Hands-On Exercise:

The R-Programming Environment and Its Application to Biometric Data and Analytics in Forensic Anthropology

Carl N. Stephan, Ph.D.

Jeffrey Lynch, MSc.

Sachin Pawaskar, Ph.D.

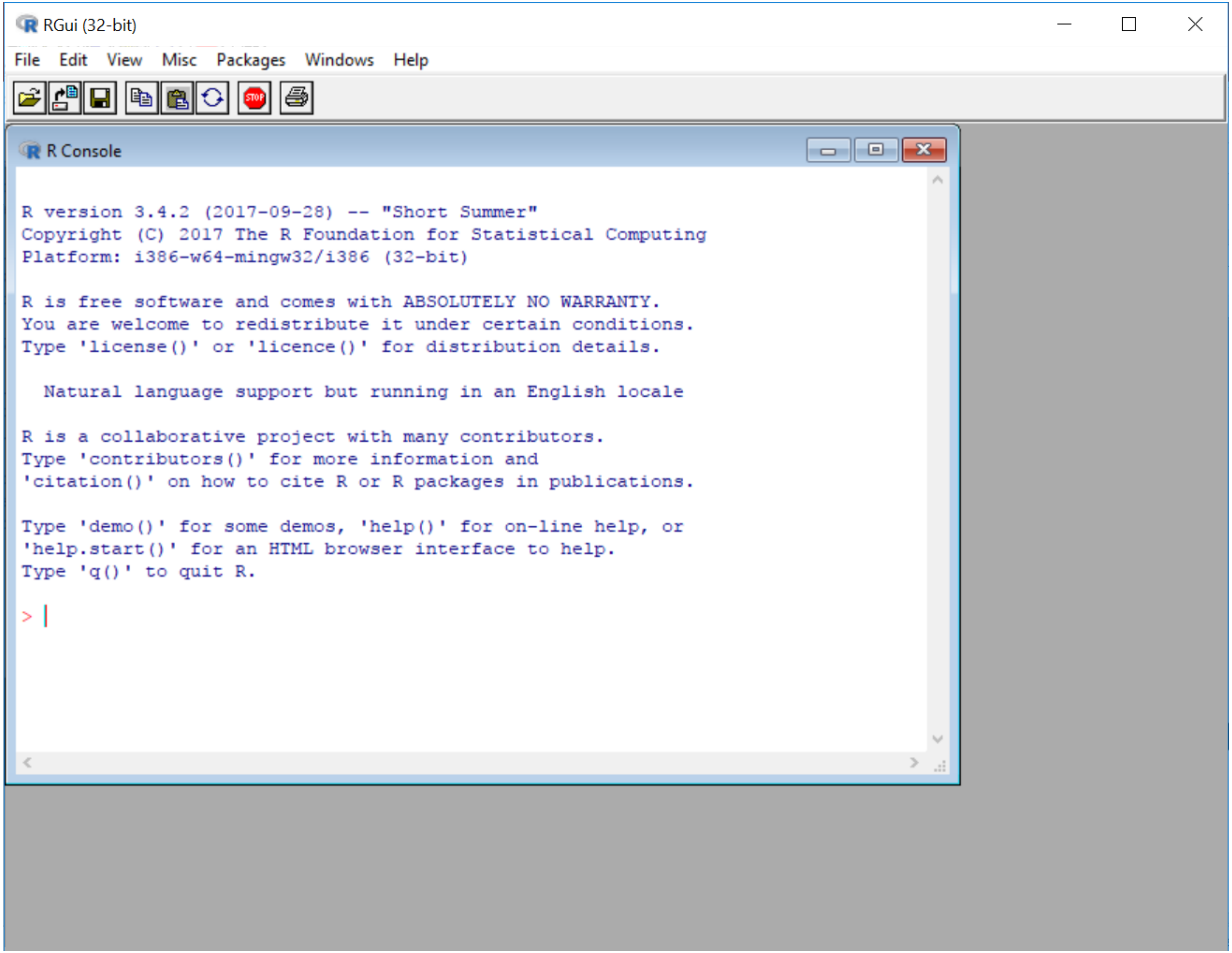
R Core Team (2013) R: A language and environment for statistical computing.
R Foundation for Statistical Computing, Vienna, Austria.
https://www.r-project.org/logo/

Disclosure

The presenters have no trade or commercial interests to disclose.

