Lesson 8H: Two Factor ANOVA

Lesson 8.4

Data

Let's explore an example with two factors. We'll use the Warpbreaks data set in `R`. Check the documentation for a description of the data by typing `?warpbreaks`.

```
data("warpbreaks")
?warpbreaks
head(warpbreaks)
```

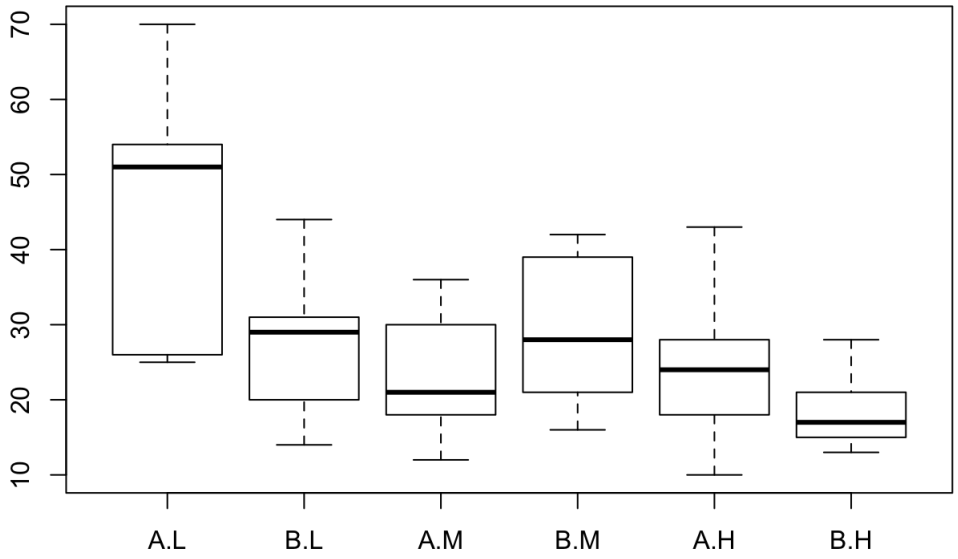
```
##   breaks wool tension
## 1     26    A       L
## 2     30    A       L
## 3     54    A       L
## 4     25    A       L
## 5     70    A       L
## 6     52    A       L
```

```
table(warpbreaks$wool, warpbreaks$tension)
```

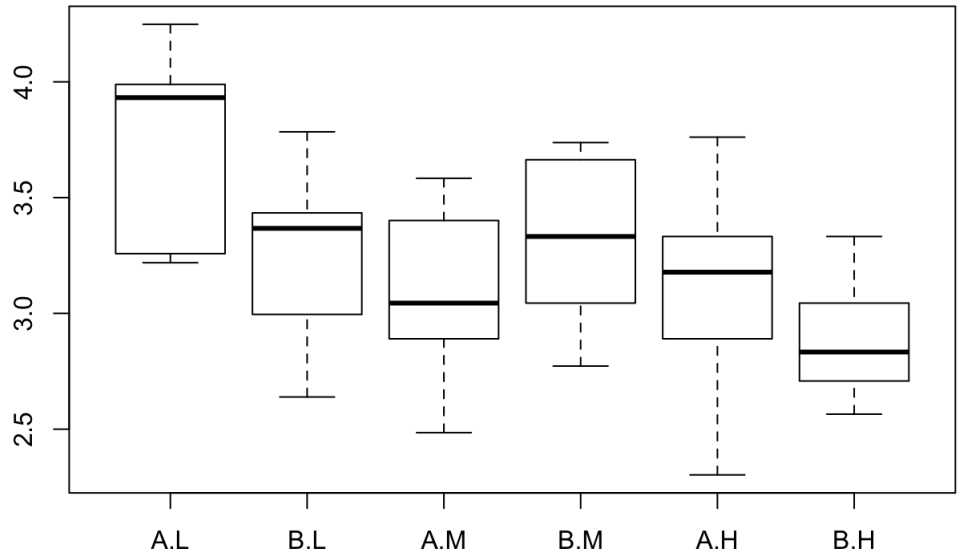
```
##
##     L M H
##   A 9 9 9
##   B 9 9 9
```

Again, we visualize the data with box plots.

```
boxplot(breaks ~ wool + tension, data=warpbreaks)
```




```
boxplot(log(breaks) ~ wool + tension, data=warpbreaks)
```



The different groups have more similar variance if we use the logarithm of breaks. From this visualization, it looks like both factors may play a role in the number of breaks. It appears that there is a general decrease in breaks as we move from low to medium to high tension. Let's start with a one-way model using tension only.

