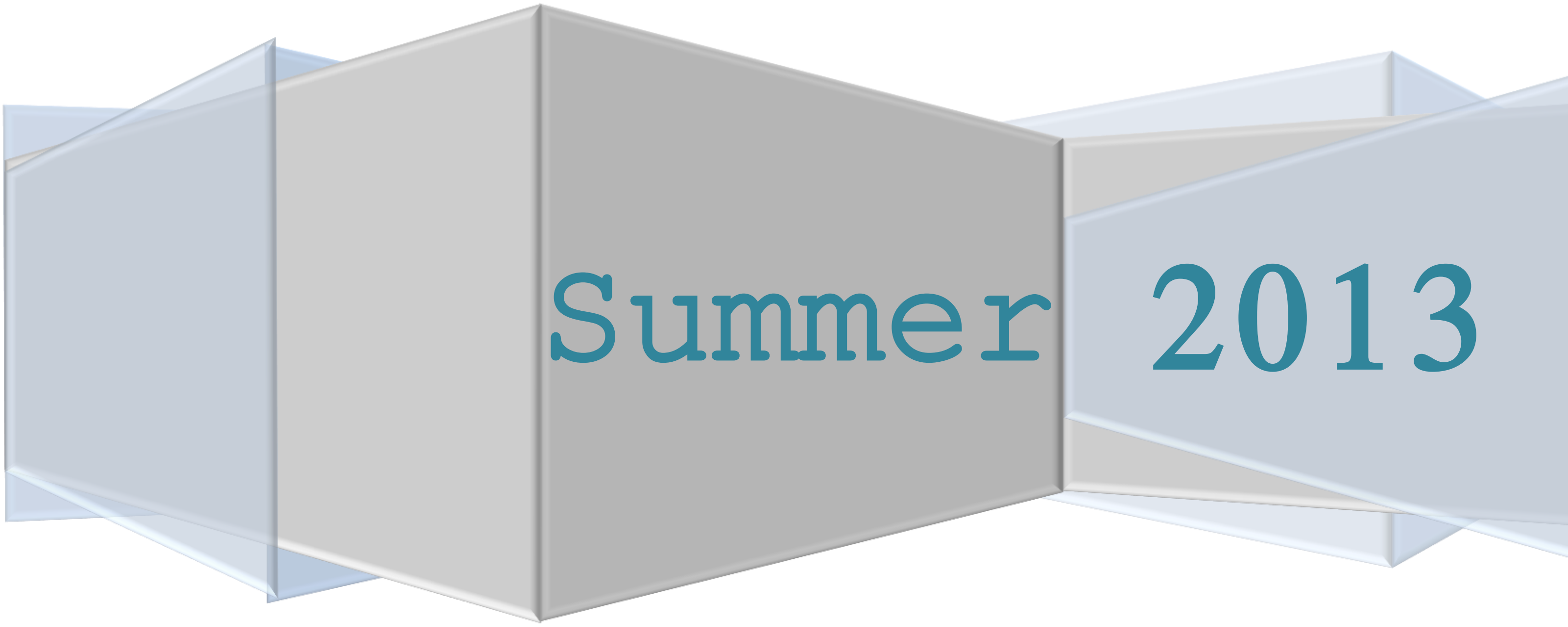
CCR **NAME: _______________________________**

R – Reference Sheet

Eric Pitman Annual Summer Workshop in Computational Science

Author: C. Ryan Mraz



Summer 2013

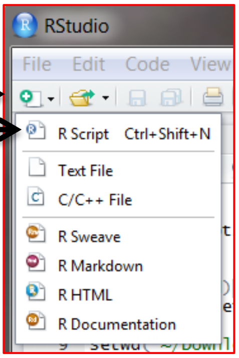
R – Reference Sheet

RStudio Tips

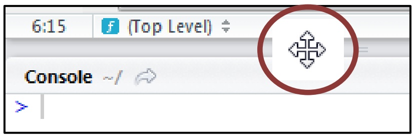
- There is no editor window until you open up a file! To do so, click:

Here:

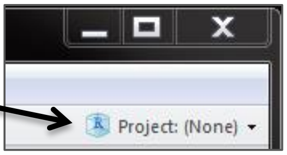
Then Here:



- To see your history (commands you have already issued), click the history pane or simply click the up arrow on your keyboard while on the command line
- To change the relative sizes of each window, hover the mouse over the window border until appears.



