added in block of *model*.

4)

We can then show the prediction for the year 2019 by plotting the histogram of the posterior predictive distribution for the number of people drowning at $\tilde{x} = 2019$.
Before we start, the whole stan code is shown below first.

```
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;
}
parameters {
real alpha;
real beta;
real<lower=0> sigma;
}
transformed parameters {
vector[N] mu;
mu = alpha + beta * x;
}
model {
beta ~ normal(0, tau);
y ~ normal(mu, sigma);
}
generated quantities {
real ypred;
ypred = normal_rng(mu, sigma);
}
```

Then the code for fitting and plotting.

```
data<-list(N=length(year),
           x=year,
           y=drownings,
           xpred=2019,
           tau=26.788)
fit<-stan(file='drownings.stan',data=data)
```

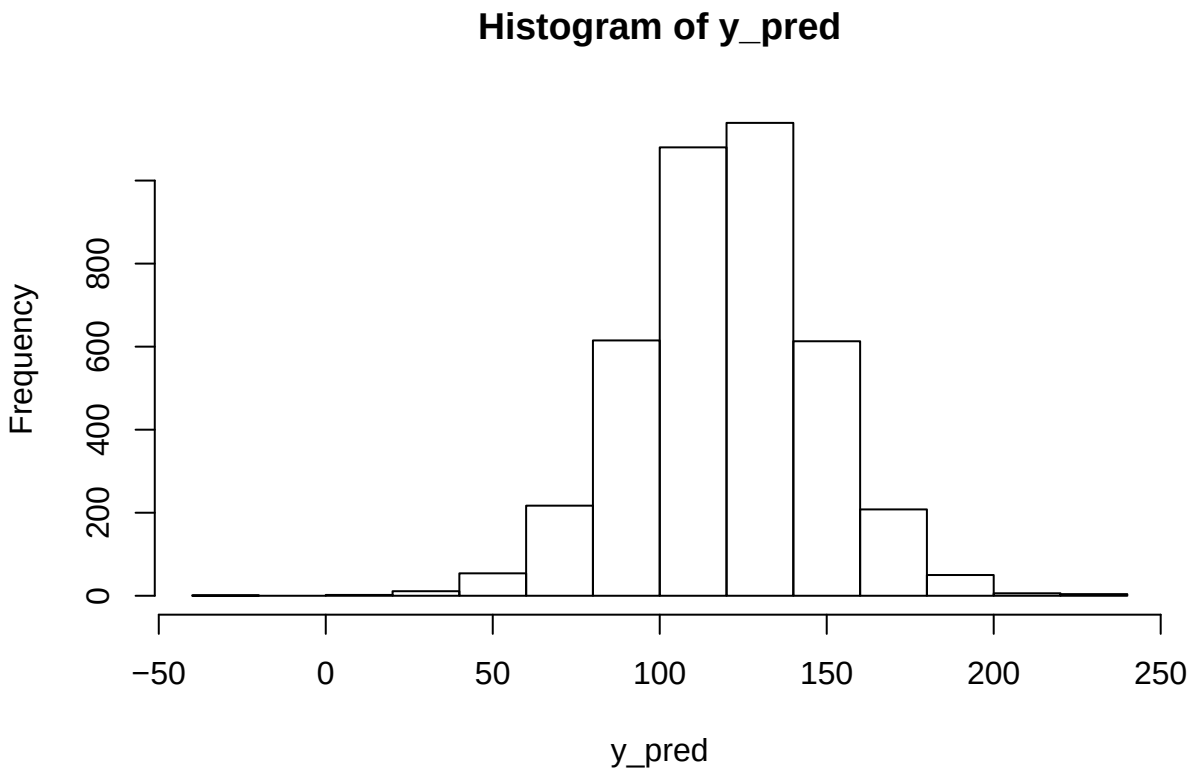
```
## Warning: There were 1092 transitions after warmup that exceeded the maximum treedepth. Increase max_
## http://mc-stan.org/misc/warnings.html#maximum-treedepth-exceeded
```

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

```
fit_ss <- extract(fit, permuted = TRUE)
y_pred <- fit_ss$ypred
cat("Mean for the prediction: ", mean(y_pred))
```

```
## Mean for the prediction:  119.8948
```

```
hist(y_pred)
```



As we can see from the above result that the prediction for 2019 is 121 drownings.

