

Fontbonne University

GriffinShare

OER Textbooks

Open Educational Resources

4-2024

A “How-To” Manual for Doing Standard Statistics in R

Elizabeth Newton

Fontbonne University

Follow this and additional works at: <https://griffinshare.fontbonne.edu/oer-books>



Part of the [Applied Statistics Commons](#), and the [Statistical Methodology Commons](#)



This work is licensed under a [Creative Commons Attribution-Noncommercial-No Derivative Works 4.0 License](#).

Recommended Citation

Newton, Elizabeth, "A “How-To” Manual for Doing Standard Statistics in R" (2024). *OER Textbooks*. 2.
<https://griffinshare.fontbonne.edu/oer-books/2>

A “How-To” Manual for Doing Standard Single-Variable Statistics in R

By Elizabeth Newton, Ph.D.

Professor Emerita, Fontbonne University

With editorial assistance from:

Valerie Slegesky and Courtney A. Iberg

Classroom pilot of original version by:

Kathleen Roy, Ph.D.

Copyright October 2019; Updated August 2024 for R Version 4.4.0

(Page intentionally left blank)

Table of Contents

Introduction: How to Use This Manual

General R Information

1. How to Save R “Programs” AND How to Create Data Files for R
2. How to Get Data into R
3. How to Find, Install and Load R Packages
4. How to Generate Random Samples and Determine Their Frequency Distributions
5. How to See Whether a Specific Value Occurs in a Data Set
6. How to Extract Particular Data Items or Sequences of Them

Graphs and Summary Statistics

7. How to Create Basic Graphs: Barplots and Pie Charts
8. How to Create a Histogram and Calculate Summary Statistics
9. How to Do Pairs of Graphs: Two Histograms or Two Boxplots on One Graph

Distributions and Critical Values

10. How to Check for Normality Using a Normal Probability Plot
11. How to Get Normal Critical Values for Common Significance Levels
12. How to Generate a t-Distribution Graph and Obtain t-Critical Values
13. How to Generate Chi-Square and F Distributions and Obtain Their Critical Values
14. How to Generate and Graph a Binomial Distribution

Parametric Methods

15. How to Run a One-Sample t-Test
16. How to Run Two-Sample t-Tests with Two Independent Samples
17. How to Run Two-Sample t-Tests with Paired Data
18. How to Perform a One-Tailed Hypothesis Test
19. How to Test a Claim about a Single Population Variance
20. How to Perform an F-Test to Compare Two Variances
21. How to Do One-Way ANOVA
22. How to Run a Repeated Measures ANOVA
23. How to Do Two-Way ANOVA with/without Interactions
24. How to Run Mauchly’s Test for Sphericity
25. How to Check Pairs of Data Values for Linear Correlation
26. How to Run a Simple Linear Regression
27. How to Obtain Residuals and Fitted Values from a Regression Line and Check the Assumptions
28. How to Run a Basic Multiple Regression

Non-Parametric Methods

29. How to Test a Hypothesis about a Proportion or Comparing Two Proportions
30. How to Run a One-Sample Mann-Whitney-Wilcoxon Test
31. How to Run a Mann-Whitney-Wilcoxon Test for Two Independent Samples
32. How to Run a Repeated Measures Mann-Whitney-Wilcoxon Test
33. How to Run a Kruskal-Wallis Test
34. How to Run Friedman's ANOVA for Repeated Measures
35. How to Create a Contingency Table and Run a Chi-Square Test
36. How to Test for Normality: Beyond Graphical Methods
37. How to Check Pairs of Data Values for Correlation Non-Parametrically: Spearman's ρ and Kendall's τ
38. How to Run a Binary Logistic Regression

Follow-Up Comparisons

39. How to Do Pairwise Comparisons of Multiple Means or Proportions
40. How to Do Pairwise Comparisons of Multiple Medians

Effect Sizes

41. How to Calculate Cohen's d : Effect Size for Difference between Two Means
42. How to Calculate Cliff's δ : Effect Size for Difference between Two Medians
43. How to Calculate Risk, Odds and the Odds Ratio: Effect Size for Proportions
44. How to Calculate Eta-Squared and Omega-Squared: Effect Size for Differences among Several Means
45. How to Calculate Effect Sizes Related to Correlation and Regression

Cross-Reference Tables

Table 1: Methods, Their Purposes, Typical Assumptions, Alternatives and Cross-Listed References

Bibliography

Software and software packages

Reference texts

Data Sets

Actual data files are available for download separately from the text.

Introduction: How to Use This Manual

First of all, you need to know how to obtain R software. Basic R and all of the packages referenced in this Manual are available as free downloads at www.r-project.org.

A Note on Packages [March 2024]

This text was prepared using version 3.6.0 of R and packages that were available under that version. The R project has recently done a major upgrade to version 4.0, and some of the packages available have changed. Most of what is in this R Manual should still work but some functionality may not be available as stated, especially when it involves installing packages from the R archives. The author is in the process of checking on this and will be updating the text as needed. Feel free to use the text but be advised of the potential for difficulties with packages.

This Manual is intended for use in two possible ways.

1. You can use this Manual to learn statistical methods along with a textbook that explains the appropriate content, either in a class or as a self-study project. The Manual does not generally explain the theory of the statistical methods; its purpose is to tell you how to execute the methods using R software. The understanding of why the methods work, or when to use them, is largely left to you to learn from another source.
2. You can use this Manual when you already know which method(s) you need to use, but are not very familiar with R and need to look up the appropriate commands.

The Manual has the topics organized into broad categories. General information about R is in sections 1 - 6. Beyond that:

- If you are using the Manual, either in a course or for self-study, to learn typical introductory statistics, you will probably want to use sections 7 - 21, 25 - 26, 35 and 41.
- If you are using the Manual, either in a course or for self-study, to learn typical analysis of variance and regression methods, you will probably want to use 21 - 28, 39, 41, 44 - 45.
- If you are using the Manual, either in a course or for self-study, to learn typical non-parametric methods, you will probably want to use sections 29 - 38, 40, 42 - 43.
- If you are using the Manual to find out how to perform one or more specific methods, then you will want to reference the sections that deal with those methods.

Table 1 at the end of the text lists each method that is covered in the Manual, its purpose, and the standard assumptions for using the method. The appropriate section numbers for the method and for checking the assumptions are then listed. So are the section numbers for alternative methods, whenever those are covered in the Manual.

(Page intentionally left blank)

General R Information: Sections 1-6

1. How to Save R “Programs” AND How to Create Data Files for R

2. How to Get Data into R

3. How to Find, Install and Load R Packages

4. How to Generate Random Samples and Determine Their Frequency Distributions

5. How to See Whether a Specific Value Occurs in a Data Set

6. How to Extract Particular Data Items or Sequences of Them

Section 1: How to Save R “Programs” AND How to Create Data Files for R

How to Save R Programs

After you have been working interactively in R, copy and paste the session into NotePad. Edit out everything but the “good” commands – that is, delete all of the > prompt symbols, the incorrect commands and their error messages, and the output. Save the remaining list of good commands as a text file.

When you open R again, you can copy/paste this file into R right after the > prompt appears. R will run all of the commands at once as a “program” and reproduce all the prior output.

Note: You can put comment lines in your R code so that when you go back to it later, you know what the code is supposed to do. To make a comment line, just put a hashtag symbol immediately after the prompt symbol. For example, you could write:

```
> #This is a comment line (and R would just ignore it)
```

How to Create Data Files for R

Here are two options.

1. Type the data into a NotePad text file. The first column should be ID values for the data items, and the first row should be headers. Save as a text file.
2. Enter the data in a spreadsheet such as Excel because it is easier to edit it there. But then you have to save it as a text file that R can use. To do so:
 - make sure the spreadsheet file only has one worksheet
 - when you save it, you will want to save it twice.

Save it once as a regular spreadsheet file, in case you want to edit some more. Save it the second time as file type “text (tab delimited)” and that will turn it into a text file with spaces between the columns. This is the form that you will use in R.

To make it more readable once it is a text file, you may want to add more spaces in order to make the columns straight. If so, use the space bar; DO NOT USE the tab key. However, straightening out the columns is optional; R will not care one way or the other.

A Few Words about Data

When you are working in Excel, it is natural to leave spaces in labels such as column headings, or in other data that is not numeric. It also seems natural to include dollar signs if the data is about currency, commas in long numbers, and similar notations. Likewise, it seems natural to use numbers as identifiers.

DO NOT DO ANY OF THOSE THINGS! SEE EXAMPLES BELOW.

They will cause various kinds of errors when R tries to read or work with the data set!

Examples

Instead of: Use: Reason:

Student ID	Student.ID	R would interpret the first version as two items, not one as you intended it to be.
\$400,000	400000	The \$ and the comma make it non-numeric, so that it cannot be used for calculations in R.
23 (<i>as an ID#</i>)	N23	R will incorrectly interpret the “23” as an actual number and not an ID number.

There are other ways to address these problems, but the simplest thing is to plan ahead when you set up your data and avoid them entirely.

Section 2: How to Get Data into R

(Uses data file: Hospitals.txt)

The prompt in R is the > symbol. When you see this symbol, you can type a command. Your first problem is to get some data into R so that you can work on it. So first read the data table into R. Assuming you have column headings in your data set, the form of the command is:

```
> Data = read.table ("Drive:/Folder/File.txt", header = TRUE)
```

For example, if the file "Hospitals.txt" is stored in a folder called "Data Files" on drive E, the command is:

```
> Data = read.table ("E:/Data Files/Hospitals.txt", header = TRUE)
```

You can choose the name that you want R to use for the whole data set and put it on the left side of the equal sign, right after the prompt symbol. You can either give it a meaningful name (for example, you could call it Hospital.Name) or just call it a generic name like "Data." The line above uses the generic name.

Once you have read the data into R, it needs to be "attached." THIS IS ESSENTIAL! Otherwise R cannot work with the data set. You specify "attach" followed by the name that you gave the data set in parentheses.

```
> attach (Data)
```

To display the data once it has been read into R, type the name that you gave the data set. For the rest of this section, assume that you gave it the generic name: Data.

```
> Data
```

The resulting output is as shown.

- The first column is a record counter generated by R; it is NOT part of the data file and is not an ID number.
- The second column is the name of the hospital; in this example, these are made-up names using Greek letters. In a real data set, you would have real names of real hospitals.
- The third is the count of its capacity in number of inpatient beds.

	Hospital	No.Beds
1	Alpha	787
2	Beta	356
3	Gamma	190
4	Delta	1252
5	Epsilon	767
6	Sigma	264
7	Theta	457
8	Kappa	154
9	Lambda	333
10	Omega	525

Section 3: How to Find, Install and Load R Packages

1. How to Find R Packages

Frequently, you will know what you want R to do but you may not know whether R has a ready-made command to do it. Or you may know such a command exists, but you may not know the exact syntax. You can search R documentation for the base package by typing a double question mark (??) after the prompt (>), and then the name of the process you want to perform. If R can match exactly what you type, it will show you documentation. Otherwise (maybe more often than not), you will have to try something else.

