R-Code for Quant Labs (B4605/B7220)

http://www.mun.ca/biology/schneider/b4605/

Code for labs 3-9

Developed by Tony McCue, December 2008

Update by Kyle Krumsick, November 2015

Appendix 1 (Kyle Krumsick) Data entry and data definition in R

Appendix 2 (Kyle Krumsick) Probability plots in R

Appendix 3 (Kyle Krumsick) Replicating the probability plot produced by Minitab

Appendix 4 Constructing a 1-Way ANOVA table

Written by Alejandro Buren and Paul Regular, October 2012

Updated by Kyle Krumsick, October 2013

Organisation of this document

Each section is divided into two segments: (1) The R Toolbox. Contained within this box are the generic codes that you will need to complete the lab. Each item consists of a description of what the code does and a breakdown of what each of the component parts does. (2) Provided Code. The code in this section is further divided into two parts. Firstly there is the defining of the data and organising it to a format which can be analyzed. Secondly there are key bits of code the correspond to Minitab code provided in the lab handouts.

Conventions used in this document.

Lines without leading # sign are executable code

Line with leading # are either comments, or describe what follows:

#C - comments and explanations

#Out - R output,

#DataDef – Define Data

#Execution – Execute analysis

New section of lab

1. Data Entry in R Studio for Quantitative Methods of Biology

1. Environment tab

Go to the pane on the top right under the Environment tab -> "Import Dataset" -> "From Text File"
Manually select the file you want to import. A dialog box will appear, with the top right "Input file" showing the native format of your file, and the bottom right "Data frame" showing how R will display/organize the file. Most of the time the default will work, but sometimes you need to adjust the dropdown boxes on the left side so the program can properly interpret your file. Click Import
* After you click import, notice the code that appears in bottom left Console pane, this is important for #2 *

2. Import Code

You can also use manually use codes to import data. These can be typed into the Source or the Console panes and executed. Here is a common example.
read.csv(file.choose()) – Defaults to header= TRUE and sep= ","
[this needs an excel file saved as filename.csv in known location]

Here are other examples.
read.table("filename")
read.delim("filename", sep= "") – Can specify separator character
Read.table can be used generally for any file, while read.csv and read.delim are variants that have convenient defaults.

Header = FALSE implies the columns you are importing do not already have names. When the columns have names, use header = TRUE.

Sep = "" indicates to separate the numbers into different columns when a space is present in the original data. This can be substituted with any symbol, such as "," or ";", should such a symbol separate the data.

Example code:
Data <- read.csv(file.choose()) # For any common excel file
Data <- read.delim("YourTextFile", sep= "") # For any common text file

3. The Clipboard

Arguably the easiest means of data entry for small data sets, this function is your copy and paste into R. Simply select the numerical values in the ".dat" or ".txt" files provided on the course website and press "Ctrl"+"C". Then proceed to R and enter the data using the following code:

```
Data <- read.delim("clipboard", header = FALSE, sep = "")
```

The imported data will have the column labels V1, V2, etc. These will need to be changed as you progress with your analysis.

d. Manual Data Entry

Overall not suggested for data entry where you have an external data file, but can be used if necessary. One may enter the data into R in the form of a list. These lists may be combined to form a table or dataframe which we can then perform the procedures outlined in the labs. For example, we can create lists as follows:

```
X <- c(1,2,3,4)
Y <- c(2,4,8,16)
```

And combine them together in a data frame to conduct our work:

```
Data <- data.frame(X,Y)
```

Where X and Y are your columns.

2. Some generally useful codes in R

These are some general codes that may help, in addition to those found in the toolbox portion of the lab documentation.