Exercise 2.

```
library(aaltobda)
data("factory")
```

1) Pooled Model

In the pooled model all the machines are considered as one entity, thus we have to combine the all the measurements into one and perform our prediction on the whole data, rather than a subset.
The model is defined as follows.

```
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;
ypred = normal_rng(mu, sigma);
}
```

Then we need to fit it and use it to predict desired values.

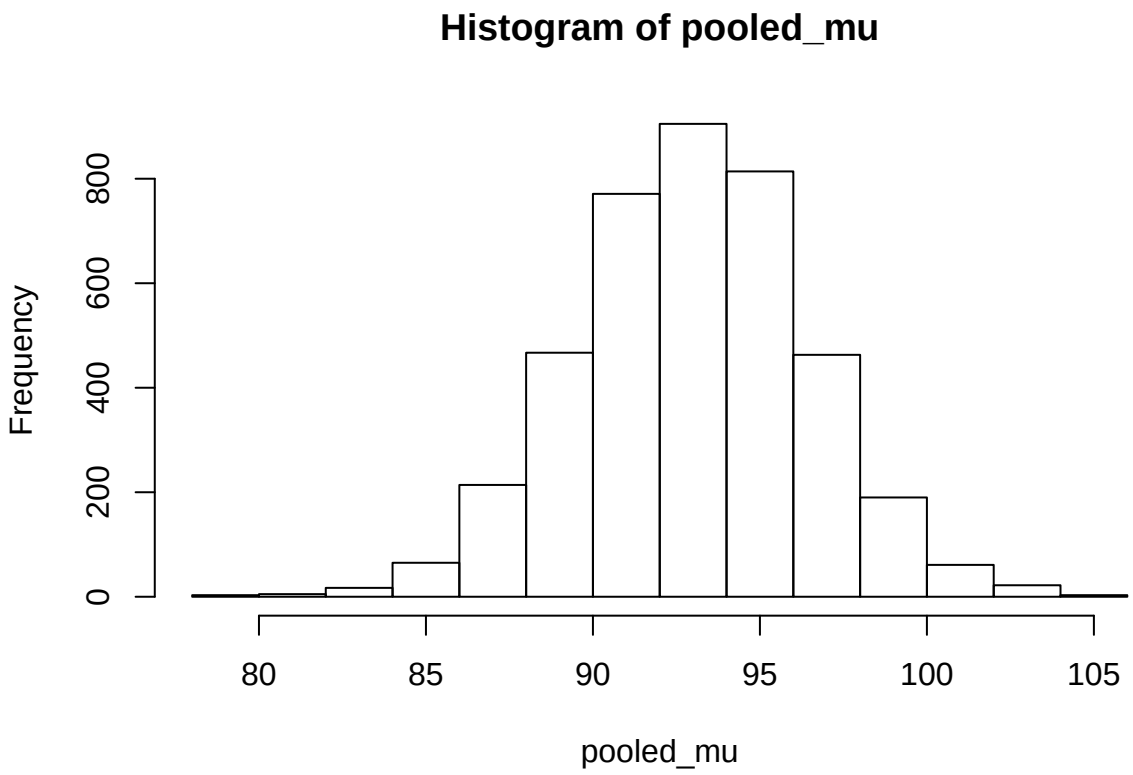
```
pooled_factory_data<-as.vector(as.matrix(factory))
pooled_data<-list(N=length(pooled_factory_data),
                  y=pooled_factory_data)
pooled_fit<-stan(file='pooled_model.stan',data=pooled_data)
```

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

```
pooled_fit_ss <- extract(pooled_fit, permuted = TRUE)
pooled_y_pred <- pooled_fit_ss$ypred
pooled_mu <- pooled_fit_ss$mu
cat("Mean for mu: ", mean(pooled_mu))
```

```
## Mean for mu:  92.9669
```

```
hist(pooled_mu)
```

