"R" in Action; Hands-On

A core ability to data-driven operations

Richard G. Lamb, PE, CPA
Tel: 832-710-0755; Email: rchrd.lamb@gmail.com
Website (educational): https://analytics4strategy.com/

Agenda:

- ❑ Purpose and approach.
- ❑ Installation, packages and sessions.
- ❑ Case from Discovering Statistics Using R, Field and Miles, 2012
  - ➢ Case
  - ➢ Loading super tables into R.
  - ➢ Know thy data.
  - ➢ Correlation and partial correlation.
  - ➢ Regression—standard and robust.
- ❑ Reference library.

Two core abilities to being data-driven is to know what can be asked and answered by analytics and entry-level skills in "R."

The purpose of the session is to achieve the entry-level.

Papers on data analytics with respect to the five types questions they allow to be asked and answered: https://analytics4strategy.com/new-age-five-questions

The approach is built upon the recognition that explanations of every analytic are plentiful, using the most powerful of analytic software—"R"—to explain them

- The presentation draws upon a published example of a universally relevant analysis—correlation—so that you can gain the full expertise in the analytic as you decide to use it in your work.
- Rather than a full explanation of correlation analysis, the explanation is how "R" is used to conduct any analysis.
- Because the session will cause you to run around in "R," you will become comfortable with following literature—as recipes—in conduct of the analytics associated to with the questions you know to ask and answer of your operations.

Literature as recipes, there is vast literature and papers to explain the analytic of interest and how to work them in "R"

- Notice the code to the analytic is placed in the explanation—as timely to the explanation.
- The text and code are "recipes" because we would substitute our variables into the demonstrated code.

Source: Discover Statistics Using R. Fields and Miles. Chapter Six
https://www.amazon.com/Discovering-Statistics-Using-Andy-Field/dp/1446200469/ref=sr_1_1?crid=2UQ4P8WGC6ZJH&keywords=discovering+statistics+using+r&qid=1568643787&sprefix=discovering+stat%2Caps%2C196&sr=8-1

correlation and its significance we will use the **pcor()** and **pcor.test()** functions respectively. These are part of the *ggm* package, so first load this:

```
library(ggm)
```

The general form of *pcor()* is:

```
pcor(c("var1", "var2", "control1", "control2" etc.), var(dataframe))
```

Basically, you enter a list of variables as strings (note the variable names have to be in quotes) using the *c()* function. The first two variables should be those for which you want the partial correlation; any others listed should be variables for which you'd like to 'control'. You can 'control' for the effects of a single variable, in which case the resulting coefficient is known as a *first-order partial correlation*; it is also possible to control for the effects of two (a *second-order partial correlation*), three (a *third-order partial correlation*), or more variables at the same time. The second part of the function simply asks for the name of the dataframe (in this case *examData2*). For the current example, we want the correlation between exam anxiety and exam performance (so we list these variables first) controlling for exam revision (so we list this variable afterwards). As such, we can execute the following command:

```
pcor(c("Exam", "Anxiety", "Revise"), var(examData2))
```

Executing this command will print the partial correlation to the console. However, I find it useful to create an object containing the partial correlation value so that we can use it in other commands. As such, I suggest that you execute this command to create an object called *pc*:

```
pc<-pcor(c("Exam", "Anxiety", "Revise"), var(examData2))
```

We can then see the partial correlation and the value of R^2 in the console by executing:

```
pc
pc^2
```

The general form of *pcor.test()* is:

```
pcor(pcor object, number of control variables, sample size)
```

Basically, you enter an object that you have created with *pcor()* (or you can put the *pcor()* command directly into the function). We created a partial correlation object called *pc*, had only one control variable (**Revise**) and there was a sample size of 103; therefore we can execute:

```
pcor.test(pc, 1, 103)
```

to see the significance of the partial correlation.

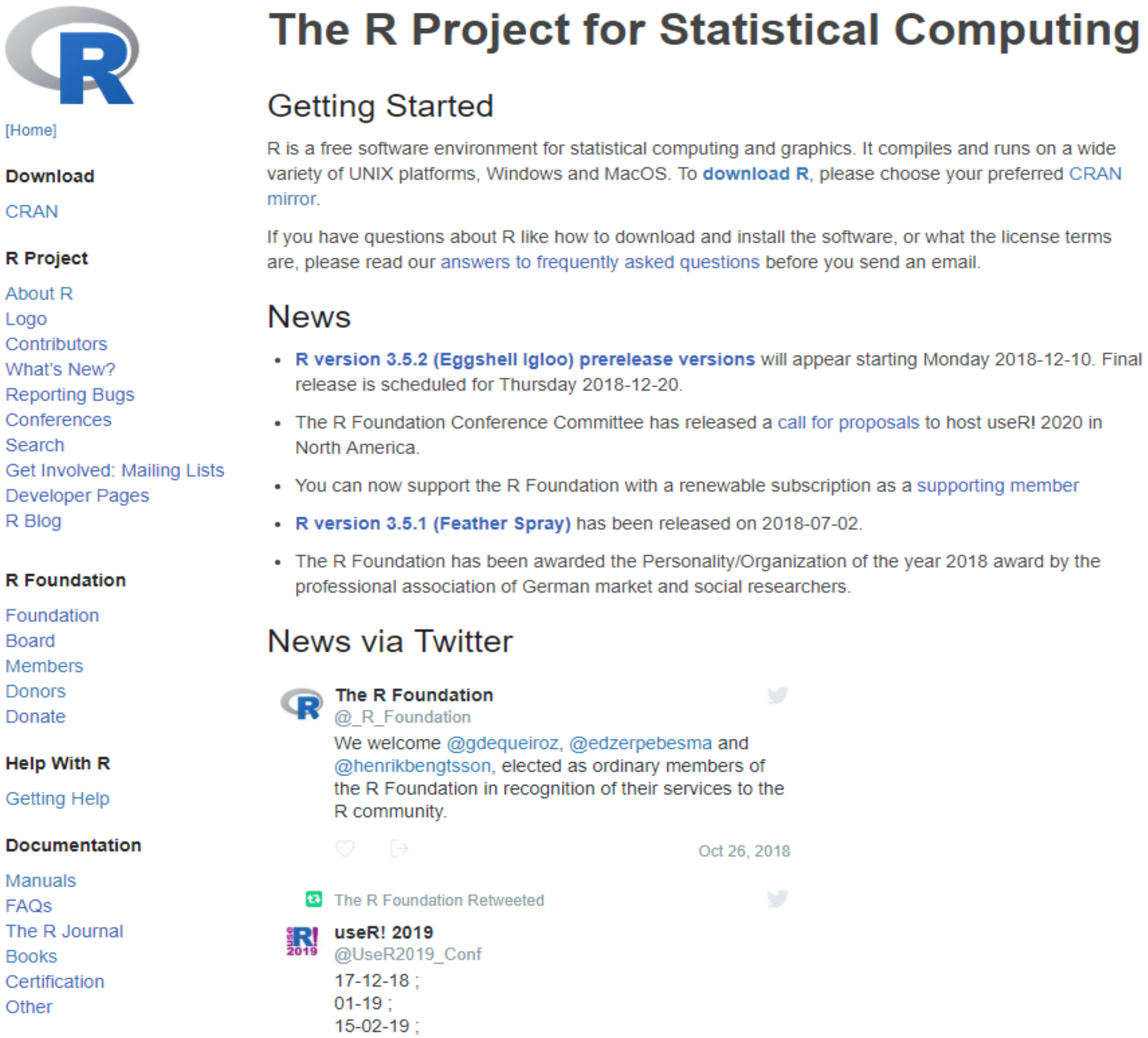
The session is accompanied with an R script and data set—acquire them as follows:

- Both are located for download at https://analytics4strategy.com/trn-rhandson
- Download both to a directory on your computer—as you would for any file.
- Script: Correlation_Cases.R.txt
  - Notice the file has an ".txt" extension so that it can be placed on the source website.
  - At download to directory remove the ".txt" extension of the file name causing the file to return to being an R script file (Correlation_Cases.R).
  - Note: Adding the extension allows us to view the script without starting a session.
- Data set: AnxietyParCorr.csv
  - You will reach for it from the "R" session.
  - Notice the code to do so in the script—based on its location in a particular computer.

Agenda:

- ❑ Purpose and approach.
- ❑ **Installation, packages and sessions.**
- ❑ Case from Discovering Statistics Using R, Field and Miles, 2012
    - ➢ Case.
    - ➢ Loading super tables into R.
    - ➢ Know thy data.
    - ➢ Correlation and partial correlation.
    - ➢ Regression—standard and robust.
- ❑ Reference library.