In that case, do an internet search using your preferred search engine. For that, you will not need to know the exact R command, but you can type a brief description of what you want to do. For example, you might search online for the following: “Test single variance in R.”

Your search engine will usually find references for you, which you can use to figure out the command that you need. The query above will fairly quickly lead you to the command “varTest,” which is in the package “EnvStats.”

That brings you to the point where you have to find out whether or not you already have that package. If you do, you just need to load it. If you don’t, you need to install it and then load it. Once the package is loaded, you can use the command. Note that, if you stop work and close R, you will need to reload the package whenever you want it again. You should not need to reinstall it however.

2. How to See Whether or Not You Already Have a Particular Package

On the menu bar, choose “Packages” and then “Load Package.” You will get a long list of packages in alphabetical order. Scroll down the list until you get to the point where the package you want should be in the list. If it shows up, you have it and can skip the “install” step and go directly to “load” step. If it doesn’t show up, you have to “install” the package.

3. How to Install an R Package

Back on the menu bar, choose “Packages” and then “Install package.” Alternatively, you can type “install.packages()” after the > prompt. Either way, you will get a list of CRAN mirrors; these are repositories of base R and R libraries. The list is alphabetical, so the U.S. repositories are near the end of the list. Choose one of them.

After you choose a CRAN mirror, you will get an alphabetical list of packages. Find the one you want and click on it. The installation process will run automatically.

4. How to Load an R Package (Once it has been installed)

If you got here directly from step 2, you have already located the package. So you just click on it and wait for R to load it. If you had to do step 3, now repeat step 2 and then click on the name of the package. Either way, when the loading is finished, you will see the R prompt (>) again and can now use the command that you wanted.

Section 4: How to Generate Random Samples and Determine Their Frequency Distributions

(Uses no data files)

The “sample()” command generates random whole numbers. It requires three items inside the parentheses: (1) the interval for the random values, written in the form “lower:upper”, (2) the number of values to be generated – that is, the sample size, and (3) a subcommand to indicate whether the sample is to be done with or without replacement.

Here are three examples. Note that your samples will differ slightly from the ones generated here.

EXAMPLE A

This example generates a random sample of 100 zeros and ones, with replacement. It represents the experiment of tossing a coin, with the notation 0 = Tails and 1 = Heads. The phrase “with replacement” means that values can be repeated within the sample. In this case, they have to be since only two values are possible and there are 100 repetitions.

```
> s1 = sample (0:1, 100, replace=TRUE)
> s1
```

The resulting output changes each time the command is run because a new sample is generated. A typical example is shown below.

```
1 0 1 1 0 1 0 1 0 1 1 1 1 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 1 1 1 0 1 0 0 1 1 0 1 1 1 0 0 0 0 0 0 0 0 1 1 1 0 0 1 0
1 1 1 0 0 0 0 0 0 1 0 1 0 0 0 1 0 0 0 0 1 1 0 1 1 1 1 1 0 0 0 1 1 0 0 0 0 0 1 1 1 0
```

You can then get a frequency distribution for the sample by using the “fable()” command. The name that you gave the sample, in this case s1, goes in the parentheses. You can also obtain a histogram with “hist()” and the name of the sample in parentheses.

```
> ftable (s1)
```

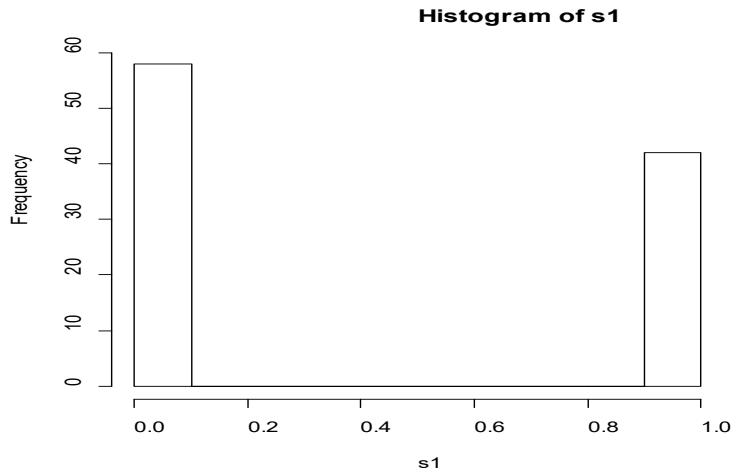
The resulting output follows; it indicates that there were 58 “tails” and 42 “heads” in the sample.

```
s1  0  1
    58 42
```

You can also generate a histogram of the sample.

```
> hist (s1)
```

You then get the following graph.



EXAMPLE B

The second example generates a random sample of 100 values 1 through 6, with replacement. This represents the experiment of rolling a fair die, one hundred times.

```
> s2 = sample (1:6, 100, replace=TRUE)
> s2
```

The resulting output changes each time the command is run because a new sample is generated. A typical example is shown below.

```
1 1 6 1 5 6 4 6 5 1 1 6 3 3 5 1 2 4 2 5 1 6 2 5 6 4 4 3 5 4 2 2 4 3 4 1 6 4 2 5 1 4 4 1 3 3 3 1 6 4 2 2 4 5 2 3 3 3
2 5 1 5 5 4 4 3 1 4 2 4 5 4 6 4 6 6 6 6 5 6 2 5 6 1 2 6 3 6 1 2 5 6 4 3 3 6 6 1 1 4
```

You can then get a frequency distribution and a histogram if you choose.

```
> ftable (s2)
```

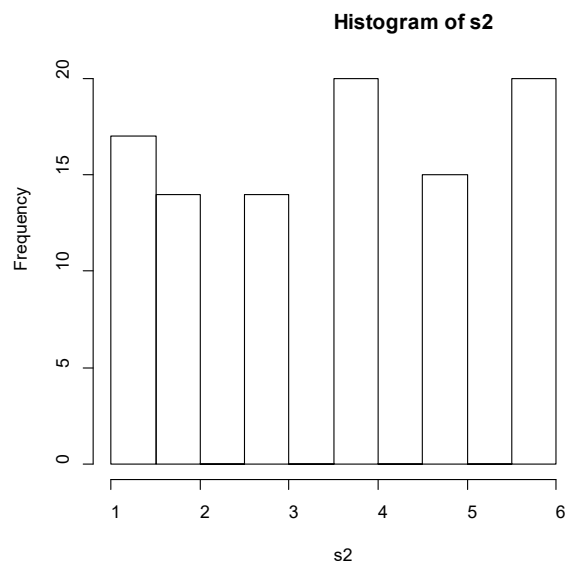
The resulting output is given below.

s2	1	2	3	4	5	6
	17	14	14	20	15	20

This output indicates that you “rolled” 17 ones, 14 twos, 14 threes, 20 fours, 15 fives and 20 sixes in the sample.

```
> hist (s2)
```

The following graph results.



EXAMPLE C

This last example illustrates sampling without replacement. That means a value cannot be repeated within the sample, so your sample size cannot exceed the number of items in the set from which you are sampling.

You can picture this example as if you had the numbers 1 through 5 written on pieces of paper and placed in a box. You then draw three of them at random.

```
> s3 = sample(1:5, 3, replace=FALSE)
> s3
```

The resulting output changes each time the command is run because a new sample is generated. A typical example is shown below.

```
4 1 5
```

Note that you if you try to generate a sample without replacement, and the sample size is bigger than the number of original items, you will get an error message. You can see what this looks like if you change the "3" in the sample command for s3 to any number larger than five.

Section 5: How to See Whether a Specific Value Occurs in a Data Set

(Uses data file: AnxietyRepeat.txt)

This example uses the data set AnxietyRepeat.txt. This data set contains scores on an Anxiety Test, repeated three times on thirty-six (fictional) students. The test is first given during their first term in college (labelled Fall1), repeated second term (labelled Spr1) and then again in their third term (labelled Fall2).

First read in the data table giving the anxiety scores and attach it. Then display the data set.

```
> Data = read.table ("E:/Data Files/AnxietyRepeat.txt", header=TRUE)
> attach (Data)
> Data
```

A partial display of the output is as follows; the spacing lines after every three scores have been inserted here for ease of reading.

	ID	Test.Session	Anx.Score
1	S1	Fall1	25
2	S1	Spr1	22
3	S1	Fall2	33
4	S2	Fall1	11
5	S2	Spr1	5
6	S2	Fall2	20
:			
103	S35	Fall1	22
104	S35	Spr1	27
105	S35	Fall2	17
106	S36	Fall1	24
107	S36	Spr1	30
108	S36	Fall2	37

For instance, suppose you want to see whether or not anyone has a score of 22.

```
> which (Anx.Score == 22)
```

The output is:

```
2 60 70 103
```

This tells you that the value “22” occurred in the list of scores at positions 2, 60, 70 and 103. You can see the first and the last of these in the partial data set displayed above.

Similarly, suppose you want to see whether or not anyone has a score of 9.

```
> which (Anx.Score == 9)
```


The output is a message:

```
integer(0)
```

This message doesn't seem to tell you much, but it means that the value you wanted does not occur in the data. That is, there is no score of "9" in this set of scores.

Section 6: How to Extract Particular Data Items or Sequences of Them

(Uses data file: AnxietyRepeat.txt)

The example uses the data set AnxietyRepeat.txt. This data set contains scores on an Anxiety Test, repeated three times on thirty-six (fictional) students. The test is given during their first term in college (labelled Fall1), repeated second term (labelled Spr1) and again in their third term (labelled Fall2).

First read in the data table giving the anxiety scores and attach it. Then display the data set.

```
> Data = read.table ("E:/Data Files/AnxietyRepeat.txt", header = TRUE)
> attach (Data)
> Data
```

A partial display of the output is as follows; the spacing lines after every three scores have been inserted here for ease of reading.

	ID	Test.Session	Anx.Score
1	S1	Fall1	25
2	S1	Spr1	22
3	S1	Fall2	33
4	S2	Fall1	11
5	S2	Spr1	5
6	S2	Fall2	20
:			
:			
103	S35	Fall1	22
104	S35	Spr1	27
105	S35	Fall2	17
106	S36	Fall1	24
107	S36	Spr1	30
108	S36	Fall2	37

Here are some examples of picking out specific data items by specifying their locations in the list.

EXAMPLE A Suppose you want to see only the scores for the student whose ID is S1; that is, the first three items in the list. All you need to do for this is to specify that you want values from Anx.Score, positions 1 through 3. The following command will do this.

```
> Student1 = Anx.Score [1:3]
```

The output is:

```
25 22 33
```

This tells you that the first three scores (the scores for student S1) are 25, 22 and 33. You can confirm this by looking at the partial display of the data set above.

