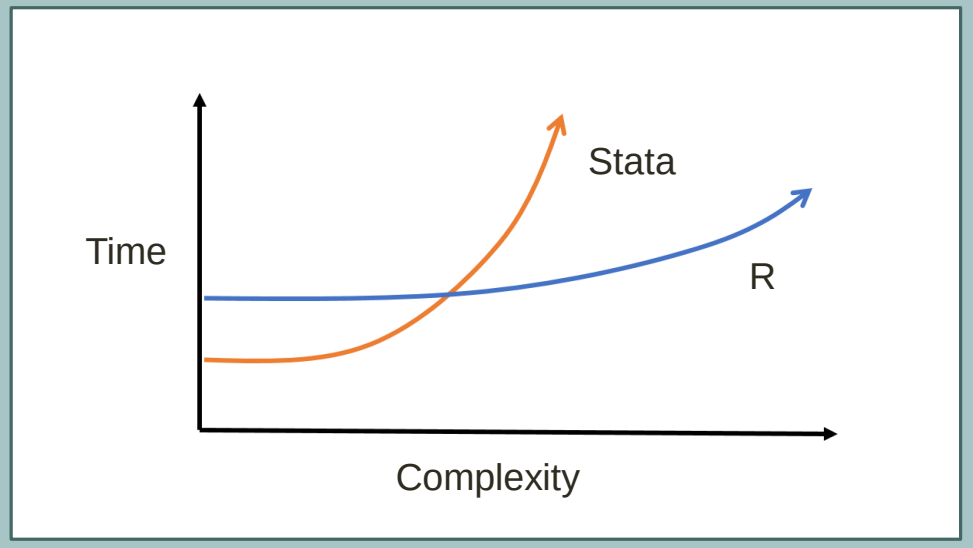
INTRO TO R

James Staley
james.staley@bristol.ac.uk



INTRODUCTION

James Staley
james.staley@bristol.ac.uk

> WHAT IS R?

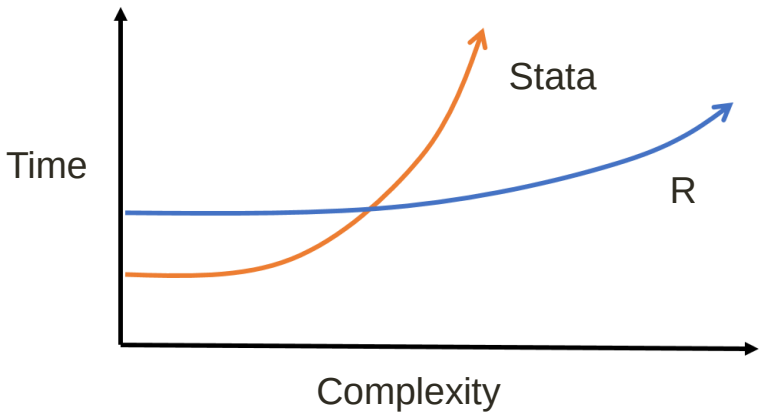
- R is a statistical programming language (based on S)
- R is open source - researchers develop packages to implement new statistical methods, plots or applications
- R runs on Windows, MacOS and UNIX

John Chambers

Dirk Eddelbuettel

> WHY USE R?

- R is free!
- R is flexible
- R is good at handling large datasets and multiple objects
- R has good plotting tools and packages for statistical analysis

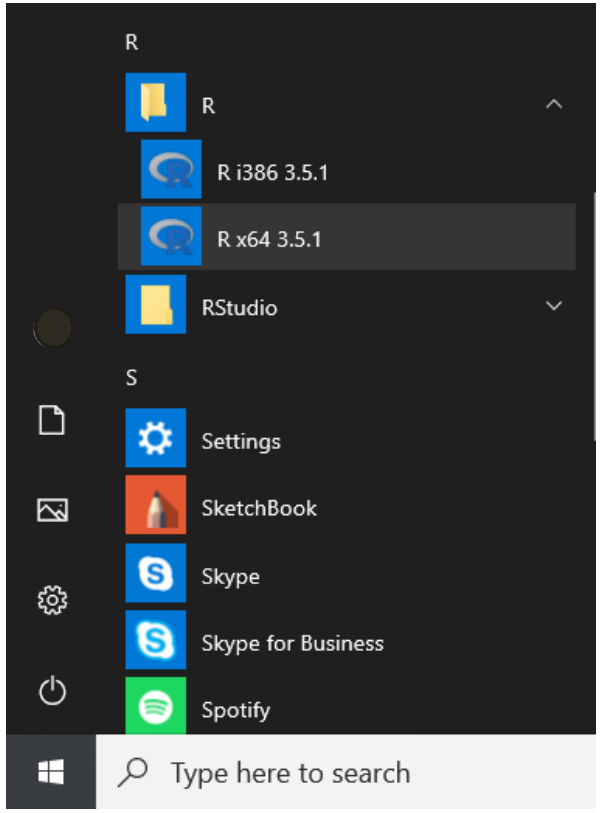


> DOWNLOADING R

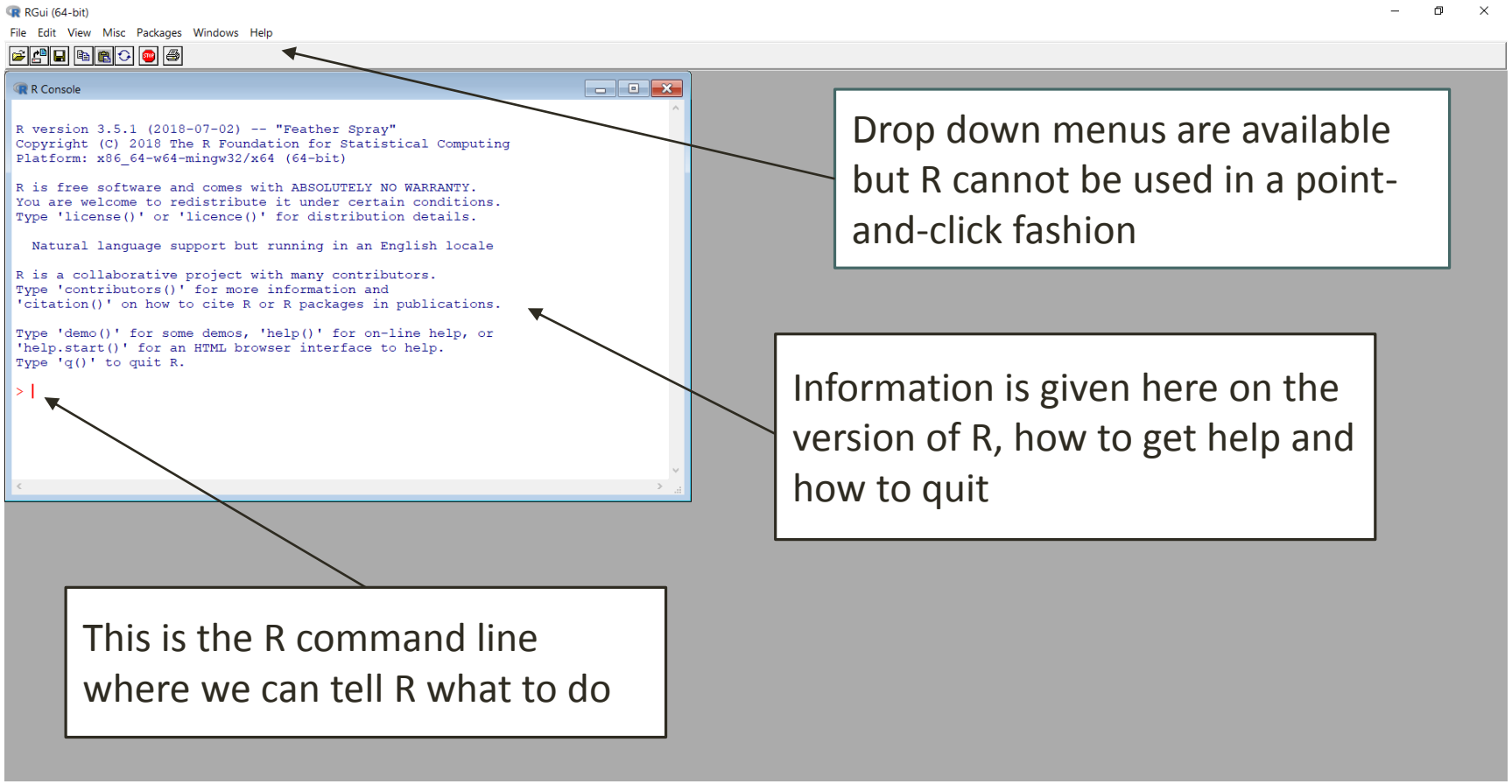
- Visit https://www.r-project.org and click “Download R”
- Choose your nearest CRAN mirror (such as http://www.stats.bris.ac.uk/R)
- Choose “Download R for [Windows/Mac/Linux]”
- Choose “base” for Windows and click on “Download R 3.5.2 for Windows”
- Choose “R-3.5.2.pkg” for Mac
- Once the .exe (Windows) or .pkg (Mac) have downloaded, run and install

> OPENING R

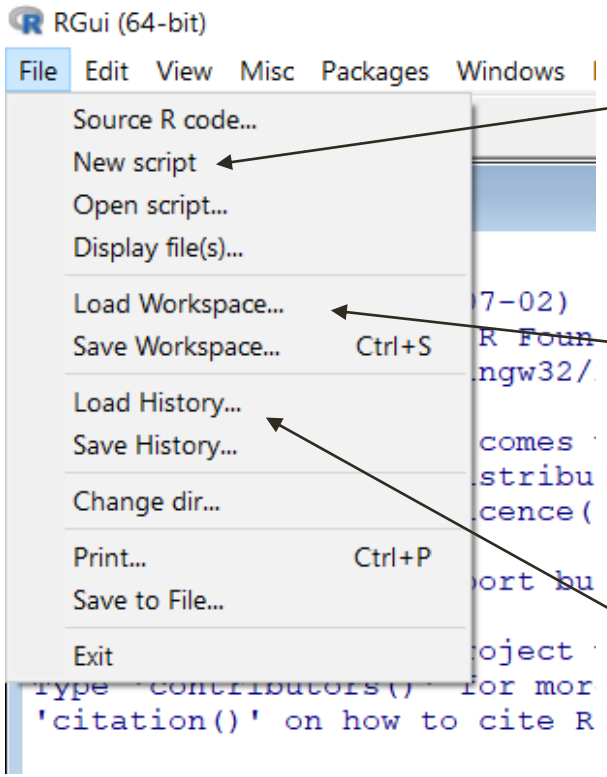
- Start → All Programs → R → R x64 3.5.2.R



