Homework 8

Biology 697

11/8/2012

1 ANOVA

Legends abound of stats classes where you have to do a multi-way ANOVA by hand. Rather than make you do this by hand, to understand what you're getting out of your anova output in R, I want you to crank through an ANOVA in R. Using the Zooplankton data from class, create an anova table that looks just like what you would get from the `anova` function with SS, Df, MSE, F, and P values. Check yourself using the output from R. You don't need to write a general function - just step through the whole thing yourself. 2 Extra points for writing a general anova function, though, and 2 more if it can use a formula.

```
library(plyr)
zoop <- read.csv("../lectures/data/18e2ZooplanktonDepredation.csv")
zoop$block <- factor(zoop$block)

#calcaulate the means of teach treatment group.
treatmentMeans <- ddply(zoop, .(treatment), summarise, avg = mean(zooplankton))
blockMeans <- ddply(zoop, .(block), summarise, avg = mean(zooplankton))

#remember to scale by the number of replicates of each treatment group
treatmentSS <- 5*sum((treatmentMeans$avg - mean(zoop$zooplankton))^2)
blockSS <- 3*sum((blockMeans$avg - mean(zoop$zooplankton))^2)

treatmentDF <- (length(levels(zoop$treatment))-1)
blockDF <- (length(levels(zoop$block))-1)

RSS <- sum((zoop$zooplankton - mean(zoop$zooplankton))^2) - treatmentSS - blockSS

ndf <- data.frame(factor = c("Treatment", "Block", "Residual"),
                  SS = c(treatmentSS, blockSS, RSS),
                  DF=c(treatmentDF, blockDF, nrow(zoop) - 1 - treatmentDF - blockDF))

ndf$MSE <- ndf$SS/ndf$DF
```

```r
ndf$F <- ndf$MSE / ndf$MSE[3]
ndf$P <- pf(ndf$F, ndf$DF, ndf$DF[3], lower.tail=F)

#compare
ndf
```

```
##      factor    SS DF    MSE      F        P
## 1 Treatment 6.857  2 3.4287 16.366 0.001488
## 2     Block 2.340  4 0.5850  2.792 0.101031
## 3  Residual 1.676  8 0.2095  1.000 0.500000
```

```r
anova(lm(zooplankton ~ treatment+block, data=zoop))
```

```
## Analysis of Variance Table
##
## Response: zooplankton
##           Df Sum Sq Mean Sq F value Pr(>F)
## treatment  2   6.86    3.43   16.37 0.0015 **
## block      4   2.34    0.58    2.79 0.1010
## Residuals  8   1.68    0.21
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
#A function using the formula interface
anovaFun <- function(model, dataFrame){
 response <- all.vars(model)[1]
 predictors <- all.vars(model)[-1]
 grandMean <- mean(dataFrame[[response]], na.rm=T)

 #get the SS and DF for each factor
 anovaTab <- sapply(predictors, function(x){
   ss <- ddply(dataFrame, x, function(adf)
    nrow(adf) * (mean(adf[[response]], na.rm=T) - grandMean)^2)

   ss <- sum(ss$V1)

   df <- length(levels(dataFrame[[x]]))-1

   ms <- ss/df

   return(c(SS = ss, DF = df, MS = ms))

 })

 #correct for shape
```

```
 anovaTab <- data.frame(t(anovaTab))

 #calculate information for residuals
 SST <- sum((dataFrame[[response]] - grandMean)^2)
 residual<- c(SST - sum(anovaTab$SS),  #SSE
              nrow(dataFrame) - 1 - sum(anovaTab$DF)) #DFE
 residual[3] <- residual[1]/residual[2]

 anovaTab <- rbind(anovaTab, residual = residual)

 #F calculations
 anovaTab$F <- anovaTab$MS/residual[3]
 anovaTab$p <- pf(anovaTab$F, anovaTab$DF, residual[2], lower.tail=F)
 anovaTab
}


#compare!
anovaFun(zooplankton ~ treatment + block, data=zoop)

##              SS DF     MS      F        p
## treatment 6.857  2 3.4287 16.366 0.001488
## block     2.340  4 0.5850  2.792 0.101031
## residual  1.676  8 0.2095  1.000 0.500000

anova(lm(zooplankton ~ treatment+block, data=zoop))

## Analysis of Variance Table
##
## Response: zooplankton
##           Df Sum Sq Mean Sq F value Pr(>F)
## treatment  2   6.86    3.43   16.37 0.0015 **
## block      4   2.34    0.58    2.79 0.1010
## Residuals  8   1.68    0.21
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

2 GLMs and Experiments

An experiment was conducted to see if seeds that occured in seed banks with a certain mucous producing worm survived better in the field than those that we in a field with no worms. These worms were very particular about where they liked to hang out. The hypothesized mechanism of the worm's effect was the anti-fungal properties of its mucus. To test this idea, the experimenter

buried bags of between 50-100 seeds in 10 separate blocks with one bag of each treatment per block. Seeds in each bag were treated with either an anti-fungal spray, nothing, or a worm-mucus spray. After 1 year, the experimenter came back, planted the seed bags in a greenhouse setting, and counted the number of plants that germinated.

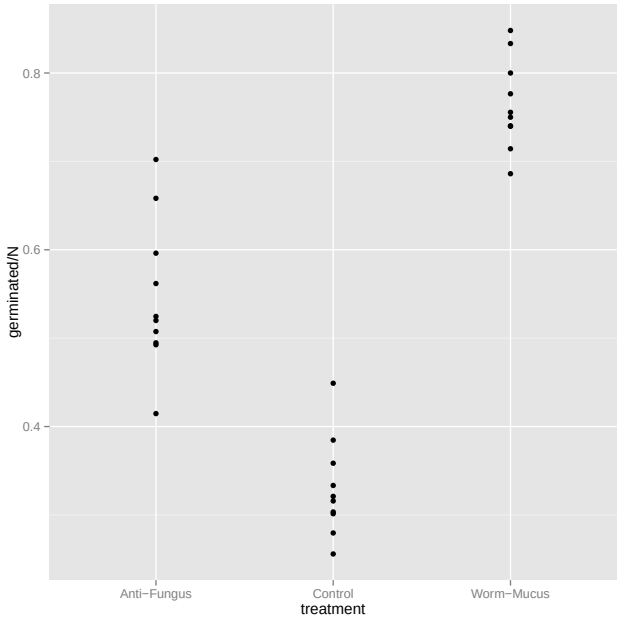
1. What kind of relationship would would describe the effects of treatment on survival and germination?

2. What kind of error distribution should the number of germinated seeds have?

3. Fit this model with the proper error distribution. Check the diagnostics. Perform the appropriate tests to determine whether worm-mucus has an effect, and if that effect appears to indicate that the mucus's anti-fungal properties may be playing a role here.

4. What conclusions can the experimenter draw about the worm mucus's effect as an anti-fungal agent given the experiment they performed? Think before you answer here.