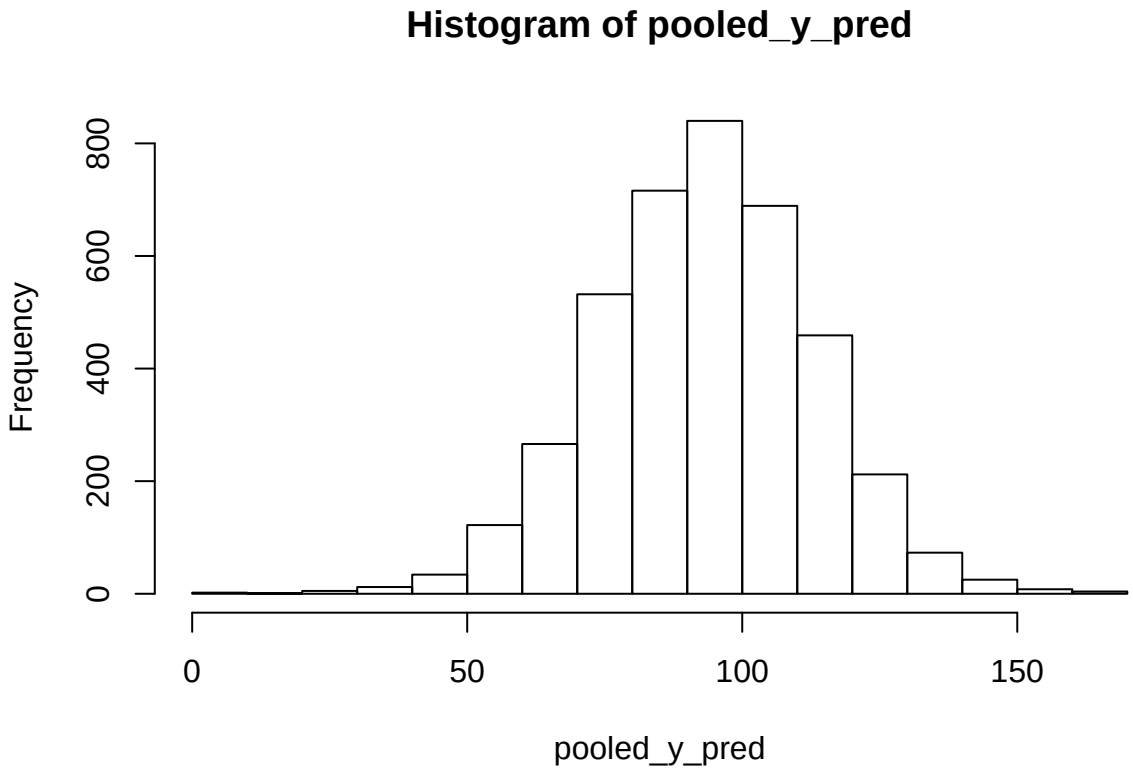


The predictive distribution for another quality measurement of the sixth machine

```
cat("Mean for the prediction: ", mean(pooled_y_pred),'\n')
```

```
## Mean for the prediction:  93.34092
```

```
hist(pooled_y_pred)
```



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

As we combined all the measurements into one, the μ value will be the same for sixth machine, seventh machine or all the machines combined. As mentioned before we don't treat each machine as a separate entity, that's exacly the reason why the posterior distribution of the mean of the quality measurements of the seventh machine should be the same as the posterior distribution of the mean of the quality measurements of the sixth machine.

2) Separate Model

In the separate model we treat every machine as an individual entity, thus when calculating σ or μ we take into consideration only a single machine. The combination of all machines should not effect σ or μ. The stan model is defined as follows.

```
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]);
}
```

Then we need to fit it and use it to predict desired values.

