CSC-20053

Keele UNIVERSITY

Introduction to R

Ram Bajpai

School of Medicine, Keele University

Install R and RStudio

To Install R - Windows

1. Open an internet browser and go to www.r-project.org.
2. Click the "download R" link in the middle of the page under "Getting Started."
3. Select a CRAN location (a mirror site) and click the corresponding link.
4. Click on the "Download R for Windows" link at the top of the page.
5. Click on the "install R for the first time" link at the top of the page.
6. Click "Download R for Windows" and save the executable file somewhere on your computer. Run the .exe file and follow the installation instructions.
7. Now that R is installed, you need to download and install RStudio.

To Install R - Mac

1. Open an internet browser and go to www.r-project.org.
2. Click the "download R" link in the middle of the page under "Getting Started."
3. Select a CRAN location (a mirror site) and click the corresponding link.
4. Click on the "Download R for (Mac) OS X" link at the top of the page.
5. Click on the file containing the latest version of R under "Files."
6. Save the .pkg file, double-click it to open, and follow the installation instructions.
7. Now that R is installed, you need to download and install RStudio.

Keele UNIVERSITY

Install R and RStudio

To Install Rstudio - Windows

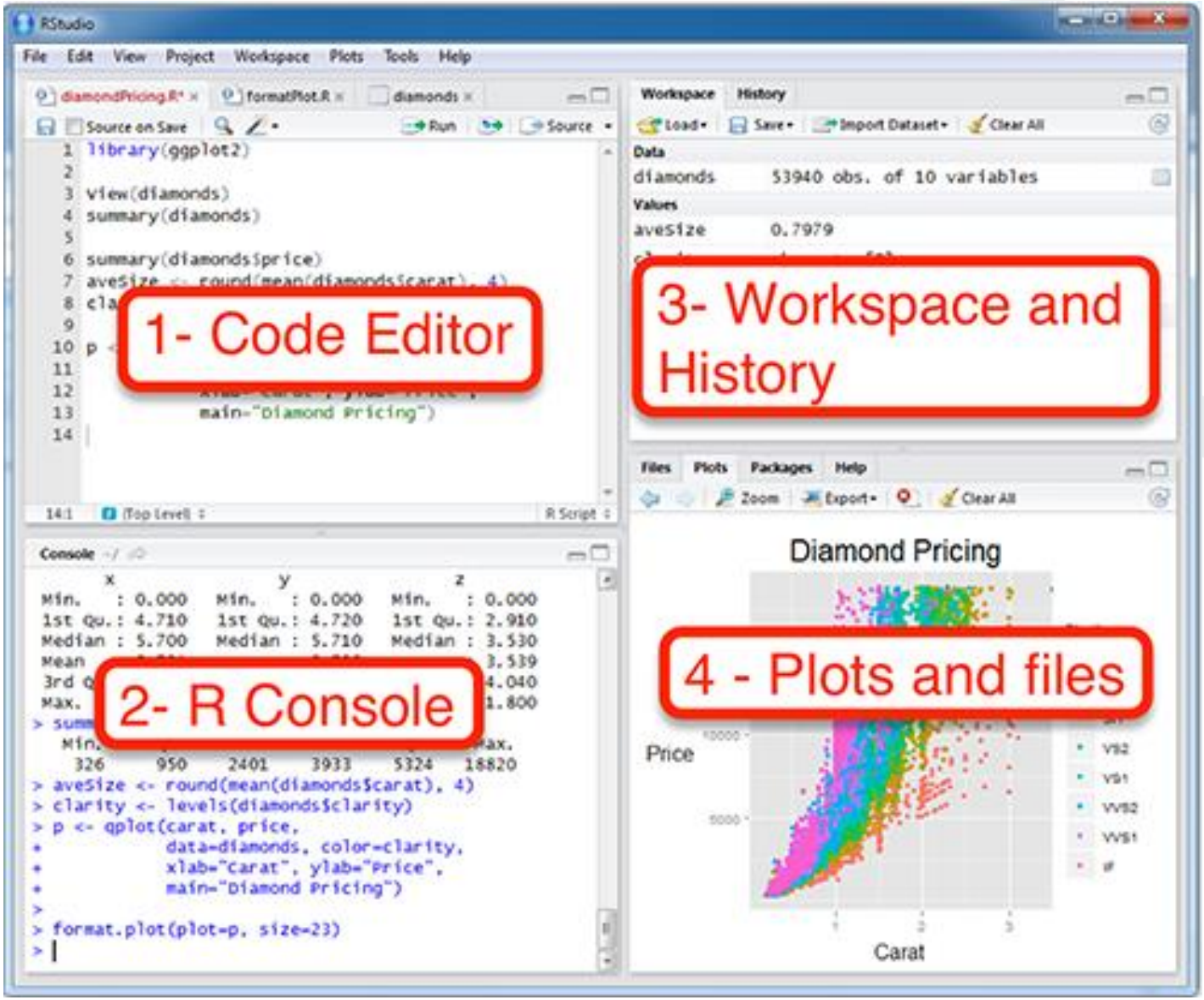
1. Go to www.rstudio.com and click on the "Download RStudio" button.
2. Click on "Download RStudio Desktop."
3. Click on the version recommended for your system, or the latest Windows version, and save the executable file. Run the .exe file and follow the installation instructions.