Almost all of the software discussed in this presentation is freely available open source software.

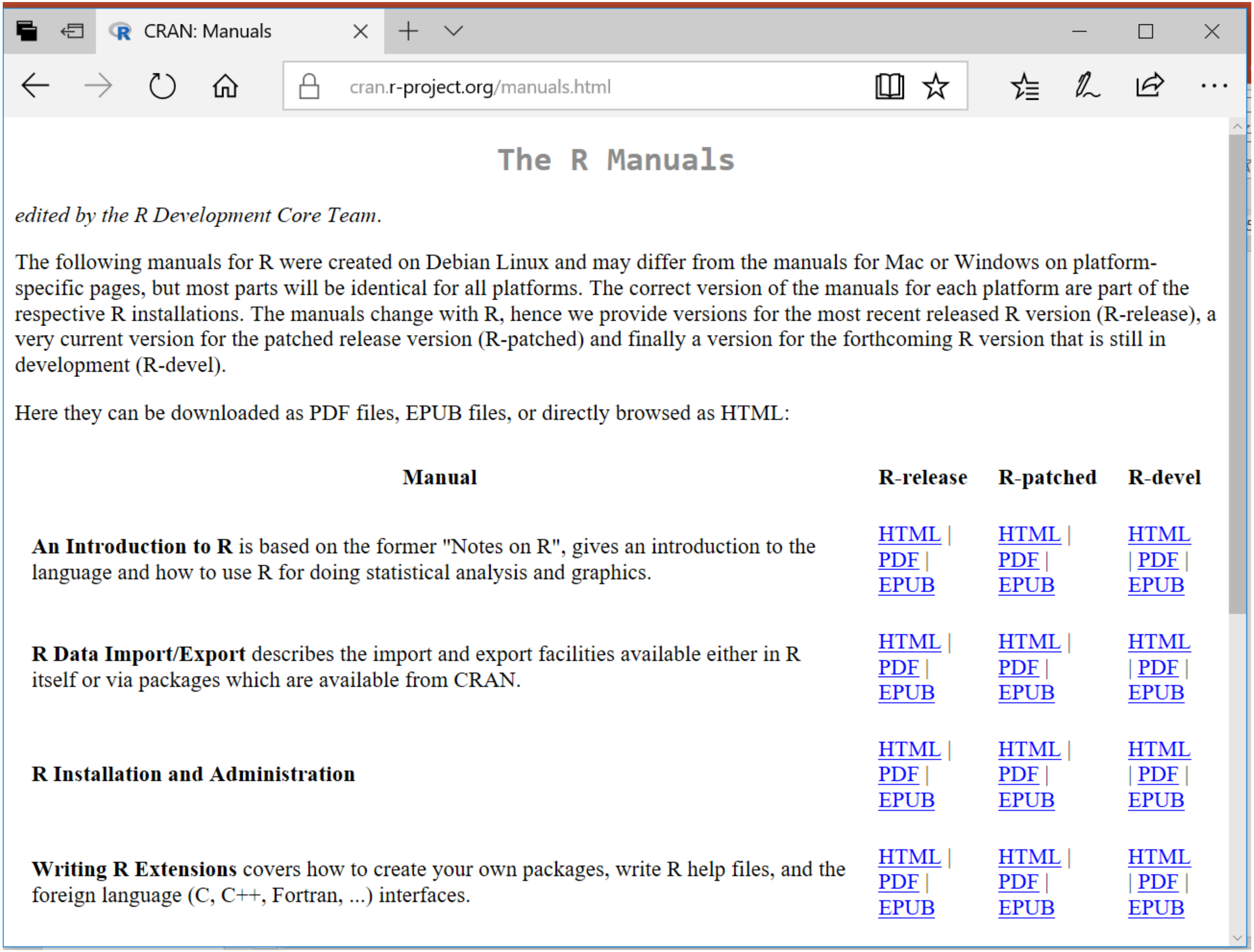
The authors do not hold any commercial interests with any of the commercially available software or hardware discussed.



```
R version 3.4.2 (2017-09-28) -- "Short Summer"
Copyright (C) 2017 The R Foundation for Statistical Computing
Platform: i386-w64-mingw32/i386 (32-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> 
```

Introduction to R

- R is an environment for statistical computing and graphics, which is highly extensible. The R language is statistics orientated, but a general purpose computer language.
- *"Many users think of R as a statistics system. We prefer to think of it of an environment within which statistical techniques are implemented." R Core Team*
- R can be considered as a different implementation of the S language.
- R is available as Free Software under the terms of the Free Software Foundation's GNU General Public License in source code form.
- Download from: https://www.r-project.org/



The R Manuals

edited by the R Development Core Team.