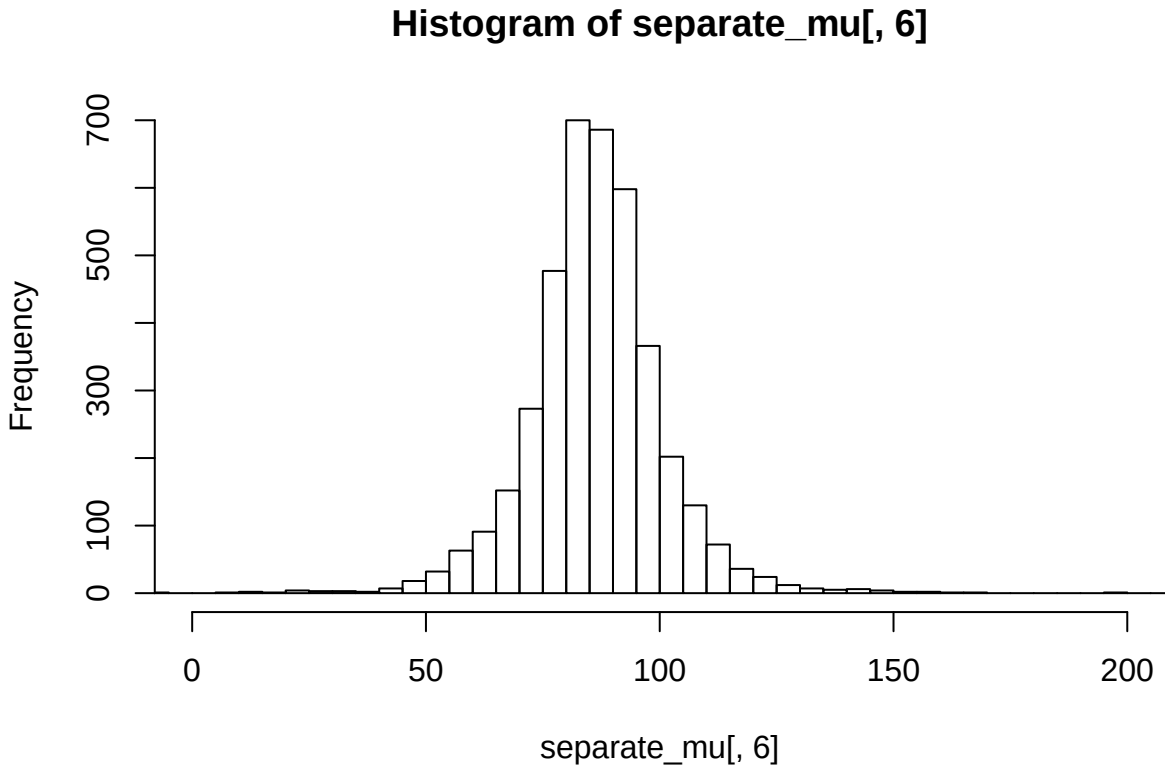
```
separate_data<-list(N=length(as.matrix(factory)),
                    K=6,
                    x=c(1, 1, 1, 1, 1,2, 2, 2, 2, 2,3, 3,
                        3, 3, 3,4, 4, 4, 4, 4,5, 5, 5, 5, 5,6, 6, 6, 6, 6),
                    y=as.vector(as.matrix(factory)))
separate_fit<-stan(file='separate_model.stan',data=separate_data)
```

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

```
separate_fit_ss <- extract(separate_fit, permuted = TRUE)
separate_y_pred <- separate_fit_ss$ypred
separate_mu <- separate_fit_ss$mu
cat("Mean for mu of the sixth machine: ", mean(separate_mu[,6]))
```

```
## Mean for mu of the sixth machine:  85.99997
```

```
hist(separate_mu[,6],breaks=50,xlim=c(0,200))
```