Environment

```
help() # type any function to pull up the documentation
install.packages() # install an R package, can also use Tools -> Install Packages… on the menu bar
library(), require() # enable or activate R packages that have already been installed
rm() # remove an object
rm(list=ls(all=TRUE)) # remove all objects and data frames from the Environment
ls() # list all objects currently stored in the Environment
View() # open a separate tab to view a data frame as a spreadsheet, can also click on the data frame in the Environment tab
names() # retrieve or assign names to an object
na.rm() # remove NA values from an object
str() # return the string of an object (class type and values)
class() # return the class type of an object
```

Vectors and Matrices

```
a <- c(0,1,2) # assign (c)oncatenated objects to 'a' as a vector (in this case, of length 3)
> a
0 1 2
b <- c(3:5) # assign (c)oncatenated all objects between first and last objects specified (x:y)
> b
3 4 5
c <- c(a, b) # assign (c)oncatenated objects to 'c' as a vector of all objects
> c
0 1 2 3 4 5
cbind() # bind objects by column
> cbind(a, b)
a b
0 3
1 4
2 5
rbind() # bind objects by row
> rbind(a, b)
a 0 1 2
b 3 4 5
```

3. Lab Documentation

Lab 3 Probability Distributions

R Toolbox

Pdf in the binomial distribution	dbinom(# of success, # of trials, probability)
Pdf in the Chisquare distribution	dchisq(G-statistic, df)
Cdf in the binomial distribution	pbinom(# of success, # of trials, probability)
Cdf in the Chisquare distribution	pchisq(G-statistic, df)
Cdf in the F distribution	pf(F-statistic, numerator df, denominator df)
Generate a scatterplot	plot(x, y, ylab = "Y Axis Label", xlab = "X Axis Label", main = "Figure Title")
Cdf in the normal distribution	pnorm(Z-statistic, mu, sigma, lower.tail=FALSE)
Cdf in the t-distribution	pt(t-statistic, df, lower.tail=FALSE)
G-statistic from a p-value	qchisq(p-value, df)
T-statistic from a p-value	qt(p-value, df)

Provided Code

BINOMIAL DISTRIBUTION

```
#Lab 3 Page 3
#DataDef
C1 <- c(0:6)
#Execution
dbinom(6, 6, 0.5)
```

```
#Lab 3 Page 5
f.x <- dbinom(x,6,0.5)
cbind(x,f.x)
plot(x,f.x,
   ylab  = "f(x)",
   main = "Figure 1")
```

CUMULATIVE FREQUENCY DISTRIBUTION

```
#Lab 3 Page 7
#Execution
F.x <- pbinom(x,6,0.25)
```

THEORETICAL FREQUENCY DISTRIBUTION

```
#Lab 3 Page 10
#DataDef
y <- c(0.5,1,2,4,8)
#C    LAB 3 CONTINUED
#Execution
pchisq(3.84,df=1)
1-pchisq(3.84,1)
f.y <- dchisq(y,1)
F.y <- pchisq(y,1)
p.y <- (1-F.y)
```

```
#Lab 3 Page 11
pf(4.56, 8,23, lower.tail=FALSE)
```

```
#Lab 3 Page 12
pnorm(1.96,mean = 0,sd = 1)
```

INVERSE PROBABILITY

```
#Lab 3 Page 13
qchisq(0.9999,1)
```

Lab 4 Randomization

R Toolbox

Absolute value of a number	abs(x)
Generate a histogram	hist(x, breaks=n, xlab = "X Axis Label", main = "Figure Title")
Mean of values	mean(x)
Names of variables in data set	names(data)=c("name1", "name2")
Replicate a function	replicate(# of replications, function)
Randomly sample from a given data set	sample(data set, number sampled, replace=TRUE/FALSE)
Numerical summary of object	summary(object)
Sort a set of numbers	sort(x)

Provided Code

```
##    LAB 4 FIRST DATASET
#DataDef
data1 <- read.delim("SRBX9_5.dat")
#C    For more information on data entry into R, see Appendices
names(data1)  = c("age1",
   "age2")
```

```
#Lab 4 Page 3
#Execution
k1 <- mean(daphnia$age1)
k2 <- mean(daphnia$age2)
k3 <- k1-k2
k3
```

```
#DataDef
all_age <- c(age1,age2)
all_age
```

```
#Lab 4 Page 5
#Execution
Random <- c(replicate(100,
   (mean(sample(all_age,7,TRUE))-mean(sample(all_age,7,TRUE)))))
summary(random>abs(k3))
```

```
##    LAB 4 SECOND DATASET
#DataDef
Data <- read.delim("SRBX1311.dat")
names(data) = c("strain_b",
   "strain_B")
```