One-way model

```
library("rjags")
```

```
## Loading required package: coda
```

```
## Linked to JAGS 4.2.0
```

```
## Loaded modules: basemod,bugs
```

```
mod1_string = " model {
    for( i in 1:length(y)) {
        y[i] ~ dnorm(mu[tensGrp[i]], prec)
    }

    for (j in 1:3) {
        mu[j] ~ dnorm(0.0, 1.0/1.0e6)
    }

    prec ~ dgamma(5/2.0, 5*2.0/2.0)
    sig = sqrt(1.0 / prec)
} "

set.seed(83)
str(warpbreaks)

data1_jags = list(y=log(warpbreaks$breaks), tensGrp=as.numeric(warpbreaks$tension))

params1 = c("mu", "sig")

mod1 = jags.model(textConnection(mod1_string), data=data1_jags, n.chains=3)
update(mod1, 1e3)

mod1_sim = coda.samples(model=mod1,
                        variable.names=params1,
                        n.iter=5e3)

## convergence diagnostics
plot(mod1_sim)

gelman.diag(mod1_sim)
autocorr.diag(mod1_sim)
effectiveSize(mod1_sim)
```

Here are the results.

```
summary(mod1_sim)
```

```
##
## Iterations = 1001:6000
## Thinning interval = 1
## Number of chains = 3
## Sample size per chain = 5000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##         Mean     SD  Naive SE Time-series SE
## mu[1] 3.5004 0.1367 0.0011162      0.0011056
## mu[2] 3.2138 0.1364 0.0011136      0.0011136
## mu[3] 3.0125 0.1348 0.0011007      0.0010936
## sig   0.5733 0.0553 0.0004515      0.0004711
##
## 2. Quantiles for each variable:
##
##         2.5%    25%    50%    75%  97.5%
## mu[1] 3.2291 3.4087 3.5011 3.5926 3.7692
## mu[2] 2.9468 3.1237 3.2142 3.3052 3.4807
## mu[3] 2.7475 2.9213 3.0129 3.1015 3.2797
## sig   0.4778 0.5347 0.5686 0.6068 0.6932
```

The 95% posterior interval for the mean of group 2 (medium tension) overlaps with both the low and high groups, but the intervals for low and high group only slightly overlap. That is a pretty strong indication that the means for low and high tension are different. Let's collect the DIC for this model and move on to the two-way model.

```
dic1 = dic.samples(mod1, n.iter=1e3)
```

Two-way additive model

With two factors, one with two levels and the other with three, we have six treatment groups, which is the same situation we discussed when introducing multiple factor ANOVA. We will first fit the additive model which treats the two factors separately with no interaction. To get the \(X\) matrix (or design matrix) for this model, we can extract it from a linear model in `R`.

```
X = model.matrix(lm(breaks ~ wool + tension, data=warpbreaks))
head(X)
```

```
##   (Intercept) woolB tensionM tensionH
## 1           1     0        0        0
## 2           1     0        0        0
## 3           1     0        0        0
## 4           1     0        0        0
## 5           1     0        0        0
## 6           1     0        0        0
```

```
tail(X)
```

```
##    (Intercept) woolB tensionM tensionH
## 49           1     1        0        1
## 50           1     1        0        1
## 51           1     1        0        1
## 52           1     1        0        1
## 53           1     1        0        1
## 54           1     1        0        1
```

