

## Chapter 2

# Data Exploration, Descriptive Statistics, and Measures of Central Tendency

**Abstract** The purpose of this lesson is to give attention to descriptive analysis, measures of central tendency, and graphical presentation of data, which are essential before any statistical analyses are conducted. Initial efforts should be placed on data exploration and specifically the use of descriptive statistics and measures of central tendency (e.g., mode, median, mean, standard deviation, etc.). A complete summary of descriptive statistics is presented in this lesson, both for factor-type object variables as well as numeric object variables of an interval or continuous nature. An initial summary of graphical presentations available through R is provided, with emphasis on publishable quality graphics deferred until later lessons.

**Keywords** Barplot • Boxplot (box-and-whiskers plot) • Boxplot statistics • Data exploration • Density plot • Descriptive statistics • Dotchart • Histogram • Interquartile range (IQR) • Length • Maximum • Maximum location • Mean • Measures of central tendency • Median • Minimum • Minimum location • Mode • Quantile-quantile plot • Quartiles • Range • Scatter plot • Sort • Standard deviation • Stem-and-leaf plot • Stripchart • Sum • Summary • Tukey's five number summary • Variance

## 2.1 Background on This Lesson

### 2.1.1 Description of the Data

This lesson on descriptive statistics and measures of central tendency is taken from a study that was conducted at a large high school in Florida, as part of a general investigation of wellness and student health. The dataset for this lesson is fairly small ( $N = 30$  subjects) and represents only a small part of a much larger dataset,

larger in terms of more subjects and larger in terms of more variables. This lesson describes the use of R for descriptive statistics and measures of central tendency, with outcomes presented as numerical statistics and simple graphical presentations.

For this lesson, consider the data gained by a school nurse who weighed all 30 students in Computer Programming III (Course Number 0201320), a Computer Science Education course offered to Grade 12 (e.g., High School Seniors, usually 17–18 years old) students. Weight was measured in pounds, with accuracy at the tenth of a pound. As the principal investigator, the school nurse is naturally concerned with overall trends as well as individual measures. What was the average weight? What was the lowest weight and what was the highest weight? What was the variance in weight? Were there any trends that need attention, either for immediate purposes or in the future? With proper analysis, this information could be used, in part, as the basis for informed decision-making on wellness, food selections in the cafeteria, policy and procedures for snack vending machines, etc.

This lesson provides an introduction, using a small sample of only 30 subjects, of how descriptive statistics and measures of central tendency have value on their own and also as indicators for the use of other statistical tests. Quite often when examining data and relationships between and among data, it is useful to offer a general view of the data. Saying this, consider the data conceivably associated with this lesson. It would be more than somewhat useful to know:

- How many students were enrolled in the class and are eligible to have their weights measured?
- How many students had their weights measured?
- What is the average weight and are there multiple definitions of the term average? If there are multiple definitions for the term average, when is it appropriate to use one view of the term average but not the other(s)?
- Did most weights cluster around the average weight, or was there a wide degree of variance in weights?
- Were there any weights that seem to be exceptionally out-of-range (e.g., outliers), demanding specific attention for these observed weights?
- Were there any weights that seem to be illogical, perhaps by accidental data entry of alphabetical characters or similar errors in an object that has otherwise been declared as a vector of numeric values?
- What was the range of weights, from the lowest (e.g., minimum) weight to the highest (e.g., maximum) weight?
- Do the weights display normal distribution, approximating a bell-shaped curve, or is the distribution skewed and if so, how? Are weights skewed to the left or are weights skewed to the right?

### **Descriptive statistics and measures of central tendency, or representation of the average:**

- **Mode:** most frequent measure (An oddity of R is that the `mode()` function has nothing to do with measures of central tendency, but there are convenient work-arounds that provide mode as an average.)

- **Median:** mid-point of an array of measures
- **Mean:** arithmetic average (Sum/N)

In the perfect bell-shaped curve, all three measures for average (e.g., mode, median, and mean) would be equivalent, but of course this level of perfection is rarely achieved.

#### **Measures of dispersion, spread, or variance in range away from the average:**

- **Variance:** the sum of squared deviations from the mean
- **SD:** the standard deviation, or the square root of the variance
- **Range:** the spread from the lowest measure to the highest measure

It is common to present in summary statistics a listing of these descriptive statistics, to give the reader a general view of the data. It is also highly desirable to provide graphical figures, visually representing trends.