> R


RGui (64-bit)
File Edit View Misc Packages Windows Help
R Console
R version 3.5.1 (2018-07-02) -- "Feather Spray"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-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.
>
Drop down menus are available but R cannot be used in a point-and-click fashion
Information is given here on the version of R, how to get help and how to quit
This is the R command line where we can tell R what to do

> R MENUS



A script is a text editor for your code (like a do file in Stata)

A workspace includes all the objects which are in R's memory at a given time. These can include data or results. To see what is in the workspace type ls()

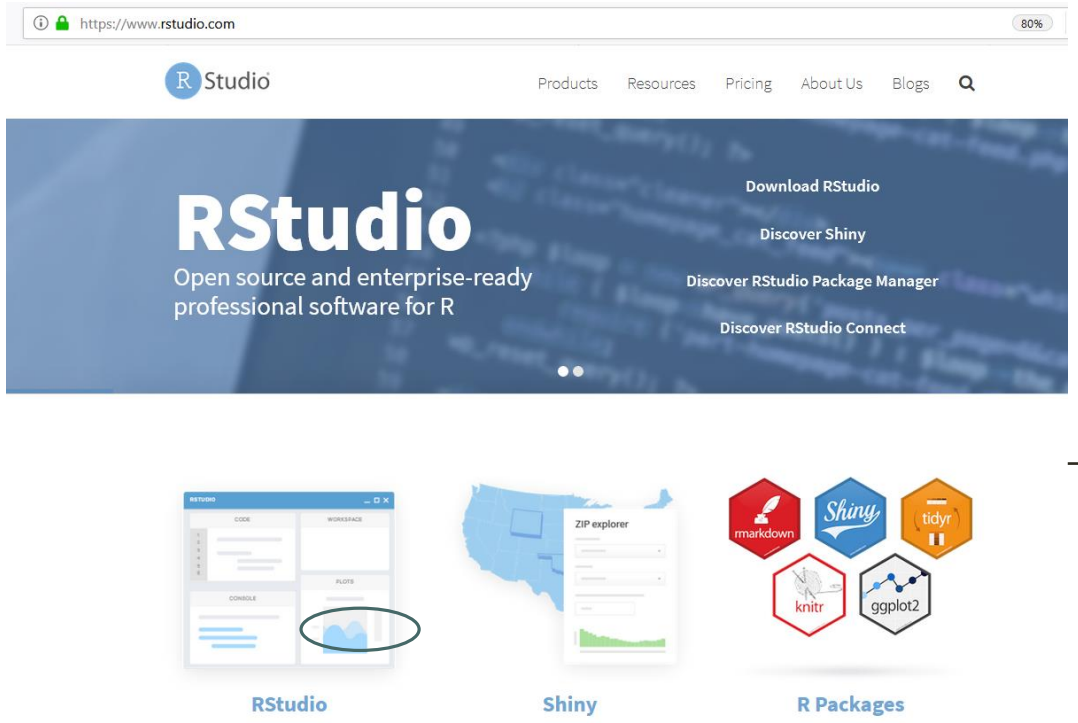
The History includes all the commands and results which are displayed during an R session

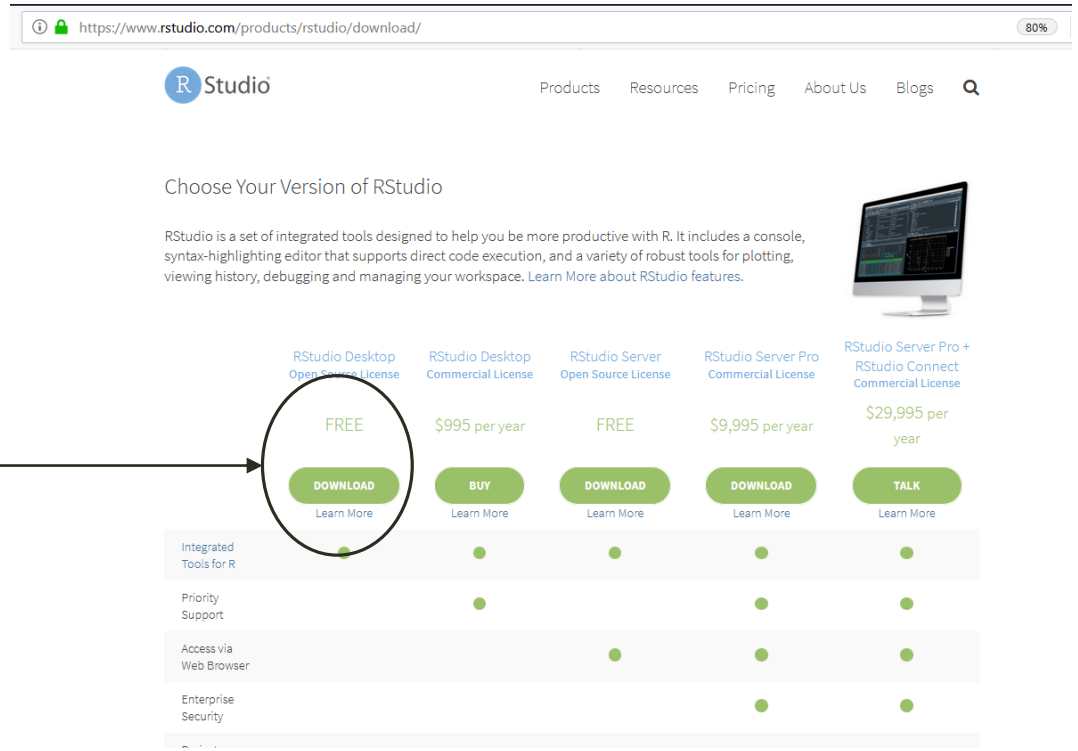
> RSTUDIO

- RStudio makes R easier to use
- It includes a code editor, debugging & visualization tools
- https://www.rstudio.com