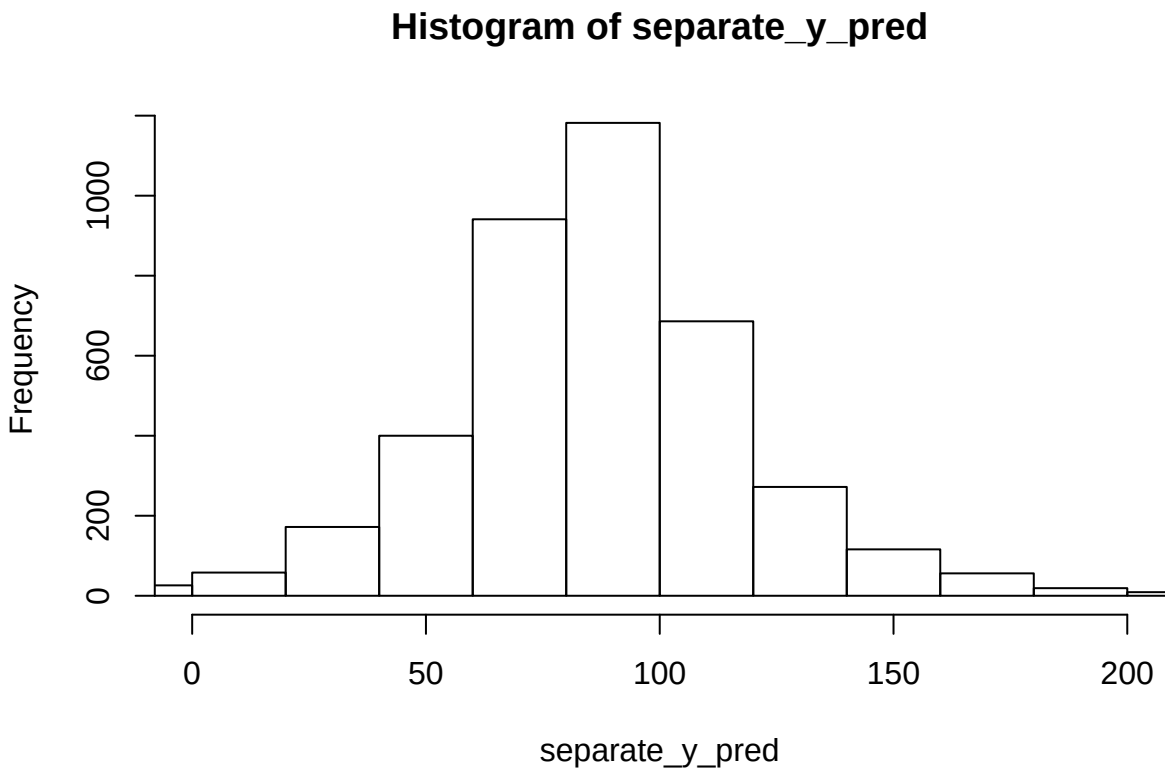


The predictive distribution for another quality measurement of the sixth machine

```
cat("Mean for the prediction: ", mean(separate_y_pred),'\n')
```

```
## Mean for the prediction:  86.14367
```

```
hist(separate_y_pred,breaks=50,xlim=c(0,200))
```



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

As it was stated before, in the separate model we treat each machine separately. Consequently, we have no any information about the seventh machine. Thus we cannot tell anything about its posterior distribution.

3) Hierarchical Model

The hierarchical model is quite interesting in the sense that it can predict measurements for the machines which have no data. For example, there is no data about the seventh machine, but this model can predict its posterior distribution.
The stan model is defined as follows.

```
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 std
}
model {
mu ~ normal(mu0, sigma0);
y ~ normal(mu[x], sigma);
}
generated quantities {
```

```
real ypred6;
real mu7;
ypred6 = normal_rng(mu[6], sigma);
mu7 = normal_rng(mu0, sigma0);
}
```

Then we need to fit it and use it to predict desired values.

```
hierarchical_fit<-stan(file='hierarchical_model.stan',data=separate_data)
```

```
## Warning: There were 13 divergent transitions after warmup. Increasing adapt_delta above 0.8 may help
## http://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
```