“R” is available to download at https://www.r-project.org/ along with instructions for download and more



[Home]

Download

CRAN

R Project

About R
Logo
Contributors
What's New?
Reporting Bugs
Conferences
Search
Get Involved: Mailing Lists
Developer Pages
R Blog

R Foundation

Foundation
Board
Members
Donors
Donate

Help With R

Getting Help

Documentation

Manuals
FAQs
The R Journal
Books
Certification
Other

The R Project for Statistical Computing

Getting Started

R is a free software environment for statistical computing and graphics. It compiles and runs on a wide variety of UNIX platforms, Windows and MacOS. To **download R**, please choose your preferred CRAN mirror.

If you have questions about R like how to download and install the software, or what the license terms are, please read our answers to frequently asked questions before you send an email.

News

- **R version 3.5.2 (Eggshell Igloo) prerelease versions** will appear starting Monday 2018-12-10. Final release is scheduled for Thursday 2018-12-20.
- The R Foundation Conference Committee has released a call for proposals to host useR! 2020 in North America.
- You can now support the R Foundation with a renewable subscription as a supporting member
- **R version 3.5.1 (Feather Spray)** has been released on 2018-07-02.
- The R Foundation has been awarded the Personality/Organization of the year 2018 award by the professional association of German market and social researchers.

News via Twitter

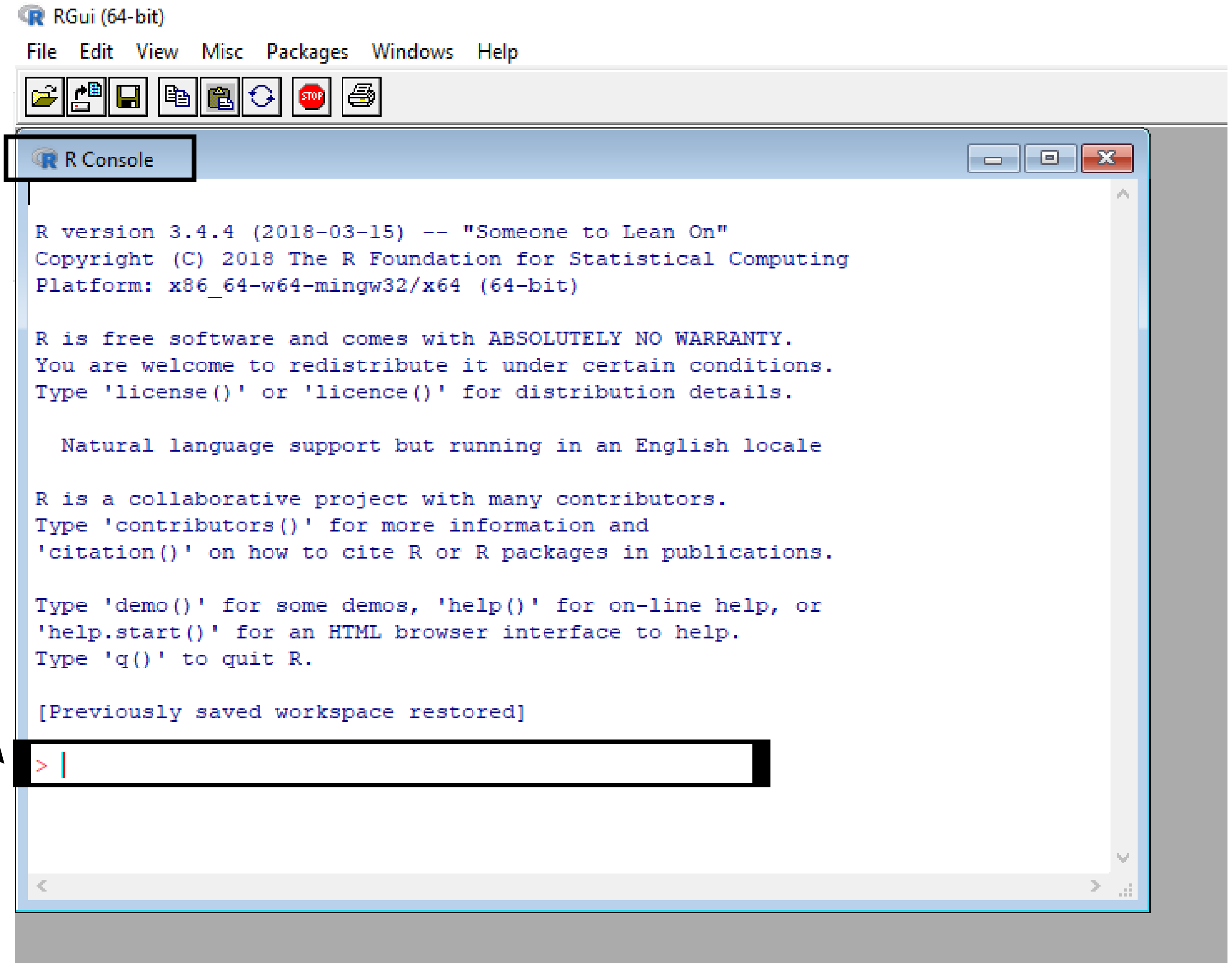
The R Foundation
@_R_Foundation
We welcome @gdequeiroz, @edzerpebesma and @henrikbengtsson, elected as ordinary members of the R Foundation in recognition of their services to the R community.
Oct 26, 2018

The R Foundation Retweeted

useR! 2019
@UseR2019_Conf
17-12-18 ;
01-19 ;
15-02-19 ;

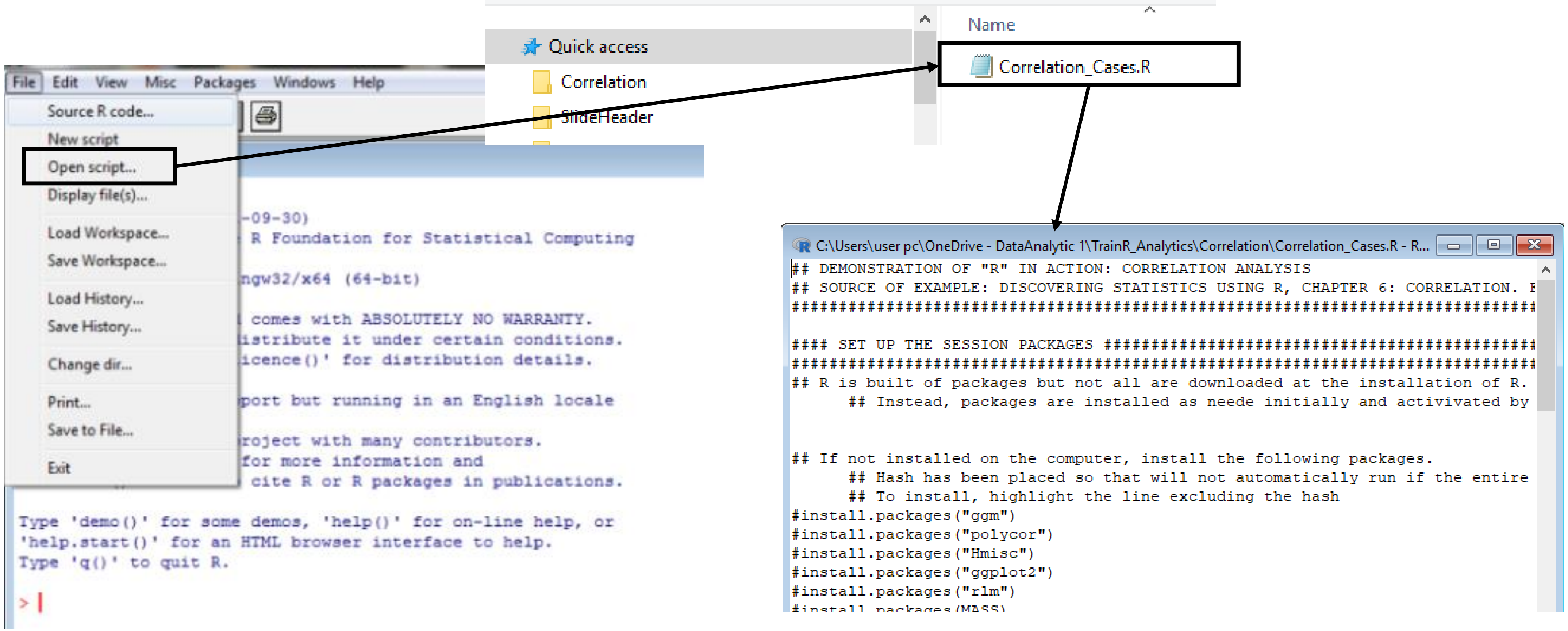
YouTube video demonstrating how to download and install R on your computer.
https://www.youtube.com/watch?v=ym8szN2Zim4

The opening view will be the “console” view in which we could work, but our code would have to be done from scratch



The nature of the Console is that entered code returns an output at each line of command.