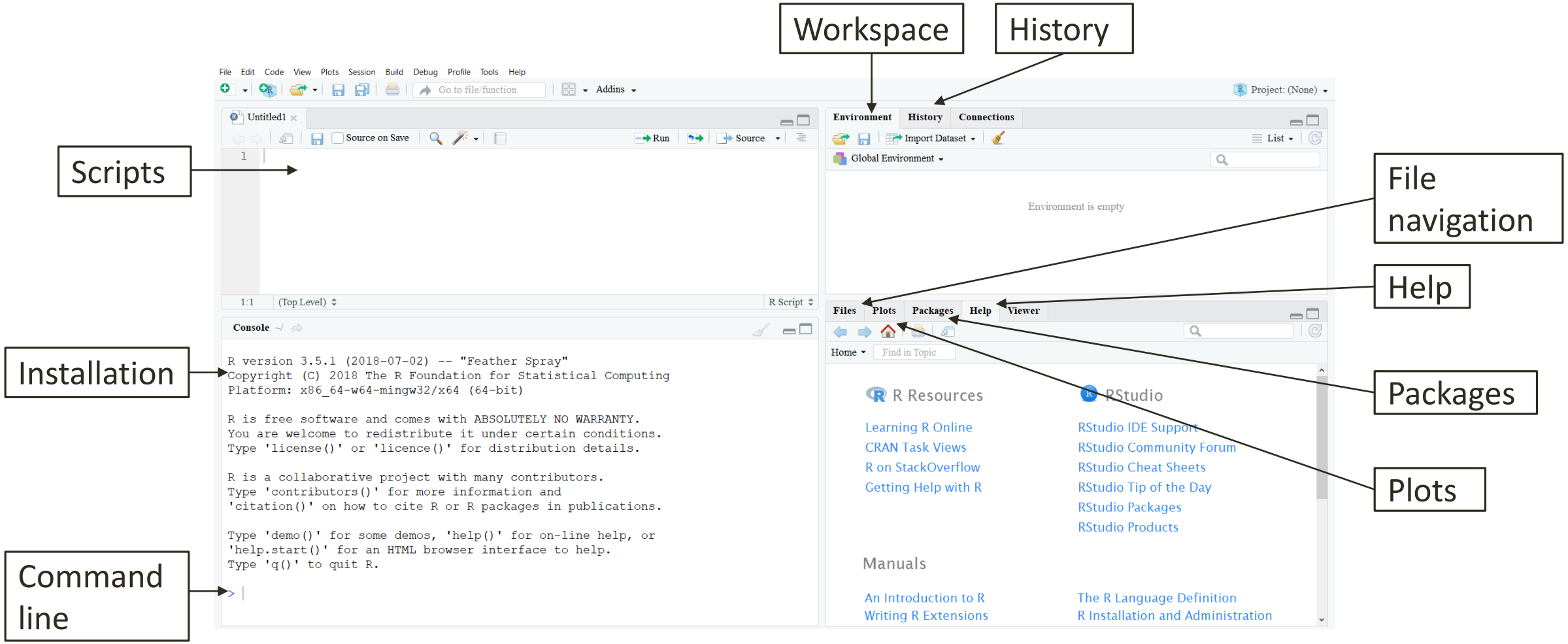
> DOWNLOADING RSTUDIO

- https://www.rstudio.com

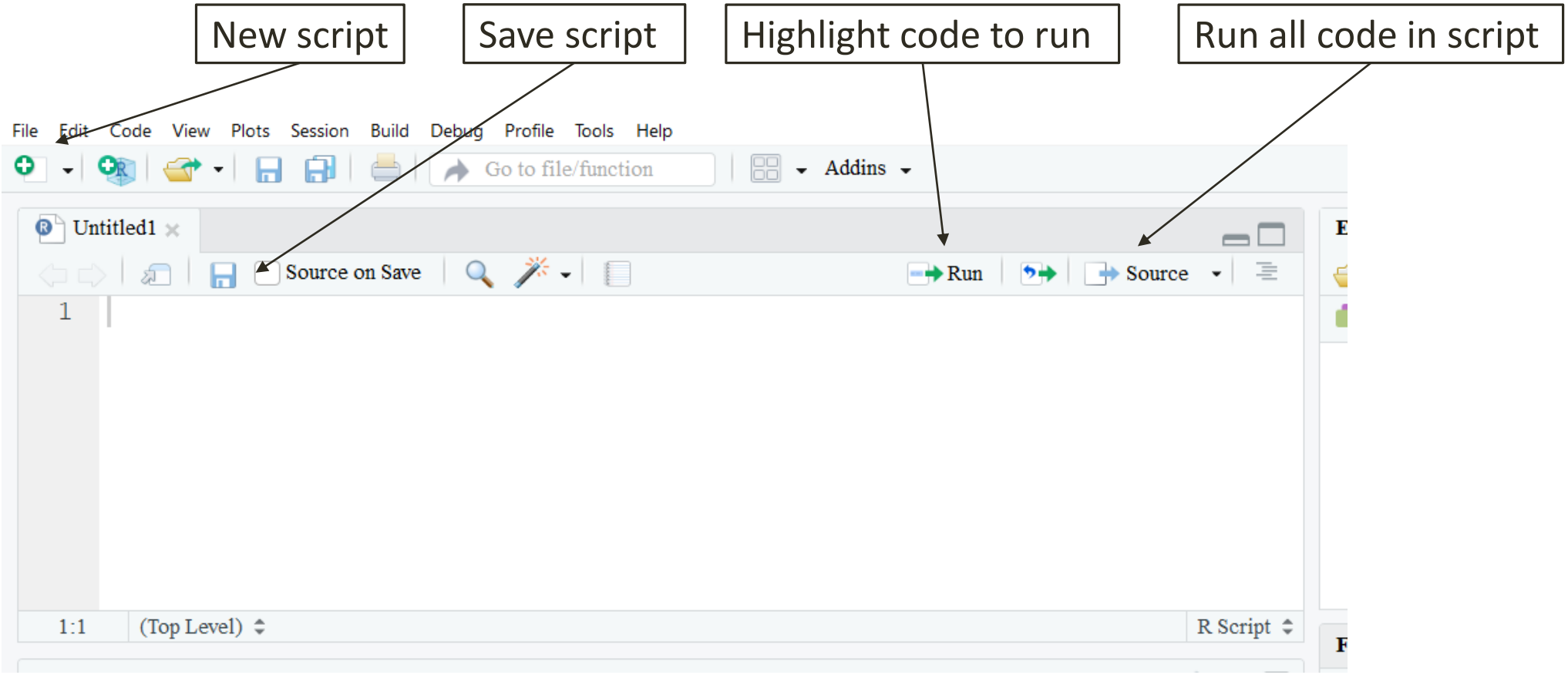


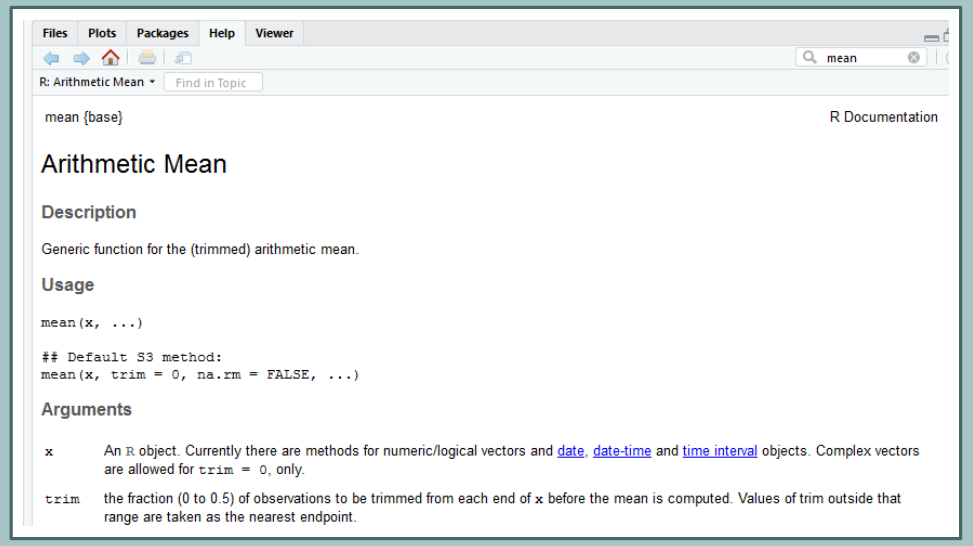


> USING RSTUDIO



> R SCRIPTS


New script
Save script
Highlight code to run
Run all code in script
File
Edit
Code
View
Plots
Session
Build
Debug
Profile
Tools
Help
Go to file/function
Addins
Untitled1
Source on Save
Run
Source
1
1:1
(Top Level)
R Script

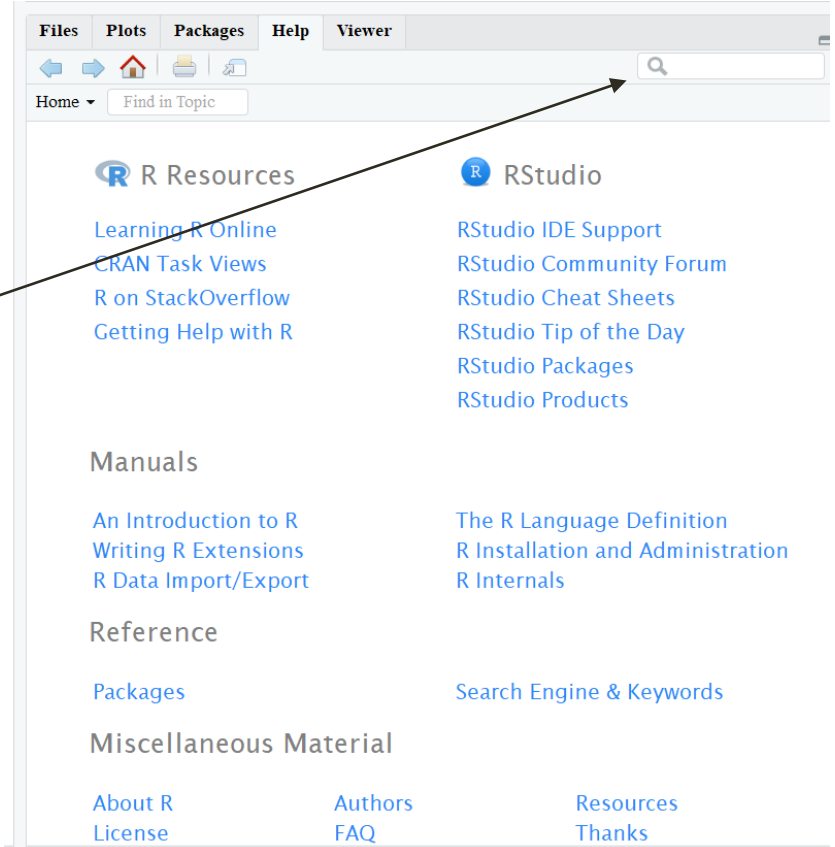


GETTING HELP

James Staley
james.staley@bristol.ac.uk

> HELP

- Use the command *help(topic)* or *?topic*
- In R this will open a web browser
- In RStudio this will open the Help tab can also use the search bar)



> HELP(MEAN)

Input and options of *mean*

Arguments of *mean*

The output of *mean*

Examples

Files | Plots | Packages | Help | Viewer

mean

R: Arithmetic Mean | Find in Topic

mean {base} R Documentation

Arithmetic Mean

Description

Generic function for the (trimmed) arithmetic mean.

Usage

```
mean(x, ...)

## Default S3 method:
mean(x, trim = 0, na.rm = FALSE, ...)
```

Arguments

`x`	An `R` object. Currently there are methods for numeric/logical vectors and date, date-time and time interval objects. Complex vectors are allowed for `trim = 0`, only.
`trim`	the fraction (0 to 0.5) of observations to be trimmed from each end of `x` before the mean is computed. Values of trim outside that range are taken as the nearest endpoint.
`na.rm`	a logical value indicating whether `NA` values should be stripped before the computation proceeds.
`...`	further arguments passed to or from other methods.

Value

If `trim` is zero (the default), the arithmetic mean of the values in `x` is computed, as a numeric or complex vector of length one. If `x` is not logical (coerced to numeric), numeric (including integer) or complex, `NA_real_` is returned, with a warning.

If `trim` is non-zero, a symmetrically trimmed mean is computed with a fraction of `trim` observations deleted from each end before the mean is computed.

References

Becker, R. A., Chambers, J. M. and Wilks, A. R. (1988) *The New S Language*. Wadsworth & Brooks/Cole.

See Also

`weighted.mean`, `mean.POSIXct`, `colMeans` for row and column means.

