Descriptive and Exploratory Techniques in R

Descriptive Techniques

- involve the use of tables, charts, and the computation of summary measures
- generalizations only apply to the data on hand
- Example: Suppose that data on the age of all participants of this workshop is collected. The average age is then computed.
- Conclusions about such average age applies only to the participants of this workshop.

Exploratory Techniques

- involve the use of visual displays and resistant statistics
- used to investigate the collected or transformed data to reveal patterns, peculiarities and relationships

What is R?

- a free software environment for statistical computing and graphics
- However, R does not have a point-and-click interface.

What is RStudio?

- An integrated development environment (IDE) for R
- Preferred since R Console is too bare-bones and is not that user-friendly

RStudio Interface

- Source
  - where the “source code” should be typed
- Console
  - where outputs appear; where codes can be run in order to generate instantaneous results
- Workspace and history
- Files, plots, packages, help

Using R

- It can be used like a calculator and we can perform basic calculations.
- We can also assign these values to “objects.”

Creating a New RScript File

- Create a new "RScript" file by pressing Ctrl+Shift+N. Input the following:
  a <- 5
  b <- 7
  A*b
  a*b
- We can run each line by pressing Ctrl+R.
- What happened upon running the 3rd line?
- Basic operations: +, -, *, /

Working in the Console

- You may use the up and down arrow keys to see the previous commands (codes) that you have used.
- Seeing the “>” symbol means that R is waiting for your input. This is called the prompt.
- If you have entered an incomplete command, R shows the “+” symbol. This gives you the chance to complete the command.
- If you just want to cancel that incomplete command, press the Esc key.

On Code Syntax

- You may add comments by using the "#" symbol.
- Including comments is good practice – it helps explain what goes on in your code.

- In most instances, R doesn't care about spacing.
- Spaces matter only when dealing with characters.

On Code Syntax

- We assign names to objects so that we can “call” them again.
- Names in R are case-sensitive.
- It is good practice to use descriptive names, but keep them short and simple.
- Spaces and special characters are not allowed. If you want to use two (or more) words in the object’s name, use an underscore, e.g. my_object.

Common Objects in R

- Vector – 1 dimension only
- Matrix – two-dimensional only
- Array – multidimensional
- Data frame
- List

Common Objects in R

- A data frame is a special form of an array while a list is a special form of a vector.
- How can you check the type of an object?

-use “typeof()”

Aside: We can check the documentation of functions or commands in R if further clarification is needed.

Modes

All objects have a certain mode. Operations can be performed depending on this mode (e.g. characters are added in a different way from numbers when using "+").

- Integer
- Numeric
- Character
- Logical

Datasets in R

Datasets in R

Two ways:

- Create
- Import

Creating a Vector

- Use “c()”

vec1 <- c(1, 2, 3)

vec2 <- c(“one”, “two”, “three”)

What is the type of vec1? Of vec2?

Note that when dealing with characters (strings), we use quotation marks.

Accessing the Elements of a Vector

- We use an element's index.
- This serves as that element's address in the vector.
- Syntax:

  - vectorname[1] will give you the first element

  - vectorname[-1] will give you the entire vector except the first element

  - vectorname[vectorname==1] will only show the elements whose value is equal to 1

Exercise

- Create a vector called "vec1" which contains the numbers 1, 2, 3, and 4.
- Create a vector called "vec2" which contains the numbers 5, 6, 7, and 8.

Creating a Matrix

Two ways:

- Matrix function
- Rbind or Cbind (row-bind/column-bind)

Example

- Type the following lines in the source window: (DO NOT RUN YET)

```
m1 <- matrix(1:8, nrow=2)
m2 <- matrix(1:8, nrow=2, byrow=TRUE)
m3 <- rbind(vec1, vec2)
m4 <- cbind(c(1,5), c(2,6), c(3,7), c(4,8))
```

Accessing the Elements of a Matrix

- Like in vectors, we can use indices to access the elements of a matrix.
- Syntax:
    - matrixname[i,j] will give you the element in the i^{th} row and j^{th} column of the matrix.
- In the previous example, what does m4[2,3] contain?

Accessing the Elements of a Matrix

- We can also access a whole row, a whole column, some rows, or some columns – not just one element at a time.
- Using m4,
  - The first row only: m4[1,]
  - The second column only: m4[,2]
  - The first and second column only: m4[,1:2]
  - The first and third column only: m4[,c(1,3)]
- These can be saved as new objects.