The important work area to most of us is the script view were we place, edit code and run code; ultimately saving it as an .R file—in this case a script file is opened rather than newly created



Notice that the script serves as a document of explanation, paper trail and code to your analysis.

There are two ways to run the script

Highlight the code to be run. . .

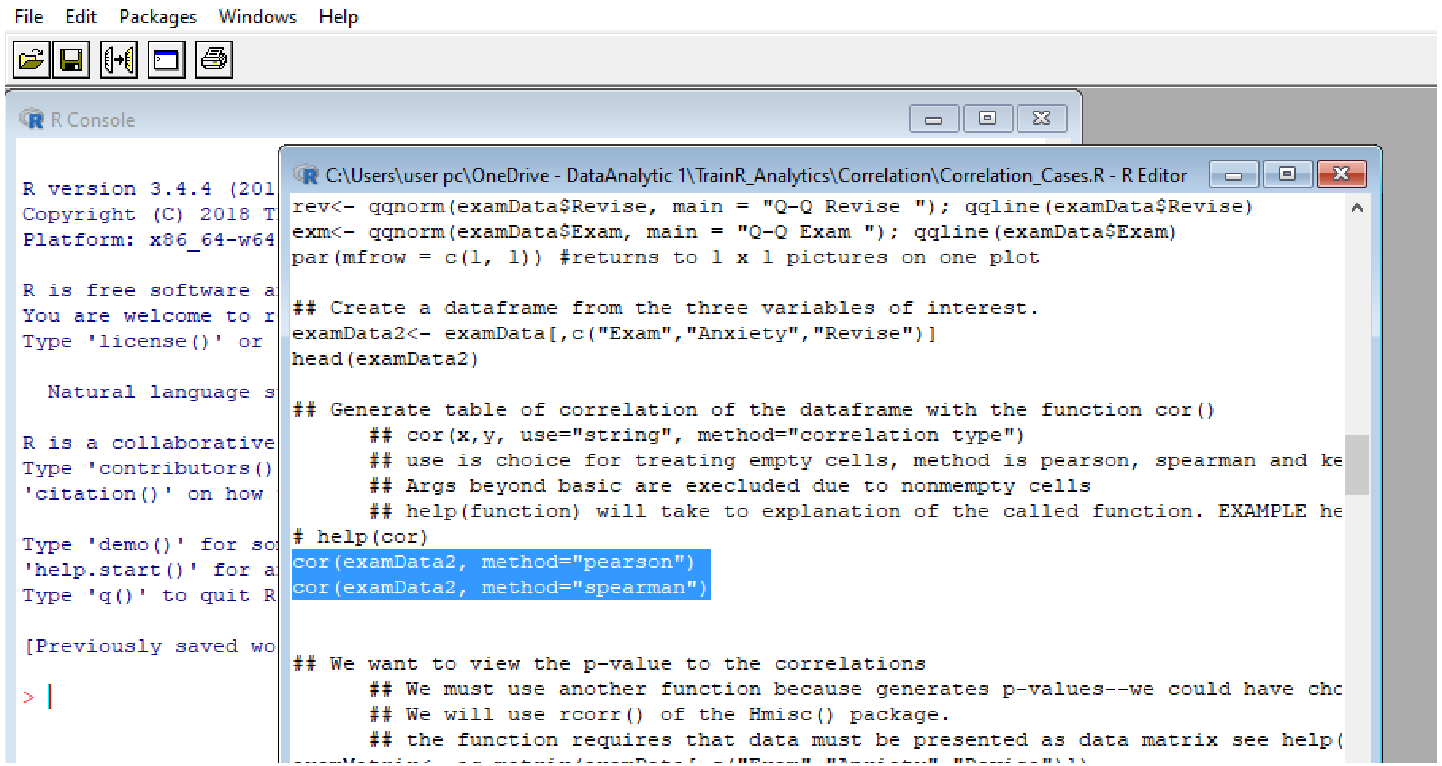
or. . .

Place cursor in line of code to be run.

Press ctrl r. . .

or. . .

Click the run icon.



After installing R, it is typically necessary to install packages

- **Almost everything in R is done as a function with arguments.**
- **The functions are contained in packages.**
- **There are 10,000 packages.**
- **The most ubiquitous are installed in the initial download, others are installed as you recognize their relevance to your current work session.**

The literature you use as your analytic recipe will tell you which packages you must have installed

Hash mark causes what follows to not be a command.

For packages if not yet installed by a previous session.

Tip:
Place hash mark in front of command to prevent it from executing automatically.

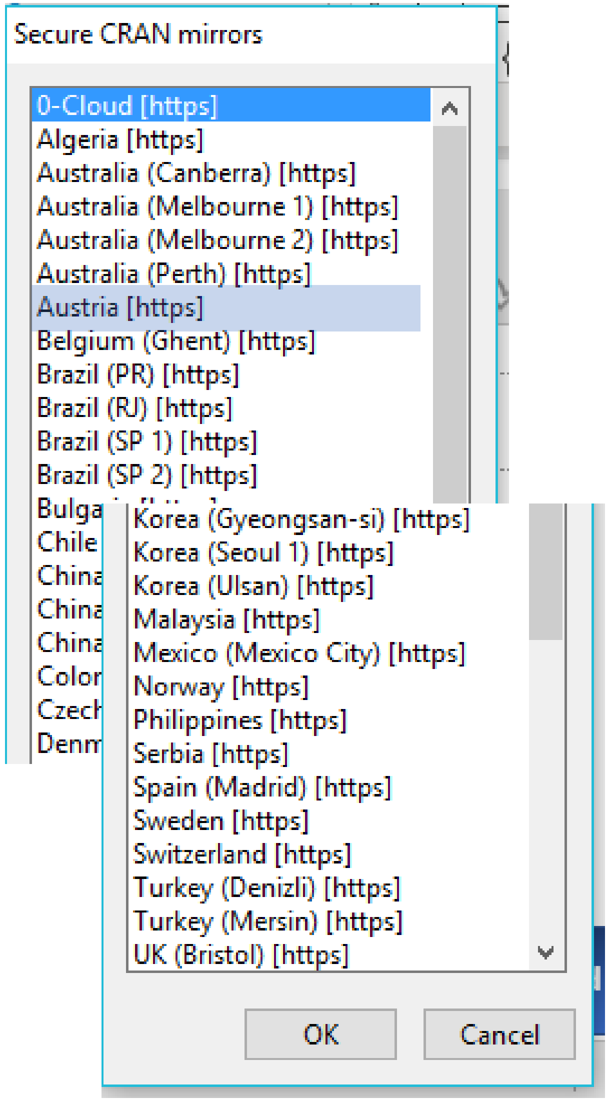
If the package is to be opened in the current session.

C:\Users\user pc\OneDrive - DataAnalytic 1\TrainR_Analytics\Correlation\Correlation_Cases.R - R Editor

```
## If not installed on the computer, install the following packages.
      ## Hash has been placed so that will not automatically run if the e
      ## To install, highlight the line excluding the hash
#install.packages("ggm")
#install.packages("polycor")
#install.packages("Hmisc")
#install.packages("ggplot2")
#install.packages("rlm")
#install.packages(MASS)

## If booting up the script for current session, execute each of the packa
      ## Hash has been placed so that will not automatically run if the e
      ## To install, highlight the line excluding the hash
#library(ggm)
#library(Hmisc)
#library(polycor)
#library(ggplot2)
#library(rlm)
#library(MASS)
```

To install a package, highlight the install.package() code, run, select a source server, click OK and the download and installation will occur

```
## If not installed on the computer, install the following packages.
                ## Hash has been placed so that will not automatically run if the entire script is run.
                ## To install, highlight the line excluding the hash
#install.packages("ggm")
#install.packages("polycor")
#install.packages("Hmisc")
#install.packages("ggplot2")
#install.packages("rlm")
#install.packages(MASS)
```



