STAT 3340 Brief Introduction to R, Rstudio, RMarkdown

For assignments, put your name here

For assignments, include Banner: B00??????

Installing R, RStudio, and RMarkdown

This is the output of an R Markdown document. You will be using R Mardown for your assignments. More on assignments later.

A RMarkdown document is an amalgam of R computer commands, which in RMarkdown are delineated by the start line "‘{r} and the finish line "‘, plain text, and interspersed latex mathematical typesetting formulas delineated by $ signs. R Markdown is a library of routines which runs withing RStudio. RStudio is a graphical user interface to R. R is the program of choice for statistical computation and statistical graphics.

Publically available computers on campus, including those in the learning commons, have R studio installed, and some of them also have the RMarkdown library available. If you want to use your own computer to do assignments, you will need to first install R, and then install R Markdown.

To install R, proceed to https://www.r-project.org/. Select a mirror. One of the Dalhousie links should be fastest. Then select the link for your operating system, Windows, MacOSX or Linux. Then select the base package. There may be a link "install R for the first time", in which case you should follow that.

Once R is installed, you should install RStudio. Follow the link to

https://www.rstudio.com/products/rstudio/download/

Follow the link for the free desktop version. You will then find links to versions for Windows, MacOSX and Linux. Choose the appropriate version.

Once you have R and RStudio installed, start up RStudio. (Some students have told me in past that they had to run RStudio as administrator [hover over the RStudio icon, then right click "Run as Administrator"] the first time they ran the program.

To install the Rmarkdown library, click on Tools, then InstallPackages, then in the Packages window, enter rmarkdown, then the Install button.

Assuming that everything has worked so far, click on the File menu, then OpenFile, and enter the following IP address:

http://chase.mathstat.dal.ca/~bsmith/stat3340/mynotes/Rintro.Rmd

which is the address of the RMarkdown file corresponding to the pdf file that you're looking at now.

If this has worked, click the Knit HTML button, which will create and open an html file which is the processed version of this document.

The following link points to an online introduction to the R program for statistical computing. This markdown file is essentially appendix A of is introduction, translated into R Markdown format. The link is essentially redundant, as the information is essentially identical to what you will find in the lower right window of the Rstudio session, under the Help tab.

https://cran.r-project.org/doc/manuals/r-release/R-intro.pdf

Start R studio by clicking on the appropriate button or link, or run from command line (on linux or MacOS).

The above link to R-intro.pdf is essentially redundant, as the information is essentially identical to what you will find in the lower right window of the Rstudio session, under the Help tab. Spend some time exploring the help facilities linked to this tab.

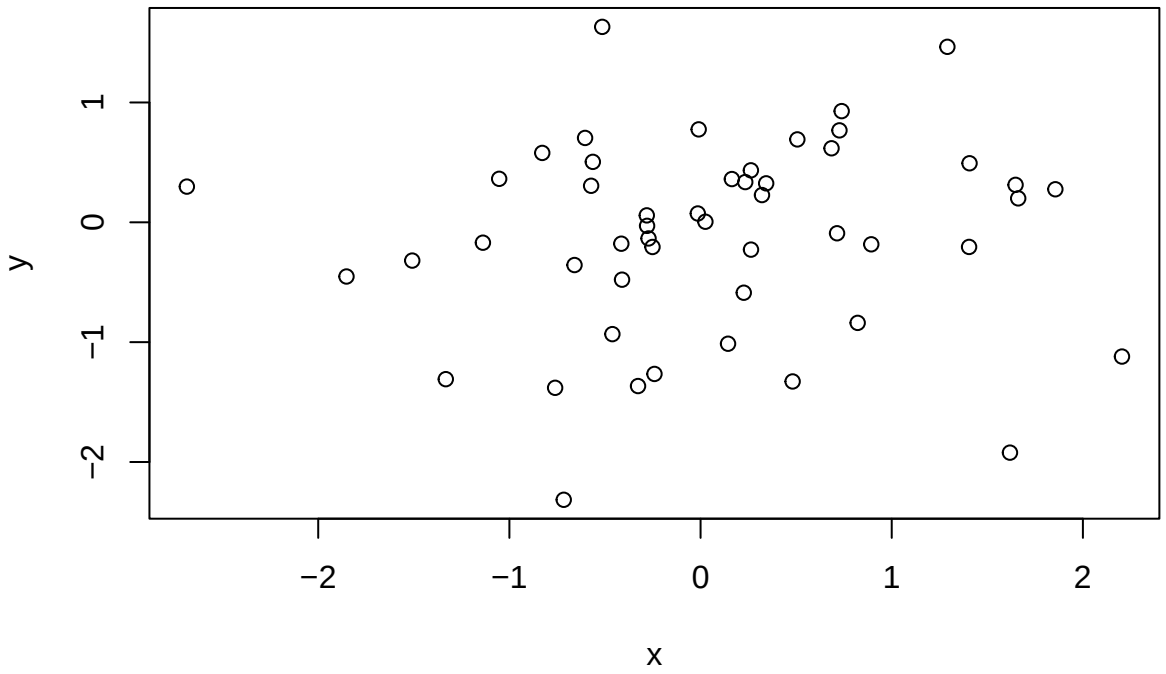
Alternatively, R can run in native model, without being invoked from Rstudio. However, it is recommended that you access R through Rstudio, at least to begin with.