Typical Workflow for Data Analysis

- Put all of your necessary files in one folder.
- Set this folder as your **working directory**.
- Make sure to save codes, results, etc. in this same folder.

Setting the Working Directory

- Syntax:

```
setwd("C:/My Documents/My Folder")
```

or

```
setwd("C:\\My Documents\\My Folder")
```

Importing Datasets

- Your files may come in various formats. Typically, you may have created or used these datasets in MS Excel, Stata, etc.
- For convenience, when exporting data from such software, export it as .csv (comma-separated file).
- This is a format that saves your data as plain text, getting rid of unnecessary formatting.

Importing Datasets

- .xlsx files may also be imported to R, but the process entails using a certain package.

Aside: Packages are “add-ons” to base R. These perform specialized or advanced procedures.

Importing Datasets

- To read .csv files, the syntax is

```
data <- read.csv("mydata.csv")
```

- The code above stores the file called "mydata.csv" into an object in R called "data."
- This will only work if the file is **in the folder of your working directory.**
- Otherwise, you would have to specify the whole path/location of your file.

Importing Datasets

- By default, read.csv() assumes that the first row contains the variable names.
- Otherwise, add header=FALSE to your code.

Built-in Datasets in R

- R comes with built-in datasets.
- To see the list of datasets,

```
data()
```

- More information about the datasets can be viewed in the "Help" pane.

Descriptive Statistics: Summary Measures

Summary Measure

- a single numeric figure that describes a particular feature of the whole collection
- provides meaning to a collection of observations

Examples

- Measures of Central Tendency
- Measures of Dispersion
- Measures of Skewness and Kurtosis

Functions for Descriptive Statistics

min()	minimum of the elements of the vector
max()	maximum of the elements of the vector
mean()	mean of the elements
median()	median of the elements
range()	the range of the vector
sd()	the standard deviation (on n-1)
var()	the variance (on n-1)

Other Helpful Functions

sum()	sums of the elements of the vector
prod()	product of the elements of the vector
sort()	sorts the vector (default: decreasing = FALSE)
length()	returns the length of the vector
summary()	returns summary statistics
unique()	returns a vector of all the unique elements of the input

Notes

- For measures of skewness and kurtosis, a package still has to be installed (“moments”).
- R does not have a built-in function for the mode.

Example

- Let us use the InsectSprays dataset.

Exercise

- Using the precip dataset, compute for the mean, median, and the coefficient of variation (CV).
- Note: CV = (standard deviation/mean) x 100%

Graphical Displays

Frequency Histograms

- serve as graphical representation of the Frequency Distribution Table
- show the shape of the distribution of the dataset

Creating a Frequency Histogram in R

```
library(MASS)
hist(data values, xlab = “label of x-axis”, main = “title”)
```

Example

- Frequency histogram for the InsectSprays dataset

Exercise

- Create a frequency histogram of the variable **len** in the **ToothGrowth** dataset.
- Create separate histograms of the same variable for each type of supplement.

Boxplots

- simple graphical summary which can display the following features of the data: (i) location, (ii) spread, (iii) symmetry, (iv) extremes, and (v) outliers.

Creating a Boxplot in R: Using the boxplot() function

boxplot(data values, ylab = “label of y-axis”, main = “title”)

If we want to create side-by-side boxplots,

boxplot(data values ~ categories, ylab = “label of y-axis”, main = “title”)

Example: Use the InsectSprays dataset.

Using the plot() Function

The arguments of the function should be a qualitative variable for x and a quantitative variable for y.

plot(categories, data values, ylab = “label of y-axis”, main = “title”)

Exercise

- Create side-by-side boxplots of the variable **len** for each type of supplement in the **ToothGrowth** dataset.
- Using the **warpbreaks** dataset, compare the distributions of the number of breaks
  - Per type of wool
  - Per level of tension