LAB 4 SECOND DATASET

```
#Lab 4 Page 6
#Execution
k1 <- mean(strain_b)
k2 <- mean(strain_B)
k3 <- k1-k2
k3
```

```
#Lab 4 Page 7
hist(random,
   breaks = 25)
```

Lab 5a Regression

R Toolbox

ANOVA table with sequential SS	`anova(model)`
Fitted values of a model	`fitted(model)`
Generate a histogram	`hist(x, breaks=n, xlab = "X Axis Label", main = "Figure Title")`
Install an R packaged, opened by library()	`install.packages(name)`
Compute a lagged version of series of numbers	`library(Hmisc) Lag(values)`
Lagged residual plot	`lag.plot(residuals, main = "Figure Title", diag = FALSE, do.lines = FALSE)`
Create a general linear model	`lm(response~explanatory, data=data1)`
Names of variables in data set	`names(data)=c("name1", "name2")`
Generate a scatterplot	`plot(x, y, ylab = "Y Axis Label", xlab = "X Axis Label", main = "Figure Title")`
Probability plot	`qqnorm(residuals, main = "Figure Title")` OR `library(e1071) probplot(residuals)`
Residuals of a model	`resid(model)`
Numerical summary of object	`summary(object)`

Provided Code

```
##    LAB 5A FIRST DATASET
#DataDef
data1 <- read.delim("SRBX14_1.dat")
names(data1)  = c("wloss",
   "humidity"
```

```
#Lab 5a Page 3
#Execution
plot(data1$humidity,data1$wloss,
   xlab  = "relative humidity (%)",
   ylab  = "weight loss (mg)",
   main  = "Figure 1")
#C Data$V specifies a variable V in dataset Data.
model1 <- lm(wloss~humidity,data = data1)
summary(model1)
```

```
##    LAB 5A FIRST DATASET
res1 <- resid(model1)
fit1 <- fitted(model1)
plot(fit1,res1,
   ylab = "residuals",
   xlab = "fitted values",
   main = "Figure 2")
```

```
#Lab 5a Page 5
lag.plot(res1,
   main = "Figure 3",
   diag = FALSE,
   do.lines = FALSE)
#C    Above approach does not offer the option to re-label axes.
#C    The following code produces a plot that allows for re-labelling the
#C    axes:
library(Hmisc)
lag.res <- Lag(res1)
plot(res1, lag.res,
   main = "Figure 3",
   ylab = "Residuals",
   xlab = "Lagged Residuals")
hist(res1,
   breaks = 9,
   xlab = "residuals",
   ylab = "frequency",
   main = "Figure 5. Tribolium weight loss")
```

```
#C Analysis #2 in Lab 2 compares the results of regression and GLM commands
#C In R the call to lm() executes any linear model (regression, GLM, etc)
#C Users of R do not need to repeat the analysis

##    LAB 5A SECOND DATASET
#DataDef
data2 <- read.delim("SRBX14_4.dat")
names(data2)  = c("arcsin_surv",
   "egg_density",
   "survival",
   "density",
   "samplesize",
   "mean(arcsin(survival))")
```

```
#Lab 5a Page 8
#Execution
model2 <- lm(survival~egg_density,data = data2)
res2 <- resid(model2)
fit2 <- fitted(model2)
```

Lab 5b Regression with Randomization

R Toolbox

ANOVA table with sequential SS	anova(model)
Combine variables into data frame	data.frame(variables)
Fitted values of a model	fitted(model)
Generate a histogram	hist(x, breaks=n, xlab = "X Axis Label", main = "Figure Title")
Install an R packaged, opened by library()	install.packages(name)
Compute a lagged version of series of numbers Lag(values)	library(Hmisc)
Lagged residual plot	lag.plot(residuals, main = "Figure Title", diag = FALSE, do.lines = FALSE)
Create a general linear model	lm(response~explanatory, data=data1)Names of
variables in data set	names(data)=c("name1", "name2")
Generate a scatterplot	plot(x, y, ylab = "Y Axis Label", xlab = "X Axis Label", main = "Figure Title")
Probability plot	qqnorm(residuals, main = "Figure Title") OR library(e1071) probplot(residuals)
Replicate a function	replicate(# of replications, function)
Residuals of a model	resid(model)
Randomly sample from a given data set	sample(data set, number sampled, replace=TRUE/FALSE)
Numerical summary of object	summary(object)