An Introduction to R

The following material is essentially what you will see in "Appendix A. A Sample Session" of "An Introduction to R".

Generate, and then plot, two pseudo-random normal vectors of x- and y-coordinates.

```
x <- rnorm(50) #generates 50 from standard normal distribution
y = rnorm(50) #note the two different assignment operators = and ->
plot(x, y)
```




```
ls() #See which R objects are now in the R workspace.
```

```
## [1] "x" "y"
```

```
rm(x, y) #Remove objects no longer needed. (Clean up).
x <- 1:20 #Make x = (1, 2, . . . , 20).
```

Make a data frame of two columns, x and y, and print the first 4 rows of it.

```
dummy <- data.frame(x=x, y= x + rnorm(x))
dummy[1:4,]
```

```
##   x        y
## 1 1 1.481264
## 2 2 2.975963
## 3 3 2.571132
## 4 4 4.185930
```

R has a fairly sophisticated storage structure. The dataframe named dummy has two variables x and y. The version of x in the dataframe is identical to the version accessible locally, but y is defined only in the dataframe.

```
ls()
```

```
## [1] "dummy" "x"
```

Fit the simple linear regression $y = \beta_0 + \beta_1 x + \epsilon$ and look at the statistical analysis of the fit.

Note how you can access latex mathematical symbols and expressions by delimiting them by $ signs in the R markdown document. Latex is a mathematical typesetting system used to produce most books and research papers in mathematics, statistics, computer science, ...

Linear regression models y as a linear function of x, with intercept (β_0) and slope (β_1) parameters to be estimated using least squares.

The R sytax for this model is "y~x", which we read as "y tilde x".

This is equivalent to the "regress y 1 x" command you may have seen in minitab.

```
fm <- lm(y ~ x, data=dummy)
summary(fm)
```

```
##
## Call:
## lm(formula = y ~ x, data = dummy)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.6743 -0.4960  0.1999  0.6107  2.1665
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.01437    0.54795   0.026    0.979
## x            1.01405    0.04574  22.169 1.62e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.18 on 18 degrees of freedom
## Multiple R-squared:  0.9647, Adjusted R-squared:  0.9627
## F-statistic: 491.4 on 1 and 18 DF,  p-value: 1.617e-14
```

Make the columns in the data frame visible as variables, and make a nonparametric local regression function.

```
attach(dummy)
```

```
## The following object is masked _by_ .GlobalEnv:
##
##     x
```

```
ls() # list objects available at command line level
```

```
## [1] "dummy" "fm"    "x"
```