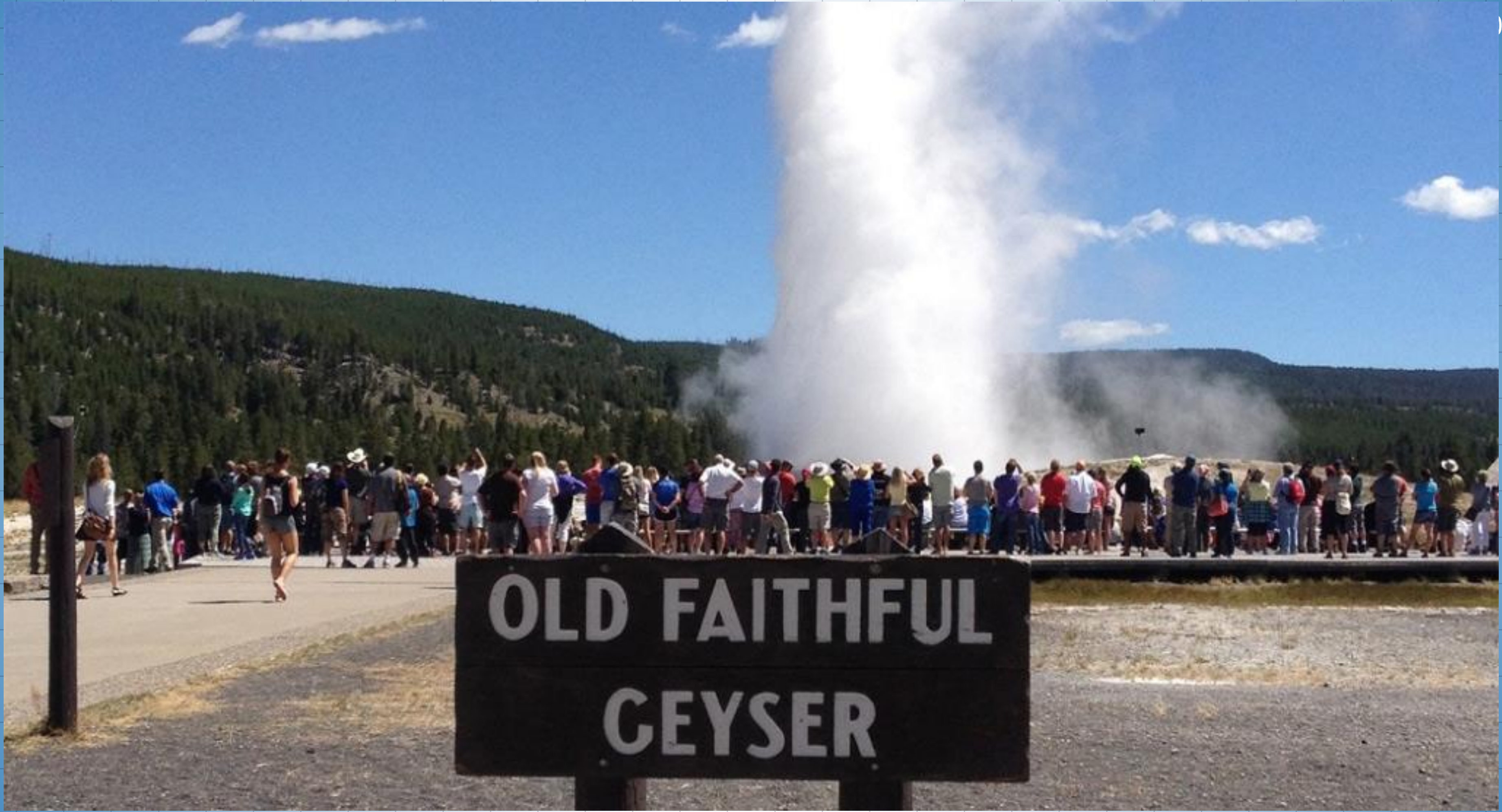
Scatterplots

- display the relationship between two quantitative variables

Creating a Scatterplot in R: Using the plot() Function

plot(variable 1, variable 2, xlab = “label of x-axis”, ylab = “label of y-axis”, main = “title”)

Example: Using the **faithful** dataset


OLD FAITHFUL
GEYSER

Example

Using the **anscombe** dataset

This is a classic example in illustrating the effectiveness of using graphical display alongside summary measures.

Exercise

- Create a scatterplot for age and circumference in the **Orange** dataset.

Quantile-Quantile Plots

- also known as Q-Q plots
- compare two distributions by plotting their respective sorted values

Common approaches:

- compare two datasets
- compare one dataset with the Normal distribution

Approach 1: Compare Two Datasets

qqplot(dataset1, dataset2, xlab = “label of x-axis”, ylab = “label of y-axis”, main = “title”)

Example: Using the **ToothGrowth** dataset

What can we conclude from the plot?

Approach 2: Checking for Normality

qqnorm(data values, main="title")

Example: Using the **trees** dataset

Exercise

- Using the **Loblolly** dataset,
  - Create a Q-Q plot comparing the distributions of height and age
  - Compare height and age (separately) against the Normal distribution

Final Exercise

- Using the **sleep** dataset, what conclusions can you form about the effect of the two types of drugs?