This lesson has been designed as a demonstration of how R can be used to provide descriptive statistics and measures of central tendency. The emphasis will be on the use of functions found in the basic R package as well as a brief introduction to the use of functions gained from external R packages. Complementary graphical representations are also provided.

This lesson should provide a fairly detailed introduction to descriptive statistics and measures of central tendency and how they are calculated and presented using R. This topic is of special importance since nearly each statistical analysis associated with parametric data (e.g., the use of interval or ratio data for Student's t-Test, Analysis of Variance, etc.) begins with descriptive statistics and measures of central tendency.

### ***2.1.2 Null Hypothesis ( $H_0$ )***

Because this lesson is specific only to descriptive statistics, there is no associated Null Hypothesis. The Null Hypothesis will be identified, however, in future lessons.

## **2.2 Data Import of a .csv Spreadsheet-Type Data File into R**

The data for this lesson are from a much larger dataset. The complete dataset was originally prepared in Gnumeric, an open source spreadsheet. After a set of manipulations (largely **Copy and Paste** and later **File and Save as**) the dataset for this lesson was put into .csv (e.g., comma-separated values) file format. The data are in ASCII format and they are separated by commas. The data are not separated by tabs and the data are not separated by spaces.

Eventually, the data were placed on an external harddrive (the F drive) in a directory marked as `R_Biostatistics`. All analyses and presentations start here.

From this starting point, note below how R is set to work in the appropriate directory and then how the `read.table()` function is used to read in the comma-separated values .csv format ASCII file that contains the data.

```
#####
# Housekeeping                                Use for All Analyses
#####
rm(list = ls())      # CAUTION: Remove all files in the working
                    # directory. If this action is not desired,
                    # use the rm() function one-by-one to remove
                    # the objects that are not needed.

setwd("F:/R_Biostatistics")
                    # Set to a new working directory.
                    # Note the single forward slash and double
                    # quotes.
                    # This new directory should be the directory
                    # where the data file is located, otherwise
                    # the data file will not be found.

getwd()             # Confirm the working directory.
search()            # Attached packages and objects.
#####
```

Create an object called `WeightG12Stu.df`. The object `WeightG12Stu.df` will be a dataframe, as indicated by the enumerated .df extension to the object name. This object will represent the output of applying the `read.table()` function against the comma-separated values file called `WeightGrade12Students.csv`. Note the arguments used with the `read.table()` function, showing that there is a header with descriptive variable names (`header = TRUE`) and that the separator between fields is a comma (`sep = ","`).

```
WeightG12Stu.df <- read.table (file =
    "WeightGrade12Students.csv",
    header = TRUE,
    sep = ",")          # Import the .csv file

getwd()                # Identify the working directory
ls()                   # List objects
attach(WeightG12Stu.df) # Attach the data, for later use
str(WeightG12Stu.df)   # Identify structure
nrow(WeightG12Stu.df)  # List the number of rows
ncol(WeightG12Stu.df)  # List the number of columns
dim(WeightG12Stu.df)   # Dimensions of the data frame
names(WeightG12Stu.df) # Identify names
colnames(WeightG12Stu.df) # Show column names
rownames(WeightG12Stu.df) # Show row names
head(WeightG12Stu.df)  # Show the head
tail(WeightG12Stu.df)  # Show the tail
WeightG12Stu.df         # Show the entire dataframe
summary(WeightG12Stu.df) # Summary statistics
```

## 2.3 Organize the Data and Display the Code Book

The dataframe `WeightG12Stu.df` is fairly simple and very little, if anything, needs to be done to organize the data. That will not be the case in later lessons, but this lesson was designed to serve as an easy-to-follow confidence-building introduction to R so in turn a simple dataset was selected for this lesson.

For this simple lesson, first the `class()` function, `str()` function, and `duplicated()` function will be sufficient first steps to be sure that data are organized as desired.

```
class(WeightG12Stu.df)
class(WeightG12Stu.df$Subject) # DataFrame$ObjectName notation
class(WeightG12Stu.df$Weight)  # DataFrame$ObjectName notation

str(WeightG12Stu.df)                # Structure

duplicated(WeightG12Stu.df$Subject) # Duplicates
```

The class for each object seems to be correct and there are no duplicate subjects in the sample. A Code Book will help with future understanding of this dataset, even if the data currently seem simple and obvious.

```
#####
# Code Book                                     #
#####
#                                               #
# Subject ..... Factor (e.g. nominal) #
#           A unique ID ranging from N0000 to N9999 #
#                                               #
# Weight ..... Numeric (e.g., interval) #
#           Weight (tenth of a pound) of Grade 12 #
#           (approximately 17-18 years) high school #
#                                               #
#                               students #
#####
```

Labels and recoding of individual object variables are not needed for this simple dataset. However, these actions will be seen in future lessons. Again, small confidence-building activities with easy-to-follow examples are used at the beginning of this set of lessons, with more complexity introduced gradually.