Provided Code

```
##    LAB 5B FIRST DATASET
#DataDef
data3 <- read.delim("Garrod.dat")
names(data3) = c("effort",
   "mortality",
   "year")
```

LAB 5B FIRST DATASET

```
#Lab 5b Page 5
#Execution
#C    Sample with replacement
rand3 <- c(replicate(100,
   data.frame(anova(lm(sample(data3$mortality,13,TRUE)~data3$effort)))[1,4]))
F <- data.frame(anova(model3))[1,4]
#C    The [1,4] indicates the value of the first row and fourth column,
#C    corresponding with the F-value on the ANOVA table.
summary(rand3>F)
```

Lab 6 ANOVA

R Toolbox	
ANOVA table with sequential SS	anova(model)
Fit an analysis of variance model	aov(response~explanatory)
Fitted values of a model	fitted(model)
Generate a histogram	hist(x, breaks=n, xlab = "X Axis Label", main = "Figure Title")
Install an R packaged, opened by library()	install.packages(name)
Compute a lagged version of series of numbers	library(Hmisc) Lag(values)
Lagged residual plot	lag.plot(residuals, main = "Figure Title", diag = FALSE, do.lines = FALSE)
Change explanatory variable to a factor	xvar <-factor(xvar)
Create a general linear model	lm(response~explanatory, data=data1)
Mean of values	mean(x)
Names of variables in data set	names(data)=c("name1", "name2")
One-way ANOVA	oneway.test(response~xvar)
Generate a scatterplot/boxplot	plot(x, y, ylab = "Y Axis Label", xlab = "X Axis Label", main = "Figure Title")
Probability plot	qqnorm(residuals, main = "Figure Title") OR library(e1071) probplot(residuals)
Residuals of a model	resid(model)
Standard deviation of values	sd(x)
Numerical summary of object	summary(object)
Apply a function to variable broken into groups	tapply(analyzed variable, categorical variable, function, e.g.mean)

Provided Code

```
##    LAB 6 FIRST DATASET
#DataDef
data1 <- read.delim("SRBX9_5.dat")
names(data1) = c("age1", "age2")
age <- with(data1,c(age1,age2))
##    LAB 6 FIRST DATASET
group <- c(1,1,1,1,1,1,1,2,2,2,2,2,2,2)
group <-factor(group)

#Execution  #Lab 6 Page 1
a1 <-  oneway.test(age~group)
a1
#Lab 6 Page 4
a2 <- aov(age~group)
summary(a2)
model2 <- lm(age~group)
anova(model2)
res2 <- resid(model2)
fit2 <- fitted(model2)

##    LAB 6 SECOND DATASET
#DataDef
data <-  read.table("SRTAB8_1.dat")
data <- stack(data)
data2 <- data.frame(cbind(data[,1], c(rep(1:7,each = 5))))
names(data2) <- c("wlength","group")
```

```
#Lab 6 Page 5
#Execution
model3 <-  lm(wlength~group, data = data2)
res3 <- resid(model3)
fit3 <- fitted(model3)
data.frame(data2$wlength,fit3,res3)
```

```
#Lab 6 Page 6
plot(fit3,res3,
   xlab = "Fits",
   ylab = "Residuals",
   main = "Figure 5")
lag.plot(res3,
   diag = FALSE,
   do.lines = FALSE
   main = "Figure 6")
hist(res3,
   breaks = 13
   main = "Figure 7")
qqnorm(res3,
   main = "Figure 8")
```

```
##    LAB 6  THIRD DATASET
#DataDef
data3 <- data.frame(read.table("FishMov.dat"))
names(data3) <- c("period","dist")
data3$period <- factor(data3$period)
summary(data3)
#C    You can create custom groupings of your own such as:
data3$period[data3$period %in% c(1,2,3)] <-  c("dawn")
data3$period[data3$period %in% c(4,5,6)] <-  c("morning")
#C    etc.  “afternoon”   “evening”
```