Examples

```
x <- c(0:10, 50)
xm <- mean(x)
c(xm, mean(x, trim = 0.10))
```

> ERROR MESSAGES

- R provides informative errors

```
> 10*a
Error: object 'a' not found
```

Command which caused the error

Error message returned by R

> ONLINE HELP

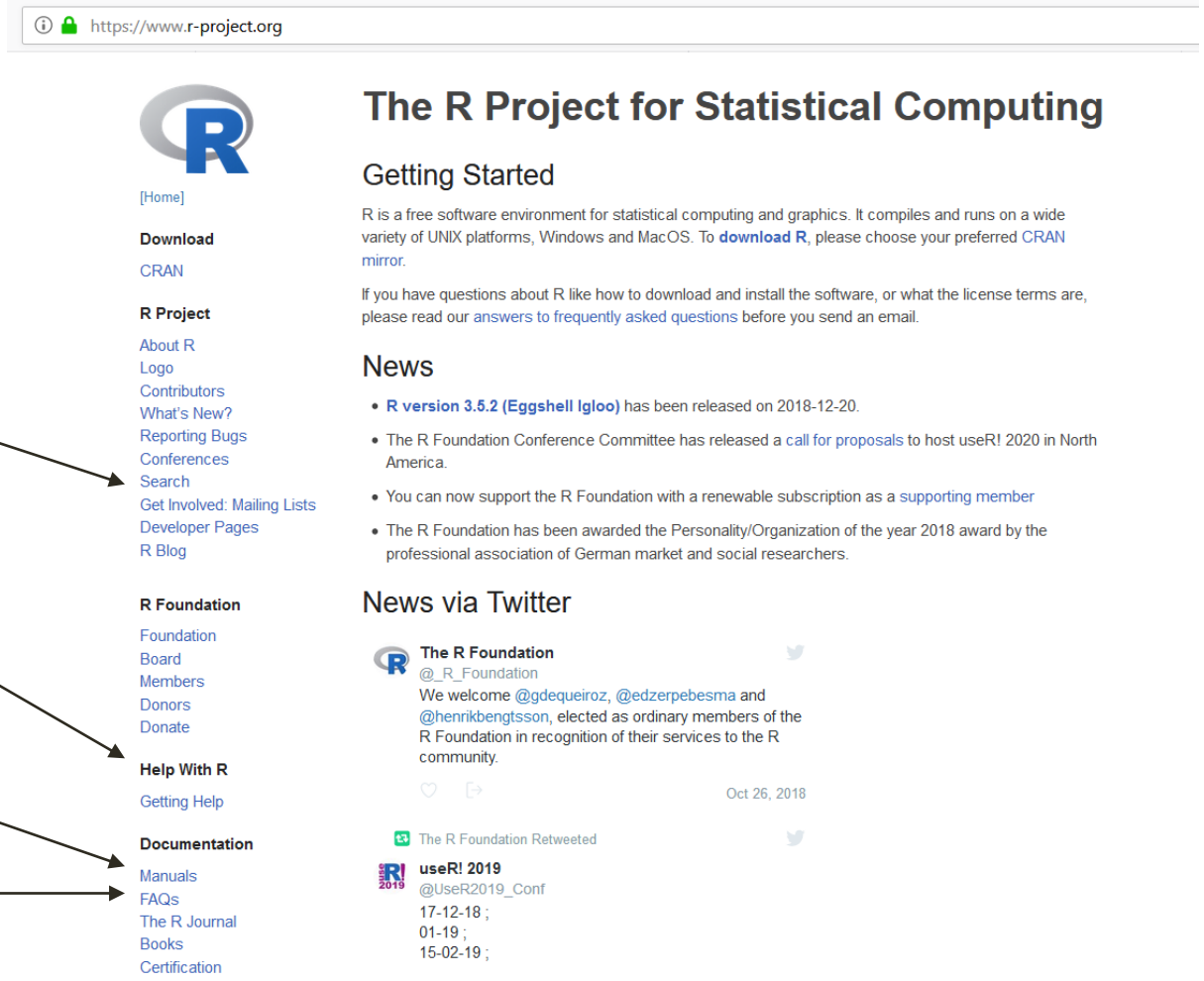
- R website: www.r-project.org

Directs you towards user forums for R

Directs you towards ways to get help in R

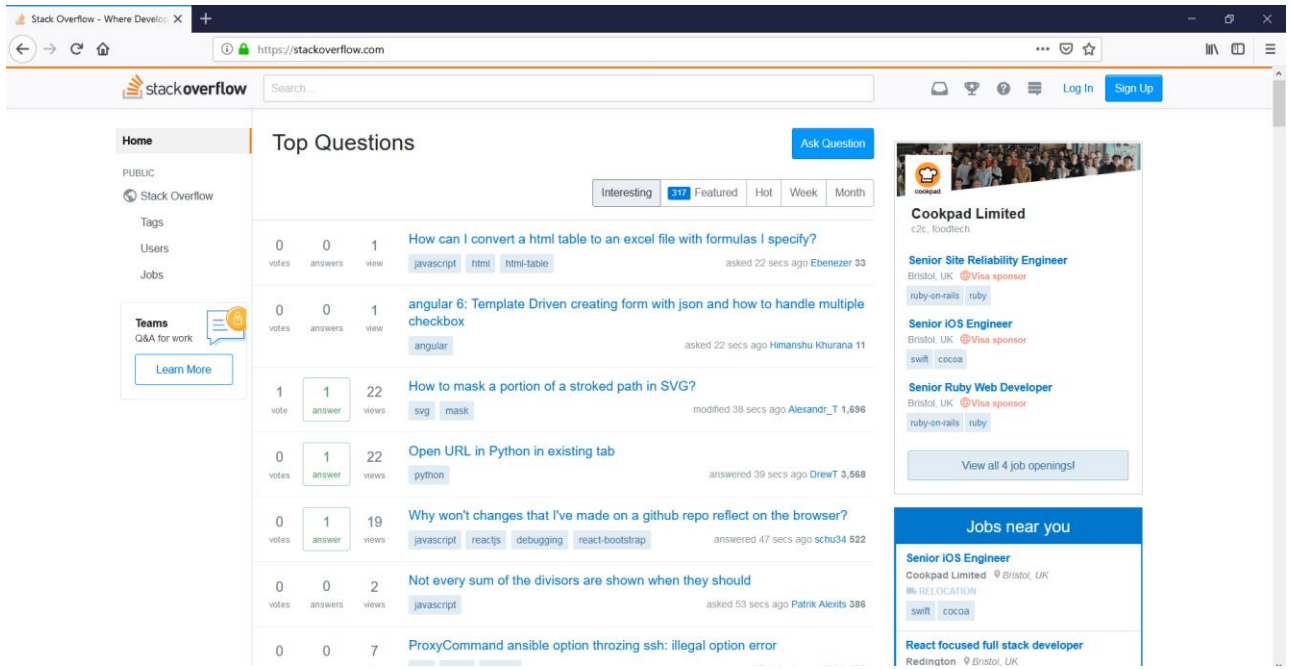
Several introductory manuals are available

List of FAQs



> FORUMS

- Mailing list archive and forum http://r.789695.n4.nabble.com
- Stack Overflow https://stackoverflow.com
- Google!



```
> x <- 11:20
> x [1] 11 12 13 14 15 16 17 18 19 20
>
>
> x[3]
[1] 13
>
>
> x[-2]
[1] 11 13 14 15 16 17 18 19 20
```

BASICS

James Staley
james.staley@bristol.ac.uk

> BRACKETS

- () are used by R functions and contain the options or arguments required to run that function
- [] are used to pick out elements, rows or columns of R objects such as matrices and vectors
- {} are used to hold a series of commands, e.g. within a user written function or a loop

> VECTORS

```
> x <- 11:20
> x
[1] 11 12 13 14 15 16 17 18 19 20
>
>
> x[3]
[1] 13
>
>
> x[-2]
[1] 11 13 14 15 16 17 18 19 20
>
>
> x[1:3]
[1] 11 12 13
```

The 3rd element of x

All except the 2nd element of x

The first 3 elements of x

> VECTOR ELEMENTS

```
> x[c(1, 5, 7)]
[1] 11 15 17
>
>
> x[-c(1, 5, 7)]
[1] 12 13 14 16 18 19 20
>
>
> x[x>13]
[1] 14 15 16 17 18 19 20
```

The 1st, 5th and 7th elements of x

All except the 1st, 5th and 7th element s of x

Those elements of x which are greater than 13

> SEQUENCES

```
> x<-seq(0, 100, by=5)
> x
[1] 0 5 10 15 20 25 30 35 40 45 50
[12] 55 60 65 70 75 80 85 90 95 100
>
>
> length(x)
[1] 21
>
>
> x<-seq(0, 100, length=5)
> x
[1] 0 25 50 75 100
```

x is a sequence from 0 to 100, in steps of 5

Returns the length of x, i.e. how many elements are in x

x is a sequence from 0 to 100 which has 5 elements (i.e. has length 5)

> REPLICATE ELEMENTS

```
> x<-c(1, 1, 1, 1, 1)
> x
[1] 1 1 1 1 1
>
>
> x<-rep(1, 5)
> x
[1] 1 1 1 1 1
```

x is the combination of 1 given five times

x is the number 1 replicated 5 times, using the *rep* function

> CREATING DATA

```
> y<-c(rep("A", 2), rep("B", 2), "C")
> y
[1] "A" "A" "B" "B" "C"
>
>
> y<-c(rep(c("A", "B"), each=2), "C")
> y
[1] "A" "A" "B" "B" "C"
>
>
> y<-c(rep(c("A", "B"), times=2), "C")
> y
[1] "A" "B" "A" "B" "C"
```

y is the combination of the letter A replicated 2 times, the letter B replicated 2 times and the letter C

rep can also be used to replicated combinations of letters, either using each or times

> COMBINING DATA

```
> z<-c(x, y)
> z
[1] "1" "1" "1" "1" "1" "A" "B" "A" "B" "C"
> z<-cbind(x, y)
> z
      x   y
[1,] "1" "A"
[2,] "1" "B"
[3,] "1" "A"
[4,] "1" "B"
[5,] "1" "C"
> z<-rbind(x, y)
> z
  [,1] [,2] [,3] [,4] [,5]
x "1"  "1"  "1"  "1"  "1"
y "A"  "B"  "A"  "B"  "C"
```

Combine x and y as a vector z

Combine x and y as columns into a matrix z

Combine x and y as rows into a matrix z

> MATRICES

```
> z<-matrix(c(x, y), nrow=5)
> z
     [,1] [,2]
[1,] "1"  "A"
[2,] "1"  "B"
[3,] "1"  "A"
[4,] "1"  "B"
[5,] "1"  "C"
> d_z<-dim(z)
> d_z
[1] 5 2
> t_z<-t(z)
> t_z
     [,1] [,2] [,3] [,4] [,5]
[1,] "1"  "1"  "1"  "1"  "1"
[2,] "A"  "B"  "A"  "B"  "C"
```

Explicitly create a matrix z, combining x and y and specifying 5 rows. Note that R fills matrices column-wise unless specified using *byrow=TRUE*.