## 2.4 Conduct a Visual Data Check

As desirable as numeric descriptive statistics and measures of central tendency may be and are therefore often our first thought, to have a full understanding of the data it is necessary to generate graphics, to actually see how data are organized. Graphics provide an essential complement to our understanding of the data. In later lessons other graphics will be demonstrated, but for initial purposes the graphical

functions of primary interest are `hist()`, `plot()` and `plot(density())`, `boxplot()`, `stem()`, `stripchart()`, `dotchart()`, and `qqnorm()`. Many arguments are available, to embellish these graphical figures, but for now the figures will be prepared in simple format.

The `par(ask=TRUE)` function and argument are used to freeze the presentation on the screen, one figure at a time. Note how the top line of the figure, under **File - Save as**, provides a variety of graphical formats to save each figure: Metafile, Postscript, PDF, PNG, BMP, TIFF, and JPEG. It is also possible to perform a simple copy and paste against each graphical image. It is also possible to save a graphical image by using R syntax.

```
par(ask=TRUE)
hist(WeightG12Stu.df$Weight)           # Histogram

par(ask=TRUE)
plot(WeightG12Stu.df$Weight)           # Plot

par(ask=TRUE)
plot(density(WeightG12Stu.df$Weight))  # Density plot

par(ask=TRUE)
boxplot(WeightG12Stu.df$Weight)        # Boxplot

stem(WeightG12Stu.df$Weight)           # Stem-and-leaf plot

par(ask=TRUE)
stripchart(WeightG12Stu.df$Weight)     # Stripchat

par(ask=TRUE)
dotchart(WeightG12Stu.df$Weight)       # Dotchart

par(ask=TRUE)
qqnorm(WeightG12Stu.df$Weight)         # Quantile-Quantile plot
```

Again, these initial graphics are simple and currently have no meaningful embellishments. They only serve as a first guide to general trends in data organization. Embellishments to the graphics will be introduced in later lessons, by demonstrating the many arguments used to present titles, prepare text and lines in bold and color, etc.

## 2.5 Descriptive Analysis of the Data

A series of functions that come with the base R software at initial download can be used to calculate a wide variety of descriptive statistics and measures of central tendency, such as `length()`, `is.na()`, `complete.cases()`, `summary()`, `mean()`, `sd()`, `var()`, `median()`, etc. A glaring omission is that the `mode()` function does not determine the most frequently occurring value but instead provides information on the storage

mode for a R-based object. A specialized function, found in an external R-based package will be used to calculate mode, when mode is viewed as one of three representations of average: mode, median, and mean.

Be sure to notice the `DataFrame$ObjectName` notation, or `WeightG12Stu.df$Weight` in this case. This type of specificity calls for a degree of strong typing, but it is a desirable practice and provides protection against unintended naming outcomes.

To learn more about the nature of each R function, use the built-in help features found in R. At the R prompt, key `help(function.name)` (e.g., `help(length)`, `help(summary)`, `help(mean)`, etc.) to learn the exact nature of each R function.

```
length(WeightG12Stu.df$Weight)
# Length or N of a vector

is.na(WeightG12Stu.df$Weight)
# Returns TRUE if indexed value is missing (e.g., NA) and
# FALSE if indexed value is not missing

complete.cases(WeightG12Stu.df$Weight)
# Returns TRUE if indexed value is not missing (e.g., NA)
# and FALSE if indexed value is missing

summary(WeightG12Stu.df$Weight)
# Descriptive statistics, including NAs if any
```

Output from this simple application of the `summary()` function follows. The output is basic and in many cases this information is more than sufficient to make judgment on data organization and quality assurance issues.

```
> summary(WeightG12Stu.df$Weight)
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  94.4   117.5   124.6   123.4   130.2   151.5
> # Descriptive statistics, including NAs if any
```