Lab 7 General Linear Mode – Multifactor ANOVA

R Toolbox

ANOVA table with adjusted SS	library(car) Anova(model1, type = "III")
Combine two sets of numbers as columns in a data.frame	cbind(column1, column2)
Fitted values of a model	fitted(model)
Install an R packaged, opened by library()	install.packages(name)
Compute a lagged version of series of numbers	library(Hmisc) Lag(values)
Lagged residual plot	lag.plot(residuals, main = "Figure Title", diag = FALSE, do.lines = FALSE)
Create a general linear model	lm(response~explanatory, data=data1)
Mean of values	mean(x)
Names of variables in data set	names(data)=c("name1", "name2")
Summarize fitted model parameter means (best for categorical variables, for regression use summary())	model.tables(aov(response~explan., data = data1), "means")
Generate a scatterplot	plot(x, y, ylab = "Y Axis Label", xlab = "X Axis Label", main = "Figure Title")
Probability plot	qqnorm(residuals, main = "Figure Title") OR library(e1071) probplot(residuals)
Repeat values	rep(numbers to repeat, # of repeats)
Residuals of a model	resid(model)
Standard deviation of values	sd(x)
Stack objects into one column	stack(data, select=c(items))
Numerical summary of object	summary(object)
Apply a function to variable broken into groups	tapply(analyzed variable, categorical variable, function, e.g.sd)

Provided Code

```
##    LAB 7 FIRST DATASET
#DataDef
a1data <- read.delim("SRBX1311.dat")

##    LAB 7 FIRST DATASET
names(a1data) <- c("b",
   "B",
   "year")
data1 <-  cbind(stack(a1data,c("b","B")),
   rep(a1data$year,2))
names(data1) <- c("ls",
   "strain",
   "year")
```

```
#Lab 7 Page 3
#Execution
model1 <- lm(ls~strain+factor(year), data = data1)
res1 <- resid(model1)
fit1 <- fitted(model1)
model.tables(aov(ls~strain+factor(year), data = data1), "means")
```

```
##    LAB 7 SECOND DATASET
#DataDef
a2data <- read.table("SRBX11_7.dat")
names(a2data) <-  c(("minutes","I",
   "II",
   "III",
   "IV")
a2data$II <- as.numeric(as.character(a2data$II))
#C    As the first value of II is a *, this column is initially coded as
#C    characters. We need to code this column as numeric, thus creating an NA
#C    for the *. YOU WILL GET AN ERROR MESSAGE indicating the introduction of
#C    NAs.
data2 <- cbind(stack(a2data,c("I","II","III","IV")),
   rep(a2data$minutes,4))
names(data2) <- c("la",
   "clutch",
   "stage")
```

Lab 8 General Linear Model – ANCOVA

R Toolbox

ANOVA table with adjusted SS	library(car) Anova(model1, type = "III")
Combine two sets of numbers as columns in a data.frame	cbind(column1, column2)
Fitted values of a model	fitted(model)
Install an R packaged, opened by library()	install.packages(name)
Compute a lagged version of series of numbers	library(Hmisc) Lag(values)
Lagged residual plot	lag.plot(residuals, main = "Figure Title", diag = FALSE, do.lines = FALSE)
Create a general linear model	lm(response~explanatory, data=data1)
Log of values	ln(x)
Mean of values	mean(x)
Names of variables in data set	names(data)=c("name1", "name2")
Summarize fitted model parameter means (best for categorical variables, for regression use summary())	model.tables(aov(response~explan., data = data1), "means")
Generate a scatterplot/boxplot	plot(x, y, ylab = "Y Axis Label", xlab = "X Axis Label", main = "Figure Title")
Probability plot OR	qqnorm(residuals, main = "Figure Title") library(e1071) probplot(residuals)
Generate a 3D scatterplot	library(scatterplot3d) scatterplot3d(x,y,z, box=FALSE, pch=16, type="h", x.ticklabs="ticklabels", ylab = "Y Axis Label", xlab = "X Axis Label", zlab = "Z Axis Label", main = "Figure Title")
Residuals of a model	resid(model)
Standard deviation of values	sd(x)
Square root of values	sqrt(x)

```
Apply a function to variable broken into groups tapply(analyzed variable,
                                                       categorical variable,
                                                       function, e.g.sd)
```