- Is your project loaded? Check the upper right Corner:



- There are two ways to load csv files in Rstudio:

1) In the RStudio Workspace:

   Select Import Dataset: From Text File

   Select a .csv file to Open

   Use Heading=Yes

2) From the command line:

   Set the Working Directory

   Load command:

```
> drop=read.csv(“drop.csv”)
```

*<u>Keep Your Projects Tidy!!</u>

- To clear the Console window, use: ctrl + L
- To clear individual items in the Workspace, use: r m(variable_name)
- To clear all items in the Workspace or plotspace, use: 🧹

R – Reference Sheet

Common functions

length()	# How many elements
dim()	# Retrieve the dimension of an object.
class()	# Class of the vector (=class of its elements)
str()	# Number of elements, type, and contents
sum()	# Sum of all element values
length()	# Number of elements
unique()	# Generate vector of distinct values
diff()	# Generate vector of first differences
sort()	# Sort elements, omitting NAs
order()	# Sort indices, with NAs last
rev()	# Reverse the element order
na.omit()	# Removes rows containing any "NA" values
which(x==#)	# Finds indices that satisfy a condition
table()	# Creates frequency or contingency tables for your data
levels()	# Displays the values that a categorical variable may hold

mean()	# Computes and Reports Average Value
median()	# Computes and Reports Median Value
range()	# Reports min and max:
min()	# Minimum value
max()	# Maximum value
var() and sd()	# Variance, standard deviation
summary()	# Reports Combination of measures
cor(X,Y)	# Reports Pearson correlation coefficient

R – Reference Sheet

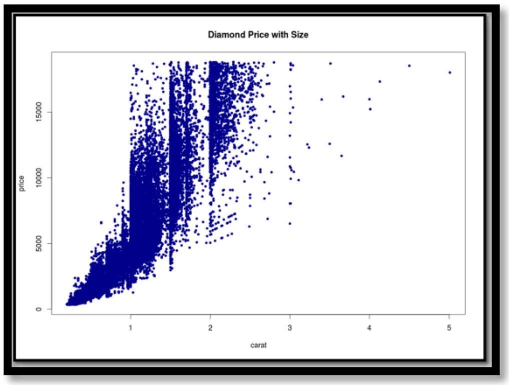
--Conditionals/Function Calls--

```
if (condition is true) {

            # do something

}


functionName =function(inputs) {

 ----------------        # do something

 return  (------)     #result

}
```

---**Common Plots**---

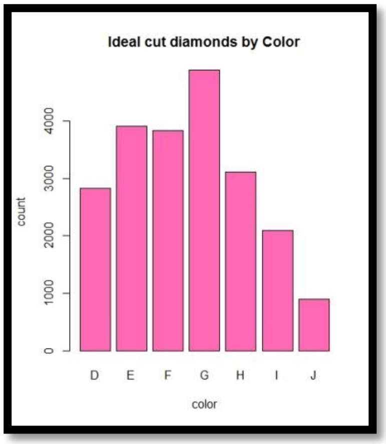
example scatterplot:

```
data(diamonds, package=”ggplot2”)

plot( formula=price~carat,

             data=diamonds,

            col=”darkblue”,

            pch=20,

             main="Diamond Price with Size"

            )
```



example barplot:

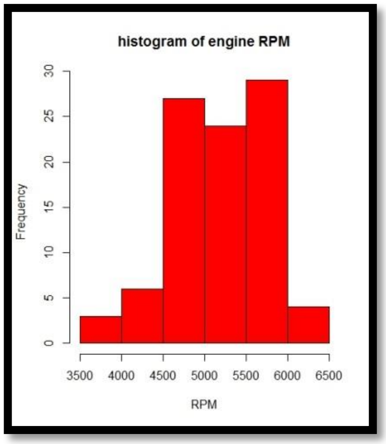
```
data(diamonds, package=”ggplot2”)

ideal=diamonds[diamonds$cut=="Ideal","color"]

barplot( table(ideal),

              xlab="color",

              ylab="count",

             main="Ideal cut diamonds by Color",

            col="hotpink" )
```