EXAMPLE B Suppose you want to see only this first student's second score, you would specify position 2 only, as follows.

```
> Student1.Score2 = Anx.Score [2]
```

The output is:

```
22
```

EXAMPLE C Continuing with this idea, suppose that the first half of the scores were from students in one major and the second half were from students in a different major. If you want to split the whole list into two separate ones, one for each major, you would type:

```
> Major1.Scores = Anx.Score [1:54]
> Major2.Scores = Anx.Score [55:108]
```

If you then display them, the Major1.Scores returned are:

```
25 22 33 11 5 20 8 31 13 17 13 18 7 20 13 24 36 43 20 15 27 27 31 15 29 50 15 18 14 24 24 18 30 14 37
35 12 26 29 45 19 40 6 5 13 10 12 18 16 14 17 42 39 17
```

And the Major2.Scores returned are:

```
20 33 19 1 26 22 13 7 13 13 39 20 15 23 13 22 27 10 27 5 5 27 33 18 18 42 12 25 15 5 23 11 21 14 37
27 23 36 30 6 8 28 31 18 34 11 32 18 22 27 17 24 30 37
```

You can then work with the scores for each major as separate data sets (named Major1.Scores and Major2.Scores, respectively) if you wish.

EXAMPLE D Now, suppose you want to obtain the sample means for each test session. First you need to split the appropriate scores out of the list by session. Looking at the data set displayed at the beginning of this section, you will notice that R has assigned counter values to the lines of data, appearing on the far left. These counters indicate that:

- the Fall1 scores are in positions 1, 4, 7, ...,
- the Spr1 scores are in positions 2, 5, 8, ...,
- the Fall2 scores are in positions 3, 6, 9,

You need to make R generate these position sequences and then use them to pick the corresponding anxiety scores out of the last column.

First you need to generate the sequences for these positions. If you name these F1, S1 and F2, respectively, the R code will be as follows. You give R the starting point, the ending point, and the step size for each.

```
> F1 = seq (from=1, to=108, by=3)
> S1 = seq (from=2, to=108, by=3)
> F2 = seq (from=3, to = 108, by=3)
```

You can see these sequences if you then type:

```
> F1, S1, F2
```

If you check the resulting output, you will see that the sequences give the lists of positions (not scores) you want.

```
1  4  7 10 13 16 19 22 25 28 31 34 37 40 43 46 49 52 55 58 61 64 67 70 73 76 79 82 85 88
91 94 97 100 103 106      ← This is the first sequence of positions that you want.

2  5  8 11 14 17 20 23 26 29 32 35 38 41 44 47 50 53 56 59 62 65 68 71 74 77 80 83 86 89
92 95 98 101 104 107      ← This is the second sequence of positions that you want.

3  6  9 12 15 18 21 24 27 30 33 36 39 42 45 48 51 54 57 60 63 66 69 72 75 78 81 84 87 90
93 96 99 102 105 108      ← And this is the last sequence of positions that you want.
```

You can then make R pick out the anxiety scores in these specific sets of positions. The code below picks out the scores from the positions listed in F1 and assigns them to a variable called F1.Scores, and does the other two similarly.

```
> F1.Scores = Anx.Score [F1]
> S1.Scores = Anx.Score [S1]
> F2.Scores = Anx.Score [F2]
```

You can view the separate lists of scores if you wish just by typing the variable names. For instance, typing the name F1.Scores will show you the list below. You can check it against the original data set to see that it is the right list. The others would be done similarly.

```
25 11 8 17 7 24 20 27 29 18 24 14 12 45  6 10 16 42 20 1 13 13 15 22 27 27 18 25 23 14 23 6 31 11 22 24
```

COMMENT: You can then work with the three individual sets of scores. For instance, you can find the means and variances for the anxiety scores of each test session separately. Here are two lines of code to find the mean and variance of the F1.Scores

```
> mean (F1.Scores)
> var (F1.Scores)
```

The resulting output is:

```
19.16667      ← This is the mean of the F1.Scores in the sample.
90.77143      ← This is the variance of the F1.Scores in the sample.
```

The other two sets of scores can be handled similarly. At that point, you will have obtained basic summary information for each test session.

(Page intentionally left blank)

Graphs and Summary Statistics: Sections 7-9

7. How to Create Basic Graphs: Barplots and Pie Charts

8. How to Create a Histogram and Calculate Summary Statistics

9. How to Do Pairs of Graphs: Two Histograms or Two Boxplots on One Graph

Section 7: How to Create Basic Graphs -- Barplots and Pie Charts

(Uses data file: Hospitals.txt)

First get the data into R by the method described in the section called: "How to Get Data into R." Then you can use the data to have R create graphs.

The first one is a barplot that shows the bed counts by hospital.

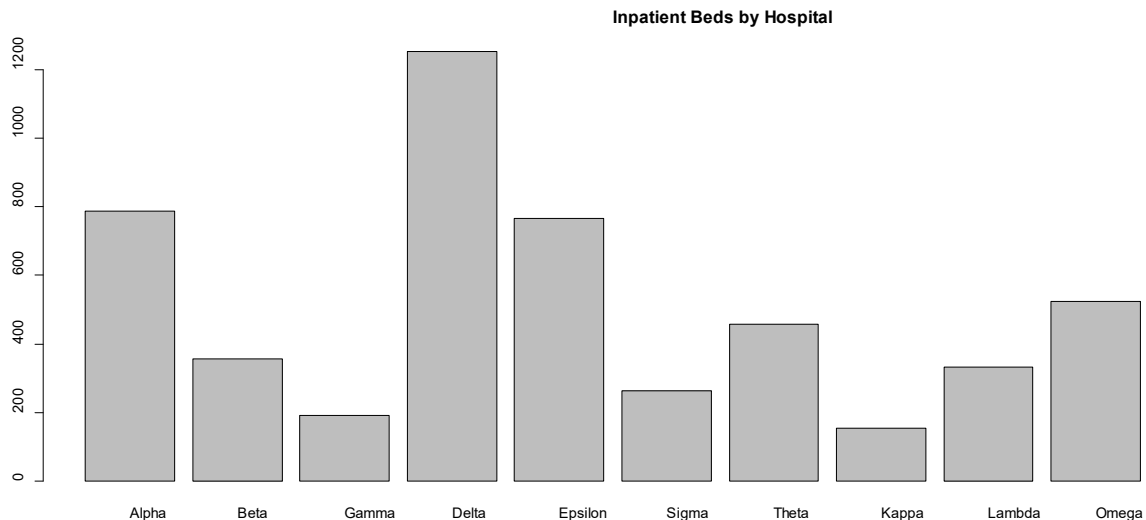
The second one is a barplot that shows the percentages of the total beds by hospital.

The third one is a pie chart that shows relative capacity by hospital.

The parts of the commands are:

1. The type of graph you want.
2. The first item inside the parentheses tells R to use the column No.Beds for the frequencies or percentages. Note that the second version of this command changes from counts to percentages by dividing by the sum of the beds in the table. The change is reflected in the second graph.
3. The second item inside the parentheses of the barplot command says "names.arg = Hospital"; this tells R to label the horizontal axis with the names in the column called Hospital from the data set. In the pie chart, the "names.arg" part of the command is replaced by "labels = Hospital."
4. The last item in the command, where it says
main = "something in quotes"
tells R the heading that you want for the whole graph.

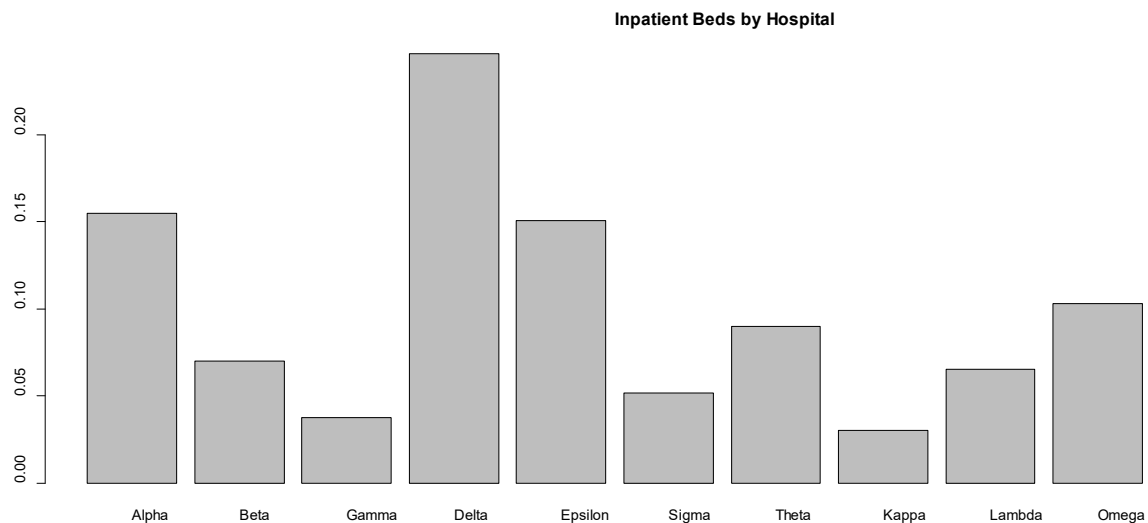
```
> barplot (No.Beds, names.arg = Hospital, main = "Inpatient Beds by Hospital")
```



Note: The graph appears in a separate window, which you can close or save. If you do not save it, you will lose it; a new graph will overwrite a previous one. To save the graph, right-click on it and then save it. If you save it as a "metafile," you can copy/paste it into another document later.

If you want a barplot with percentages instead of counts of numbers of beds, you change the first item in parentheses.

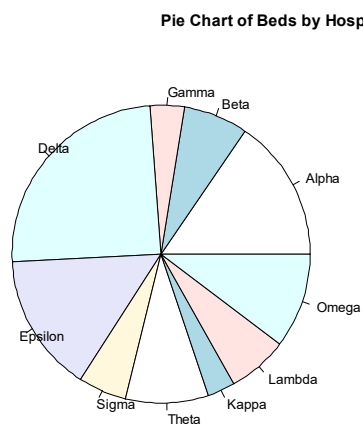
```
> barplot (No.Beds/sum(No.Beds), names.arg = Hospital, main = "Inpatient Beds by Hospital")
```



Note that this graph looks like the previous one, except that the scale on the vertical axis is no longer in terms of bed count. Now it is percentage of the total.

Finally, suppose you want a pie chart based on the number of beds. The command is similar to the “barplot” command, but the portion that said “names.arg” is replaced simply by “labels.”

```
> pie (NoBeds, labels = Hospital, main = "Pie Chart of Beds by Hospital")
```



Section 8: How to Create a Histogram and Calculate Summary Statistics

(Uses data file: Hospitals.txt)

Read in the data file, attach it and then display it.