```
survival <- read.csv("./seed_survival.csv")
survival$block <- as.factor(survival$block)
qplot(treatment, germinated/N, data=survival)
```



```r
aglm <- glm(germinated/N ~ treatment + block, data=survival, weights=N, family=binomial(link
summary(aglm)
```

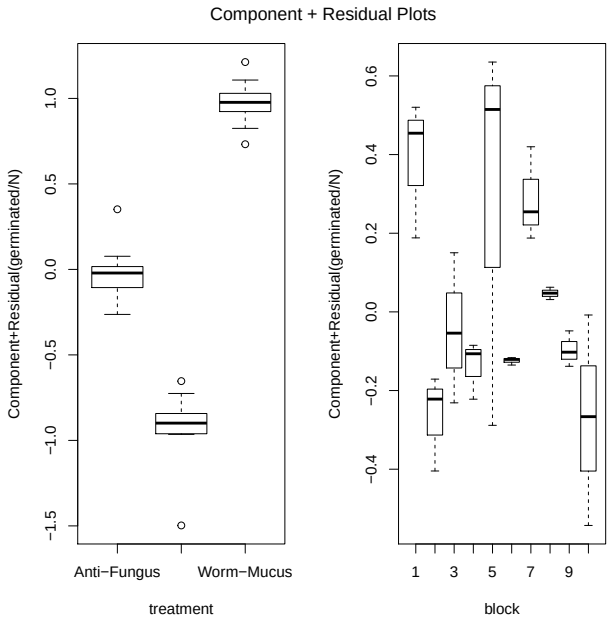
```
##
## Call:
## glm(formula = germinated/N ~ treatment + block, family = binomial(link = "logit"),
##     data = survival, weights = N)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.5723  -0.1328   0.0421   0.2260   1.7787
##
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)            0.620      0.149    4.16  3.1e-05 ***
## treatmentControl      -0.925      0.108   -8.59  < 2e-16 ***
## treatmentWorm-Mucus    0.982      0.113    8.71  < 2e-16 ***
## block2                -0.686      0.193   -3.55  0.00038 ***
## block3                -0.487      0.202   -2.41  0.01588 *
## block4                -0.571      0.191   -2.99  0.00283 **
## block5                -0.154      0.191   -0.81  0.41941
## block6                -0.543      0.197   -2.76  0.00584 **
## block7                -0.144      0.206   -0.70  0.48603
## block8                -0.371      0.198   -1.87  0.06140 .
## block9                -0.518      0.207   -2.51  0.01223 *
## block10               -0.718      0.196   -3.65  0.00026 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 346.702  on 29  degrees of freedom
## Residual deviance:  16.131  on 18  degrees of freedom
## AIC: 178.3
##
## Number of Fisher Scoring iterations: 3
```