The following manuals for R were created on Debian Linux and may differ from the manuals for Mac or Windows on platform-specific pages, but most parts will be identical for all platforms. The correct version of the manuals for each platform are part of the respective R installations. The manuals change with R, hence we provide versions for the most recent released R version (R-release), a very current version for the patched release version (R-patched) and finally a version for the forthcoming R version that is still in development (R-devel).

Here they can be downloaded as PDF files, EPUB files, or directly browsed as HTML:

Manual	R-release	R-patched	R-devel
An Introduction to R is based on the former "Notes on R", gives an introduction to the language and how to use R for doing statistical analysis and graphics.	HTML | PDF | EPUB	HTML | PDF | EPUB	HTML | PDF | EPUB
R Data Import/Export describes the import and export facilities available either in R itself or via packages which are available from CRAN.	HTML | PDF | EPUB	HTML | PDF | EPUB	HTML | PDF | EPUB
R Installation and Administration	HTML | PDF | EPUB	HTML | PDF | EPUB	HTML | PDF | EPUB
Writing R Extensions covers how to create your own packages, write R help files, and the foreign language (C, C++, Fortran, ...) interfaces.	HTML | PDF | EPUB	HTML | PDF | EPUB	HTML | PDF | EPUB

Vectors

What are Vectors?

Numerical vectors are the basic data structure in R.

Numerical vectors are ordered collection of numbers
(like a long line of digits):

e.g.,   [1,2,3,4,5]

Vectors are created using the concatenate function c().

e.g., c(1,2,3,4,5)

This can be shortcut to: c(1:5)

Type:
c(1,2,3,4,5)

Naming Vectors

Vectors can be named or un-named.

To name a vector an assignment symbol is used (<-).

This is referred to as ‘points to’ and is represented by an arrow “<-” composed of “<“ and “-” symbols.

“=“ does the same but is not considered best practice R language.

The name for your vector can be anything you desire.

Type the black font:

```
vector1 <- c(2, 3, 4, 5, 6) #c() concatenates numbers into a vector
vector1 #Access all elements
```

Indexing Vectors

Each digit in a vector is called an ‘element’.

Elements can be accessed with a vector using indexing.

Square brackets [] are used to define the index position.

e.g., vector1[3] would index the third element in vector1, which equals 4.

Type the black font:

```
vector1[1] #Access only the first element

vector1[1:3] #Access the first three elements
```

Data structures in R

Numerical vectors can be ‘reformatted’ into different types of data structure in R:

- Lists
- Matrix
- Arrays
- Data frames

Examples using vector1 follow on the next slide.

Data structure Examples

(using vector1)

Simple vector:

[2,3,4,5,6]

List:

[[1]]

[2,3,4,5,6]

[[2]]

[2,3,4,5,6]

Matrix:

	[,1]	[,2]	[,3]	[,4]	[,5]
[1,]	2	2	2	2	2
[2,]	3	3	3	3	3
[3,]	4	4	4	4	4
[4,]	5	5	5	5	5
[5,]	6	6	6	6	6

Data frame:

	v1	v2	v3	v4]	v5
1	2	2	2	2	2
2	3	3	3	3	3
3	4	4	4	4	4
4	5	5	5	5	5
5	6	6	6	6	6

Functions for Generating Data Structures

Matrix

cbind() #column bind

rbind() #row bind

matrix() #generate cells for matrix

Lists

list()

Data frame

data.frame()

Change data to different formats:

as.list()

as.matrix(); data.matrix()

as.data.frame()

Scatterplot of two vectors

Type:

```
x <- c(0:150)
y <- x^4
plot(x,y)
```

Basic Graphical User Interfaces in R (GUIs)

- R has lots of pre-existing functions/code, making program development rapid
- One such library is tcltk, pronounced "Tickle Tee Kay"

To load type:

```
library(tcltk)
```

- tcltk enables fast generation of graphical user interfaces (GUIs), which you might know as 'windows'
- tcltk is its own language. R shoots messages to (inbuilt) tcltk using an interpreter, so the syntax used in R is an modification of the tcltk syntax.