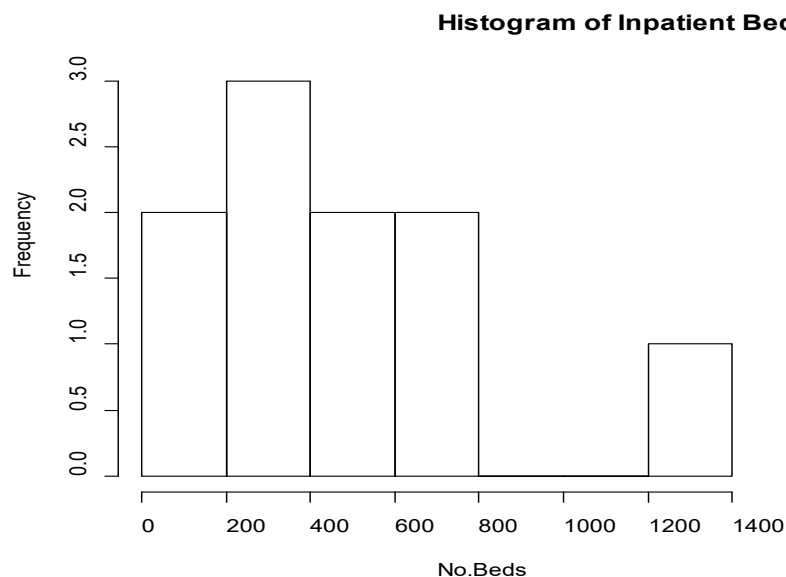
```
> Data = read.table ("E:/Data Files/Hospitals.txt", header = TRUE)
> attach (Data)
> Data
```

The file is as follows.

	Hospital	No.Beds
1	Alpha	787
2	Beta	356
3	Gamma	190
4	Delta	1252
5	Epsilon	767
6	Sigma	264
7	Theta	457
8	Kappa	154
9	Lambda	333
10	Omega	525

You can create a histogram using the column called No.Beds as input data. R will automatically group the data and count the frequencies. The second item inside the parentheses give the graph a title.

```
> hist (No.Beds, main= "Histogram of Inpatient Beds in Hospitals")
```



The next series of commands calculates the mean, median, deviations, variance, standard deviation and range of the data column called No.Beds. Most of these only require single lines of R code. Whenever two lines are needed, the first line defines the function and the second displays the result. First find the mean number of beds.

```
> mean (No.Beds)
```

The output is:

```
508.5
```

Next find the median number of beds.

```
> median (No.Beds)
```

The output is:

```
406.5
```

Find the deviations. These are defined to be the numbers of beds minus the mean.

```
> devs = No.Beds – mean (No.Beds)
> devs
```

The output is:

```
278.5 -152.5 -318.5 743.5 258.5 -244.5 -51.5 -354.5 -175.5 16.5
```

Next find the variance.

```
> var (No.Beds)
```

The output is:

```
115672.3
```

Then find the standard deviation of the sample.

```
sd (No.Beds)
```

The output is:

```
340.1063
```

Finally, if you want the range, it is defined to be the maximum minus the minimum.

```
> range = max (No.Beds) – min (No.Beds)
> range
```

The output is:

```
1098
```

Section 9: How to Do Pairs of Graphs

Two Histograms or Two Boxplots on One Graph

(Uses data file: Solar Eclipses.txt)

Suppose you have two numeric data sets that you want to compare graphically. The most common comparisons are: (1) comparing their histograms, or (2) comparing their boxplots. Obviously, you could do one graph at a time and then look at the results. However, it may be easier to see the comparisons if you put both histograms on one graph, or both boxplots side-by-side. This section shows you how to do that.

The data set is: Solar Eclipses.txt. It contains the duration (in seconds) of a sample of annular and total solar eclipses, where duration is the length of time the shadow of the moon is completely in front of the sun.

As usual, read in the data set and attach it. Then you can display it if you wish.

```
> Data = read.table ("E:/Data Files/Solar Eclipses.txt", header = TRUE)
> attach (Data)
> Data
```

A portion of the data set appears as follows.

	Annular	Total
1	661	389
2	113	132
3	487	380
:		
:		
60	33	226
61	241	153
67	701	249
68	373	143

To put two histograms on the same graph, you first want to see which data set has a greater maximum value. That is because you will want to list that one first when you create the graphs, so that the scaling of the horizontal axis works out nicely.

```
> max (Annular); max (Total)
```

The output from this command is:

```
729
428
```

You can see that the annular eclipses in the data set have a larger maximum. Therefore, you will create the histogram of annular eclipse durations first. The first command does this. The second command overlays the histogram of total eclipse durations by “adding” it to the first graph.

A few comments are needed to explain these commands in the general case. The first command has the following syntax:

```
> hist (First variable, main = "Title for Graph", col="color for bars", xlab = "Labels for x-axis")
```

NOTE: In xlab, plan ahead to specify that the frequencies of first variable you are graphing will show up as one color of bars on the histogram, and the second will show up as a different color. A reader would not know that, so you have to explain it in the labeling. To see what this means, continue below to look at the example.

The second command is then:

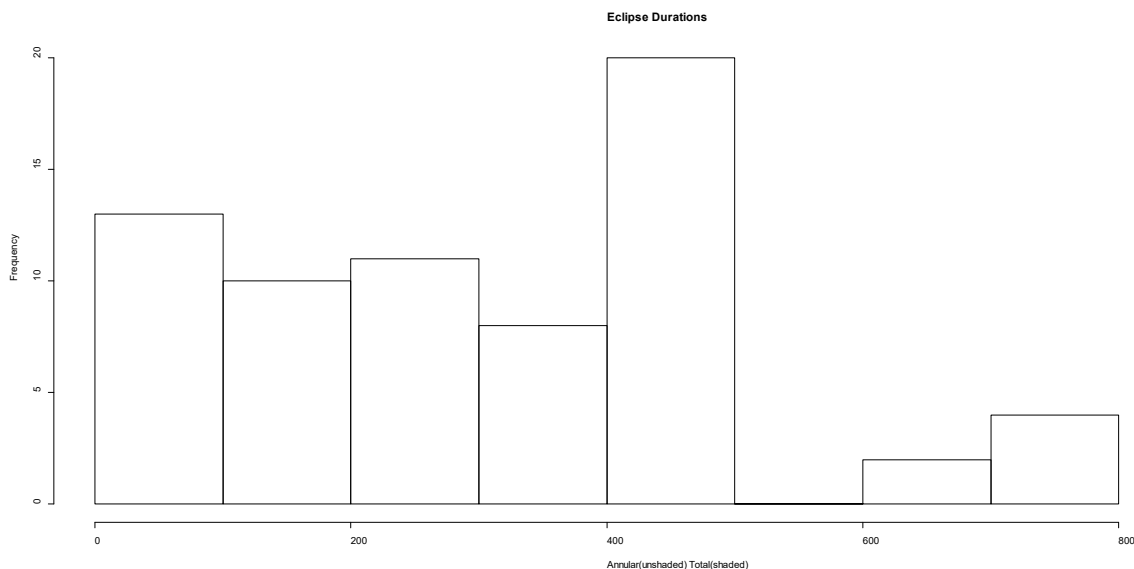
```
> hist (Second variable, col="a different color", add = TRUE)
```

NOTE: The instruction "add = TRUE" tells R to put the second histogram on the same graph as the first.

For this example, assume you want annular eclipse data done with white bars and total eclipse data with light gray bars. You will proceed as follows.

```
> hist (Annular, main = "Eclipse Durations", col="white", xlab = "Annular (unshaded)  
Total(shaded)")
```

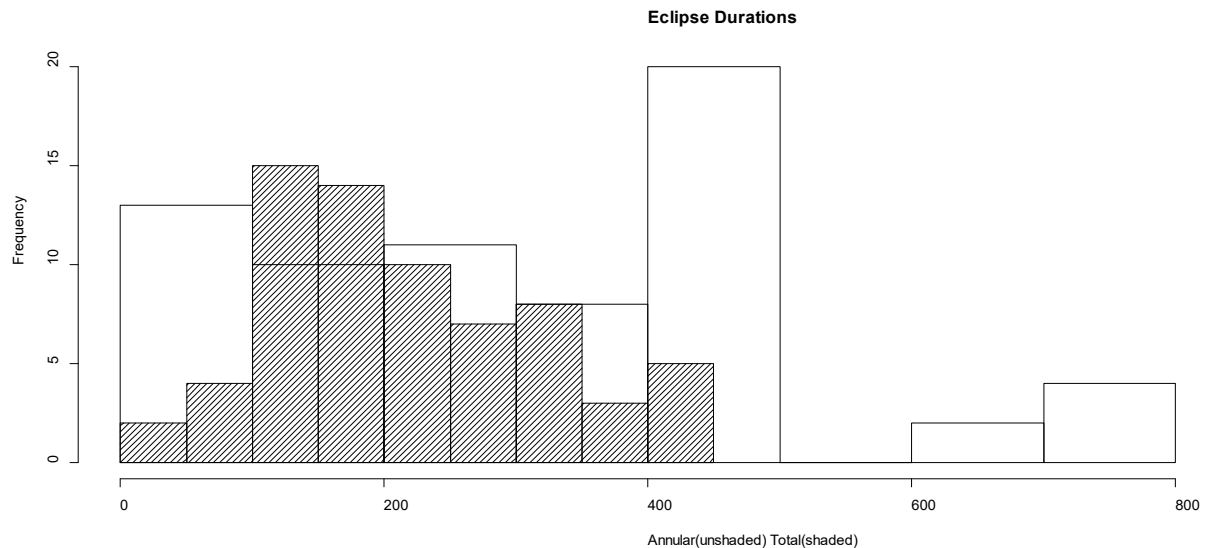
At this point, the histogram looks like this, showing only the annular eclipse durations. That is all that you have told R to produce so far, even though you gave it labelling for both types of eclipses. The labelling is really just a label, and does not get involved in producing the intervals used for the graph.



Leave the graph window open so you can overlay the histogram for the total eclipses on it.

```
> hist (Total, col="lightgray", add = TRUE)
```

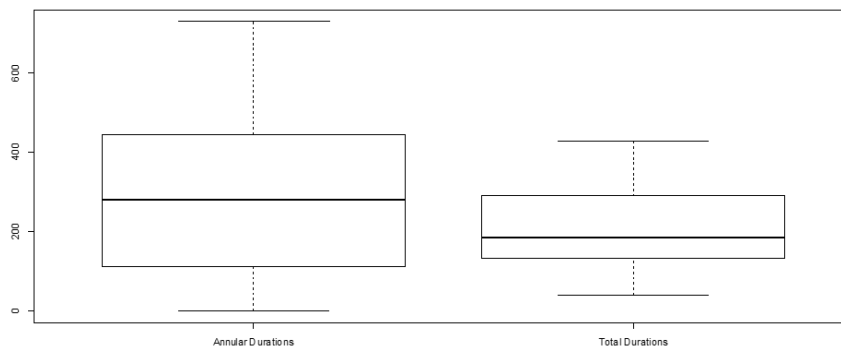
Here is the resulting graph.