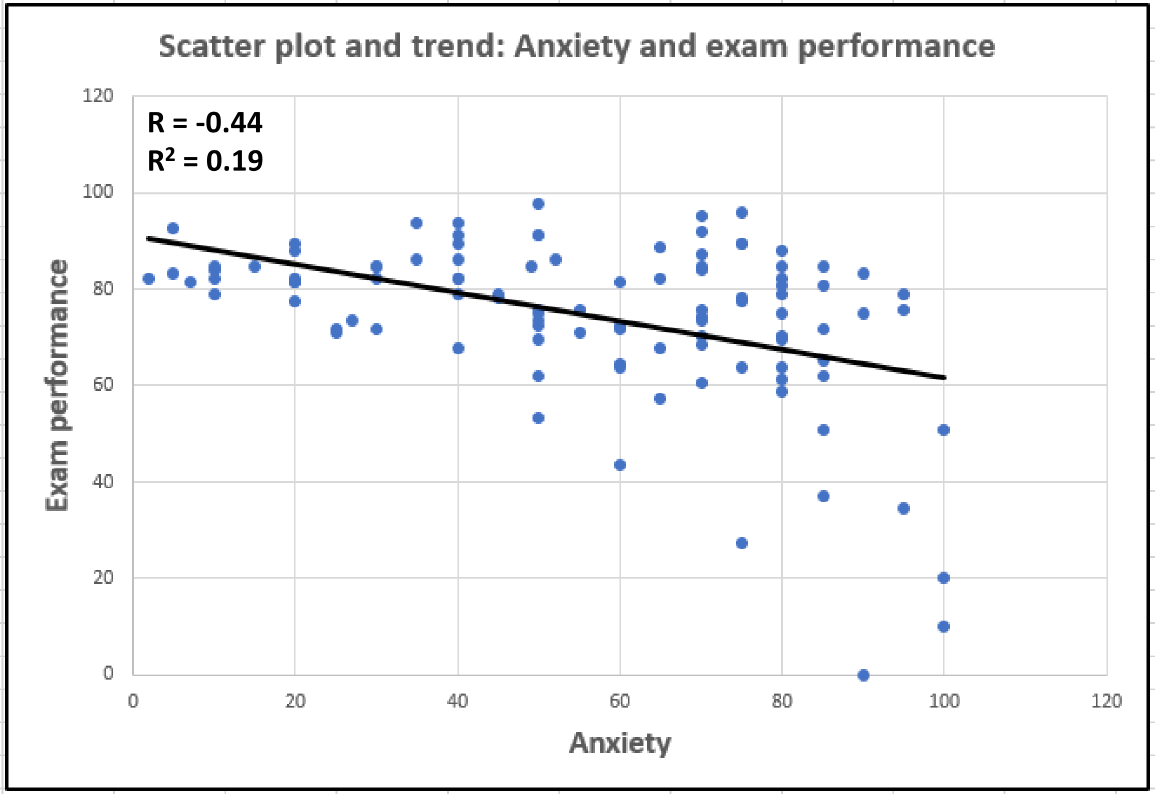
Notes:

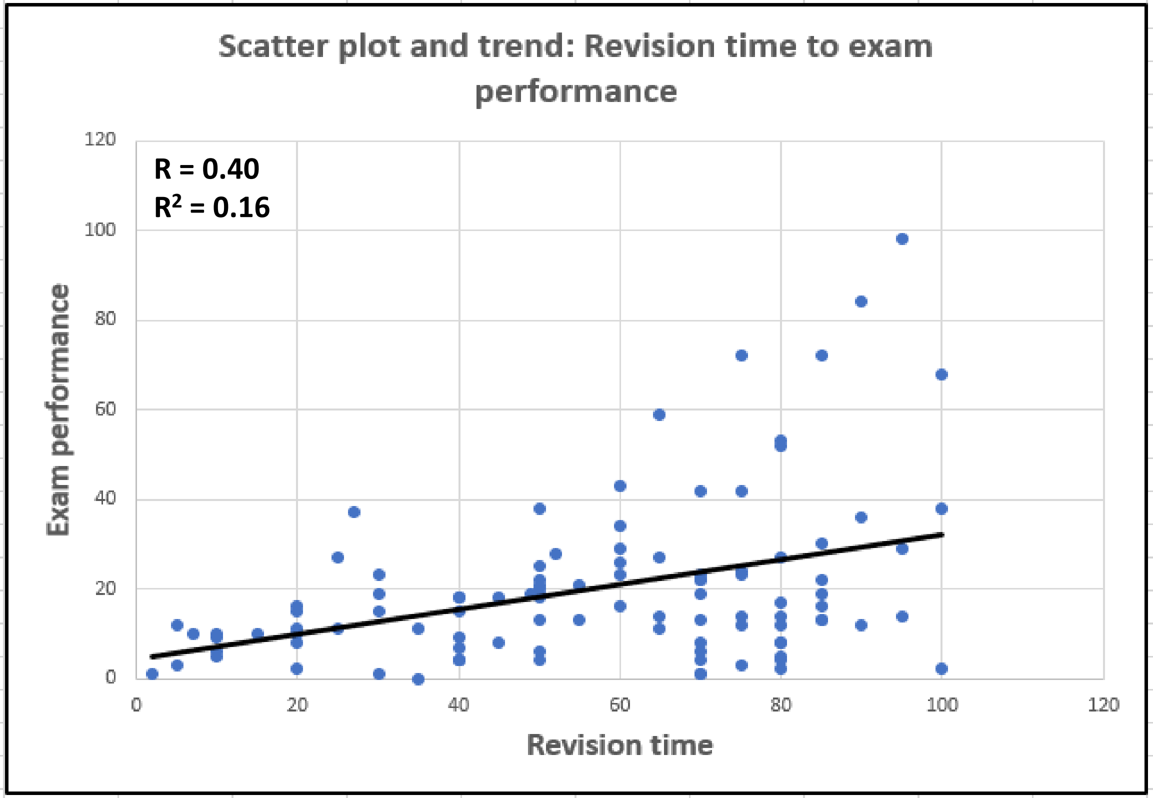
- The globally distributed servers are mirror servers; they all contain the same content.
- In our organizations, administrators typically must download the software and packages.

Agenda:

- ❑ Purpose and approach.
- ❑ Installation, packages and sessions.
- ❑ Case from Discovering Statistics Using R, Field and Miles, 2012
  - ➢ **Case**
  - ➢ Loading super tables into R.
  - ➢ Know thy data.
  - ➢ Correlation and partial correlation.
  - ➢ Regression—standard and robust.
- ❑ Reference library.

We are given two Excel scatter plot and trendline charts and asked to explore the correlation and relation between anxiety and review time to exam performance





We will demonstrate the fundamentals of the "R" software by answering the following questions

- What is the statistical nature of the data—cases, average, median, etc?
- Is the provided data normally distributed?
- Are anxiety and review time significantly correlated to exam performance?
- How much of the variance in performance shared with anxiety and review time?
- Does the association hold up if we isolate the correlation to what is unique between each combination—partial correlation?
- Are anxiety and review time predictive of exam performance?

The full explanation of correlation and the herein case are available in the text, Discover Statistics Using R. Fields and Miles. Chapters 6 and 7.

Agenda:

- ❑ Purpose and approach.
- ❑ Installation, packages and sessions.
- ❑ Case from Discovering Statistics Using R, Field and Miles, 2012
  - ➢ Case.
  - ➢ **Loading super tables into R.**
  - ➢ Know thy data.
  - ➢ Correlation and partial correlation.
  - ➢ Regression—standard and robust.
- ❑ Reference library.

Reading data into the session is done with a function, by which we can demonstrate several fundamental aspects of “R”

Location strings from windows typically have backward slashes, they must be changed to two backwards slashed and to forward slashes for R.

The source data is specified by the path, thus, leaving a record for where the data is stored.

```
## Read the csv file into R

examData <- read.csv("C:/Users/user pc/OneDrive - DataAnalytic 1/TrainR_Analytics/Correlation/AnxietyParCorr.csv", header=TRUE, sep=",")
```

Read.csv() is a function with three arguments between the (), file and location, table includes headers and data separator is “,”

The “<-” assigns what happens to the right to whatever is to the left. Our data is now called as examData.

Note:

- **For xlsx files, use the read.xlsx package (**https://cran.r-project.org/web/packages/xlsx/xlsx.pdf**)**
- **For txt files, use the read.delim function (**https://cran.rstudio.com/web/packages/readtext/vignettes/readtext_vignette.html**).**
- **For MS Access files, use the RODBC package (**https://cran.r-project.org/web/packages/ImportExport/ImportExport.pdf**)**

Comments on tables

- **Tables can be built from scratch in R; with the SQL() function or mainstream coding as R-type tables.**
- **Most typically you would build super tables in MS Access (**https://analytics4strategy.com/train-builddatatables**) as needed for the intended analytic and import them to "R.**
- **The imported data of the imported super table will be manipulated in the conduct of the analytic; but the super table remains the same.**
- **Tables returned by the analytic can be exported with a write() function.**

Agenda:

- Purpose and approach.
- Installation, packages and sessions.
- Case from Discovering Statistics Using R, Field and Miles, 2012
  - Case.
  - Loading super tables into R.
  - **Know thy data.**
  - Correlation and partial correlation.
  - Regression—standard and robust.
- Reference library.

Descriptive, graphic and statistical insight into the source data is generated through various R functions

- This section demonstrates analysis that should be standard to working with the data to any analysis.
- The functions demonstrated can be run as shown for each body of data.
- See Field and Miles, Chapter 4: Exploring Data with Graphs, for recipes to visualize your data in the many ways regarded as best practice.
- Save the codes of the section and the chapter in a .R script, ready to activate and feed the subject data.