Other functions for descriptive statistics have value, however, and a few of the many functions associated with descriptive statistics and measures of central tendency are demonstrated below.

```
mean(WeightG12Stu.df$Weight)
# Mean or arithmetic average

sd(WeightG12Stu.df$Weight)
# Standard Deviation

var(WeightG12Stu.df$Weight)
# Variance

median(WeightG12Stu.df$Weight)
# Median or midpoint

range(WeightG12Stu.df$Weight)
# Range, minimum and maximum
```

```
min(WeightG12Stu.df$Weight)
# Minimum

which.min(WeightG12Stu.df$Weight)
# Location (e.g., index) of the first occurrence of the
# minimum value

max(WeightG12Stu.df$Weight)
# Maximum

which.max(WeightG12Stu.df$Weight)
# Location (e.g., index) of the first occurrence of the
# maximum value

quantile(WeightG12Stu.df$Weight)
# Quantiles, or values at: 0%, 25%, 50% 75%, and 100%

sort(WeightG12Stu.df$Weight)
# Sort or order values in a vector

sum(WeightG12Stu.df$Weight)
# Arithmetic sum of all values in a vector

boxplot.stats(WeightG12Stu.df$Weight)
# Produce values for a vector related to a boxplot:
# lower whisker, lower hinge, median, upper hinge, upper
# whisker, N, and outliers

fivenum(WeightG12Stu.df$Weight)
# Tukey's five number summary for a vector: minimum,
# lower-hinge, median, upper-hinge, and maximum

IQR(WeightG12Stu.df$Weight)
# Interquartile range of a vector (e.g., a measure of
# dispersion that is equal to the difference between the
# upper quartile and the lower quartile

table(WeightG12Stu.df$Weight)
# Contingency table (e.g., crosstab) of counts for each
# combination of vector values v factor levels
```



## 2.6 Summary

Based on the descriptive statistics associated with this lesson, it is evident that the typical student in Computer Programming III weighs approximately 124 pounds, but of course there is variance in weight:

```
N ..... 30
Missing ..... 0
Median ..... 124.6
Mean ..... 123.3533
SD ..... 12.90337
Minimum ..... 94.4
Maximum ..... 151.5
```

A review of the histogram and density plot provides assurance that there is fairly normal distribution of weights, following a broad approximation of the bell-shaped curve. Further, the `boxplot.stats()` function indicated the presence of outliers, both for the minimum weight and the maximum weight. A diligent researcher would look more closely at the outliers, to be sure that the outlier-specific data are correct and that these data do not represent an error in either measurement or data entry.

Quite simply, the descriptive statistics and measures of central tendency for this sample of 30 subjects follows along with useful outcomes and given the approximation of normal distribution of weights, there should be a fair degree of confidence that the students in this sample could be used for other analyses from the larger dataset for any statistical tests that demand normal distribution.

## 2.7 Addendum: Specialized External Packages and Functions

To use the somewhat humorous expression from a set of well-known American television commercials, *But wait! There's more!* To be specific, there are possibly more than 3,000 external R-based packages available. From these packages there are thousands of specialized functions to supplement the set of functions available when the base R software is initially downloaded. A few specialized functions specific to descriptive statistics and measures of central tendency are demonstrated below.