Create a Basic GUI

```
library(tcltk)

tkmessageBox(title = "My First GUI", message = “R Rocks!", icon = "info", type = "ok")
```

Mac OS users will need to download X Windows first:

http://planspace.blogspot.com.au/2013/01/fix-r-tcltk-dependency-problem-on-mac.html

Custom Designing GUIs

- You can build your own GUIs from the ground up.
- The grid() function can be used to build GUIs row-by-row (or column-by-column)
- You can then specify where items within the GUI are placed, such as text and buttons.

Build Your Own Stature Calculator

Type the black font:

```
library(tcltk) #load tcltk library

dlg <- tktoplevel() #create window

tkwm.title(dlg, "Stature Calculator") #name the window

tkwm.geometry(dlg,"230x100+120+50") #define window size and position

a <- tclVar() #generate a variable to take entry text

entry.A <- tkentry(dlg, width= "5",textvariable = a) #generate a text entry box

OnOK <- function () {

            femur <- as.numeric(tclvalue(a))

            result <- round(0.20740*femur+72.17) #Trotter & Gleser 1952 stature regression for Black males

            tkgrid(tklabel(dlg,text=result), row=4, column=3, sticky="e") #grid result

            } #accept the entered number, use in calculation and write result to window

Button1<-tkbutton(dlg,text="  Calculate  ", command=OnOK) #create button

tkgrid(tklabel(dlg,text="")) #grid a blank line

tkgrid(tklabel(dlg,text="Enter Femur Length (mm):"),entry.A, columnspan=3, sticky="e") #grid some text

tkgrid(Button1, row=3, column=3, sticky="e") #grid the button

tkgrid(tklabel(dlg,text="Estimated Stature (cm):"), column=2, sticky="e") #grid some more text

tkgrid(tklabel(dlg,text="")) #grid a blank line
```

Calculate Stature for a Large Sample

In this example you will calculate statures across a large sample rather than for a single entry.

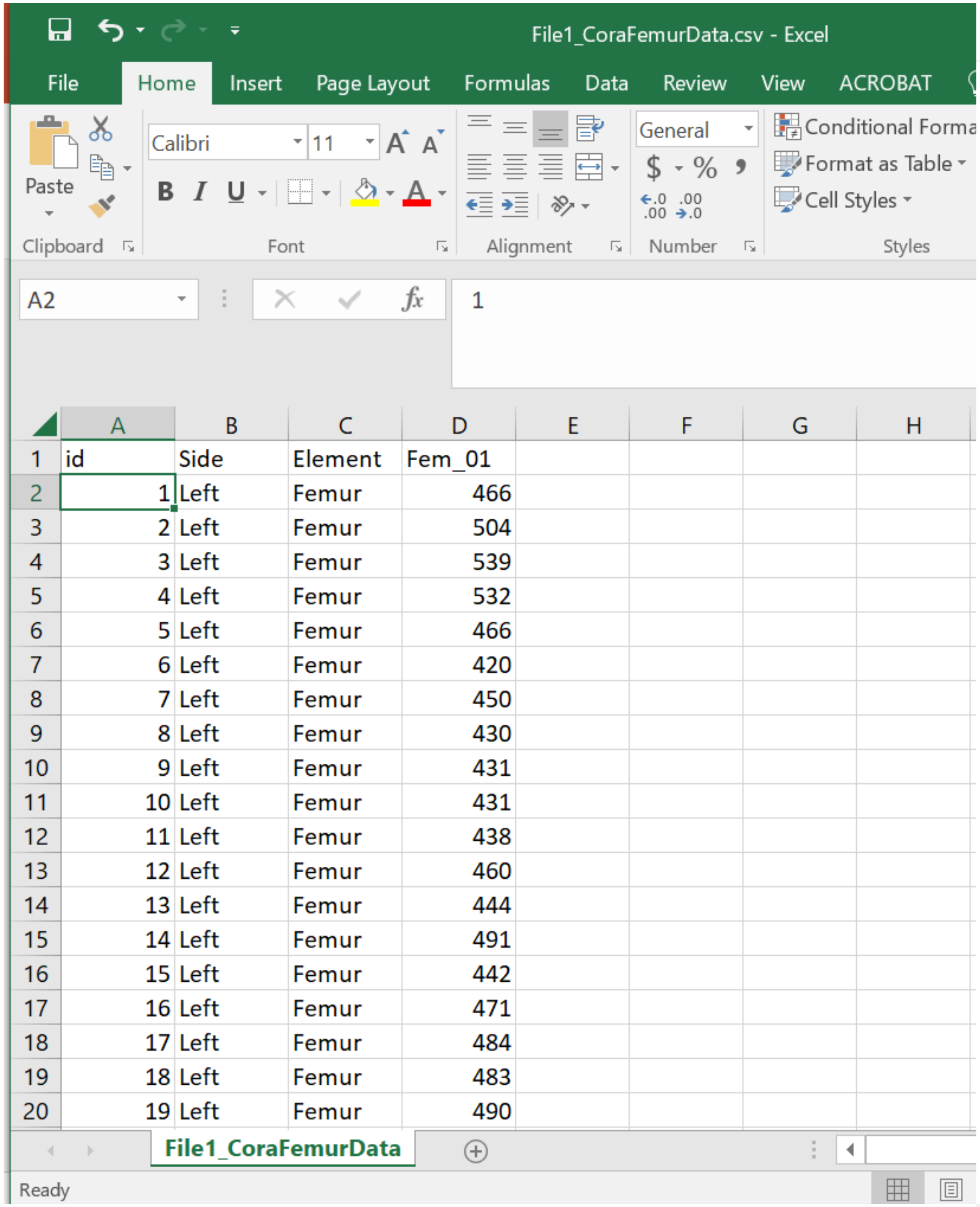
The data is contained in the presupplied .csv file called "File1_CoraFemurData.csv"