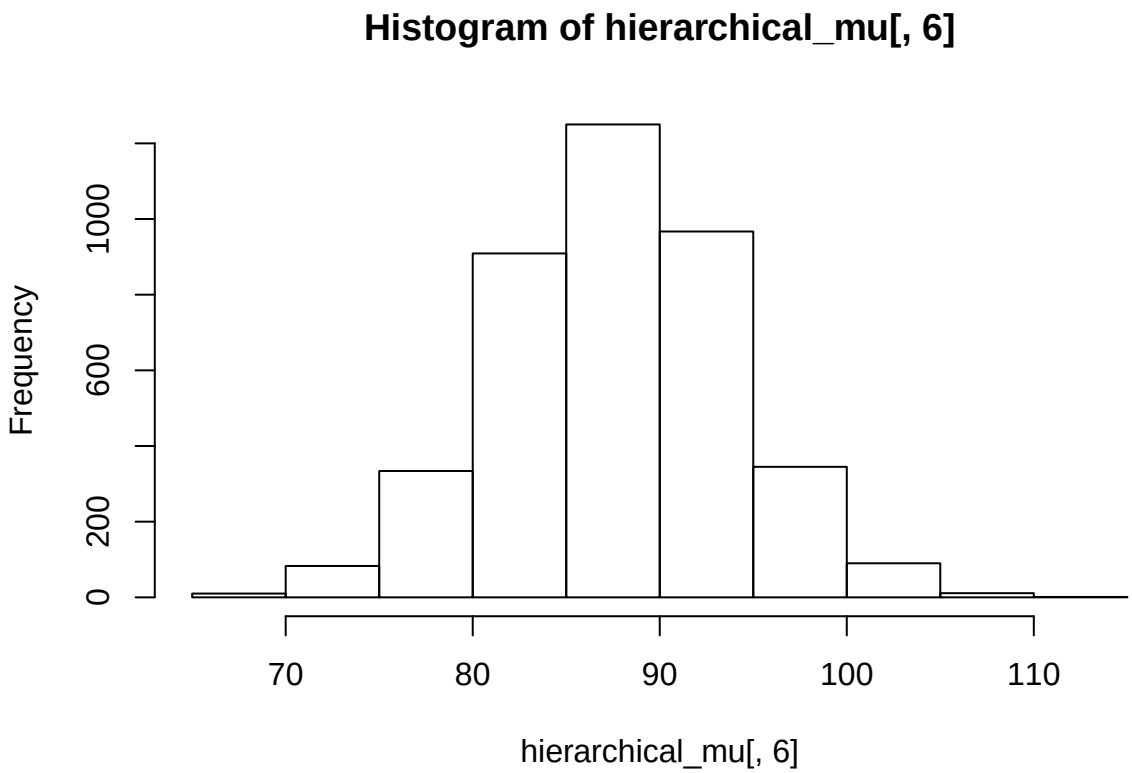
```
## Warning: Examine the pairs() plot to diagnose sampling problems
```

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

```
hierarchical_fit_ss <- extract(hierarchical_fit, permuted = TRUE)
hierarchical_y_pred6 <- hierarchical_fit_ss$ypred6
hierarchical_mu <- hierarchical_fit_ss$mu
hierarchical_mu7 <- hierarchical_fit_ss$mu7
cat("Mean for mu the sixth machine: ", mean(hierarchical_mu[,6]))
```

```
## Mean for mu the sixth machine:  87.68507
```

```
hist(hierarchical_mu[,6])
```