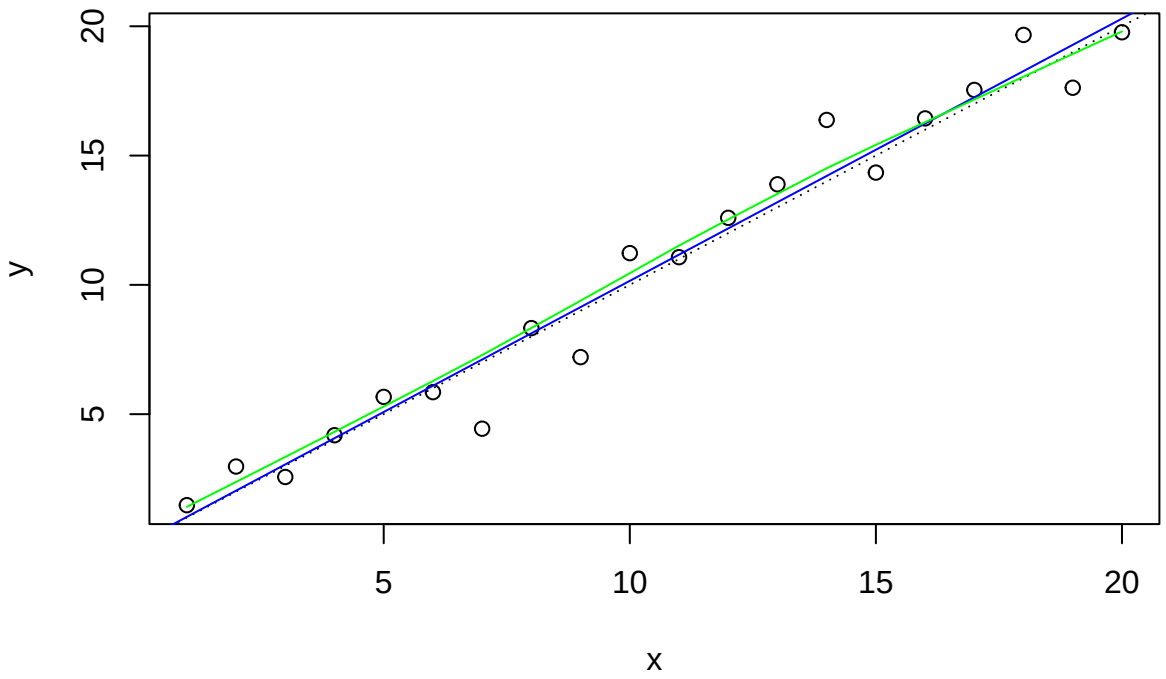
```
ls(pos=2) #list objects in position 2, which is where "dummy" is opened
```

```
## [1] "x" "y"
```

```
lrf <- lowess(x, y)
```

Plot the data points, the true regression line, the least squares regression line, and the the local regression curve.

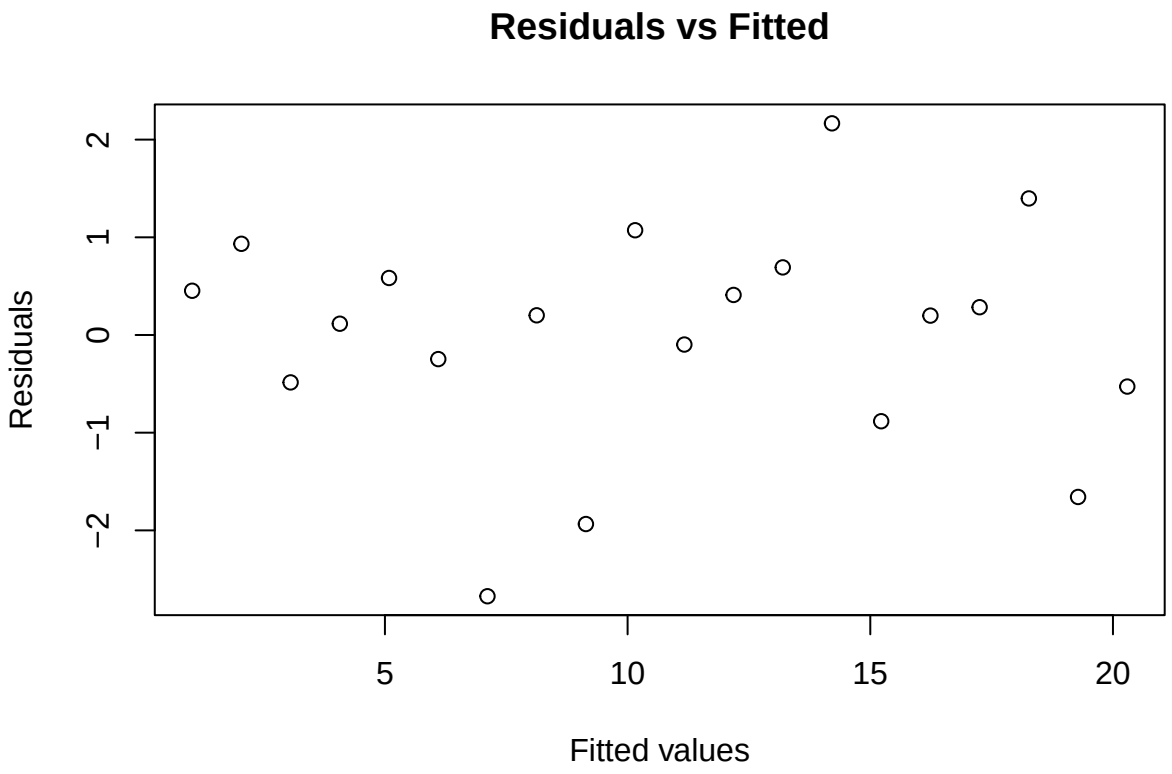
```
plot(x, y)
abline(0, 1, lty=3) #the true regression line: (intercept 0, slope 1).
abline(coef(fm), col="blue") #Unweighted regression line.
lines(x, lrf$y, col="green") #local regression curve
```




```
detach() #Remove data frame from the search path.
```