Getting two boxplots side by side is simpler; it only requires one command. You use the “boxplot” command, specify both variables in the order that you want them, and use the “names” subcommand to give R a list of the labels to use for each.

```
> boxplot (Annular, Total, names = c ("Annular Durations", "Total Durations") )
```

Here is graph that results. Now you can visually compare the two boxplots and see, for instance, that the median total duration is less than the median annular duration, but the first quartiles are almost the same.



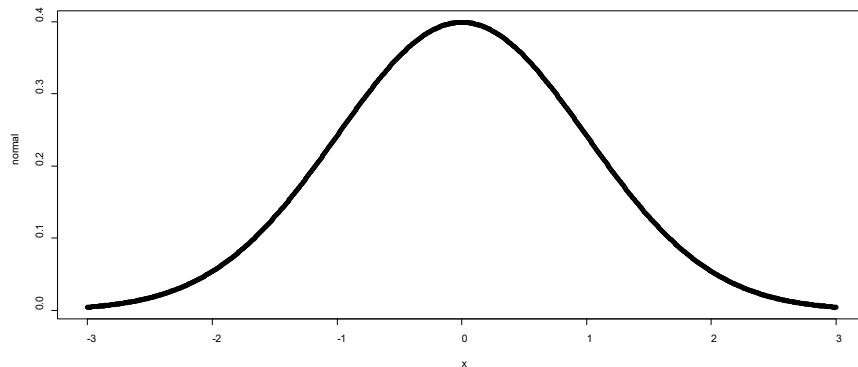
Distributions and Critical Values: Sections 10 - 14

10. How to Check for Normality Using a Normal Probability Plot
11. How to Get Normal Critical Values for Common Significance Levels
12. How to Generate a t-Distribution Graph and Obtain t-Critical Values
13. How to Generate Chi-Square and F Distributions and Obtain Their Critical Values
14. How to Generate and Graph a Binomial Distribution

Section 10: How to Check for Normality Using a Normal Probability Plot

(Uses data file: MO Life Expectancy.txt)

There are several commonly used distributions in elementary statistics – usually the binomial, the normal, the Student's-t, the chi-squared and the F-distributions. By far, the most used distribution is the normal. This follows the traditional “bell-curve” as shown below.



Most of the procedures that are in an elementary statistics course require that the sample come from a population that is approximately normal. R is very good for helping you visualize your data. You can visually assess your data set for normality in two ways: (1) create a histogram and see if it looks reasonably normal, or (2) create a “normal Q-Q plot.” This plots the quantiles of a normal distribution on the horizontal axis, and plots the quantiles of your data on the vertical axis. If the sample comes from a normal population, the points should fall approximately along a straight line.

The following example uses life-expectancy data for the counties of Missouri in the period 2004-2012. (Source: <http://health.mo.gov/data/lifeexpectancy/>).

First, read the data into R. attach it, and display it if you choose.

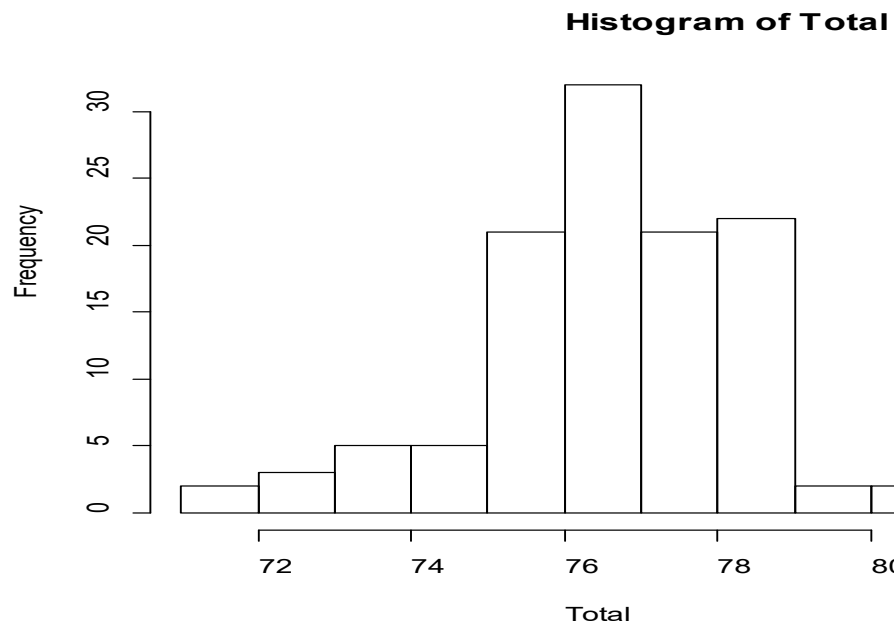
```
> LifeExp = read.table ("E:/Data Files/MO Life Expectancy.txt", header = TRUE)
> attach (LifeExp)
> LifeExp
```

The beginning and the end of data set is shown below. The example will use the column called “Total.”

	County	Total	Male	Female
1	Adair	77.7	75.4	79.9
2	Andrew	77.8	75.1	80.3
3	Atchison	78.3	75.5	81.2
4	Audrain	77.2	73.3	80.7
:				
114	Worth	78.7	75.3	81.8
115	Wright	75.5	72.7	78.2

Make R create the histogram of the Total variable.

```
> hist (Total)
```

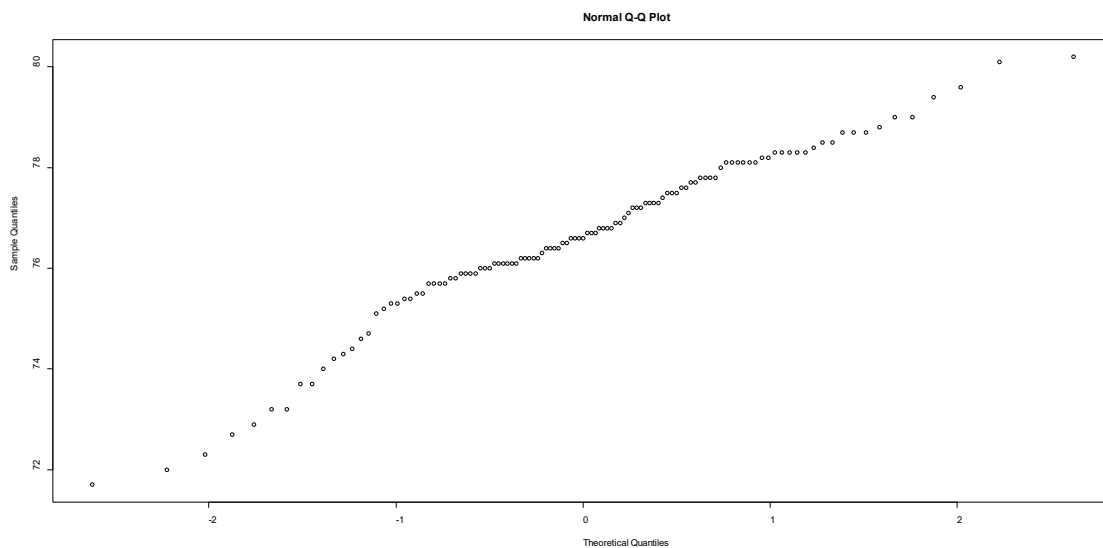


The histogram appears to be slightly skewed left, but not too badly.

Now make R create the normal probability plot of the variable Total. This requires two commands. The first command will show just the data points in the graph below. The second command will make R superimpose the line that corresponds to a perfect normal distribution on top of the plot.

```
> qqnorm (Total)
```

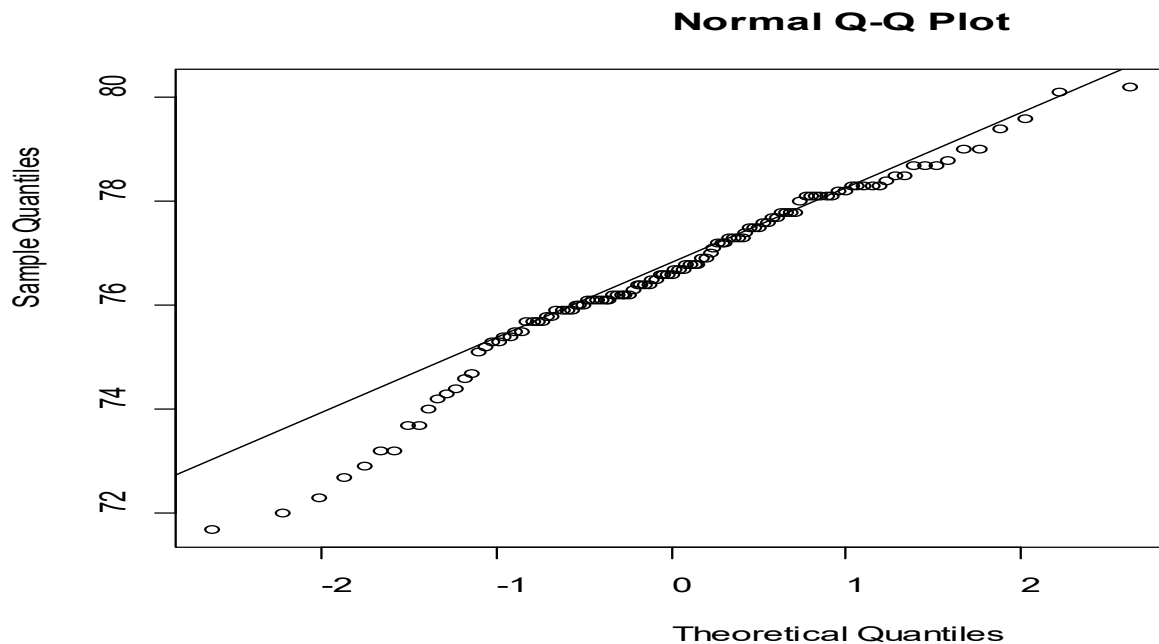
After this first command, the graph looks like the following.



Now the second command will superimpose the line on this graph.

```
> qqline (Total)
```

The resulting graph, showing both the data and the line, is a normal QQ-plot.



At first glance, the line looks similar to the one that you probably think of as the graph of “ $y=x$.” However, if you look at the axis labels, the plot shows sample quantiles vs. theoretically normal quantiles. Therefore, normally distributed data should fit snugly around this line.