To Install Rstudio - Mac

1. Go to www.rstudio.com and click on the "Download RStudio" button.
2. Click on "Download RStudio Desktop."
3. Click on the version recommended for your system, or the latest Mac version, save the .dmg file on your computer, double-click it to open, and then drag and drop it to your applications folder.

Keele UNIVERSITY

RStudio



Keele UNIVERSITY

Learning Outcomes

These workshops will provide you with the practical skills in:

- basic programming in R
- data management
- data analytics

Basis for

- Statistical Data Analytics

Keele UNIVERSITY

Background

R

Created in 1992 by Ross Ihaka and Robert Gentleman at the University of Auckland, New Zealand.

- Free (or opensource) statistical programming language
- Functional programming language
- Object oriented

Rstudio

- RStudio is an integrated development environment (IDE) for R.
- It includes a console, syntax-highlighting editor that supports direct code execution, as well as tools for plotting, history, debugging and workspace management.

Keele UNIVERSITY

Basic Data Types

- R provides three core data types
  - Numeric
    - both integer and real numbers
  - Character
    - i.e., text, also called *string*
  - Logic
    - TRUE or FALSE

Keele UNIVERSITY

Numerical Operators

Operator	Meaning	Example	Output
+	Addition	5 + 2	7
–	Substruction	5 – 2	3
*	Multiplication	5 * 2	10
/	Division	5 / 2	2.5
^	Power	5^2	25