The demonstrated analysis will begins with commands to view the data of the super table from different perspectives as shown by the code line to each output

```
> head(examData)##Shows first five cases
  Code Revise Exam Anxiety Gender
1    1      4   40  86.298   Male
2    2     11   65  88.716 Female
3    3     27   80  70.178   Male
4    4     53   80  61.312   Male
5    5      4   40  89.522   Male
6    6     22   70  60.506 Female
```

```
> str(examData)  ##Shows make up of the variables
'data.frame':   103 obs. of  5 variables:
 $ Code   : int  1 2 3 4 5 6 7 8 9 10 ...
 $ Revise : int  4 11 27 53 4 22 16 21 25 18 ...
 $ Exam   : int  40 65 80 80 40 70 20 55 50 40 ...
 $ Anxiety: num  86.3 88.7 70.2 61.3 89.5 ...
 $ Gender : Factor w/ 2 levels "Female","Male": 2 1 2 2 2 1 1 1 1 1 ...
```

Look at the data with summary(), describe(), str() and head()

head(examData)	**##Shows first five cases**
summary(examData)	**##Descriptive statistics**
describe(examData)	**##Table of descriptive statistics for each variable**
str(examData)	**##Shows make up of the variables**

One of the three tables to be returned

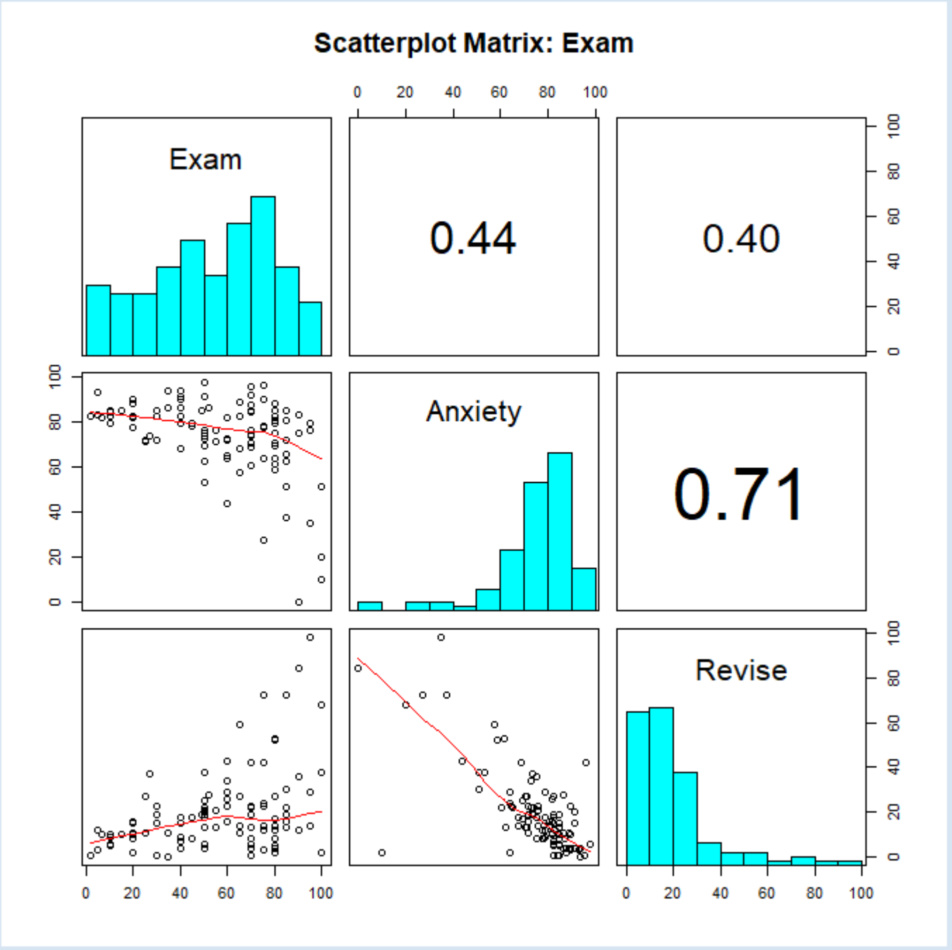
```
> summary(examData)##Descriptive statistics
      Code           Revise            Exam           Anxiety          Gender
 Min.   :  1.0   Min.   : 0.00   Min.   :  2.00   Min.   : 0.056   Female:51
 1st Qu.: 26.5   1st Qu.: 8.00   1st Qu.: 40.00   1st Qu.:69.775   Male  :52
 Median : 52.0   Median :15.00   Median : 60.00   Median :79.044
 Mean   : 52.0   Mean   :19.85   Mean   : 56.57   Mean   :74.344
 3rd Qu.: 77.5   3rd Qu.:23.50   3rd Qu.: 80.00   3rd Qu.:84.686
 Max.   :103.0   Max.   :98.00   Max.   :100.00   Max.   :97.582
```

```
> describe(examData)##Table of descriptive statistics for each variable
examData

 5  Variables      103  Observations
--------------------------------------------------------------------------------
Code
       n  missing distinct     Info     Mean      Gmd      .05      .10
     103        0      103        1       52    34.67      6.1     11.2
     .25      .50      .75      .90      .95
    26.5     52.0     77.5     92.8     97.9

lowest :   1   2   3   4   5, highest:  99 100 101 102 103
--------------------------------------------------------------------------------
Revise
```

The pairs grid is an example of finding a code by googling with an awareness of what is out there—scatter plot matrix—copy-paste to the session and insert my own variables

The function is available in the script to the slides under the section titled, "FUNCTION TO CREATE PAIRS PLOT--ONLY RUN INITIALLY."

```
## Pairs() creates scatter plot matrix--see graphics window
                ##Must have run the function earlier in the script
pairs(Exam ~ Anxiety + Revise, data = examData,
                main = "Scatterplot Matrix: Exam", diag.panel = panel.hist,
                lower.panel = panel.smooth, upper.panel = panel.cor)
```

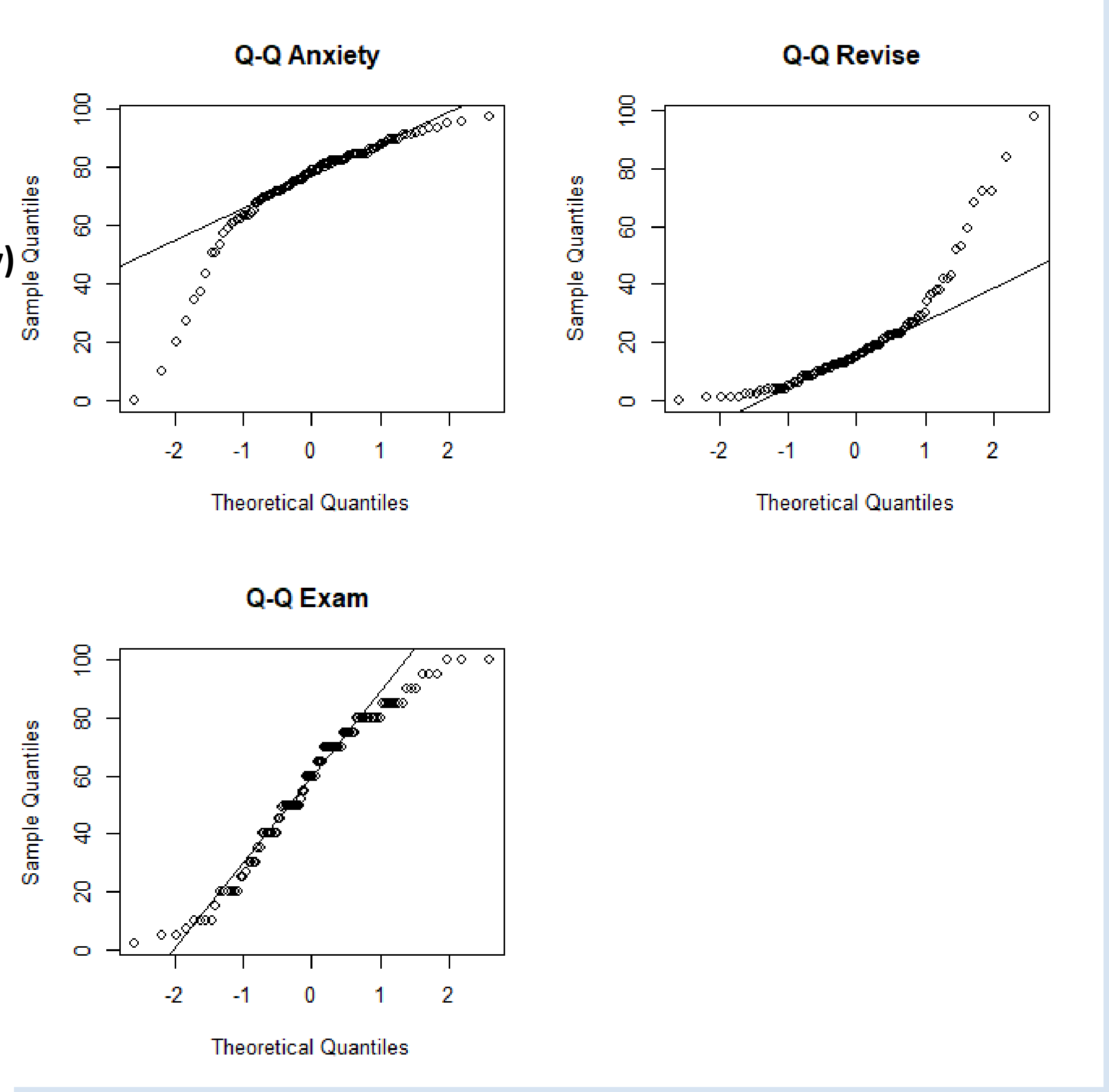


The matrix is an example of layered charting.