.csv is a transportable stock standard format. (Forget your .xls extensions!)

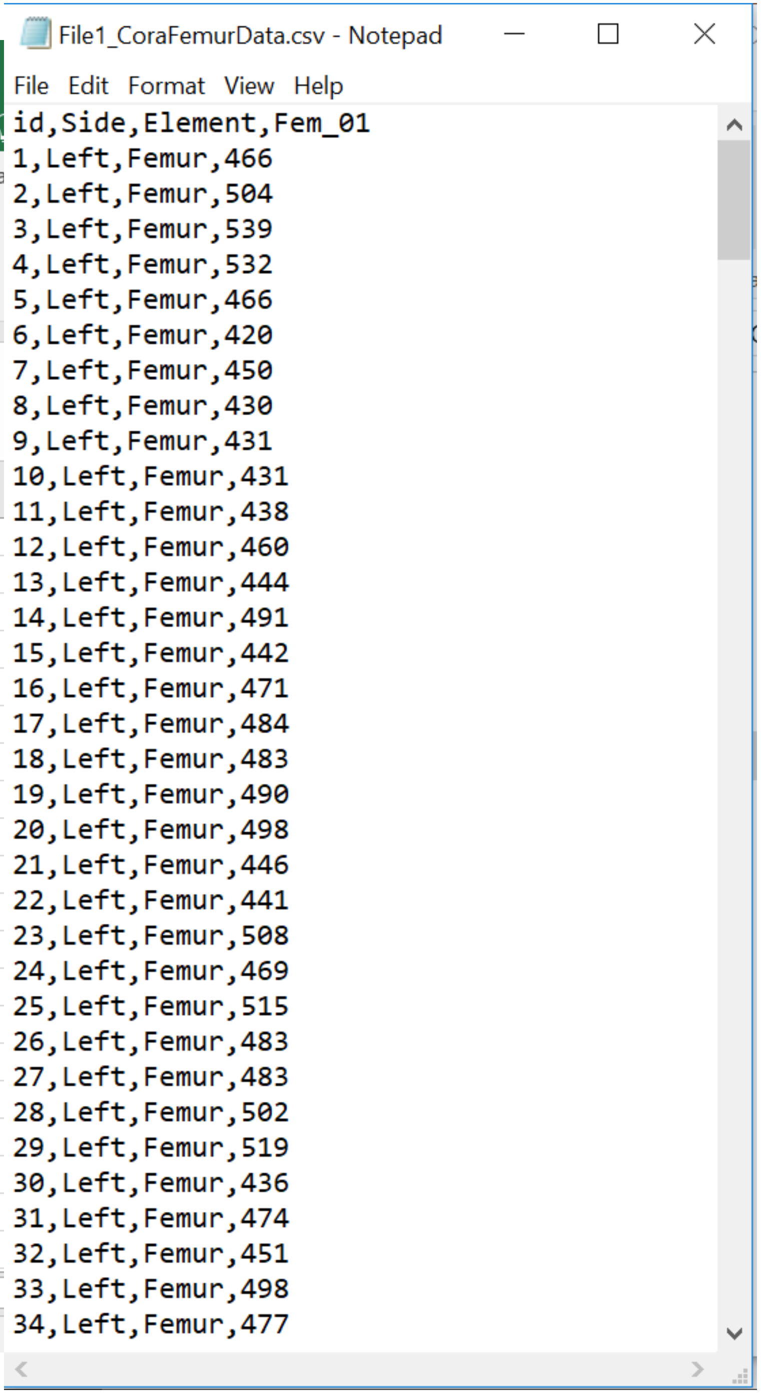
- "csv" stands for comma separated values