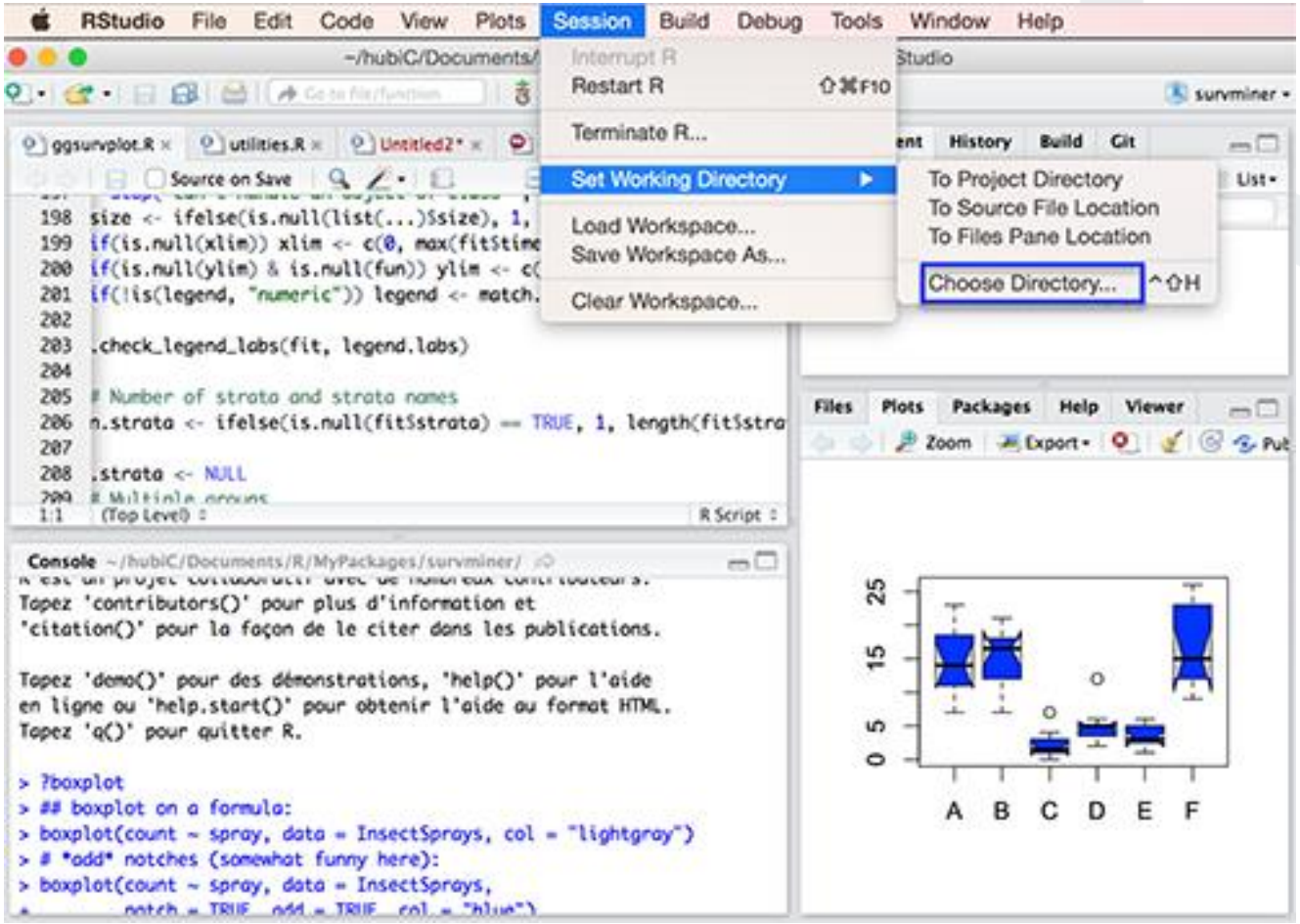
R Console

```
> 5 + 2
[1] 7
```

Keele UNIVERSITY

Setting Working Directory

setwd("C:/Users/OneDrive - University of Keele/KEELE TEACHING/Statistical Techniques for Data Analytics/Data science R practical")



Keele UNIVERSITY

Variables

Variables **store data** and can be defined

- using an *identifier* (e.g., var1 or x or a)
- on the left of an *assignment operator* <-
- followed by the object to be linked to the identifier such as a *value* (e.g., 1)

```
var1 <-  1
```

The value of the variable can be invoked by simply specifying the identifier.

```
var1
```

```
##  [1]  1
```

Keele UNIVERSITY

Algorithms and Functions

- *An **algorithm** or effective procedure is a mechanical rule, or automatic method, or programme for performing some mathematical operations* (Cutland, 1980).
- A program is a specific set of functions that implement an abstract algorithm.
- The definition of an algorithm (and thus a program) can consist of one or more functions.
  - set of instructions that perform a task
- Programming languages usually provide pre-defined functions that implement common algorithms (e.g., to find the square root of a number or to calculate a linear regression).