What are the dimensions of matrix z (rows then columns)

Transpose the matrix z

> MATRIX ELEMENTS

```
> z
     [,1] [,2]
[1,] "1"  "A"
[2,] "1"  "B"
[3,] "1"  "A"
[4,] "1"  "B"
[5,] "1"  "C"
>
> z[1, 2]
[1] "A"
>
> z[2,]
[1] "1" “B"
>
> z[,-2]
[1] "1"  "1"  "1"  "1"  "1"
```

Element in the 1st row and 2nd column of z

All elements in the 2nd row of z

Remove the second column

> MATRIX ROWS & COLUMNS

```
> z[1:3,]
     [,1] [,2]
[1,] "1"  "A"
[2,] "1"  "B"
[3,] "1"  "A"
>
> z[c(1, 4),]
     [,1] [,2]
[1,] "1"  "A"
[2,] "1"  "B"
>
> z[-c(1, 4),]
     [,1] [,2]
[1,] "1"  "B"
[2,] "1"  "A"
[3,] "1"  "C"
```

First 3 rows of z

1st and 4th rows of z

Remove the 1st and 4th rows of z

> DATA FRAMES

```
> z <- data.frame(x=x, y=y)
> z[1:3,]
  x y
1 1 A
2 1 A
3 1 B
> z$x
[1] 1 1 1 1 1
```

Create a data frame with variables x and y

Use $ to access the x column of z

> LISTS

```
> x <- 1
> y <- diag(2)
> w <- list(x=x, y=y)
> w
$x
[1] 1
$y
     [,1] [,2]
[1,]    1    0
[2,]    0    1
```

Create an array of data using R lists

Use $ to access the a component of w

> FUNCTIONS

- Are commands
- Are of the form: *function*(*input, options*)
- Examples: *help*, *cor*, *lm*
- Users can write their own functions, e.g. a function for multiplication:

```
> multiply <- function(a, b){
+   res <- a*b
+   return(res)
+ }
> multiply(2,3)
[1] 6
```

> LOGICAL STATEMENTS

Command	Operator
And	&
Or	|
Not	!
Is equal to	==
Is not equal to	!=
Is less than	<
Is less than or equal to	<=
Is greater than	>
Is greater than or equal to	>=

> CLASS STATEMENTS

Command	Operator
Are any values in x missing?	is.na(x)
Are any values in x not missing?	!is.na(x)
Is the variable x numeric?	is.numeric(x)
Is the variable x a string?	is.character(x)
Is the variable x a factor?	is.factor(x)
Coerce the variable x to be a factor	as.factor(x)

> OTHER USEFUL FUNCTIONS

Command	Operator
Addition, subtraction, multiplication and division	+, -, *, /
Matrix operations: transpose, inverse, multiplication	t, solve, %*%
List R objects in session	ls
Class, length and dimensions	class, length, dim
Paste	paste
Row and column names (matrices)	rownames, colnames
Row and column names (data frames)	row.names, names
Combine, row bind and column bind	c, rbind, cbind
Subset dataset	subset

```
> read.csv("bmi.csv")
   id age  bmi  sex diet
1   1 32.0 25.0 M    0
2   2 35.0 32.0 M    0
3   3 41.0 27.0 M    0
4   4 29.0 29.0 M    0
5   5 33.5 28.0 M    1
6   6 33.2 29.1 M    1
7   7 32.9 29.4 M    1
8   8 32.6 29.7 F    1
9   9 32.3 30.0 F    1
```

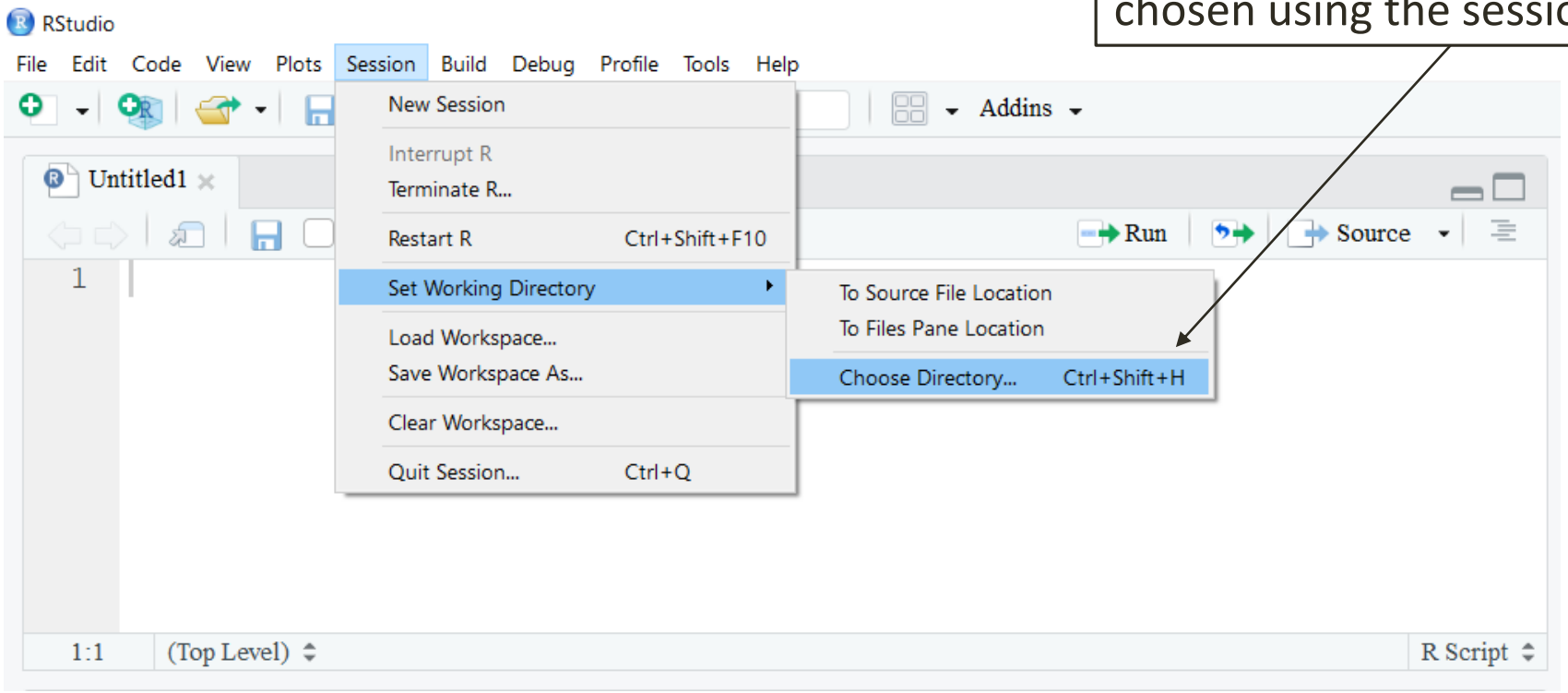
READING AND WRITING DATA

James Staley
james.staley@bristol.ac.uk

> SETTING THE WORKING DIRECTORY

```
> setwd("O:/Documents")
```

The working directory can also be chosen using the session tab



> READING DATA - CSV

```
> read.csv("bmi.csv")
   id age  bmi  sex diet
1   1 32.0 25.0 M   0
2   2 35.0 32.0 M   0
3   3 41.0 27.0 M   0
4   4 29.0 29.0 M   0
5   5 33.5 28.0 M   1
6   6 33.2 29.1 M   1
7   7 32.9 29.4 M   1
8   8 32.6 29.7 F   1
9   9 32.3 30.0 F   1
10 10 32.0 30.3 F   1
…
> data_bmi<-read.csv("bmi.csv")
> data_bmi<-read.table("bmi.csv", header=T, sep=",")
```

read.csv will read and display the bmi data but not store it

To allow us to use the bmi data, we assign the .csv file to an object called data_bmi

read.table can also read in CSV files and is more generic than read.csv, e.g. can open tab-delimited text files

> USING THE DATA

```
> data_bmi[, 3]
[1] 25.0 32.0 27.0 29.0 28.0 29.1 29.4 29.7 30.0 30.3
[11] 30.6 30.9 31.2 31.5 31.8 32.1 32.4 32.7 33.0 33.3
>
> data_bmi[, "bmi"]
[1] 25.0 32.0 27.0 29.0 28.0 29.1 29.4 29.7 30.0 30.3
[11] 30.6 30.9 31.2 31.5 31.8 32.1 32.4 32.7 33.0 33.3
>
>
> data_bmi$bmi
[1] 25.0 32.0 27.0 29.0 28.0 29.1 29.4 29.7 30.0 30.3
[11] 30.6 30.9 31.2 31.5 31.8 32.1 32.4 32.7 33.0 33.3
```

Access the 3rd column of data_bmi data

Access the bmi column from data_bmi

Use $ to access the bmi column of data_bmi

> WRITING DATA - CSV

```
> data_bmi<-read.csv("bmi.csv")
> data_bmi[1:5,]
   id age  bmi  sex diet
1   1 32.0 25.0 M   0
2   2 35.0 32.0 M   0
3   3 41.0 27.0 M   0
4   4 29.0 29.0 M   0
5  5 33.5 28.0 M   1
> write.csv(data_bmi, "data_bmi.csv", row.names=F, quote=F)
> write.table(data_bmi, "data_bmi.csv", row.names=F, quote=F, sep=",")
```

Read in the .csv and assign to object data_bmi

Save the object data_bmi to a .csv file at the given path

> READING & WRITING DATA - STATA

- The *foreign* package allows R to read and write Stata 11 and 12 datasets
- There are other packages that read and write datasets for Stata 13, 14 and 15
- *read.dta* and *write.dta* are the Stata dataset equivalents of *read.csv* and *write.csv*

```
> cor(data_bmi$bmi, data_bmi$age)
[1] -0.56091
>
>
> lm(bmi~age, data=data_bmi)
Call:
lm(formula = bmi ~ age, data = data_bmi)

Coefficients:
(Intercept)          age
   44.5249      -0.4419
```

STATISTICAL ANALYSIS

James Staley
james.staley@bristol.ac.uk

> BASIC FUNCTIONS

```
> mean(data_bmi$bmi)
[1] 30.45
> median(data_bmi$bmi)
[1] 30.75
> sd(data_bmi$bmi)
[1] 2.124419
> min(data_bmi$bmi)
[1] 25
> max(data_bmi$bmi)
[1] 33.3
> quantile(data_bmi$bmi, probs=0.25)
25% 29.325
```

Mean

Median

Standard deviation

Minimum

Maximum

Lower quartile

> SUMMARISE DATA

summary can be used on a variable or the whole dataset

```
> summary(data$bmi)
Min. 1st Qu. Median Mean 3rd Qu. Max.
25.00 29.32 30.75 30.45 32.02 33.30
>
> summary(data)
     id              age              bmi          sex         diet
min.   : 1.00    Min. :29.00    Min. :25.00    F:13    Min. :0.00
1st Qu.: 5.75  1st Qu.:30.12  1st Qu.:29.32    M: 7    1st Qu.:0.00
Median :10.50  Median :31.55  Median :30.75            Median :1.00
Mean   :10.50    Mean :31.85    Mean :30.45            Mean :0.55
3rd Qu.:15.25  3rd Qu.:32.67  3rd Qu.:32.02            3rd Qu.:1.00
Max.   :20.00    Max. :41.00    Max. :33.30            Max. :1.00
>
> table(data$sex)
 F  M
13  7
```

> CORRELATION AND REGRESSION

```
> cor(data$bmi, data$age)
[1] -0.56091
>
>
> model <- lm(bmi~age, data=data)
> model
Call:
lm(formula = bmi ~ age, data = data)

Coefficients:
(Intercept)          age
   44.5249      -0.4419
```