  = comma delimited file

  represents a text file with data values separated by commas.
- In Europe they often use semicolons so csv is said to represent "character delimited files".



	A	B	C	D	E	F	G	H
1	id	Side	Element	Fem_01				
2	1	Left	Femur	466				
3	2	Left	Femur	504				
4	3	Left	Femur	539				
5	4	Left	Femur	532				
6	5	Left	Femur	466				
7	6	Left	Femur	420				
8	7	Left	Femur	450				
9	8	Left	Femur	430				
10	9	Left	Femur	431				
11	10	Left	Femur	431				
12	11	Left	Femur	438				
13	12	Left	Femur	460				
14	13	Left	Femur	444				
15	14	Left	Femur	491				
16	15	Left	Femur	442				
17	16	Left	Femur	471				
18	17	Left	Femur	484				
19	18	Left	Femur	483				
20	19	Left	Femur	490				



```
id,Side,Element,Fem_01
1,Left,Femur,466
2,Left,Femur,504
3,Left,Femur,539
4,Left,Femur,532
5,Left,Femur,466
6,Left,Femur,420
7,Left,Femur,450
8,Left,Femur,430
9,Left,Femur,431
10,Left,Femur,431
11,Left,Femur,438
12,Left,Femur,460
13,Left,Femur,444
14,Left,Femur,491
15,Left,Femur,442
16,Left,Femur,471
17,Left,Femur,484
18,Left,Femur,483
19,Left,Femur,490
20,Left,Femur,498
21,Left,Femur,446
22,Left,Femur,441
23,Left,Femur,508
24,Left,Femur,469
25,Left,Femur,515
26,Left,Femur,483
27,Left,Femur,483
28,Left,Femur,502
29,Left,Femur,519
30,Left,Femur,436
31,Left,Femur,474
32,Left,Femur,451
33,Left,Femur,498
34,Left,Femur,477
```

Type the black font:

```
sample <- read.csv(file="C:/Rwork/File1_CoraFemurData.csv" , header = TRUE) #Imports reference data
sample #print the file to the console

sample <- na.omit(sample) #remove NA values from ‘empty’ cells
sample #print the file to the console

stature <- 0.20740*sample$Fem_01+72.17 #calculate statures
sample <- cbind(sample, stature) #add the statures to the main dataset
sample #print the file to the console

mean(sample[,5]) #calculate the stature mean
boxplot(sample[,5]) #boxplot the stature data
```

Calculate a Stature Model from Scratch & Use it to Calculate Stature

Load the second file called
“File2_ReferenceData.csv”

Refer to previous example for help.