Observe how well your data does so. This plot looks reasonably normal, although the life expectancies on the lower end fall a bit below where they should in a normal distribution. This could mean that some counties have something influencing them that causes them to differ from normal (e.g., very poor access to health care, proximity to an environmental problem such as lead or radioactive materials, substandard nutrition, etc.). You cannot determine a reason from the data available in this data set.

Section 11: How to Get Normal Critical Values for Common Significance Levels (Uses no data files)

There are certain common levels of significance used in hypothesis tests, commonly denoted by the Greek letter alpha (α). The most common is $\alpha = 0.05$; other common values are 0.01 and 0.10. Corresponding to any specific value for α , there is a critical value. When you are working with a normal distribution, a critical value is a z-score that defines the boundary between occurrences that “rare” and those that are “not rare.” (Critical values may also be obtained when you are using other distributions; see sections 12 and 13 for other cases.)

To obtain a critical value for a standard normal distribution, you use: “qnorm (area to the left, 0, 1).” The “0” and “1” are the mean and standard deviation for the standard normal distribution, respectively.

To obtain two-tailed standard normal critical values, assign the value for α , then proceed as shown. This example uses $\alpha = 0.05$. Since you want two-tailed critical values, α is divided by two.

For the lower critical value, you want the area to the left to be $\alpha/2$. So you type the following.

```
> Alpha = 0.05
> LowerCV = qnorm (Alpha/2, 0, 1)
> LowerCV
```

Then R displays the lower critical value.

```
-1.959964
```

Now get the upper critical value. For the upper critical value, the area to the right should be $\alpha/2$, so the area to the left is $1 - \alpha/2$.

```
> UpperCV = qnorm (1-Alpha/2, 0, 1)
> UpperCV
```

Then R displays the upper critical value.

```
1.959964
```

Try a new level of significance, say $\alpha = .01$. You only need to change the line that assigns the value of α .

```
> Alpha = .01
> LowerCV = qnorm (Alpha/2, 0, 1)
> LowerCV
```

The lower critical value is displayed.

```
-2.575829
```

Now get the upper critical value.

```
> UpperCV = qnorm (1-Alpha/2, 0, 1)
> UpperCV
```

R displays the following upper critical value.

```
2.575829
```

What about one-tailed critical values?

Suppose you want just an upper critical value with all of α in the upper tail. As before, first specify the value you want to use for α . Then you use the command above for "UpperCV" except that you do not divide α by 2.

Finally, suppose you want just a lower critical value with all of α in the lower tail. Use the command above for "LowerCV" but do not divide α by 2.

NOTE: If you are working with a non-standard normal distribution, it will have either a different mean, a different standard deviation, or both. In that case, put its mean in place of the "0" and its standard deviation in place of the "1." The commands are otherwise the same.

Section 12: How to Generate a t-Distribution Graph and Obtain t-Critical Values

(Uses no data files)

Another distribution that you will need frequently is the Student's-t distribution. This looks almost like a normal distribution but is more variable. It requires that you enter a parameter called the degrees of freedom. It is traditionally abbreviated as "df." This value is dependent on the sample size, but can vary depending upon exactly what you are testing.

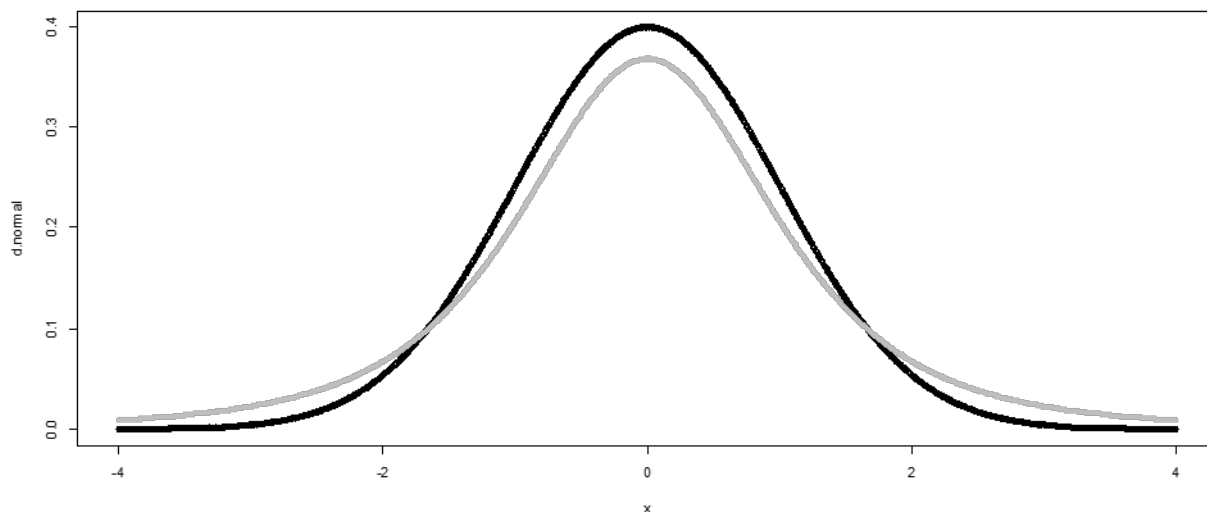
For the example below, the degrees of freedom parameter (named `degreefree`) is set equal to three. The random variable `x` is along the horizontal axis, which is scaled from -4 to 4 in increments of .005.

The normal distribution shows as the heavy black line; the t-distribution with `df=3` is the thin gray line. You access the normal distribution with "`dnorm (x)`" and the t-distribution with "`dt (x, degreefree)`." These are assigned to the variables names `d.normal` and `d.tdist`, respectively, which can then be used in the plots.

Note: in the last line of the code, setting "`lwd=5`" controls the thickness of the gray curve. You can play with changing this number if you want the actual curve to be plotted with a thicker or thinner line.

```
> # The first three lines plot the standard normal curve (in black)
> x = seq(-4, 4, .005)
> d.normal = dnorm (x)
> plot (x, d.normal)
> # The next three lines add the t-distribution curve (in gray)
> degreefree = 3
> d.tdist = dt (x, degreefree)
> lines (x, d.tdist, col="gray", lwd=5)
```

Here is the graph that results.



The fact that the t-distribution is more variable than the normal can be seen in the graph. The tails of the t-distribution (shown in gray) are “fatter” and the hump in the center is lower than it is in the normal (shown in black). If you do more plots with the df-values getting larger, the t-distribution graph approaches the same shape as the normal.

There are certain common levels of significance used in hypothesis tests, commonly denoted by the Greek letter alpha (α). The most common is $\alpha = 0.05$; other common values are 0.01 and 0.10. Corresponding to any specific value for α , there is a critical value. When you are using the t-distribution, a critical value is a t-score that defines the boundary between occurrences that “rare” and those that are “not rare.”

If you want to obtain critical values of the t-distribution, you specify alpha (α) as you did for the normal, and also a value for the degrees of freedom. The general syntax is: “qt (area to left of critical value, df).” Therefore, if you want α in the left tail, then you use α for the area to the left. If you want α in the right tail, you use $1 - \alpha$ as the area to the left.

For example, suppose you want a critical value with all of $\alpha = .05$ in the upper tail and $df = 3$. Since α is the area to the right of the critical value, then $1 - \alpha = .95$ is the area to its left. So the command is:

```
> UCV = qt (.95, 3)
> UCV
```

Then R returns the upper critical value.

```
2.353363.
```

The critical values returned by R match those in the tables supplied with most textbooks (but carry more decimal places). The advantage of using R to obtain critical values is that the tables only cover a few values of α and degrees of freedom, but R can handle others that are not included in the tables.

Section 13: How to Generate Chi-Square and F Distributions and Obtain Their Critical Values

(Uses no data files)

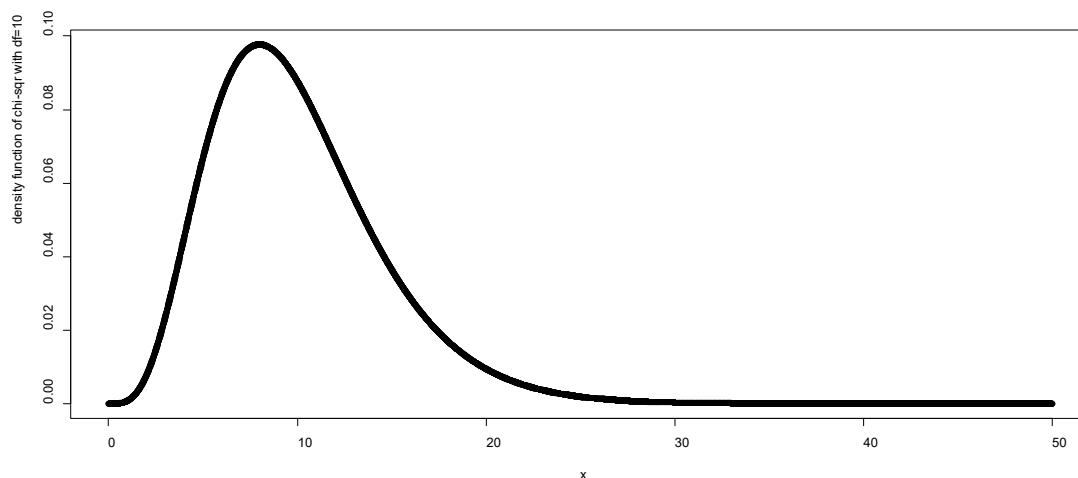
Two distributions that you will probably need are the chi-square distribution and the F-distribution. The chi-square distribution is used in some non-parametric tests and also in testing hypotheses about a variance. The F-distribution is used in ANOVA and in testing hypotheses that compare two variances.

Both require you to supply values for degrees of freedom. For the F-distribution, you will actually need to supply two df values – one goes with the numerator of a fraction involved in the testing process, and the other goes with the denominator. For now, the df-values will be randomly selected for purposes of illustration.

First consider the chi-square distribution; here is a typical graph. You need three commands to produce it. The first command scales the random variable x , on the horizontal axis, from 0 to 50 in increments of .005. The second command accesses the chi-square distribution using “`dchisq(x, df)`” -- for this example, using $df = 10$. In the example, the result is assigned to the variable `chidist`, which is then used in the third command for the vertical axis in the plot. The “`ylab`” part of the command lets you label the vertical axis.

```
> x = seq(0, 50, .005)
> chidist = dchisq(x, 10)
> plot(x, chidist, ylab = "density function of chi-sqr with df=10")
```

The resulting graph is:



There are certain common levels of significance used in hypothesis tests, commonly denoted by the Greek letter alpha (α). The most common is $\alpha = 0.05$; other common values are 0.01 and 0.10. Corresponding to any specific value for α , there is a critical value. When you are using a chi-square distribution, a critical value is a chi-square score that defines the boundary between occurrences that “rare” and those that are “not rare.”