Pearson correlation between bmi and age

Linear regression for outcome bmi and exposure age

Intercept and slope estimates from the regression

> SUMMARISE REGRESSION MODELS

summary gives you more information on the regression model

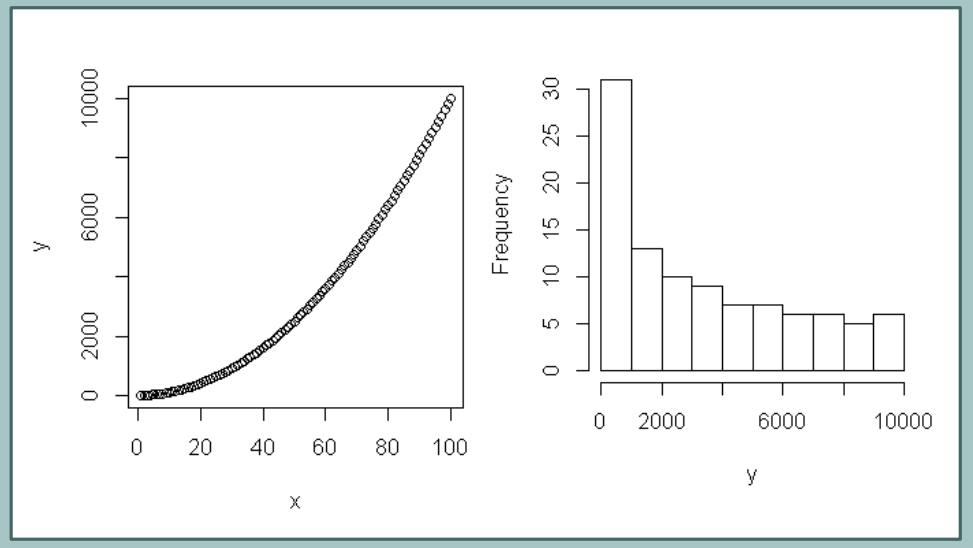
```
> summary(model)
Call:
lm(formula = bmi ~ age, data = data)

Residuals:
Min 1Q Median 3Q Max
-5.3837 -0.4604 0.3349 0.9627 2.9420

Coefficients:
            Estimate   Std. Error   t value   Pr(>|t|)
(Intercept) 44.5249    4.9131       9.063     3.97e-08 ***
age         -0.4419    0.1537      -2.875     0.0101 *
---
Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

> USING RESULTS