R – Reference Sheet

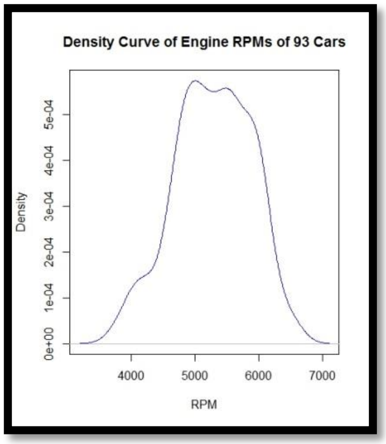
example histogram:

```
data(Cars, package=”MASS”)
hist( Cars93$RPM,
          breaks = 4,
           xlab="RPM",
           main="histogram of engine RPM",
           col="red"
          )
```



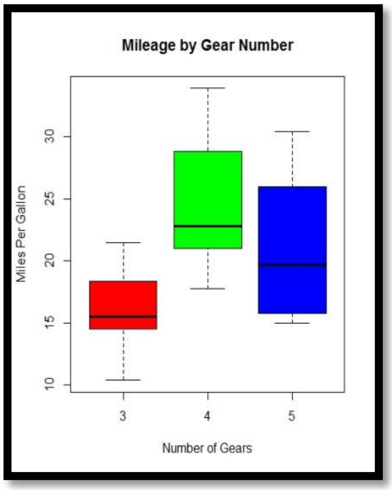
example density plot:

```
data(Cars, package=”MASS”)
plot( density(Cars93$RPM,bw=200),
          main="Density Curve of Engine RPMs of 93 Cars",
          xlab="RPM",
          col="blue"
          )
```



example boxplot:

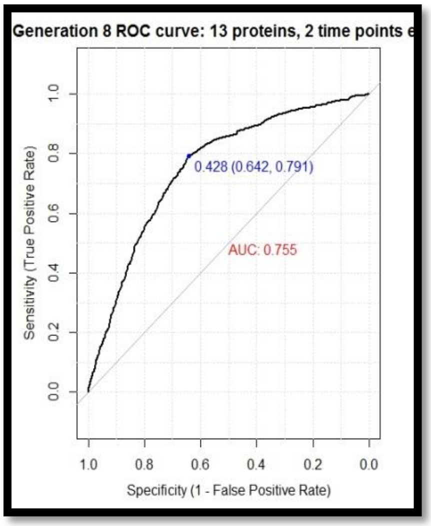
```
boxplot(formula=mpg~gear,
           data=mtcars,
            main="Mileage by Gear Number",
            xlab="Number of Gears",
           ylab="Miles Per Gallon",
          col=c("red","green","blue")
           )
```



R – Reference Sheet

example ROC Curve:

```
library(pROC)

plot.roc( roc(exp$human_crystal, exp$class3_crystal),

        ylab="Sensitivity (True Positive Rate)",

        xlab="Specificity (1 - False Positive Rate)",

        print.auc = TRUE,

        print.auc.col="red",

        main='Generation 8 ROC curve: 13 proteins, 2 time points each',

        print.thres=TRUE,

        print.thres.col="blue",

        grid=TRUE

        )
```



#--**Common Graph Modifiers**---

```
abline(lm(y~x))               # prints linear regression line on graph

pch=#                         # Chooses the type of point character to plot

cex = #                       # Magnifies text or labels on a graph/chart [smaller<(default=1)<larger]

par(mfrow=c(rows,collumns))   # Prints multiple graphs/charts on one sheet

par(mar=c(#,#,#,#))           # Changes margins' sizes starting at bottom


legend(x="location",                          # location of legend

        title = "---",

        c("Label.1","Label.2",etc.),          # separation labels

        fill = c("Color.1","Color.2",etc.)

        )
```

*N.B. There are practically endless possibilities for making graphs and plots pretty!! Play around and find out how!!

R – Reference Sheet

#---**Apply Family**---

# There are many types of the function *apply,* but for our purposes, we will only be using *sapply*.

sapply

The apply() family of functions can be used to call some other function multiple times on a dataset, with several different arguments. sapply() returns a vector or matrix result. You can use sapply() on a native R function, or on a function you wrote yourself.

EXAMPLE:

```
> u=c(33,45,37,50)        # Creating Vector u
> v=c(2,5,8,11)           # Creating Vector v
> d=data.frame(u=u,v=v)   # Creating Data frame d from Vectors u and v
>
> d                       # This is what our data frame looks like:
    u   v                 # 4 rows of 2 columns
1   33   2
2   45   5
3   37   8
4   50  11
>
>
> sapply( d, mean)        # Here, we apply the mean function to our data frame
                          # using sapply

    u     v               # sapply applies the mean function to each column of
41.25   6.50              # the data frame and outputs each answer in a user-
                          # friendly format
```