You can obtain critical values of a chi-square distribution using “qchisq (area to the left, df).” For example, if you want the one-tailed, lower critical value that cuts off $\alpha=.025$ in the lower tail of the distribution in the graph above, the command would be as show below.

```
> LCV = qchisq (.025, 10)
> LCV
```

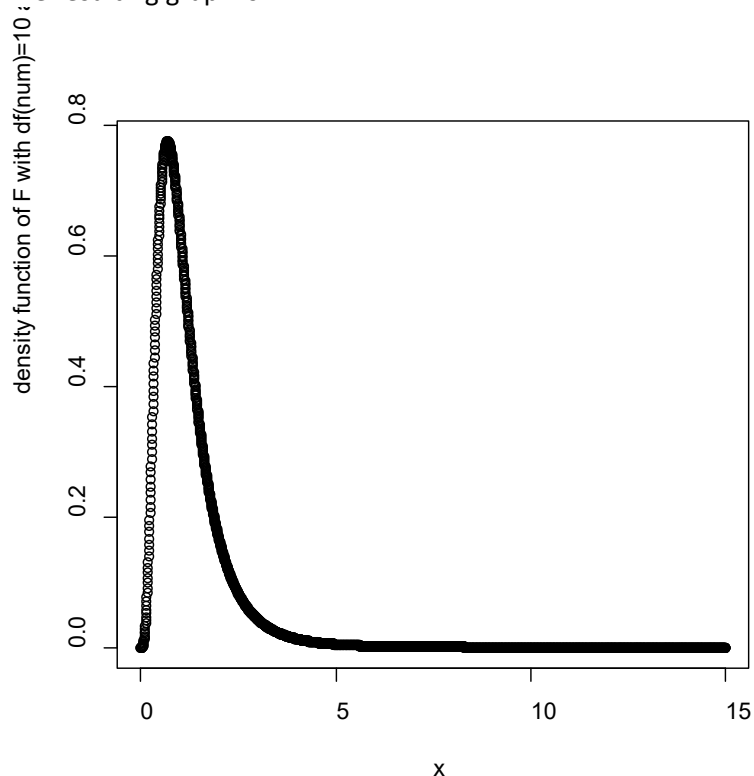
The critical value that results is:

```
3.246973
```

The F distribution can be handled similarly. For example, the following commands plot an F-distribution with degrees of freedom 10 and 12. The first number, 10 in this example, is the number of degrees of freedom for the numerator. The second number, 12 in this example, is the number of degrees of freedom for the denominator. The random variable x is on the horizontal axis, scaled from 0 to 15 in increments of .005. The F distribution is accessed using “df (x, 10, 12).” The results are assigned to the variable Fdist, which is then used in the plot. Again, the “ylab” part of the command lets you label the vertical axis.

```
> x = seq (0, 15, .005)
> Fdist = df (x, 10, 12)
> plot (x, Fdist, ylab = "density function of F with df(num)=10 and df(denom)=15")
```

The resulting graph is:



Critical values may be obtained with a similar process to that used before. For example, to get two-tailed critical values of the distribution in the graph with $\alpha = .05$, split α in half to account for the two tails. This means you would put an area of .025 in each tail. Consequently, the upper critical value is obtained by using $1-.025=.975$ as the area to its left. So you do the following.

```
>UCV = qf (.975, 10, 12)
>UCV
```

R returns the following upper critical value.

```
3.373553
```

Similarly, you enter the following two lines to get the lower critical value.

```
>LCV = qf (.025, 10, 12)
>LCV
```

R then produces the following lower critical value.

```
0.276171
```


Section 14: How to Generate and Graph a Binomial Distribution

(Uses no data files)

To generate a binomial distribution, you use the “`dbinom ()`” command. Inside the parentheses, you will need three items: (1) the name of the random variable that you are using for the number of “successes,” (2) the number of trials of the experiment, and (3) the probability of a success on any single trial. What R actually generates is the set of probabilities corresponding to the possible numbers of successes. The form of the command is:

```
> dbinom (variable name, size = no. of trials, prob = probability of a success on a single trial)
```

Here is an example.

Suppose twenty basketball players each take one turn at shooting a free throw. Based on previous attempts, the probability that an individual player will make the shot is 70%, or 0.7. Since there are twenty players, it is possible that none of them, all of them, or any number between, will make the shot. Assume that you want to obtain the probabilities corresponding to the numbers (0, 1, 2, ..., 19, 20) of players successfully making the shot.

First create a random variable that stands for the number of successes. In this case, a success is a player making the shot. Have R assign the values 0 through 20 to this variable.

```
> y = 0:20
```

Next assign the probability for success (making the shot) to another variable.

```
> p = .7
```

Now you can use these variables to create the appropriate binomial distribution. The first command defines the specific binomial function to generate the distribution and assigns the probabilities to the variable `Prob.y`. The second command displays the results.

```
> Prob.y = dbinom (y, size=20, prob=p)
> Prob.y
```

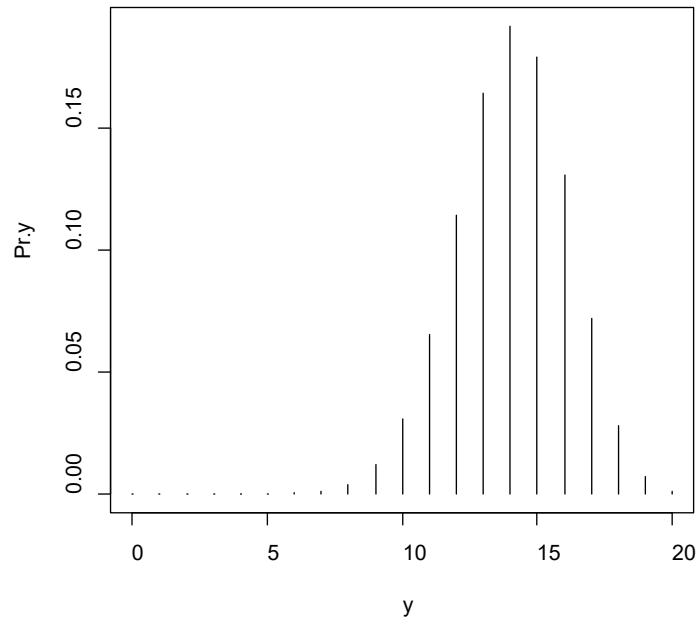
The output is the list of probabilities for 0 through 20 successes, in that order, as shown below. Note that these probabilities are in scientific notation, so the last one, for instance, means 7.97×10^{-4} .

```
3.486784e-11  1.627166e-09  3.606885e-08  5.049639e-07  5.007558e-06
3.738977e-05  2.181070e-04  1.017833e-03  3.859282e-03  1.200665e-02
3.081708e-02  6.536957e-02  1.143967e-01  1.642620e-01  1.916390e-01
1.788631e-01  1.304210e-01  7.160367e-02  2.784587e-02  6.839337e-03
7.979227e-04
```

Now you can graph the distribution if you wish. Use the “`plot ()`” command. In parentheses, you need: (1) the name of the variable for the number of successes, (2) the name of the variable holding the probabilities you just generated, and (3) the subcommand: `type="h"`).

```
> plot(y, Prob.y, type="h")
```

You get the following graph.



You can mentally, or with a pencil, draw a smooth curve over the upper ends of the bars to emphasize the shape of the distribution.

(Page intentionally left blank)

Parametric Methods: Sections 15-28

- 15. How to Run a One-Sample t-Test
- 16. How to Run a Two-Sample t-Test with Independent Samples
- 17. How to Run a Two-Sample t-Test with Paired Data
- 18. How to Perform a One-Tailed Hypothesis Test
- 19. How to Test a Claim about a Single Population Variance
- 20. How to Perform an F-Test to Compare Two Variances
- 21. How to Do One-Way ANOVA
- 22. How to Run a Repeated Measures ANOVA
- 23. How to Run a Two-Way ANOVA with or without Interactions
- 24. How to Run Mauchly's Test for Sphericity
- 25. How to Check Pairs of Data Values for Linear Correlation
- 26. How to Run a Simple Linear Regression
- 27. How to Obtain Residuals and Fits from a Regression Line and Check the Assumptions
- 28. How to Run a Basic Multiple Regression

Section 15: How to Run a One-Sample T-Test

(Uses data file: MO Life Expectancy.txt)

This example will test the hypothesis that the mean overall U.S. life expectancy is 76 years. You will use the Missouri life expectancy by county as your sample. This may not be random enough to provide a representative sample for the whole country, but proceed as if it is for this example.

First read in the data table, and check it for normality – see Section 10: “How to Check for Normality Using a Normal Probability Plot.” A portion of the data set is displayed there. There you will find the graphs for this data set. You can see there that this data set looks fairly normal, although there is some doubt about the lower end. But for purposes of this example, assume normality holds.

Now your hypotheses are: H_0 : Mean US life expectancy is 76 years
 H_1 : Mean US life expectancy is not 76 years
 Pre-select your level of significance; this example uses alpha (α) = .05

Run a t-test. In the command, you have to specify the variable name and the hypothesized mean. In this example, the column containing the overall MO life expectancies by county is called “Total” in the data table. The hypothesized mean is 76. The test is intended as a two-tail test.

```
> t.test (Total, mu = 76)
```

The resulting output is as shown at left. Explanations are added on the right.

One Sample t-test	
data: Total	
t = 3.8405, df = 114, p-value = 0.0002022	<p>←-This line tells you that the test statistic is 3.8405.</p> <p>df = sample size -1 = 115 -1 = 114 in this example.</p> <p>The p-value is the probability of getting a sample at least as extreme as yours if the null hypothesis is true. Note that this p-value is much smaller than your α level.</p>
alternative hypothesis: true mean is not equal to 76	<p>← Indicates a two-tail test; a one-tail would specify “greater than 76” or “less than 76”</p>
95 percent confidence interval:	
76.29051 76.90949	<p>← Tells you that you can be 95% sure that the the true mean lies between about 76.3 and 76.9.</p>
sample estimates:	
mean of x 76.6	<p>← Calculated sample mean</p>

The R output does not produce the t-critical value(s), but you can do that now. Specify the quantiles desired and the degrees of freedom to be used. In this example, since it is intended to be a two-tail test with $\alpha=0.05$, split α in half and use half for each of the lower and upper critical values. The lower critical value will be the 2.5th percentile; the upper critical value will be the 97.5th percentile. The degrees of freedom (df) value appears in the output above, and is 114 in this case.

To get the lower critical value, use the following.

```
> LCV = qt (.025, 114)
> LCV
```

The result that is returned is:

```
-1.980992
```

The next two lines will get you the upper critical value.

```
> UCV = qt (.975, 114)
> UCV
```

The output is:

```
1.980992
```

Since the test statistic 3.8405 falls outside the interval between the two critical values, you will reject the null hypothesis. Also, since the p-value = 0.0002022 is less than α , you will reject the null hypothesis.