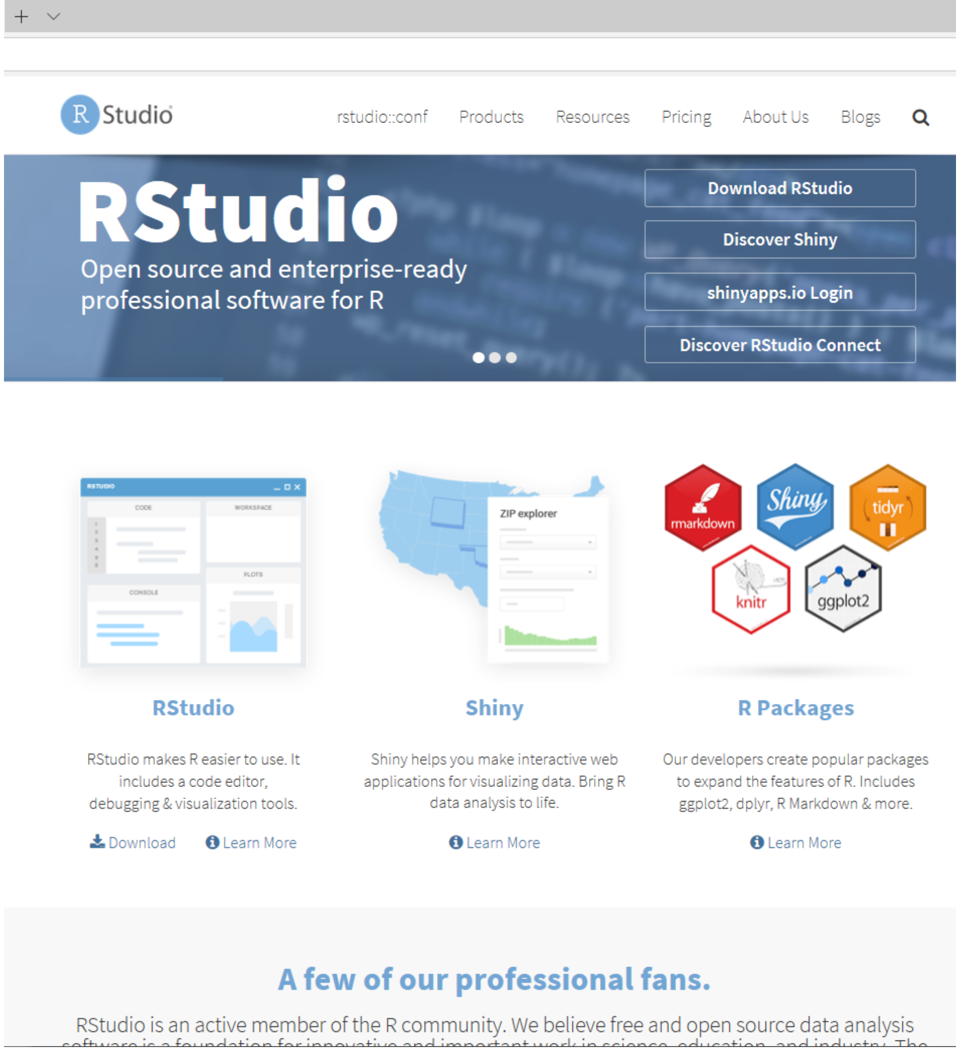
Type the black font:

```
ref <- ref[ref$sex == "M",] #Filters reference data by males

ref <- ref[ref$population == “B",] #Filters reference data by white population


lm1 <- lm(stature ~ Fem_01, data = ref) #Calculate regression model


pm1 <- predict(object = lm1, newdata = data.frame(Fem_01 = sample$Fem_01), interval = "predict", level = 0.95) #Calculate 95% point estimates and prediction intervals


sample <- cbind(sample, pm1) #Column bind the new results back into sample

sample

boxplot(sample[,5], sample[,6]) #Plot both stature estimation results
```

To go further, the following may be helpful…

R Studio, https://www.rstudio.com/

Notepad++, https://notepad-plus-plus.org/



RStudio

Open source and enterprise-ready professional software for R

RStudio

RStudio makes R easier to use. It includes a code editor, debugging & visualization tools.