Provided Code

```
## LAB 8 DATASET
#DataDef
data3 <- read.table("SREX1412.dat")
names(data3) <- c("weight",
   "age",
   "diet")
#C    LAB 8 categorical age
cat.age <- c(1,2,3,4,4,1,2,3,4,4,1,2,2,3,4,1,2,2,3,4)
data3 <- cbind(data3,cat.age)
```

Lab 9 Problem Solving with General Linear Model

R Toolbox	
ANOVA table with adjusted SS	library(car) Anova(model1, type = "III")
Combine two sets of numbers as columns in a data.frame	cbind(column1, column2)
Combine variables into data frame	data.frame(variables)
Fitted values of a model	fitted(model)
Install an R packaged, opened by library()	install.packages(name)
Compute a lagged version of series of numbers	library(Hmisc) Lag(values)
Lagged residual plot	lag.plot(residuals, main = "Figure Title", diag = FALSE, do.lines = FALSE)
Create a general linear model	lm(response~explanatory, data=data1)
Summarize fitted model parameter means (best for categorical variables, for regression use summary())	model.tables(aov(response~explan., data = data1), "means")
Names of variables in data set	names(data)=c("name1", "name2")
Generate a scatterplot/boxplot	plot(x, y, ylab = "Y Axis Label", xlab = "X Axis Label", main = "Figure Title")
Probability plot	qqnorm(residuals, main = "Figure Title") OR library(e1071) probplot(residuals)
Stacks 2 datasets, matching the columns	rbind(data1, data2)
Repeat values	rep(numbers to repeat, # of repeats)
Residuals of a model	resid(model)
Generate a 3D scatterplot	library(scatterplot3d) scatterplot3d(x,y,z, box=FALSE, pch=16, type="h", x.ticklabs="ticklabels", ylab = "Y Axis Label", xlab = "X Axis Label", zlab = "Z Axis Label", main = "Figure Title")

Stack objects into one column	stack(data, select=c(items))
Numerical summary of object	summary(object)
Apply a function to variable broken into groups	tapply(analyzed variable, categorical variable, function, e.g.sd)