- **Charts to multiple variable sets.**
- **Scatter plots.**
- **Trend line to pairs.**
- **Histogram of each variable.**
- **Correlation to each pair.**

Use the qqnorm() and qqline() functions for each variable to determine if the variable has a normal distribution—those shown do not because they do not lie along the straight line

```
#### We have assumed normal distribution in the data ##################
        ##Test data with the qqnorm() function
        ##The points must lie along the qqline
par(mfrow = c(2, 2)) # 2 x 2 pictures on one plot

anx<- qqnorm(examData$Anxiety, main = "Q-Q Anxiety "); qqline(examData$Anxiety)
rev<- qqnorm(examData$Revise, main = "Q-Q Revise "); qqline(examData$Revise)
exm<- qqnorm(examData$Exam, main = "Q-Q Exam "); qqline(examData$Exam)

par(mfrow = c(1, 1)) #returns to 1 x 1 pictures on one plot
```



The code to the qq test plots demonstrates some fundamentals

Cause the three graphs to display in a two-by-two layout rather than single separate plots.

Shows "one" method for identifying variables—Anxiety variable of the examData table.

```
#### We have assumed normal distribution in the data #################
            ##Test data with the qqnorm() function
            ##The points must lie along the qqline
par(mfrow = c(2, 2))      # 2 x 2 pictures on one plot

anx<- qqnorm(examData$Anxiety, main = "Q-Q Anxiety "); qqline(examData$Anxiety)
rev<- qqnorm(examData$Revise, main = "Q-Q Revise "); qqline(examData$Revise)
exm<- qqnorm(examData$Exam, main = "Q-Q Exam "); qqline(examData$Exam)

par(mfrow = c(1, 1))      #returns to 1 x 1 pictures on one plot
```

Reverses the previous par() command, returning to presenting graphs individually

There are two plots, qqline layered over the qqnorm plot of points.

Agenda:

- ❑ Purpose and approach.
- ❑ Installation, packages and sessions.
- ❑ Case from Discovering Statistics Using R, Field and Miles, 2012
  - ➢ Case.
  - ➢ Loading super tables into R.
  - ➢ Know thy data.
  - ➢ **Correlation and partial correlation.**
  - ➢ Regression—standard and robust.
- ❑ Reference library.

The first step need not be to form a table just for the purpose of conducting correlation analysis, but doing so allows the demonstration of working with R tables

```
> head(examData)##Shows first five cases
  Code Revise Exam Anxiety Gender
1    1      4   40  86.298   Male
2    2     11   65  88.716 Female
3    3     27   80  70.178   Male
4    4     53   80  61.312   Male
5    5      4   40  89.522   Male
6    6     22   70  60.506 Female
```

```
## Crate a dataframe from the three variables of interest.
examData2<- examData[,c("Exam","Anxiety","Revise")]
head(examData2)
```

```
> head(examData2)
  Exam Anxiety Revise
1   40  86.298      4
2   65  88.716     11
3   80  70.178     27
4   80  61.312     53
5   40  89.522      4
6   70  60.506     22
```

When data is assigned to an object—examData—the object looks like a table, but in R is called a data frame which will now be reduced to the data on which the cor() would calculate correlation and other measures

The code to form the table for correlation analysis presents two very fundamental basics to working in "R"—brackets and the combine function, c()

Crate a dataframe from the three variables of interest.

examData2<- examData[,c("Exam","Anxiety","Revise")]

```
> head(examData)##Shows first five cases
  Code Revise Exam Anxiety Gender
1    1      4   40  86.298   Male
2    2     11   65  88.716 Female
3    3     27   80  70.178   Male
4    4     53   80  61.312   Male
5    5      4   40  89.522   Male
6    6     22   70  60.506 Female
```

```
> head(examData2)
  Exam Anxiety Revise
1   40  86.298      4
2   65  88.716     11
3   80  70.178     27
4   80  61.312     53
5   40  89.522      4
6   70  60.506     22
```

- No code at the left of the comma specifies to transfer all rows to the new table.
- Can filter rows by any commanded category and calculation.

Row

[,]

Column

- The combine function, c("Exam","Anxiety","Revise"), specifies the columns to the new table, examData2
- Notice order has been changed by the order of c()
- Could have also coded as c(3, 4, 2)

Note:
For a deeper explanation of working with tables, see sections 3.5 and 3.9 of Field and Miles.

This is a good place to demonstrate how that to find one's way through an analytic—one only needs to know there is a solution function and to go looking for it—use the help() function or google it

The following and much more can be found via the script help(cor) http://127.0.0.1:30843/library/stats/html/cor.html

cor(x, y, use = "everything", method = "correlation type")

Arguments

X	A numeric variable, matrix or data frame.
Y	NULL (default) or a vector, matrix or data frame with compatible dimensions to x.
Use	An optional character string giving a method for computing in the presence of missing values. This must be (an abbreviation of) one of the strings "everything", "all.obs", "complete.obs", "na.or.complete", or "pairwise.complete.obs." (1)
Method	A character string indicating which correlation coefficient is to be computed. One of "pearson" (default), "kendall", or "spearman": can be abbreviated.

(1) See page 216 of Field and Miles for definitions of each case to the "Use" argument.

We will use the cor() function with methods as pearson and spearman—with our dataframe examData2 as the x argument and y as accordingly not necessary

```
## Generate table of correlation of the dataframe with the function cor()
        ## cor(x,y, use="string", method="correlation type")
        ## use is choice for treating empty cells, method is pearson, spearman and kedall
        ## Args beyond basic are execluded due to nonmempty cells
        ## help(function) will take to explanation of the called function. EXAMPLE help(cor)
# help(cor)
cor(examData2, method="pearson")
cor(examData2, method="spearman")
```

```
> cor(examData2, method="pearson")
              Exam    Anxiety     Revise
Exam     1.0000000 -0.4409934  0.3967207
Anxiety -0.4409934  1.0000000 -0.7092493
Revise   0.3967207 -0.7092493  1.0000000
> cor(examData2, method="spearman")
              Exam    Anxiety     Revise
Exam     1.0000000 -0.4046141  0.3498948
Anxiety -0.4046141  1.0000000 -0.6219694
Revise   0.3498948 -0.6219694  1.0000000
```

- Notice that the tables return different correlations because our data is not a normal distribution—as seen previously.
- For the non-normal, the spearman uses a non-parametric method.
- The strength of correlations of the more realistic Spearman method are shown to be less than the Pearson reports.

Now to review the significance as p-value less than some percent—e.g., 5 percent—in this case the correlations are significant because "P" for both are very small

```
## We want to view the p-value to the correlations
        ## We must use another function to generate p-values--we could have chosen initially
        ## We will use rcorr() of the Hmisc() package.
        ## the function requires that data must be presented as data matrix see help(matrix)
examMatrix<- as.matrix(examData[,c("Exam","Anxiety","Revise")])
head(examMatrix)
# help(rcorr)
rcorr(examMatrix, type = "pearson")
rcorr(examMatrix, type = "spearman")
```

- P-value <= .05 indicate the correlations under both methods are significant
- In two pairs, the Spearman shows significance is less than the Pearson.

```
> rcorr(examMatrix, type = "pearson")
         Exam Anxiety Revise
Exam     1.00   -0.44   0.40
Anxiety -0.44    1.00  -0.71
Revise   0.40   -0.71   1.00

n= 103


P
        Exam Anxiety Revise
Exam          0       0
Anxiety  0            0
Revise   0    0
> rcorr(examMatrix, type = "spearman")
         Exam Anxiety Revise
Exam     1.00   -0.40   0.35
Anxiety -0.40    1.00  -0.62
Revise   0.35   -0.62   1.00

n= 103


P
        Exam   Anxiety Revise
Exam           0e+00   3e-04
Anxiety 0e+00          0e+00
Revise  3e-04  0e+00
```

And “R” Matrix and Data frame look-alike, but are different in nature.