By default, `R` has chosen the mean for wool A and low tension to be the intercept. Then, there is an effect for wool B, and effects for medium tension and high tension, each associated with dummy indicator variables.

```
mod2_string = " model {
    for( i in 1:length(y)) {
        y[i] ~ dnorm(mu[i], prec)
        mu[i] = int + alpha*isWoolB[i] + beta[1]*isTensionM[i] + beta[2]*isTensionH[i]
    }

    int ~ dnorm(0.0, 1.0/1.0e6)
    alpha ~ dnorm(0.0, 1.0/1.0e6)
    for (j in 1:2) {
        beta[j] ~ dnorm(0.0, 1.0/1.0e6)
    }

    prec ~ dgamma(3/2.0, 3*1.0/2.0)
    sig = sqrt(1.0 / prec)
} "

data2_jags = list(y=log(warpbreaks$breaks), isWoolB=X[,"woolB"], isTensionM=X[,"tensionM"], isTensionH=X[,"tens
ionH"])

params2 = c("int", "alpha", "beta", "sig")

mod2 = jags.model(textConnection(mod2_string), data=data2_jags, n.chains=3)
update(mod2, 1e3)

mod2_sim = coda.samples(model=mod2,
                        variable.names=params2,
                        n.iter=5e3)

## convergene diagnostics
plot(mod2_sim)

gelman.diag(mod1_sim)
autocorr.diag(mod1_sim)
effectiveSize(mod1_sim)
```

Let's summarize the results, collect the DIC for this model, and compare it to the first one-way model.

```
summary(mod2_sim)
```

```
##
## Iterations = 1001:6000
## Thinning interval = 1
## Number of chains = 3
## Sample size per chain = 5000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##            Mean     SD  Naive SE Time-series SE
## alpha   -0.1530 0.1226 0.0010008      0.0016977
## beta[1] -0.2844 0.1500 0.0012249      0.0023705
## beta[2] -0.4867 0.1495 0.0012203      0.0023432
## int      3.5745 0.1205 0.0009837      0.0023147
## sig      0.4541 0.0446 0.0003642      0.0003899
##
## 2. Quantiles for each variable:
##
##            2.5%     25%     50%      75%    97.5%
## alpha   -0.3951 -0.2350 -0.1528 -0.07088  0.08634
## beta[1] -0.5789 -0.3839 -0.2846 -0.18506  0.01238
## beta[2] -0.7828 -0.5870 -0.4875 -0.38741 -0.19070
## int      3.3355  3.4939  3.5753  3.65500  3.80973
## sig      0.3768  0.4222  0.4505  0.48190  0.55303
```

```
(dic2 = dic.samples(mod2, n.iter=1e3))
```

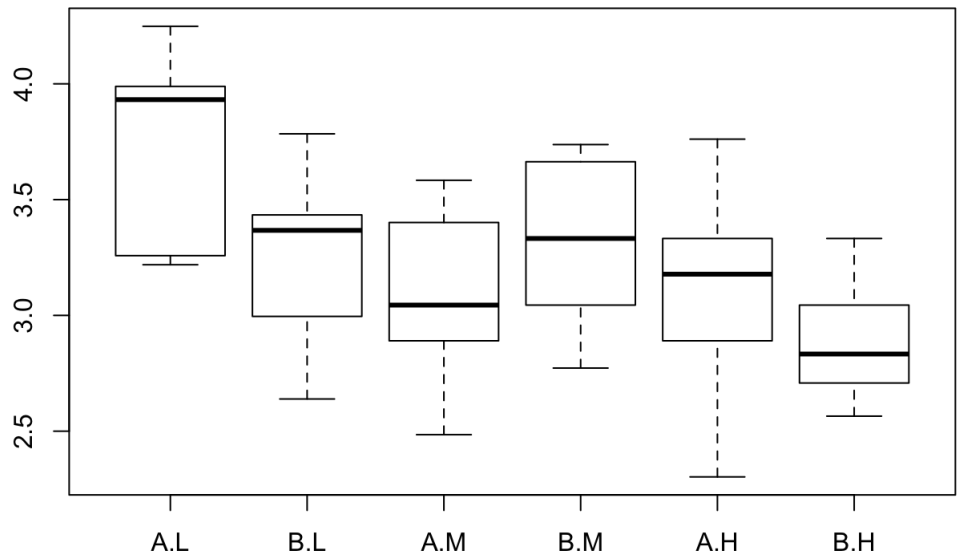
```
## Mean deviance:  55.5
## penalty 5.103
## Penalized deviance: 60.61
```

```
dic1
```

```
## Mean deviance:  66.36
## penalty 3.956
## Penalized deviance: 70.32
```

This suggests there is much to be gained adding the wool factor to the model. Before we settle on this model however, we should consider whether there is an interaction. Let's look again at the box plot with all six treatment groups.

```
boxplot(log(breaks) ~ wool + tension, data=warpbreaks)
```