Shiny

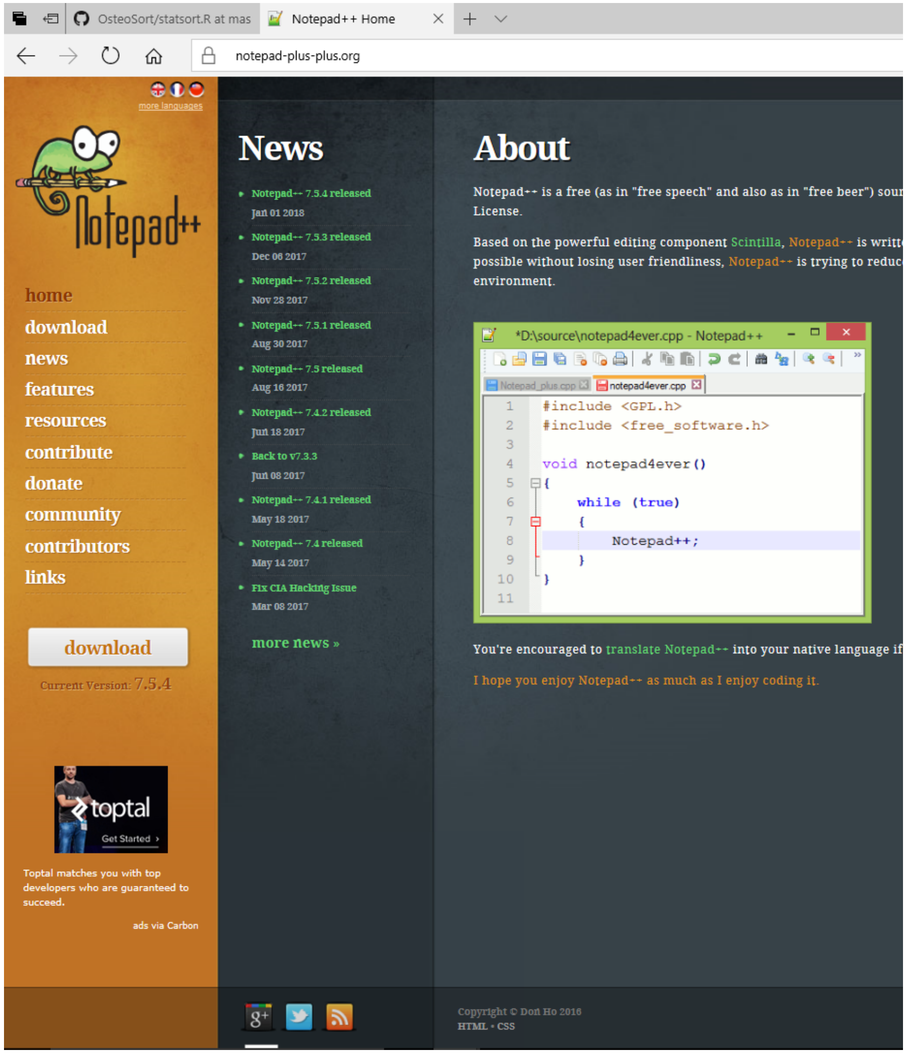
Shiny helps you make interactive web applications for visualizing data. Bring R data analysis to life.

R Packages

Our developers create popular packages to expand the features of R. Includes ggplot2, dplyr, R Markdown & more.

A few of our professional fans.

RStudio is an active member of the R community. We believe free and open source data analysis software is a foundation for innovative and important work in science, education, and industry. The



Current Version: 7.5.4

News

- Notepad++ 7.5.4 released
  Jan 01 2018
- Notepad++ 7.5.3 released
  Dec 06 2017
- Notepad++ 7.5.2 released
  Nov 28 2017
- Notepad++ 7.5.1 released
  Aug 30 2017
- Notepad++ 7.5 released
  Aug 16 2017
- Notepad++ 7.4.2 released
  Jun 18 2017
- Back to v7.3.3
  Jun 08 2017
- Notepad++ 7.4.1 released
  May 18 2017
- Notepad++ 7.4 released
  May 14 2017
- Fix CIA Hacking Issue
  Mar 08 2017

more news »

About

Notepad++ is a free (as in "free speech" and also as in "free beer") sour License.

Based on the powerful editing component Scintilla, Notepad++ is writt possible without losing user friendliness, Notepad++ is trying to reduc environment.

*D:\source\notepad4ever.cpp - Notepad++

Notepad_plus.cpp | notepad4ever.cpp

```
#include <GPL.h>
#include <free_software.h>

void notepad4ever()
{
    while (true)
    {
        Notepad++;
    }
}
```

You're encouraged to translate Notepad++ into your native language if

I hope you enjoy Notepad++ as much as I enjoy coding it.

HTML • CSS