Functions and Variables

Functions can execute complex operations and can be invoked

- Specifying the function name
- The arguments (input values) between simple brackets

> sqrt(5) [1] 2.236068 > round(2.236068, digits = 2) [1] 2.24	> sqrt <- sqrt(5) > sqrt [1] 2.236068 > round(sqrt, digits = 2) [1] 2.24
> log(5) [1] 1.609438 > round(1.609438, digits = 2) [1] 1.61	> log <- log(5) > log [1] 1.609438 > round(log, digits = 2) [1] 1.61

Keele UNIVERSITY

Name Space

When creating an identifier for a variable or function

- R is a case sensitive language
  - UPPER and lower case are not the same
  - a_variable is different from a_VARIABLE

- names can include
  - Alphanumeric symbols
  - . and _

- names must start with
  - a letter

Keele UNIVERSITY

Libraries

Once a number of related, reusable functions are created, they can be collected and stored in **libraries** (a.k.a. packages). For example, ggplot2 for enhanced graphics.

- *install.packages ()* is a function that can be used to install libraries (i.e., download to your computer)
- *library ()* is a function that loads a package for a given analysis.
- libraries can be of any size and complexity

Keele UNIVERSITY

Vectors

- Vectors are ordered list of values
- Vectors can be of any data type
  - numeric
  - character
  - logic
- All items in a vector have to be of the same type
- Vectors can be of any length

Keele UNIVERSITY

Vectors - Example

```
> city  <- c("Birmingham", "Derby", "Leicester", "Lincoln", "Nottingham",
            "Wolverhampton", "Stoke")

> city

[1] "Birmingham"    "Derby"        "Leicester"    "Lincoln"      "Nottingham"
"Wolverhampton" "Stoke“


> city[c(5, 3)]

[1] "Nottingham" "Leicester“


> number <- seq(1, 30, by = 2)

> number

 [1]  1  3  5  7  9 11 13 15 17 19 21 23 25 27 29
```

Keele UNIVERSITY

Functions on Vectors

Functions can be used on a vector variable directly

```
> a <- 1:5

> a
[1]  1  2  3  4  5

> a + 10
[1] 11 12 13 14 15

> sqrt(a)
[1] 1.000000 1.414214 1.732051 2.000000 2.236068

> a >= 3
[1] FALSE FALSE  TRUE  TRUE  TRUE
```

Keele UNIVERSITY

Factors

A factor is a data type similar to a vector. However, the values contained in a factor can only be selected from a set of levels.

```
> houses_vector <- c("Bungalow", "Flat", "Flat", "Detached", "Flat", "Terrace", "Terrace")
> houses_vector
[1] "Bungalow" "Flat"     "Flat"     "Detached" "Flat"     "Terrace"  "Terrace"
```

```
> houses_factor <- factor (c("Bungalow", "Flat", "Flat", "Detached", "Flat", "Terrace", "Terrace"))
> houses_factor
[1] Bungalow Flat     Flat     Detached Flat     Terrace  Terrace
Levels: Bungalow Detached Flat Terrace
```

Keele UNIVERSITY

Table

The function table can be used to obtain a tabulated count for each level.

```
> houses_factor <- factor (c("Bungalow", "Flat", "Flat", "Detached", "Flat", "Terrace", "Terrace"))

> houses_factor

[1] Bungalow Flat     Flat     Detached Flat     Terrace  Terrace

Levels: Bungalow Detached Flat Terrace


> table(houses_factor)

houses_factor

Bungalow Detached     Flat  Terrace

       1        1        3        2
```

Keele UNIVERSITY

Matrices

Matrices are collections of numeric data arranged in a two-dimensional rectangular layout

- the first argument is a vector of values
- the second specifies number of rows and columns
- R offers operators and functions for matrix algebra

```
a_matrix <- matrix(c(3, 5, 7, 4, 3, 1), c(3, 2))

a_matrix

##      [,1] [,2]
## [1,]    3    4
## [2,]    5    3
## [3,]    7    1
```

Keele UNIVERSITY

Arrays

Variables of the type array are higher dimensional matrices.