```r
Anova(aglm)
```

```
## Analysis of Deviance Table (Type II tests)
##
## Response: germinated/N
##           LR Chisq Df Pr(>Chisq)
## treatment    302.3  2     <2e-16 ***
## block         27.7  9     0.0011 **
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
crPlots(aglm)
```




```
library(contrast)
contrast(aglm,
         list(treatment = "Worm-Mucus", block=levels(survival$block)),
         list(treatment = c("Control"), block=levels(survival$block)),
         type="average")

## glm model parameter contrast
##
##   Contrast   S.E. Lower Upper     t df Pr(>|t|)
## 1    1.907 0.1161 1.663 2.151 16.42 18        0
```

```
We cannot draw strong conclusions, as the mucus may
have affected fungal infection, or it may do something
else that facilitates germination in the seed bank. We
also do not know how the anti-fungal agent affects
germination of plants per se.  It may remove fungus
but hinder germination.  There's a lot here still to explore.
```

3 Components and Residuals in MLR

The component + residual plot is absolutely essential to understanding the multiple linear regression. But what if we didn't have a car library to depend on? Let's implement our own crplot, using ggplot2. Write a function that, given a model with continuous predictors, will give you a component+residual plot in ggplot2. Implement it for the keeley data we looked at in class.

There are two approaches to this, each of which requires one new tool for you to play with. One approach is to create 3 separeate ggplots in a list. Once you do this, the function `grid.arrange` from the gridExtra package along with the function `do.call` can allow you to generate a multi-panel set of ggplots. For example, `do.call(grid.arrange, listOfGGplots)` where listOfGGplots is 3 different ggplot2 objects will give you a 3-panel figure with each of those graphs. Note, as we're talking about lists, lapply may come in handy. Also, note that a fit linear model is a list - so, for a model fit called alm, alm$model actually returns the data that is used to fit the model.

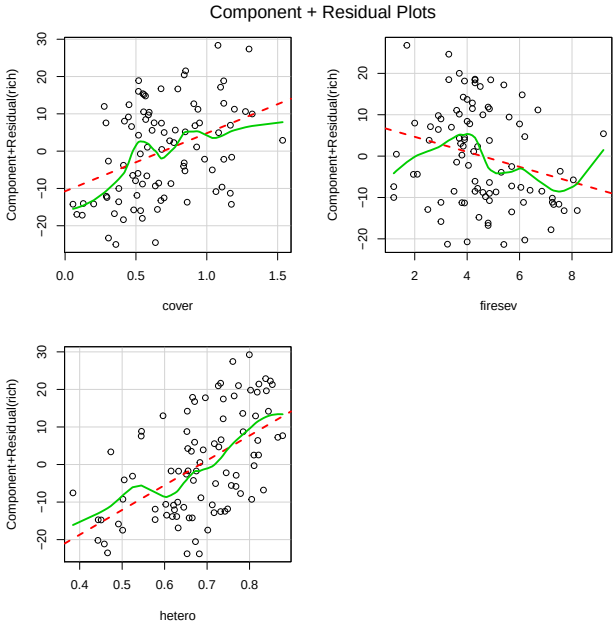
The second way is to use the `melt` function from the `reshape2` package. Try melting a data frame, and see what happens. You'll need to be include the residuals in your data frame that you melt, and use them as your id.vars. This way is a little less intuitive at first, but ultimately you can plot everything using a single call to ggplot2.

Both will require a mixture of lapply, for loops, and others at the right time and place to loop over things. Just to keep you in practice. There are also likely other solutions here. Feel free to try whatever way you would like.

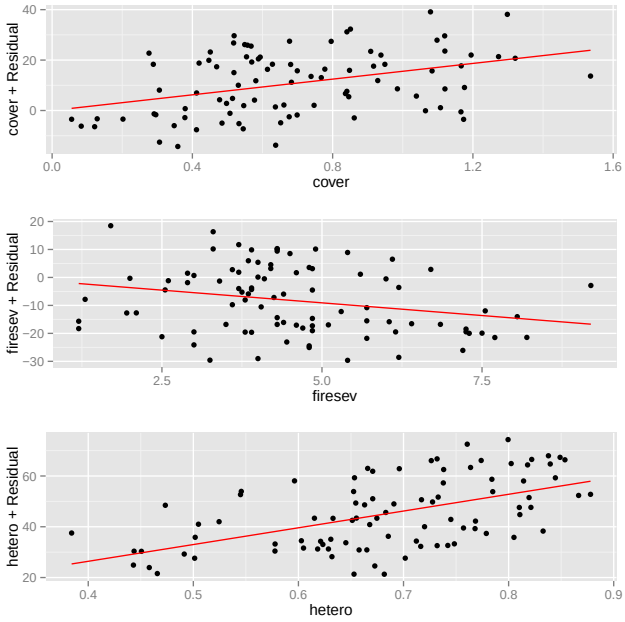
```
keeley <- read.csv("../lectures/data/Keeley_rawdata_select4.csv")
klm <- lm(rich ~ cover + firesev + hetero, data=keeley)

