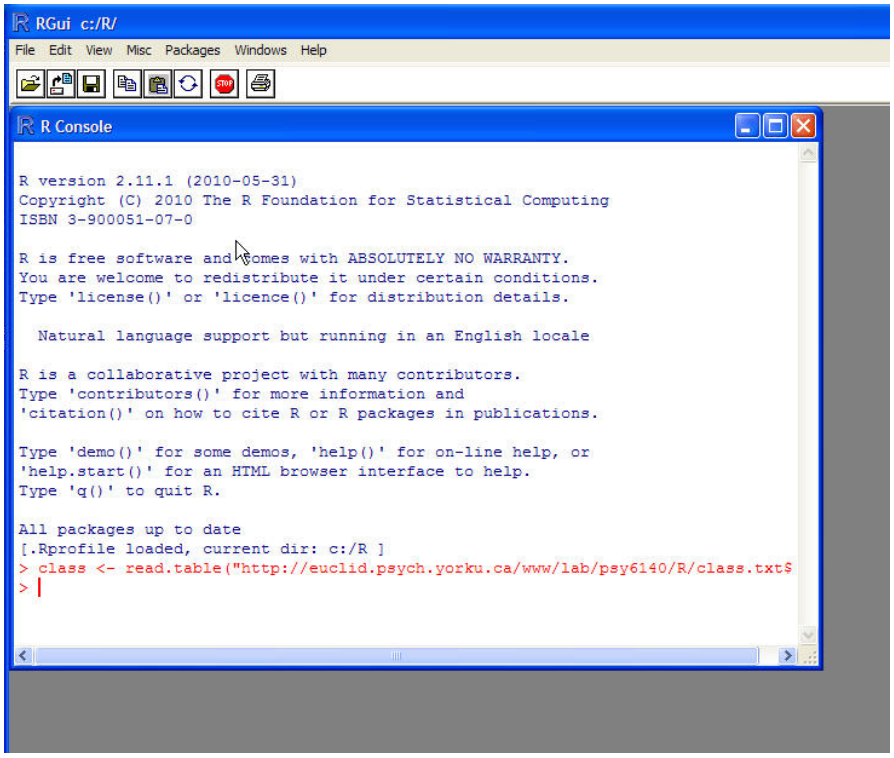
Introduction to R

This is a most brief and basic introduction to statistical computing and graphics with R. We use the basic R application ("Rgui") today. Starting next week, we will use R Studio, a much more convenient and advanced R application.

1. **Start R:** double-click on the R icon
2. **Windows:** by default, R works with multiple windows contained within the RGui window. You can work with R via copy/paste from an external editor (Notepad, Notepad++, etc.), but here we'll just work by typing commands into the R Console window, after the '>' prompt.



3. **Reading data:** Let's start by reading in some data from a text file. The file `class.txt` looks like this (first few lines). The first line specifies variable names.

```
         sex age height weight
Alfred     M  14   69.0  112.5
Alice      F  13   56.5   84.0
Barbara    F  13   65.3   98.0
Carol      F  14   62.8  102.5
   ...
```

There are quite a few R functions for reading such data, in various formats, and others for importing data from SAS, SPSS, excel, etc. Here we will use `read.table()`. Enter the lines below to read the class.txt file and create a

'data.frame' called class. *Note: in the R script below, # signals a comment. You don't have to type those lines.*

```
# read a data table from a local file (NB: always use '/' not '\')
class <- read.table("N:/psy6140/R/class.txt")
# or, read the same data from a web URL ...
class <-read.table("http://euclid.psych.yorku.ca/www/psy6140/R/class.txt")
# print the data
class
```

- **Getting help:** Documentation for any R function is obtained with `help()` or `?`. Almost all have examples at the end.

```
?read.table
```

4. **Working with data:** Try the following and see what they do.

```
# 'class' is a data.frame; look at it's structure
str(class)
# data.frames usually have row and column names
colnames(class)
rownames(class)
# print subsets
subset(class, sex=='M')
subset(class, sex=='F')
```

5. **Creating new variables:** The following lines show two different ways to calculate body mass index from height and weight and growth rate from height and age. Note how variables are referred to with the $ operator, as in `class$weight`. With the `transform()` function, calculations take place within the data.frame.

```
# add new measures BMI; NB: '$' references a column in a data.frame
class$BMI <- 703 * class$weight / (class$height)^2
class$growth <- class$height / class$age

# easier way: using transform()
class <- transform( class,
   BMI= 703 * weight / height^2,
   growth.rate = height / age
   )
```

6. **Simple summaries for any R object:** Most R objects have a summary() method

```
# get some simple summaries
```

```
summary(class)
```

7. **Plotting:** R has an extensive collection of graphical methods. Try the following to see what they do.

```
# plot method for a data.frame: scatterplot matrix
plot(class)

boxplot(weight ~ sex, data=class)

# plotting functions
plot(sin, -pi, 5*pi)

# scatterplots
plot(weight ~ height, data=class)
```

8. **Fitting and graphing linear models:** All regression and ANOVA models can be fit in R with a single function: lm(). The result is an lm object, which has special print, summary, plot and other methods.

```
# fitting linear models
mod1 <- lm(weight ~ height, data=class)
mod1
anova(mod1)
abline(mod1, col="red")
# regression diagnostics (NB: produces 4 plots by default)
plot(mod1)
```

9. **More complex models:** R has a simple comprehensive syntax for expressing linear models. In this syntax, ~ means "is modeled by," + means "and" and * expands to include main effects and interactions.

```
mod2 <- lm(weight ~ height + age + sex, data=class)
anova(mod2)

mod3 <- lm(weight ~ height*sex + age, data=class)
anova(mod3)

# comparing models
anova(mod1, mod2, mod3)
```

10. **Ending an R session:** Type q( ) to end, or simply close the R Gui window.