- An R matrix is a collection of numbers arranged into a fixed number of rows and columns.
- An R data frame is a collection of characters and numbers as compared to fully numeric matrix.
- The function we will choose to determine p-values, requires a matrix rather than able to work with our data frame.
- To get there we convert the data frame to a matrix with the as.matrix() function and assign to a table object, examMatrix.

```
examMatrix<- as.matrix(examData[,c("Exam","Anxiety","Revise")])
```

or

```
examMatrix<- as.matrix(examData2)
```

Note:
For a deeper explanation of working with tables, see sections 3.5 and 3.9 of Fields and Miles.

We are using the cor.test() function to determine the upper and lower limits of the correlations—default of 95 percent

```
# View the conficent intervals to what has shown to be significant
                ## See help(cor.test)for details of the function
                ## Must view for each combination
#help(cor.test)
cor.test(examData$Anxiety, examData$Exam, method="pearson")
cor.test(examData$Anxiety, examData$Exam, method="spearman")
cor.test(examData$Anxiety, examData$Revise, method="pearson")
cor.test(examData$Anxiety, examData$Revise, method="spearman")
cor.test(examData$Revise, examData$Exam, method="pearson")
cor.test(examData$Revise, examData$Exam, method="spearman")
```

```
> cor.test(examData$Anxiety, examData$Exam, method="pearson")

        Pearson's product-moment correlation

data:  examData$Anxiety and examData$Exam
t = -4.938, df = 101, p-value = 3.128e-06
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.5846244 -0.2705591
sample estimates:
       cor
-0.4409934

> cor.test(examData$Anxiety, examData$Exam, method="spearman")

        Spearman's rank correlation rho

data:  examData$Anxiety and examData$Exam
S = 255790, p-value = 2.245e-05
alternative hypothesis: true rho is not equal to 0
sample estimates:
       rho
-0.4046141

Warning message:
In cor.test.default(examData$Anxiety, examData$Exam, method = "spearman") :
  Cannot compute exact p-value with ties
>
```

As expected for a small p-value, “0” does not appear between the limits.

Notice the spearman does not produce confidence limits.

Before going further it is important to distinguish between the correlation coefficients **R** and **R²**

- **Correlation confidents (R) indicates the extent to which two variables move together with respect to their respective means.**
- **R^2 is the extent that the variance in on variable is shared by another.**

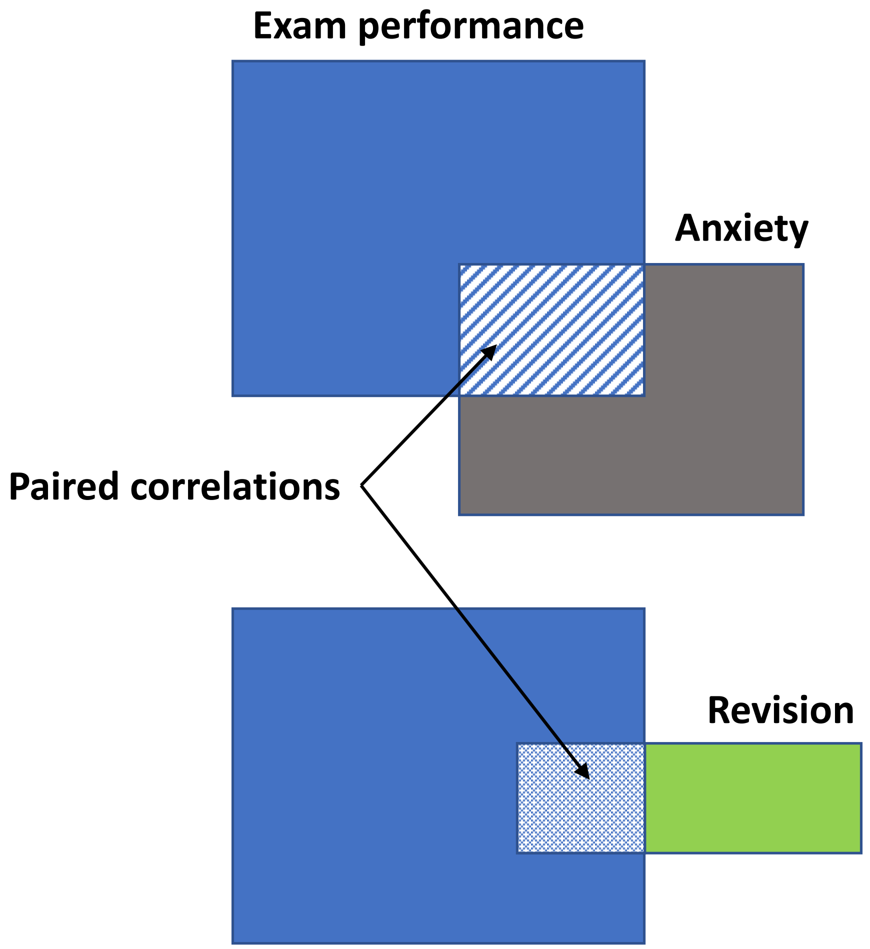
The extent that the variances in one variable are shared by another is a simple calculation—the cor() squared

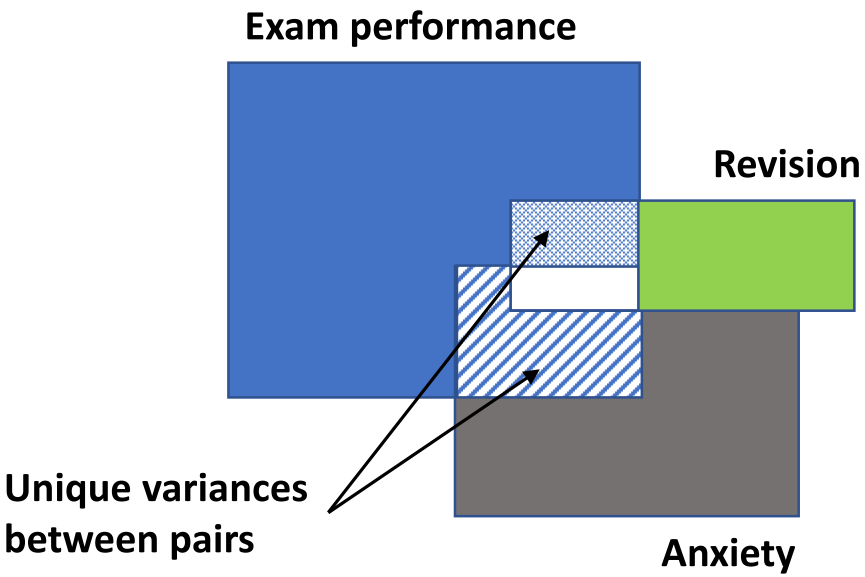
```
## Measure the extent that the variance in on variable is shared by another--R squared
            ##Square the previous cor()
cor(examData2, method="pearson")^2
cor(examData2, method="spearman")^2
```

Once again the more realistic result upon non-normal distribution is notably less.

```
> cor(examData2, method="pearson")^2
              Exam   Anxiety    Revise
Exam     1.0000000 0.1944752 0.1573873
Anxiety  0.1944752 1.0000000 0.5030345
Revise   0.1573873 0.5030345 1.0000000
> cor(examData2, method="spearman")^2
              Exam   Anxiety    Revise
Exam     1.0000000 0.1637126 0.1224264
Anxiety  0.1637126 1.0000000 0.3868459
Revise   0.1224264 0.3868459 1.0000000
> 
```

There is a need to determine the extent of correlation—and if significant—that is unique between two variables rather than including the correlation shared with one or more other variables





It is readily apparent that paired correlation and R^2 can be misleading, if we disregard the associations of other variables.