Our two-way model has a single effect for wool B and the estimate is negative. If this is true, then we would expect wool B to be associated with fewer breaks than its wool A counterpart on average. This is true for low and high tension, but it appears that breaks are *higher* for wool B when there is medium tension. That is, the effect for wool B is not consistent across tension levels, so it may appropriate to add an interaction term. In `R`, this would look like:

```
lmod2 = lm(log(breaks) ~ .^2, data=warpbreaks)
summary(lmod2)
```

```
##
## Call:
## lm(formula = log(breaks) ~ .^2, data = warpbreaks)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.81504 -0.27885  0.04042  0.27319  0.64358
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)      3.7179     0.1247  29.824  < 2e-16 ***
## woolB           -0.4356     0.1763  -2.471  0.01709 *
## tensionM        -0.6012     0.1763  -3.410  0.00133 **
## tensionH        -0.6003     0.1763  -3.405  0.00134 **
## woolB:tensionM   0.6281     0.2493   2.519  0.01514 *
## woolB:tensionH   0.2221     0.2493   0.891  0.37749
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.374 on 48 degrees of freedom
## Multiple R-squared:  0.3363, Adjusted R-squared:  0.2672
## F-statistic: 4.864 on 5 and 48 DF,  p-value: 0.001116
```

Adding the interaction, we get an effect for being in wool B and medium tension, as well as for being in wool B and high tension. There are now six parameters for the mean, one for each treatment group, so this model is equivalent to the full cell means model. Let's use that.

Two-way cell means model

In this new model, \(\mu\) will be a matrix with six entries, each corresponding to a treatment group.

```
mod3_string = " model {
    for( i in 1:length(y)) {
        y[i] ~ dnorm(mu[woolGrp[i], tensGrp[i]], prec)
    }

    for (j in 1:max(woolGrp)) {
        for (k in 1:max(tensGrp)) {
            mu[j,k] ~ dnorm(0.0, 1.0/1.0e6)
        }
    }

    prec ~ dgamma(3/2.0, 3*1.0/2.0)
    sig = sqrt(1.0 / prec)
} "

str(warpbreaks)

data3_jags = list(y=log(warpbreaks$breaks), woolGrp=as.numeric(warpbreaks$wool), tensGrp=as.numeric(warpbreaks$
tension))

params3 = c("mu", "sig")

mod3 = jags.model(textConnection(mod3_string), data=data3_jags, n.chains=3)
update(mod3, 1e3)

mod3_sim = coda.samples(model=mod3,
                        variable.names=params3,
                        n.iter=5e3)
mod3_csim = as.mcmc(do.call(rbind, mod3_sim))

plot(mod3_sim, ask=TRUE)

## convergence diagnostics
gelman.diag(mod3_sim)
autocorr.diag(mod3_sim)
effectiveSize(mod3_sim)
raftery.diag(mod3_sim)
```

Let's compute the DIC and compare with our previous models.

```
(dic3 = dic.samples(mod3, n.iter=1e3))
```

```
## Mean deviance:  52.15
## penalty 7.379
## Penalized deviance: 59.53
```

```
dic2
```

```
## Mean deviance:  55.5
## penalty 5.103
## Penalized deviance: 60.61
```

```
dic1
```

```
## Mean deviance:  66.36
## penalty 3.956
## Penalized deviance: 70.32
```

This suggests that the full model with interaction between wool and tension (which is equivalent to the cell means model) is the best for explaining/predicting warp breaks.