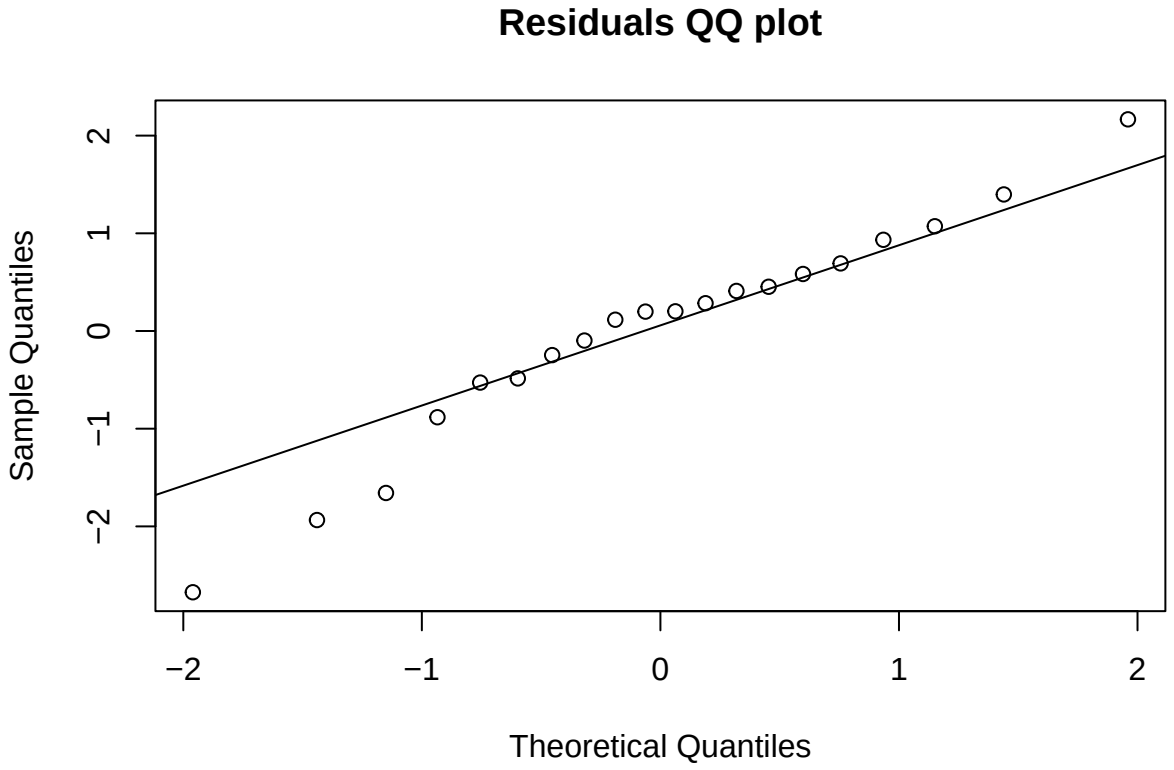
Plot residuals vs fitted values. This is a standard regression diagnostic plot to check if the variability of the residuals changes with fthe fitted values, which is a situation known as heteroscedasticity. When y was defined, the theoretical variance of the deviations ϵ was constant, so we don't expect to see any evidence of heteroscedasticity.

```
plot(fitted(fm), resid(fm), xlab="Fitted values", ylab="Residuals", main="Residuals vs Fitted")
```



Do a normal QQ plot, used plot to check for normality, skewness, kurtosis and outliers. It's not very useful here, as you can't see much in these plots with a small number of observations. The data was generated with normal errors, so this plot gives an idea of what a QQ plot might look like when the assumption of normality IS satisfied.

```
qqnorm(resid(fm), main="Residuals QQ plot")
qqline(resid(fm))
```




```
rm(fm, lrf, x, dummy) #Clean up again.
```