Provided Code

```
##    LAB 9 FIRST DATASET
## LAB 9 FIRST DATASET
#DataDef
install.packages("car")
data1 <- read.delim("Wworm1.dat")
names(data1) = c("T","N",
      "T.1","N.1",
      "T.2","N.2",
      "T.3","N.3",
      "T.4","N.4")
wt1 <- stack(data1,
select = c(T,T.1,T.2,T.3,T.4))
wn1 <- stack(data1,
select = c(N,N.1,N.2,N.3,N.4))
data1 <- data.frame(cbind(wn1[,1],
      wt1[,1],
      c(rep(1:5,each = 5)),
      c(rep(1:5,5))))
names(data1) <- c("n",
      "trt",
      "col",
      "row")
```

```
#Lab 9 Page 3
#Execution
plot(data1$trt,data1$n,
      ylab = "count",
      xlab = "trtment")
tapply(data1$n, data1$row, summary)
tapply(data1$n, data1$row, sd)
tapply(data1$n, data1$col, summary)
tapply(data1$n, data1$col, sd)
model1 <- lm(n~factor(trt)+factor(row)+factor(col), data = data1)
library(car)
Anova(model1,type = "III")
model1 <- lm(n~factor(row)+factor(col), data = data1)
Anova(model1,type = "III")
```

```
## Lab 9 SECOND DATASET
#DataDef
data2 <- read.delim("Wworm2.dat",nrows = 5)
names(data2) = c("T","N",
      "T.1","N.1",
      "T.2","N.2",
      "T.3","N.3",
      "T.4","N.4")
## Lab 9 SECOND DATASET
```

```
wt2 <- stack(data2,
select = c(T,T.1,T.2,T.3,T.4))
wn2 <- stack(data2,
      select = c(N,N.1,N.2,N.3,N.4))
data2 <- data.frame(cbind(wn2[,1],
      wt2[,1],
      c(rep(1:5,each = 5)),
      c(rep(1:5,5)),
      c(rep(2,25))))
names(data2) <- c("n",
      "trt",
      "col",
      "row",
      "year")
data1 <- data.frame(cbind(data1,
      c(rep(1,25))))
names(data1) <- c("n",
      "trt",
      "col",
      "row",
      "year")
data2 <- data.frame(rbind(data1,data2))
```

```
#Lab 9 Page 4
#Execution
plot(data2$trt,data2$n,
      ylab = "count",
      xlab = "trtment")
```

```
#Lab 9 Page 5
tapply(data2$n, data2$row, summary)
tapply(data2$n, data2$row, sd)
tapply(data2$n, data2$col, summary)
tapply(data2$n, data2$col, sd)
```

```
## LAB 9 THIRD DATASET
#DataDef
data3 <- read.delim("Leprosy.dat",nrows = 10)
names(data3) = c(
      "B","A",
      "B.1","A.1",
      "B.2","A.2")
b <- data.frame(stack(data3,
      select = c(B,B.1,B.2)))
a <- data.frame(stack(data3,
      select = c(A,A.1,A.2)))
data3 <- data.frame(cbind(a[,1],
      b[,1]))
data3$trt<-rep(c("TrI","TrII","Ctrl"),
      each = 10))
names(data3) <- c("a",
      "b",
      "trt")


## LAB 9 THIRD DATASET
```

```
#Lab 9 Page 6
#Execution
summary(data3)
plot(data3$A~data3$B,
      xlab=c("bTRI"),
      ylab=c("aTRI"))
```

```
#Lab 9 Page 7
plot(data3$A.1~data3$B.1,
      xlab=c("bTRII"),
      ylab=c("aTRII"))
plot(data3$A.2~data3$B.2,
      xlab=c("bCONT"),
      ylab=c("aCONT"))
```

Lab 10: Logistic Regression

R Toolbox	
Combine variables into data frame	data.frame(variables)
Create a general linear model	glm(response~explanatory, family= error dist'n (link=model link), weights=workingweights, data=data1)
Generate a scatterplot/boxplot	plot(x, y, ylab = "Y Axis Label", xlab = "X Axis Label", main = "Figure Title")
Probability plot	qqnorm(residuals, main = "Figure Title") OR library(e1071) probplot(residuals)
Residuals of a model	resid(model, "type")
Numerical summary of object	summary(object)

Provided Code

```
#DataDef
BP<-c(111.5, 121.5, 131.5,141.5, 151.5, 161.5, 176.5, 191.5)
N<-c(156, 252, 284, 271, 139, 85, 99, 43)
Nhd<-c(3, 17, 12, 16, 12, 8, 16, 8)
data<-data.frame(BP, N, Nhd)
```

```
#Lab 10 Page 3
#Execution
HdModel <- glm(Nhd/N ~ bP,
      family = binomial(link = logit),
      weights = N,
      data = Cornfield)
summary(HdModel)
plot(HdModel)
resid(HdModel, "deviance")
```

APPENDICES

Probability plots: qqnorm(), probplot(), and qqline()