Results

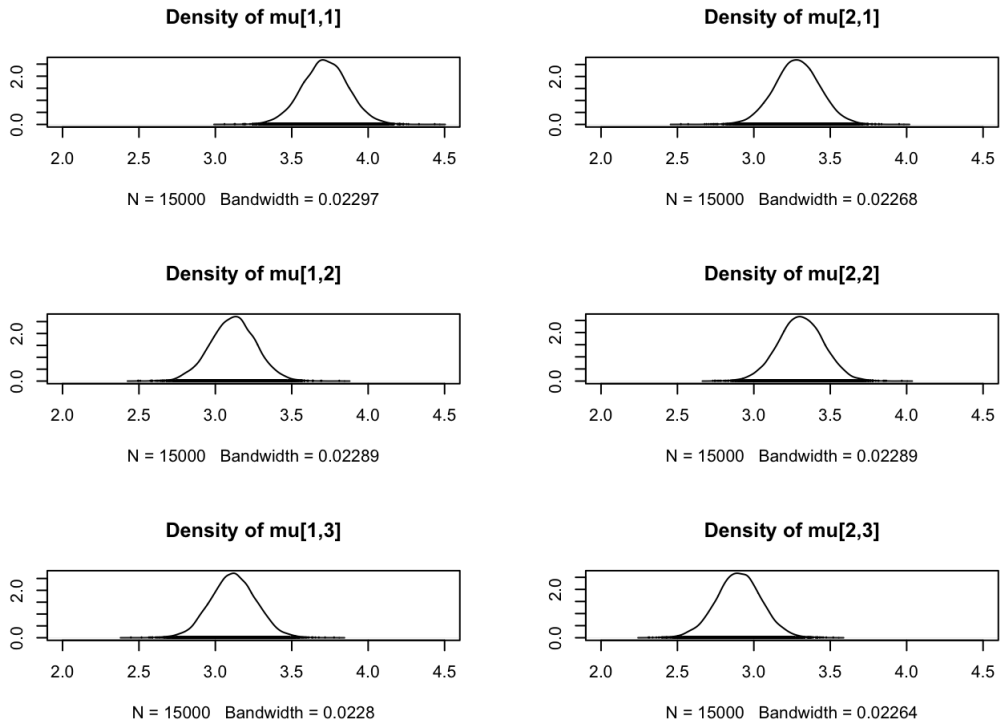
```
summary(mod3_sim)
```

```
##
## Iterations = 1001:6000
## Thinning interval = 1
## Number of chains = 3
## Sample size per chain = 5000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##          Mean      SD  Naive SE Time-series SE
## mu[1,1] 3.719 0.14980 0.0012231      0.0012303
## mu[2,1] 3.282 0.14847 0.0012122      0.0012597
## mu[1,2] 3.116 0.14821 0.0012101      0.0012288
## mu[2,2] 3.308 0.14859 0.0012132      0.0012713
## mu[1,3] 3.117 0.14869 0.0012140      0.0012141
## mu[2,3] 2.902 0.14865 0.0012137      0.0012213
## sig     0.443 0.04486 0.0003663      0.0004047
##
## 2. Quantiles for each variable:
##
##          2.5%    25%    50%    75%  97.5%
## mu[1,1] 3.420 3.6188 3.7186 3.8175 4.0143
## mu[2,1] 2.991 3.1857 3.2827 3.3818 3.5744
## mu[1,2] 2.819 3.0173 3.1173 3.2153 3.4067
## mu[2,2] 3.015 3.2097 3.3072 3.4076 3.5981
## mu[1,3] 2.827 3.0181 3.1169 3.2153 3.4077
## mu[2,3] 2.603 2.8039 2.9011 2.9998 3.1939
## sig     0.364 0.4114 0.4401 0.4714 0.5397
```

```
HPDinterval(mod3_csim)
```

```
##             lower     upper
## mu[1,1] 3.4180926 4.0088513
## mu[2,1] 2.9926109 3.5757322
## mu[1,2] 2.8151067 3.4015734
## mu[2,2] 3.0209483 3.6018741
## mu[1,3] 2.8260308 3.4070867
## mu[2,3] 2.6039687 3.1940863
## sig     0.3555993 0.5274215
## attr(,"Probability")
## [1] 0.95
```

```
par(mfrow=c(3,2)) # arrange frame for plots
densplot(mod3_csim[,1:6], xlim=c(2.0, 4.5))
```



It might be tempting to look at comparisons between each combination of treatments, but we warn that this could yield spurious results. When we discussed the statistical modeling cycle, we said it is best not to search your results for interesting hypotheses, because if there are many hypotheses, some will appear to show "effects" or "associations" simply due to chance. Results are most reliable when we determine a relatively small number of hypotheses we are interested in beforehand, collect the data, and statistically evaluate the evidence for them.

One question we might be interested in with these data is finding the treatment combination that produces the fewest breaks. To calculate this, we can go through our posterior samples and for each sample, find out which group has the smallest mean. These counts help us determine the posterior probability that each of the treatment groups has the smallest mean.

```
prop.table( table( apply(mod3_csim[,1:6], 1, which.min) ) )
```

```
##
##         2         3         4         5         6
## 0.0178000 0.1158667 0.0106000 0.1136000 0.7421333
```

The evidence supports wool B with high tension as the treatment that produces the fewest breaks.