- – the first argument is a vector containing the values
- – the second argument is a vector specifying the depth of each dimension

```
a3dim_array <- array(1:24, dim=c(4, 3, 2))

a3dim_array
```

```
, , 1

     [,1] [,2] [,3]
[1,]    1    5    9
[2,]    2    6   10
[3,]    3    7   11
[4,]    4    8   12

, , 2

     [,1] [,2] [,3]
[1,]   13   17   21
[2,]   14   18   22
[3,]   15   19   23
[4,]   16   20   24
```

Keele UNIVERSITY

List

We can assign names to list members, and reference them by names instead of numeric indexes.

For example, in the following, v is a list of two members, named "bob" and "john".

```
> v = list(bob=c(2, 3, 5), john=c("aa", "bb"))
> v
$bob
[1] 2 3 5

$john
[1] "aa" "bb"
```

Keele UNIVERSITY

Data Frame

A data frame is used for storing data tables. It is a list of vectors of equal length.

```
> n = c(2, 3, 5)
> s = c("aa", "bb", "cc")
> b = c(TRUE, FALSE, TRUE)
> df = data.frame(n, s, b)      # df is a data frame
> head(df)
```

Built-in Data Frame

A number of data frames are built-in R. For example, mtcars.

```
> mtcars
> head(mtcars)
```

Keele UNIVERSITY

Importing Data

You can directly import data frames if they are prepared in text, excel, SPSS, Stata, or SAS.

```
> library(readxl)
```

```
> sample <- read_excel("C:/Users/prf49/OneDrive - University of Keele/KEELE TEACHING/Statistical Techniques for Data Analytics/Data science R practical/sample.xlsx")
```

```
> view(sample)
```

```
> head(sample)
```

Keele UNIVERSITY

Creating a Graph

Types of graphs (e.g., age, hight, weight)

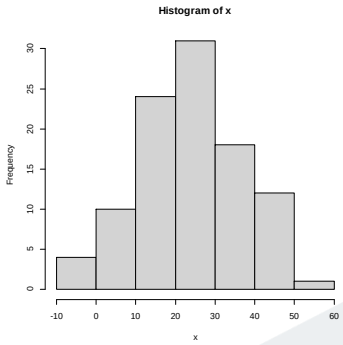
- Continuous data
  - Histogram
  - Density plot
  - Box plot
  - Scatter plot
  - Line plot
- Categorical data (e.g., blood group, gender)
  - Pie diagram
  - Bar diagram

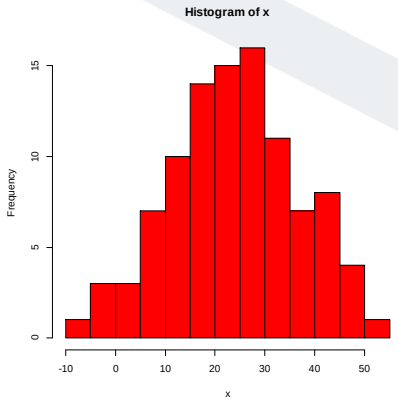
Keele UNIVERSITY

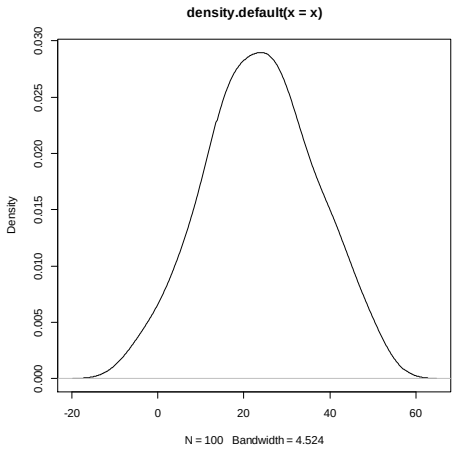
Creating a Graph - Example

- Histogram

```
x <- rnorm(n = 100, mean = 25, sd = 15)

# Simple Histogram

hist(x)

# Colored Histogram with Different Number of Bins

hist(x, breaks=12, col="red")

# Kernel Density Plot

d <- density(x) # returns the density data

plot(d) # plots the results
```







Keele UNIVERSITY

Keele UNIVERSITY

Thank you