#using grid.arrange
crPlot.ggplot2 <- function(fitMod){
  #extract the data and coefs
  ndf <- fitMod$model[,-1]
  coefs <- coef(klm)[-1]

  #loop over it all to get the component*coef, then add residual
  for(i in 1:ncol(ndf)) ndf[,i] <- ndf[,i] * coefs[i]
  ndf <- ndf+residuals(fitMod)
  predictorDF <- fitMod$model[,-1]

  #create a list of ggplots
```

```
  plotList <- lapply(1:length(coefs), function(i){
    qplot(predictorDF[,i], ndf[,i], geom="point") +
      stat_smooth(method="lm", se=F, col="red") +
      ylab(paste(names(coefs)[i], "+ Residual")) +
      xlab(names(coefs)[i])

  })

  #use grid.arrange to put it all together
  do.call(grid.arrange, plotList)

}

#A function that uses melt
crPlot.ggplot2.melted <- function(fitMod){
  ndf <- cbind(fitMod$model[,-1], residual = residuals(fitMod))

  #use melt to get the value of a residual
  #and the value of the component in one place
  ndf <- melt(ndf, id.vars="residual")
  coefs <- coef(klm)[-1]

  ndf$cr <- rep(NA, nrow(ndf))

  #now fill in a new column with component*coef + residual
  for(i in 1:length(coefs)) {
    idx <- which(ndf$variable == names(coefs)[i])
    ndf$cr[idx] <- coefs[[i]] * ndf[idx,3] + ndf[idx,1]

  }

  #plot it with ggplot2
  qplot(value, cr, data=ndf) +
      facet_wrap(~variable, scale="free_x") +
      stat_smooth(method="lm", se=F) +
      ylab("Component + Residual")

}
```

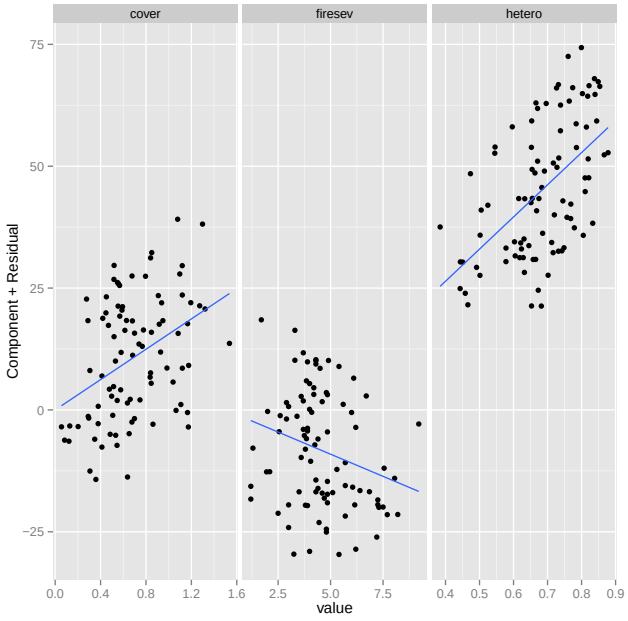
```
crPlots(klm)
```




```
crPlot.ggplot2(klm)
```



```
crPlot.ggplot2.melted(klm)
```



4 Extra Credit: Remember Bayes Theorem?

Assuming that the probability of the sun going super-nova at any given moment is 0.00001, why did the Bayesian statistician make the bet see in today's xkcd? `http://xkcd.com/1132/`