Be sure to notice how some specialized functions provide not only numerical statistics of immediate use but they also provide a graphical image, to further reinforce the organization of data in question. Function arguments are typically used to embellish graphical output, but in this lesson, function arguments have only been used to any meaningful degree to embellish output from the `epicalc::summ()` function, to provide a glimpse of potentials that will be enhanced in future lessons.

```
install.packages("asbio")
library(asbio)           # Load the asbio package.
help(package=asbio)      # Show the information page.
sessionInfo()           # Confirm all attached packages.

asbio::Mode(WeightG12Stu.df$Weight)
# Mode, as average (e.g., mode, median, and mean) and not as
# storage mode

> asbio::Mode(WeightG12Stu.df$Weight)
[1] 120.9
> # Mode, as average (e.g., mode, median, and mean) and not as
> # storage mode

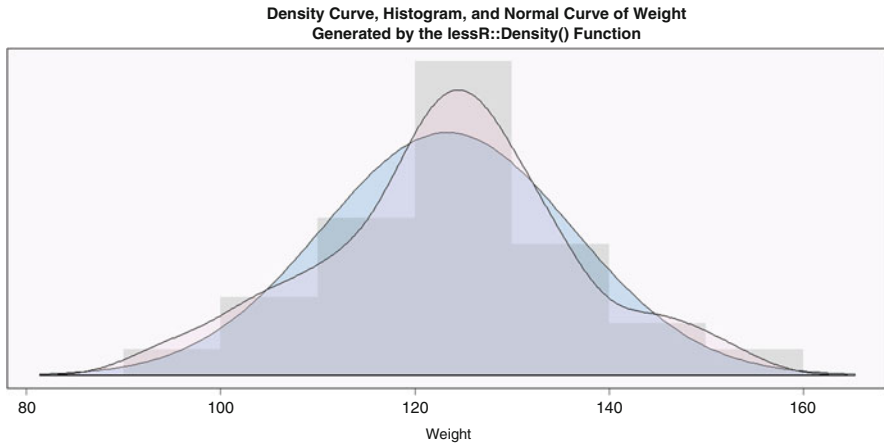
install.packages("lessR")
library(lessR)           # Load the lessR package.
help(package=lessR)      # Show the information page.
sessionInfo()           # Confirm all attached packages.

lessR::SummaryStats(Weight, dframe=WeightG12Stu.df)
# Provide a wide variety of summary statistics and identify
# outliers, if any

par(ask=TRUE)
lessR::BoxPlot(Weight, dframe=WeightG12Stu.df,
  main="Boxplot of Weight Generated by the
  lessR::BoxPlot() Function")
# Produce a boxplot and accompanying descriptive statistics
# about the boxplot and add a title to the figure

par(ask=TRUE)
lessR::Histogram(Weight, dframe=WeightG12Stu.df,
  main="Histogram of Weight Generated by the
  lessR::Histogram() Function")
# Produce a histogram and accompanying descriptive statistics
# about the histogram and add a title to the figure

par(ask=TRUE)
lessR::Density(Weight, dframe=WeightG12Stu.df,
  main="Density Curve, Histogram, and Normal Curve of Weight
  Generated by the lessR::Density() Function")
# Produce a density curve, histogram, and normal curve,
# identify accompanying descriptive statistics about the
# density curve, and add a title to the figure
```



```
install.packages("epicalc")
library(epicalc)           # Load the epicalc package.
help(package=epicalc)     # Show the information page.
sessionInfo()             # Confirm all attached packages.

par(ask=TRUE)
epicalc::summ(WeightG12Stu.df$Weight,
  by=NULL,                # No breakout statistics.
  graph=TRUE,             # Use graph=TRUE, if desired.
  pch=20, ylab="auto",
  main="Sorted Dotplot of Weight Generated by the
  epicalc::summ() Function",
  cex.X.axis=1.25, # Note X axis label size.
  cex.Y.axis=1.25, # Note Y axis label size.
  font.lab=2, dot.col="auto")
# Produce a sorted dotplot and accompanying descriptive
# statistics
```

## 2.8 Prepare to Exit, Save, and Later Retrieve This R Session

It is common to prepare R syntax in a separate file, using a simple ASCII text editor. If time permits, experiment with Crimson Editor, Tinn-R, or vim, but there are many other possible selections.

Use the following set of actions to exit from the current R session.

```
getwd()          # Identify the current working directory.
ls()             # List all objects in the working
                # directory.
ls.str()         # List all objects, with finite detail.
list.files()     # List files at the PC directory.
```

```
save.image("R_Lesson_DescriptiveStatistics.rdata")

getwd()          # Identify the current working directory.
ls()             # List all objects in the working
                 # directory.
ls.str()         # List all objects, with finite detail.
list.files()     # List files at the PC directory.

alarm()          # Alarm, notice of upcoming action.
q()              # Quit this session.
                 # Prepare for Save workspace image? query.
```

Use the R Graphical User Interface (GUI) to load the saved rdata file: **File** and then **Load Workspace**. Otherwise, use the `load()` function, keying the full pathname, to load the .rdata file and retrieve the session.

Recall, however, that it may be just as useful to simply use a R script file (typically saved as a .txt ASCII-type file) and recreate the analyses and graphics, provided the data files remain available.

Introduction to Data Analysis and Graphical  
Presentation in Biostatistics with R  
Statistics in the Large

MacFarland, Th.W.

2014, VII, 167 p. 16 illus., 14 illus. in color., Softcover

ISBN: 978-3-319-02531-5