Written by Kyle Krumsick, October 2013

The premise of both of the functions qqnorm() and probplot() is to compare the cumulative distribution of the residuals of a presented model to a normal cumulative distribution. The purpose of such plots is to test the assumption of normality of residuals for a GLM.
The probplot() function produces a plot similar to the minitab output observed during class. The x-axis are your raw residuals. The y-axis is produced from normal scores for each observation in the data set resulting from fitting a cumulative frequency distribution to the observed residuals.
The qqplot() function compares the standardized residuals to the quantiles from normal distribution. Residuals are standardized by dividing the individual residuals by the standard deviation of the residuals. These standardized residuals are compared to normal scores of distribution around $\mu = 0$ and $\sigma = 1$.
The minitab probability plots additionally produce a line representing a straight-line of the sigmoid-shaped normal cumulative distribution. Minitab calculates this line by fitting a normal distribution to the presented residuals (diagramed in the following section of the appendix).
The lines produced by default in R may be misleading, particularly when using small data sets. The probplot() function by default produces a straight line and the qqplot() will produce the line upon the addition of the function qqline(). While Minitab generates this line based on the mean and standard deviation of a normal distribution fitted to the residuals, R draws a line between the first and third quantiles for the sample and theoretical quantiles. With small data sets these quantiles may provide an inaccurate representation of the entire data set (e.g. *Tribolium* weights in lab 5a) and adjusting the two quantiles upon which this line is based is advisable.
Students wishing to add this theoretical normal line to their qqnorm() plots may do so utilizing the following function:

```
qqline(x, probs = c(a,b))
```

where x is the residuals of the generated model. The a and b represent the quantiles (between 0 and 1) which R will draw the line between. By default the first and third quantiles are utilized, represented as 0.25 and 0.75 for a and b.

Replicating the Minitab Probability Plot in R

Written by Kyle Krumsick, September 2013

```
#C    Two Packages are required for this code to work
install.packages("ggplot2")
install.packages("MASS")
library(MASS)
library(ggplot2)

#C    Now, let's replicate the Minitab probability plot. The input res1
#C    represents the residuals of constructed model.
df <- data.frame(res1 = sort(res1), y = qnorm(ppoints(length(res1))))
probs <- c(0.01, 0.05, seq(0.1,0.9, by = 0.1), 0.95, 0.99)
qprobs <- qnorm(probs)
p <- ggplot(data = df, aes(x = res1, y = y))+ geom_point()+
scale_y_continuous(limits = range(qprobs), breaks = qprobs, labels =
100*probs)+labs(y = "Percent", x = "Data")
fd <- fitdistr(res1,"normal")
xp_hat <- fd$estimate[1]+qprobs*fd$estimate[2]
v_xp_hat <- fd$sd[1]^2+qprobs^2*fd$sd[2]^2+2*qprobs*fd$vcov[1,2]
xpl <- xp_hat+qnorm(0.025)*sqrt(v_xp_hat)
xpu <- xp_hat+qnorm(0.975)*sqrt(v_xp_hat)
df.bound <- data.frame(xp = xp_hat, xpl = xpl, xpu = xpu, nquant = qprobs)
p + geom_line(data = df.bound, aes(x = xp, y = nquant))+
geom_line(data = df.bound, aes(x = xpl, y = nquant))+
geom_line(data = df.bound, aes(x = xpu, y = nquant))
```

How To Manually Construct a One-Way ANOVA Table

Written by Alejandro Buren and Paul Regular, October 2012
Updated by Kyle Krumsick, October 2013

```
#C    These steps are simply setting up the data
#DataDef
data <-  read.table("SRTAB8_1.dat")
data <- stack(data)
data1 <- data.frame(cbind(data[,1], c(rep(1:7,each = 5))))
names(data2) <- c("wlength","group")

#C    The ANOVA table is calculated as follows
#Execution
total.df <- length(data2$wlength)-1 #n-1
model.df <- length(unique(flies$group))-1 #ngroup-1
res.df <- total.df-model.df
total.SS <- sum((flies$wlength-mean(flies$wlength))^2) #Calculate total SS
res.SS <- sum(flies$res^2) #residual SS
model.SS <- sum((flies$fits-mean(flies$wlength))^2) #model SS
res.MS <- res.SS/res.df #residual MS
model.MS <- model.SS/model.df #model MS
F.val <- model.MS/res.MS #F value
p.val <-  1-pf(F.val, model.df, res.df) # p-value from F distribution.
ANOVA.tab <- data.frame(DF = c(model.df, res.df), SS = c(model.SS, res.SS),
MS = c(model.MS, res.MS), F = c(round(F.val,5),””),P = c(round(p.val,5),””))
row.names(ANOVA.tab) <- c(“Model”,”Residuals”)
ANOVA.tab
#C    compared to
anova(lm(wlength~group, data = data2)
#C    They are the same!
```