####The Michelson-Morely data

The next section will look at data from the classical experiment of Michelson to measure the speed of light. This dataset is available in the morley object, but we will read it to illustrate the read.table function.

```
#Get the path to the data file.
filepath <- system.file("data", "morley.tab" , package="datasets")
file.show(filepath)
```

Read in the Michelson data as a data frame, and look at it. There are five experiments (column Expt) and each has 20 runs (column Run) and sl is the recorded speed of light, suitably coded.

```
mm <- read.table(filepath)
mm [1:10,]
```

```
##     Expt Run Speed
## 001    1   1   850
## 002    1   2   740
## 003    1   3   900
## 004    1   4  1070
## 005    1   5   930
## 006    1   6   850
## 007    1   7   950
## 008    1   8   980
## 009    1   9   980
## 010    1  10   880
```

Change Expt and Run into factors, which means they are qualitative variables having different levels, but that labelling of the levels doesn't matter, only that they are different.

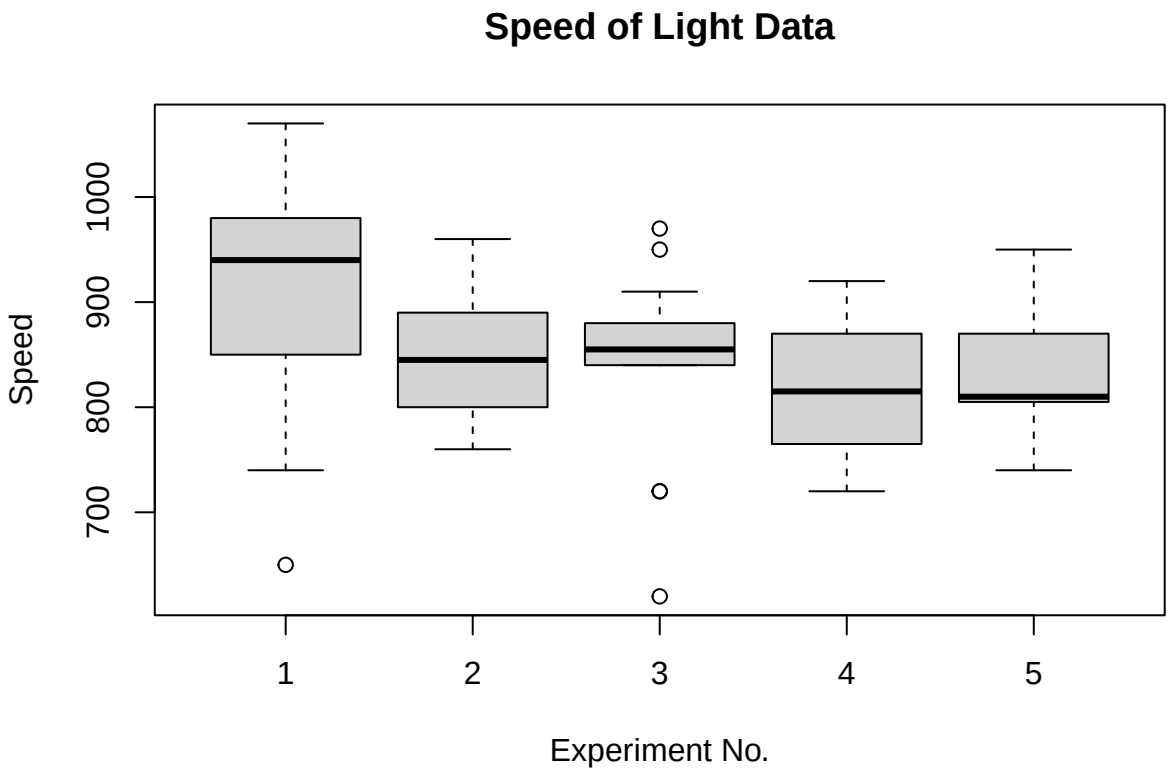
```
mm$Expt <- factor(mm$Expt)
mm$Run <- factor(mm$Run)
```

Make the data frame visible.

```
attach(mm)
```

Compare the five experiments with simple boxplots.

```
boxplot(Speed~Expt, main="Speed of Light Data", xlab="Experiment No.")
```



Clean up before moving on.

```
detach() #detach the data frame mm
ls()
```

```
## [1] "filepath" "mm"
```

```
rm(list=ls())
ls()
```

```
## character(0)
```