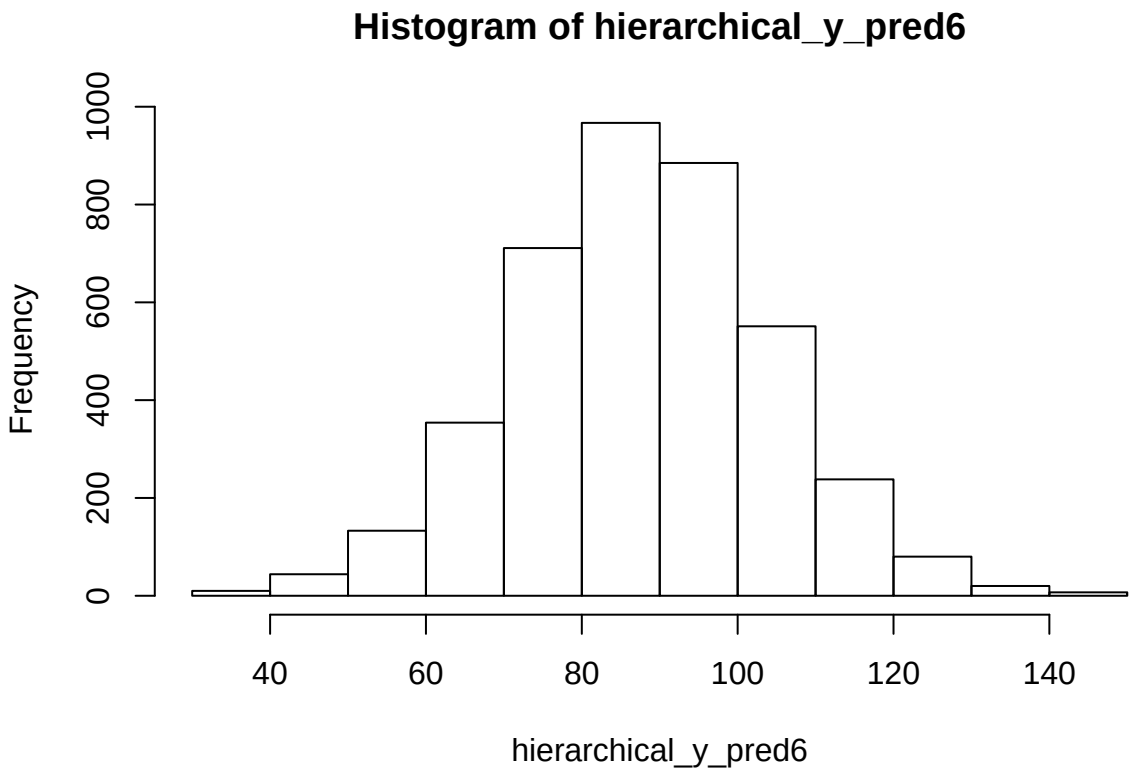


The predictive distribution for another quality measurement of the sixth machine

```
cat("Mean for the prediction: ", mean(hierarchical_y_pred6),'\n')
```

```
## Mean for the prediction:  87.80343
```

```
hist(hierarchical_y_pred6)
```

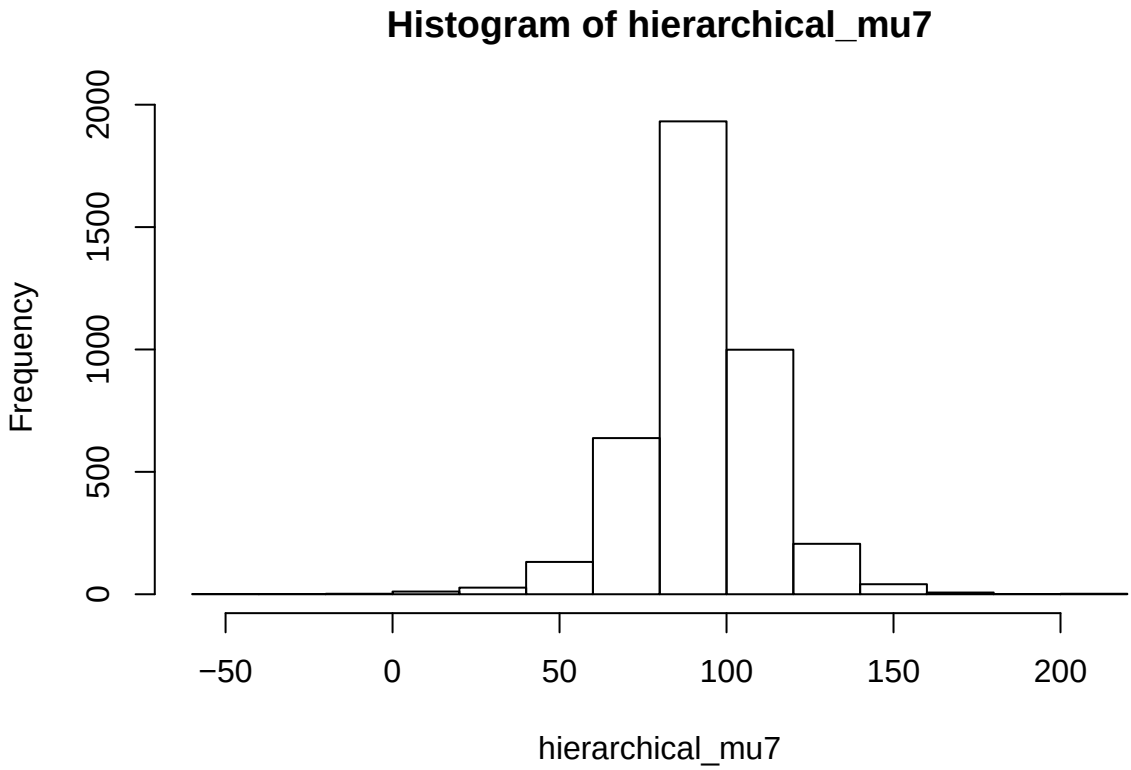


The posterior distribution of the mean of the quality measurements of the seventh machine

```
cat("Mean for mu7: ", mean(hierarchical_mu7),'\n')
```

```
## Mean for mu7:  92.78267
```

```
hist(hierarchical_mu7)
```


Histogram of hierarchical_mu7
Frequency
0
500
1000
1500
2000
−50
0
50
100
150
200
hierarchical_mu7