```
> names(model)
[1] "coefficients" "residuals" "effects" "rank"
[5] "fitted.values" "assign" "qr" "df.residual“
[9] "xlevels" "call" "terms" "model"
> names(summary(model))
[1] "call" "terms" "residuals" "coefficients"
[5] "aliased" "sigma" "df" "r.squared"
[9] "adj.r.squared" "fstatistic"    "cov.unscaled"
> model$coefficients
(Intercept)         age
44.524864     -0.441911
```

names function allows us to see the various components within “model”

The $ command allows us to extract a component within “model”



PLOTS

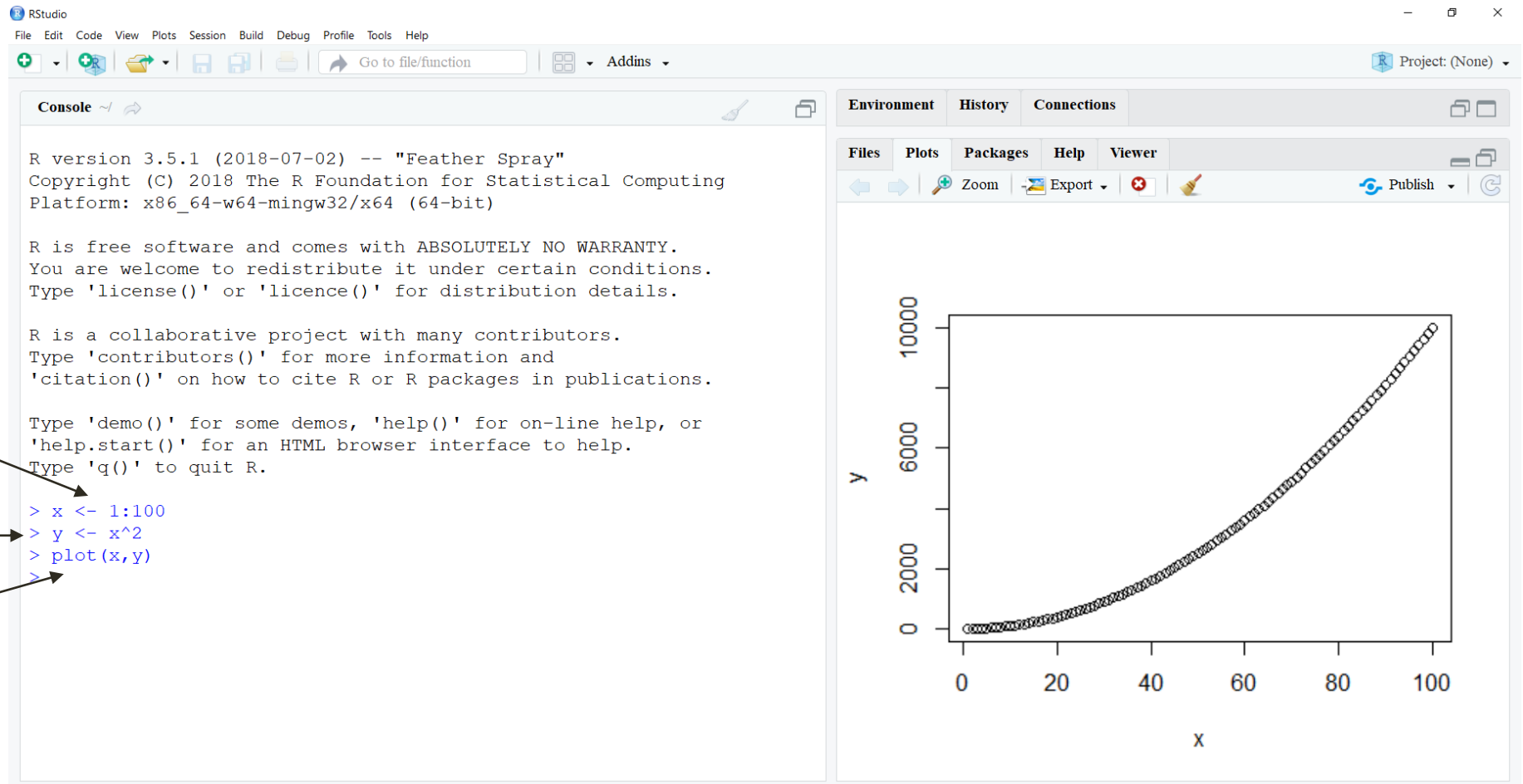
James Staley
james.staley@bristol.ac.uk

> SCATTERPLOT

x is a vector of the numbers 1 to 100

y is x squared

plot(x,y) produces a scatterplot

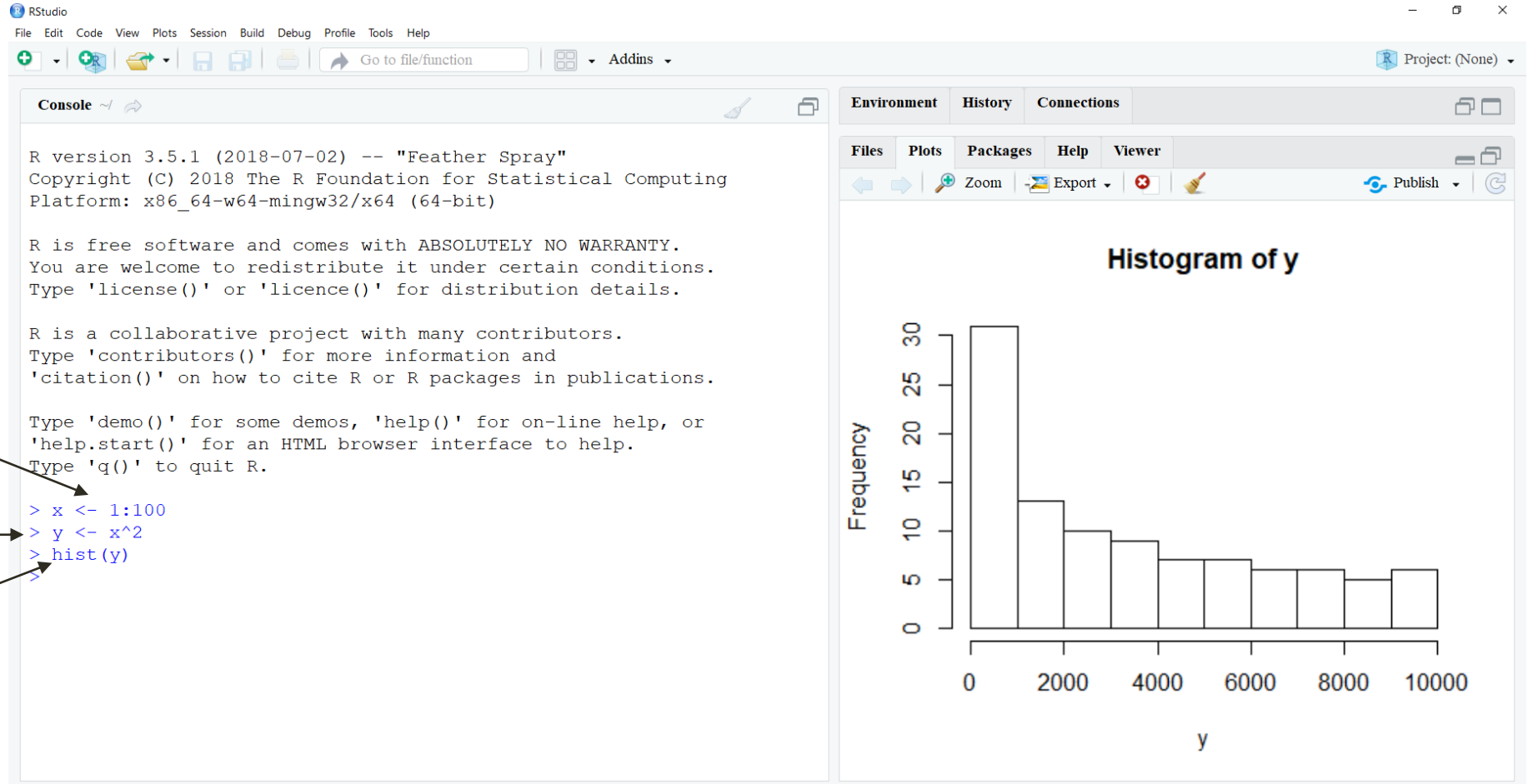


> HISTOGRAM

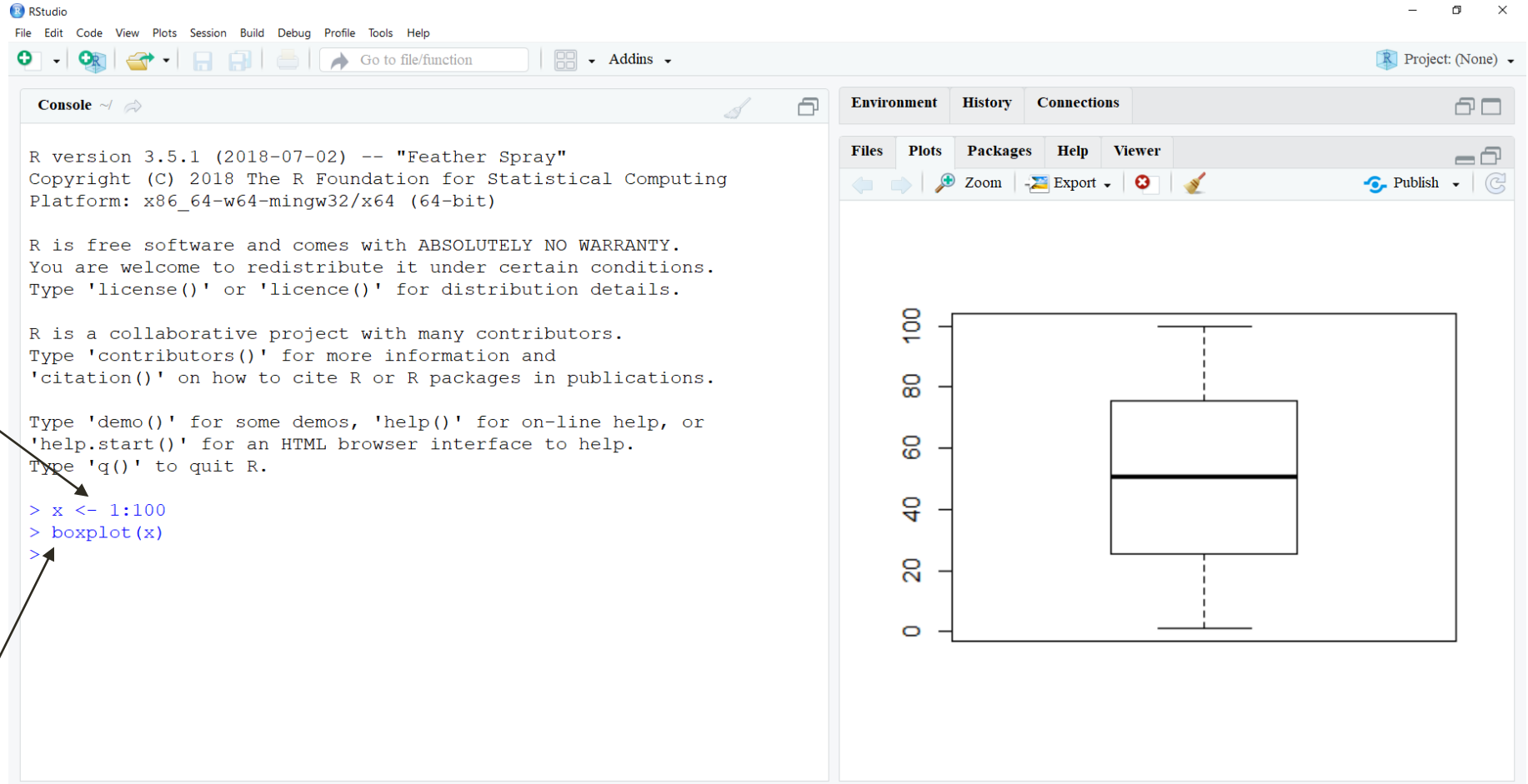
x is a vector of the numbers 1 to 100

y is x squared

hist(x,y) produces a histogram of y



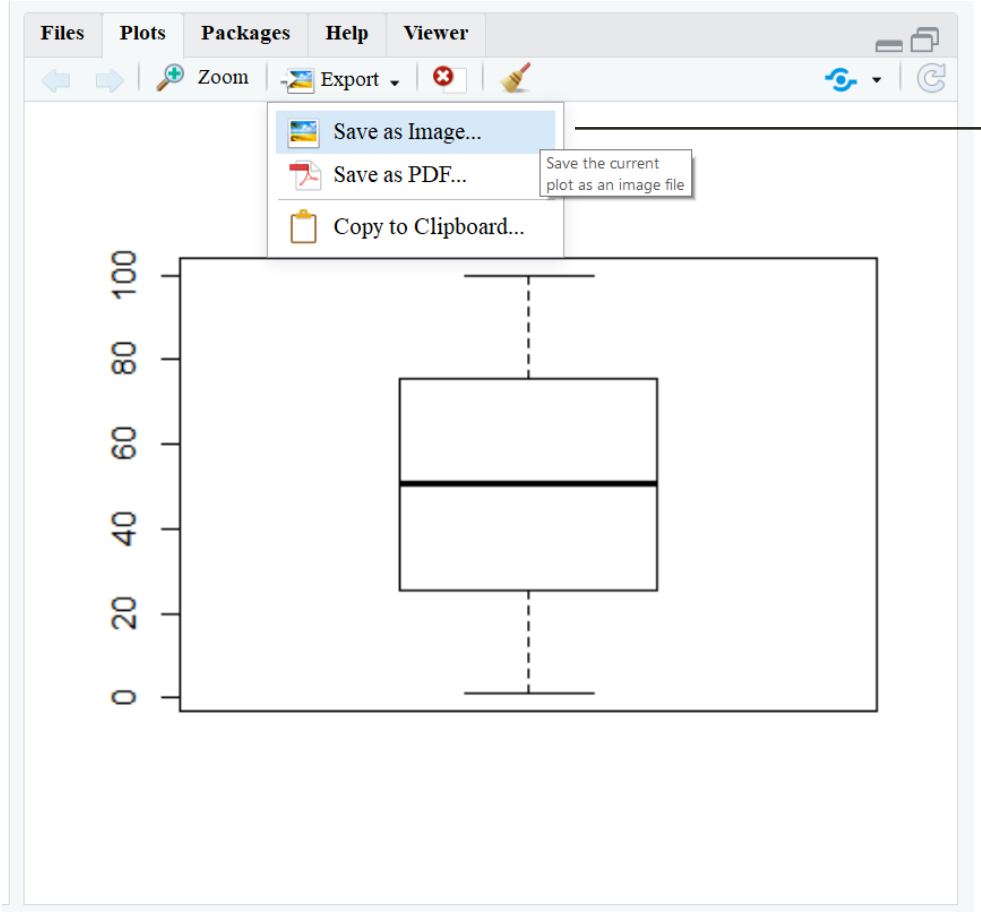
> BOXPLOT

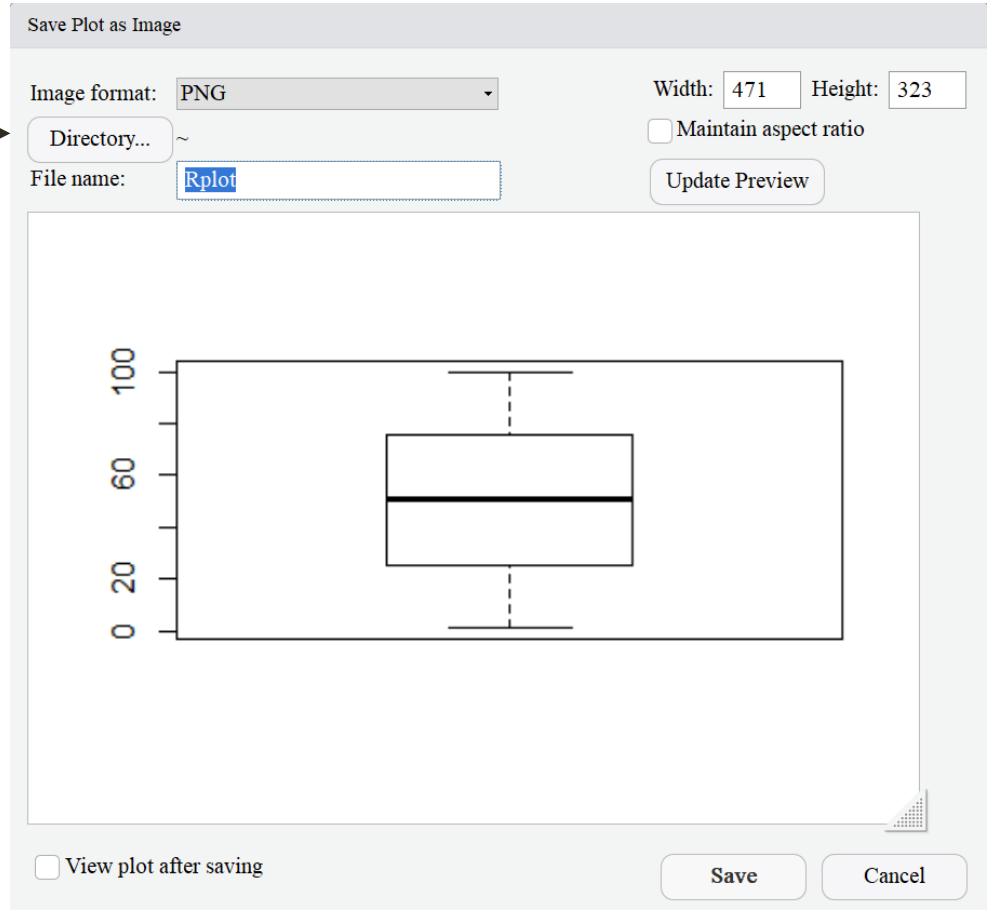


x is a vector of the numbers 1 to 100

boxplot(x) produces a box plot of x

> SAVING PLOTS - DRIECTLY





> SAVING PLOTS - INLINE