There are of course a functions for that—one to determine partial correlation and another to determine significance—let's look at the pairs and partials between Exam and Anxiety

```
## Partial correlation
        ## Use the pcor() and pcor.test() OF  the ggm package
        ## Must be used for pairs, controled variables can be one or multiple
cor(examData2, method="pearson")
        #### Partial Exam to Anxiety
pcExAn<- pcor(c("Exam", "Anxiety", "Revise"), var(examData2))
pcExAn
pcExAn^2
        ## The second arg of pcor.test is number of controlled variables,
        ## the third is number of cases
pcor.test(pcExAn, 1, 103)
```

Can be more than one control variable

```
> ## Partial correlation
> ## Use the pcor() and pcor.test() OF  the ggm package
> ## Must be used for pairs, controled variables can be one or multiple
> cor(examData2, method="pearson")
              Exam    Anxiety     Revise
Exam     1.0000000 -0.4409934  0.3967207
Anxiety -0.4409934  1.0000000 -0.7092493
Revise   0.3967207 -0.7092493  1.0000000
> #### Partial Exam to Anxiety
> pcExAn<- pcor(c("Exam", "Anxiety", "Revise"), var(examData2))
> pcExAn
[1] -0.2466658
> pcExAn^2
[1] 0.06084403
> ## The second arg of pcor.test is number of controled variables,
> ## the third is number of cases
> pcor.test(pcExAn, 1, 103)
$tval
[1] -2.545307

$df
[1] 100

$pvalue
[1] 0.01244581
```

Note the contrast

The partial correlation is significant (<=0.05%)

Type	Exam-Anxiety			Exam-Revise		
	R	R^2	P	R	R^2	P
Paired	-0.44	0.19	<0.001	0.40	0.16	<0.001
Partial	-0.24	0.06	0.012	0.13	0.02	0.18

Correlation does not hold (P>0.05%) when recognizing presence of other variables

Agenda:

- Purpose and approach.
- Installation, packages and sessions.
- Case from Discovering Statistics Using R, Field and Miles, 2012
  - Case.
  - Loading super tables into R.
  - Know thy data.
  - Correlation and partial correlation.
  - **Regression—standard and robust.**
- Reference library.

Let’s demonstrate linear regression in contrast to correlation analysis as an opportunity to point to the structure of “formula” that appears often in the “R” functions of most analytic models

- **There are multiple steps to linear regression in order to:**
  - **Determine best predictor variables.**
  - **Discover and work around issues to the model until it can be considered as valid.**
- **There is a clear distinction between the coefficient of regression and correlation.**
  - **Correlation coefficients indicate the extent to which two variables move together with respect to their respective means.**
  - **Regression indicates the impact of a unit change in a predictor variable will show in the predicted variable.**
- **Refer to Chapters 7 and 8 of Field and Miles for a full explanation of the principles to linear and logistic regression, demonstrated with R.**

The regression has a set-up common to many other models and some basic interpretation—let's look at the basic structure

The computation is assigned to an object, our model.

The desired function: lm() with our data as input

```
examPred <- lm(Exam ~ Anxiety + Revise, examData)
summary(examPred)
confint(examPred)
```

Calls up the model.

Calls up confidence intervals to prediction variables.

The formula of the model

- "~" is equivalent to "=“
- To right of "~" are the predictor variables, distinguish by the "+"
- An "*" instead of "+" will structure the formula to cause the interaction between predictors

Note:
The lm() function is the machine learning (fit) stage linear regression, the predict() function is to conduct the artificial intelligence stage to most model types.

An early-on first step after determining the predictive variables is to test for interaction between variables with a simple element in the formula

```
## Test for interaction between predictor variables
examPredIntr <- lm(Exam ~ Anxiety * Revise, examData)
summary(examPredIntr)
confint(examPredIntr)
```

"*" rather than "+" causes the interaction Anxiety:Revise to be a predictor variable.

- The interaction is returned along with the estimate and test of the main variables.
- In this, p-value is >0.05—interaction is not significant—drop the variable.

- Other marker of significance is if the sign changes between upper and lower limits.
- Because p-value > 0.05, our signs change.

```
> examPredIntr <- lm(Exam ~ Anxiety * Revise, examData)
> summary(examPredIntr)

Call:
lm(formula = Exam ~ Anxiety * Revise, data = examData)

Residuals:
    Min      1Q  Median      3Q     Max 
-44.790 -13.208   0.653  20.721  40.210 

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)    
(Intercept)    100.959573  19.031389   5.305 6.89e-07 ***
Anxiety         -0.683887   0.230505  -2.967  0.00377 ** 
Revise           0.139686   0.309041   0.452  0.65226    
Anxiety:Revise   0.007343   0.004854   1.513  0.13352    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 23.16 on 99 degrees of freedom
Multiple R-squared:  0.2265,	Adjusted R-squared:  0.2031 
F-statistic: 9.665 on 3 and 99 DF,  p-value: 1.181e-05

> confint(examPredIntr)
                      2.5 %       97.5 %
(Intercept)    63.197167425 138.72197829
Anxiety        -1.141257979  -0.22651521
Revise         -0.752889743   0.47351756
Anxiety:Revise -0.002288171   0.01697387
> 
```

Now to determine if there is a relationship, and how strong, between Anxiety and Revise to Exam outcome

Model with interaction removed

examPred <- lm(Exam ~ Anxiety + Revise, examData)

summary(examPred)
confint(examPred)

Revise time is NOT a significant predictor of exam performance, but Anxiety is.

```
> ##Conduct linear model
> examPred <- lm(Exam ~ Anxiety + Revise, examData)
> summary(examPred)

Call:
lm(formula = Exam ~ Anxiety + Revise, data = examData)

Residuals:
    Min      1Q  Median      3Q     Max
-46.181 -16.949  -0.834  21.757  40.556

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  87.8326    17.0469   5.152 1.3e-06 ***
Anxiety      -0.4849     0.1905  -2.545  0.0124 *
Revise        0.2413     0.1803   1.339  0.1837
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 23.31 on 100 degrees of freedom
Multiple R-squared:  0.2087,    Adjusted R-squared:  0.1928
F-statistic: 13.18 on 2 and 100 DF,  p-value: 8.285e-06

> confint(examPred)
                 2.5 %      97.5 %
(Intercept) 54.0120991 121.6530779
Anxiety     -0.8628957  -0.1069428
Revise      -0.1163331   0.5989376
```

Recall that the data is not a normal distribution—calling for “robust” linear regression made possible by the rlm() function of the Hmisc package

```
## Run a robust linear model with rlm() function
rlm.examlm <- rlm(Exam ~ Anxiety + Revise, data = examData)
summary(rlm.examlm)    ##call the model run by the previous line
```

Revise once again does not hold as significant.

- The Student’s T tells the story for both predictors.
- T-value for 0.05 or less with 100 degrees of freedom is >=1.962.
- Anxiety exceeds the cut-off, Revise does not.
- The strength of significance of relationship is less when we take the distribution in count.

```
> summary(rlm.examlm)##call the model run by the previous line

Call: rlm(formula = Exam ~ Anxiety + Revise, data = examData)
Residuals:
    Min      1Q  Median      3Q     Max
-46.567 -16.891  -0.924  21.456  40.283

Coefficients:
            Value   Std. Error t value
(Intercept) 88.3062 18.3553     4.8109
Anxiety     -0.4859  0.2051    -2.3685
Revise       0.2320  0.1941     1.1955

Residual standard error: 28.95 on 100 degrees of freedom
```

Notice the found strength of relationship of the robust regression is less than the parametrized regression.

Agenda:

- Purpose and approach.
- Installation, packages and sessions.
- Case from Discovering Statistics Using R, Field and Miles, 2012
    - Case.
    - Loading super tables into R.
    - Know thy data.
    - Correlation and partial correlation.
    - Regression—standard and robust.
- **Reference library.**

How-to guidance for "R"

Knowledge and skills	Texts, papers and session slides
Download software	▪ **Website: Download at** https://r-project.org ▪ **YouTube: How to install** https://www.youtube.com/watch?v=ym8szN2Zim4
Coding—general	▪ **R for Data Science, Grolemund, Wickham, 2017. Free E-Book** https://r4ds.had.co.nz/index.html ▪ **R for Dummies, de Vries, Meys, 2015.** ▪ **Art of Programing R, Matloff, 2011.**
Coding in context of conducting analytics (1)	▪ **Discovering Statistics Using R, Field and Miles, 2012** ▪ **Multilevel Modeling Using R, Holmes, 2014** ▪ **Introductory Time Series with R, Cowpertwait and Metcalfe, 2009** ▪ **Event History Analytics with R, Bostrom, 2012** ▪ **Machine Learning with R, Lantz, 2015**

Library: Continued

Knowledge and skills		Papers and presentations	Texts or equivalents
Five analytic questions	Relationship	[Find What Matters with Relationship Questions of Operations]()	▪ Discovering Statistics Using R, Field and Miles, 2012 ▪ Multilevel Modeling Using R, Holmes, 2014
	Difference	[Know that Improvements Work by Asking Difference Questions]()	
	Time series	[Explore What Did and May Happen with Time Series Questions]()	▪ Introductory Time Series with R, Cowpertwait and Metcalfe, 2009 ▪ R Package "tsoutliers," Javier López-de-Lacalle, 2017
	Duration	[Find the Time That is Money by Asking Duration Questions]()	▪ Event History Analytics with R, Bostrom, 2012 ▪ New Weibull Handbook, Abernathy, 2007 ▪ R Package "WeibullR" Weibull Analysis for Reliability Engineering, Silkworth & Symynck, 2018
	Apparency	[Dive Below the Surface of Process Functioning with Apparency Questions]()	Machine Learning with R, Lantz, 2015
Machine learning, AI	Methodology	None available	