```
> x<-1:100
>
> png(filename="example_boxplot.png", width=10, height=10,
units="cm", res=500)
>
> boxplot(x)
>
> dev.off()
null device
       1
```

Opens a device in which to save the plot (in this case a .png file)

Create plot

Once the plot is complete we close the device

```
> library(Hmisc)
Loading required package: lattice
Loading required package: survival
Loading required package: Formula
Loading required package: ggplot2

Attaching package: ‘Hmisc’

The following objects are masked from
‘package:base’:

          format.pval, round.POSIXt,
trunc.POSIXt, units
```

R PACKAGES

James Staley
james.staley@bristol.ac.uk

> INSTALLING PACKAGES

```
> install.packages("Hmisc")
Installing package into 'O:/Documents/R/win-library/3.5'
(as 'lib' is unspecified)
also installing the dependencies 'backports', 'scales', 'checkmate',
      'viridisLite', 'survival', 'ggplot2', 'htmlTable', 'viridis'
.
.
.
package 'Hmisc' successfully unpacked and MD5 sums checked
The downloaded binary packages are in < path >
```

Install packages

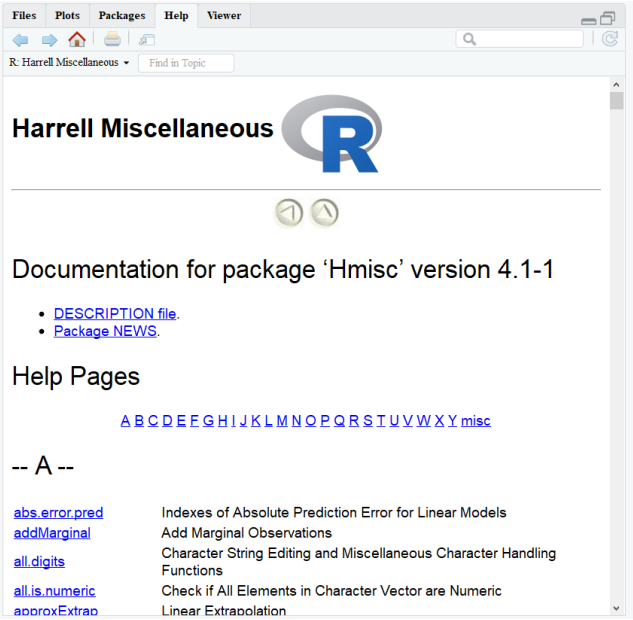
> LOADING PACKAGES

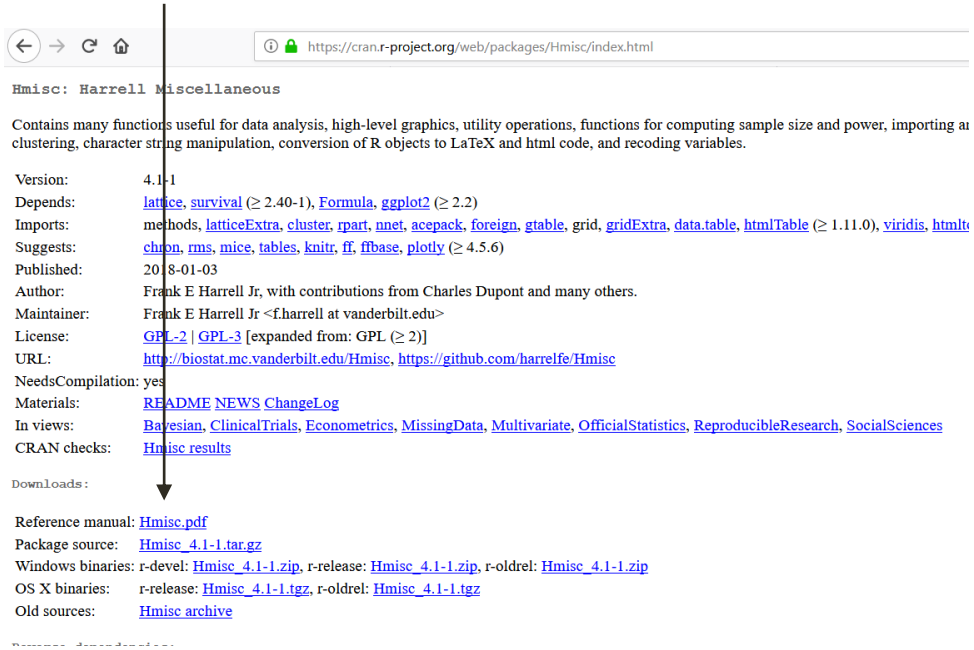
```
> library(Hmisc)
Loading required package: lattice
Loading required package: survival
Loading required package: Formula
Loading required package: ggplot2

Attaching package: ‘Hmisc’

The following objects are masked from ‘package:base’:

       format.pval, round.POSIXt, trunc.POSIXt, units
```

Command to load the *Hmisc* package into our current R session

> HELP FOR PACKAGES

- The *help* function can be used to open the documentation for an R package, e.g. help(package="Hmisc")
- Search the package list on CRAN https://cran.r-project.org



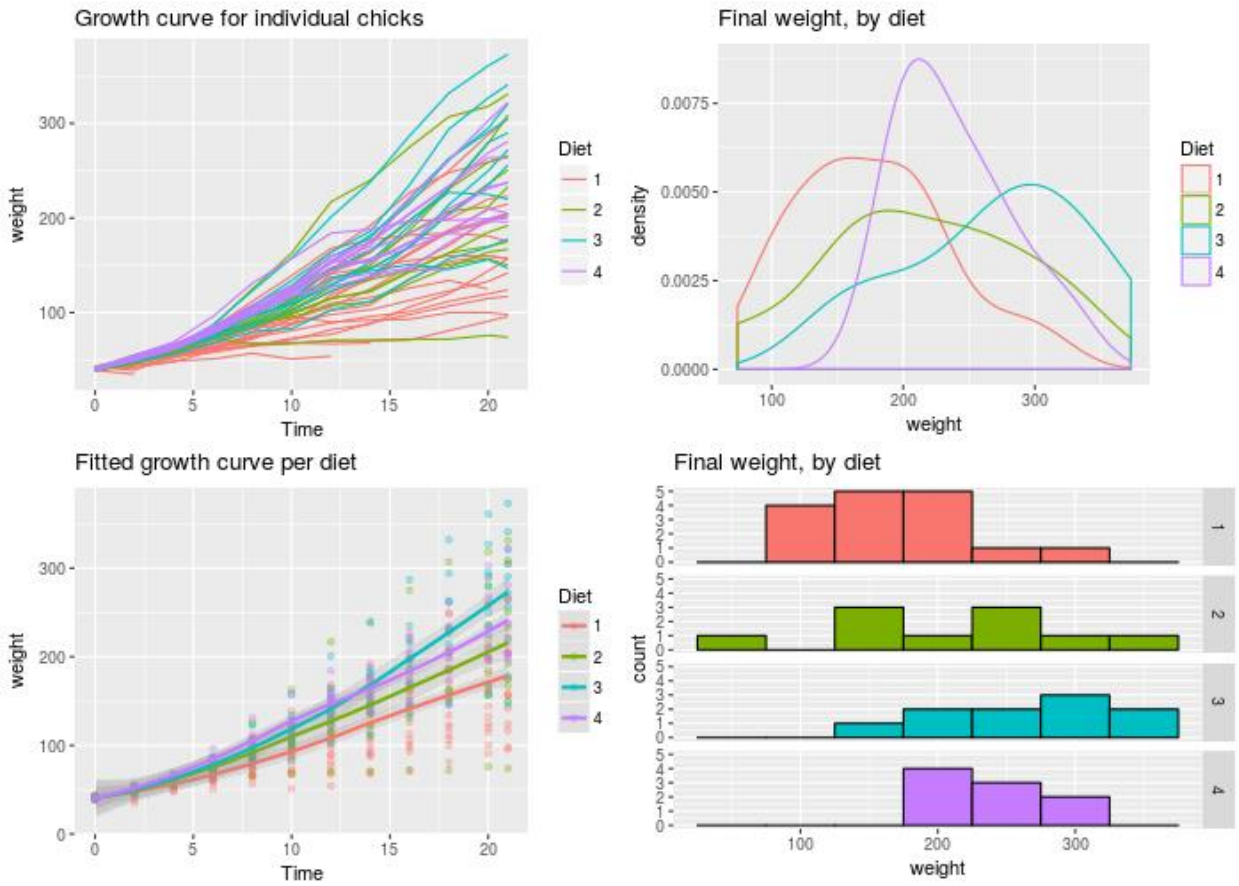


> LEARN R - SWIRL

```
> install.packages("swirl")
> library(swirl)
| Hi! Type swirl() when you are ready to begin.
```

swirl (http://swirlstats.com/students.html) allows you to learn R within R itself

> PLOTS - GGPLOT2



ggplot2 (http://ggplot2.tidyverse.org) allows you to create publication quality plots

> NOTEBOOK INTERFACE - R MARKDOWN

Reusable Version in rCharts

As I mentioned above, this visualization works well with any cumulative growth time series, so let's apply it to the `managers` dataset supplied by the `PerformanceAnalytics` package.

Get Data and Transform

```
#get the data and convert to a format that we would expect from melted xts
#will be typical
#also original only uses a single value (val) and not other
require(reshape2)
require(PerformanceAnalytics)

data(managers)
managers <- na.omit(managers)
managers.melt <- melt(
  data.frame( index( managers ), coredata(cumprod( managers+1 )*100 ) ),
  id.vars = 1
)
colnames(managers.melt) <- c("date", "manager","val")
managers.melt[,"date"] <- format(managers.melt[,"date"],format = "%Y-%m-%d")
```

Draw The Graph

```
require(rCharts)
p2 <- rCharts$new()
p2$setLib('libraries/widgets/nyt_home')
p2$setTemplate(script = "libraries/widgets/nyt_home/layouts/nyt_home.html")

p2$set(
  description = "This data comes from the managers dataset included in the R package PerformanceAnalytics.",
  data = managers.melt,
  groups = "manager"
)
cat(noquote(p2$html()))
```

rmarkdown (http://rmarkdown.rstudio.com) provides an interface to put together narrative text, code and output

> WEB APPS - SHINY

shiny (http://shiny.rstudio.com) allows you to create interactive web applications

Harriet Mills

Andrew Simpkin

ACKNOWLEDGMENTS

James Staley
james.staley@bristol.ac.uk