NOTE: Comparing the test statistic against the t-critical value(s), and comparing the p-value to α , should both yield the same conclusion.

INTERPRETATION: The sample provides significant evidence that the US overall life expectancy is not 76 years. It does not, by itself, indicate whether the national mean is higher or lower than that.

Section 16: How to Run Two-Sample T-Tests (Independent Samples)

(Uses data file: Life Exp by Gender – nonpaired.txt)

This is an example of a t-test to compare means of two independent (unpaired) samples. The context is as follows. Male life expectancy in MO was recorded for 20 randomly selected counties. Female life expectancy in MO was recorded for 20 randomly selected counties, not necessarily the same ones. That is, the male and female data **is not** in pairs matched by county

The hypotheses to be tested are that mean life expectancy is the same for both genders vs. the alternative that it is not. Set up your null and alternative hypotheses and your level of significance.

Now your hypotheses are: H_0 : Mean MO life expectancies by county for both genders are equal (that is, the mean difference is zero).

H_1 : Mean MO life expectancy by county is not the same for both genders.

Pre-select your level of significance; this example uses alpha (α) = .05

```
> LifebyGender = read.table ("E:/Data Files/Life Exp by Gender - nonpaired.txt", header=TRUE)
> attach (LifebyGender)
> LifebyGender
```

A portion of the data set is shown below.

	Gender	Years
1	M	75.4
2	M	75.1
3	M	75.5
4	M	73.3
:		
20	M	72.6
21	F	79.6
22	F	81.4
23	F	78.5
24	F	80.7
:		
40	F	79.6

In general, the command to run a t-test to compare means of two groups is: “t.test (variable ~ group).” The ~ symbol means that the variable is being considered as a function of the group. So here, since the variable is Years and the group is Gender, you use the following statement.

```
> t.test (Years~Gender)
```

The resulting output is on the left; interpretation of the output is added on the right. Note that name of the two-sample t-test for independent samples is the “Welch Two Sample t-test.”

Welch Two Sample t-test

data: Years by Gender

t = 9.9765, df = 35.6, p-value = 7.479e-12

← Test statistic is 9.9765

df is calculated by a more complicated formula than in most textbooks.

p-value is about $7.5 \times 10^{-12} < \alpha$.

alternative hypothesis: true difference in means is
not equal to 0

← Indicates a two-tailed test. A one-tail test would specify "greater than" or "less than."

95 percent confidence interval:

4.134533 6.245467

← Tells you that you can be 95% confident that the interval between about 4.1 and 6.2 contains the true mean difference in life expectancy by gender.

sample estimates:

mean in group F mean in group M
79.295 74.105

← Means for the two samples.

Note that the df value that would be calculated by most textbooks is simpler, using the formula
(Male sample size - 1) + (Female sample size - 1) = (20 - 1) + (20 - 1) = 38.

The value produced by R involves a more complicated calculation but the two df values should be close.

The output does not provide the t-critical values, but you can make R find them. Put half of α in each tail, since this is a two-tailed test. Use the df value from the output above, that is, use 35.6.

Therefore, you use the following two lines to get the lower critical value.

```
> LCV = qt (.025, 35.6)
> LCV
```

The resulting lower critical value is:

```
-2.028886
```

Similarly, you use the next two lines to get the upper critical value.

```
> UCV = qt (.975, 35.6)
> UCV
```

Then R returns the following upper critical value.

2.028886

Since the test statistic 9.9765 is outside the interval bounded by these two critical values, you will reject the null hypothesis. Since the p-value is smaller than α , this also indicates that you will reject the null hypothesis.

INTERPRETATION: The sample provides sufficient evidence to conclude that mean life expectancy by county in Missouri is not the same for both genders. Note that this was a two-tailed test, so it does not indicate the direction of the difference. But you can compare the two sample means and observe the sample direction – on average, women live longer than men.

Section 17: How to Run Two-Sample T-Tests with Paired Data

(Uses data file: Life Exp by Gender – paired.txt)

This is an example of a t-test to compare the means of paired data. The comparison is done by running a one-sample t-test on the difference between them. The null hypothesis is that the mean difference is zero, i.e., the means are equal.

The context of the example is a comparison of male and female life expectancy in Missouri in the period from 2004 through 2012. A random sample of twenty counties is taken, and the male and female life expectancies for these counties are recorded in pairs. The difference is calculated: Female expectancy minus male expectancy.

Note that you could also calculate the difference the other way: Male expectancy minus female expectancy. But once you decide on a direction, you have to be consistent.

The null hypothesis to be tested is that the difference is zero, vs. the alternative that the difference is non-zero.

Now your hypotheses are: H_0 : Mean MO life expectancies by county for both genders are equal.

(Equivalently, the mean difference is zero.)

H_1 : Mean MO life expectancy by county is not the same for both genders.

(Equivalently, the mean difference is not zero.)

Pre-select your level of significance; this example uses $\alpha = .05$

```
> LifeExp = read.table("E:/Data Files/Life Exp by Gender - paired.txt" , header = TRUE)
> attach (LifeExp)
> LifeExp
```

A portion of data set appears as follows.

	County	Male	Female	Difference
1	Adair	75.4	79.9	4.5
2	Andrew	75.1	80.3	5.2
3	Cole	76.1	80.5	4.4
4	Cooper	75.3	79.4	4.1
5	Franklin	74.1	79.3	5.2
	:			
23	Worth	75.3	81.8	6.4
24	Wright	72.7	78.2	5.5

Now you can run the t-test with the hypothesized mean of Difference = 0.

```
> t.test (Difference, mu=0)
```

The resulting output is on the left; interpretive comments are added at right.

One Sample t-test

data: Difference

t = 24.9683, df = 23, p-value < 2.2e-16

← Test statistic is 24.9683.

df=number of pairs – 1 = 24 – 1 = 23.

The p-value = 2.2×10^{-16} , which is smaller than α .

alternative hypothesis: true mean is not equal to 0

← Indicates that this is a two-tail test. A one-tail test would specify “greater than” or “less than.”

95 percent confidence interval:

5.071067 5.987266

← So you that you can be 95% confident that the interval between about 5.07 and 5.99 contains the true population mean difference.

sample estimates:

mean of x 5.529167

← Calculated sample mean difference.

The output does not include the t-critical values. You can obtain these in the usual way.

The following two lines will get you the lower critical value; use df = 23 from the output above.

```
> LCV = qt (.025, 23)
> LCV
```

R returns the following:

```
-2.068658
```

Similarly, the next two lines will get you the upper critical value.

```
> UCV = qt (.975, 23)
> UCV
```

R returns the following:

```
2.068658
```

The test statistic t = 24.9683 is outside the interval bounded by the critical values, so you should reject the null hypothesis. Also, the p-value is less than α , so the null hypothesis is rejected on that basis.

INTERPRETATION: There is sufficient evidence to indicate that average (by county) male and female life expectancies are different in Missouri.

COMMENT: As presented here, the difference is already in the data set as its own column. If it was not, before running the test, you would have to make R calculate it by typing:

```
> Difference = Female – Male.
```

Section 18: How to Perform a One-Tailed Hypothesis Test

(Uses data files: Life Exp by Gender – paired.txt, Life Exp by Gender – nonpaired2.txt)

The default alternative hypothesis in R is always “not equal to,” that is, R defaults to a two-tailed hypothesis test. You can override this by specifying a direction. This goes into the “test” command after stating the value to be used for the null hypothesis. Also, if you are using a one-tailed test to compare two populations, you should arrange your data in two columns instead of one.

Here are two examples that you have already seen as two-tailed tests, but they are now presented as one-tailed tests. The advantage to the latter is that the direction of the difference in means is specified as part of the test.

EXAMPLE A (Paired Data)

The context of the first example is a comparison of male and female life expectancy in Missouri (2004-2012). A random sample of twenty counties is taken, and the male and female life expectancies for these counties are recorded. Then the difference is calculated: female expectancy minus male expectancy. The hypotheses to be tested are that the difference is zero, vs. the alternative that the difference is greater than zero. You can see the data set partially displayed in Section 17.

Your hypotheses are: H_0 : Mean MO life expectancies by county for both genders are equal.
(Equivalently, the mean difference is zero.)

H_1 : Mean MO life expectancy by county is greater for women than men.
(Equivalently, the mean difference is greater than zero.)

Pre-select your level of significance; this example uses alpha (α) = .05

```
> t.test (Difference, mu=0, alternative = "greater")
```

The output is as shown below on the left; interpretative comments are added on the right.

One Sample t-test

data: Difference

t = 24.9683, df = 23, p-value < 2.2e-16

← Test statistic is 24.9683.

Degrees of freedom=23.

The p-value is some value < 2.2×10^{-16}

alternative hypothesis: true mean is greater than 0

← Indicates upper tail test.

95 percent confidence interval:

5.149634 Inf

← One-sided confidence interval: you can be 95% sure that the true mean difference is > 5.149634

sample estimates:

mean of x 5.529167

← Calculated sample mean difference.

The output, as usual, does not include the t-critical value. You can obtain this in the usual way, but remember that all of $\alpha = .05$ belongs in the upper tail. Use $df = 23$ from the previous output.

```
> UCV=qt (.95, 23)
> UCV
```

The critical value for the upper tail test is then returned by R.

```
1.713872
```

The test statistic of 24.9683 exceeds the upper critical value, and the p-value is less than α . Both of these indicate that you should reject the null hypothesis.

INTERPRETATION: There is sufficient evidence to indicate that the mean female life expectancy is greater than the mean male life expectancy in Missouri.

COMMENT: For a lower-tailed test, you would have specified: `alternative = "less."`

EXAMPLE B (Unpaired, i.e., Independent Data)

This is an example of a t-test to compare means of two independent (unpaired) samples. The context is as follows. Male life expectancy in MO was recorded for 20 randomly selected counties. Female life expectancy in MO was recorded for 20 randomly selected counties, not necessarily the same ones. That is, the male/female data is not in pairs matched by county.

You can read in and display the data set in the usual way.

```
> LifeExp = read.table ("E:/Data Files/Life Exp by Gender – nonpaired2.txt" , header = TRUE)
> attach (LifeExp)
> LifeExp
```

A portion of the data set is shown below.

MYears	FYears
75.4	79.6
75.1	81.4
75.5	78.5
73.3	80.7
73.1	78.9
75.5	80.5
:	
75.9	79.3
73.6	79.0
69.5	81.3
75.9	80.3
72.6	79.6

Your hypotheses are: H_0 : Mean MO life expectancies by county for both genders are equal.
 H_1 : Mean MO life expectancy by county is less for men than women.
 Pre-select your level of significance; this example uses alpha (α) = .05

```
> t.test (MYears, FYears, alternative="less")
```

The resulting output, with comments added at right, is as follows.

```
Welch Two Sample t-test

data: MYears and FYears

t = -9.9765, df = 35.6, p-value = 3.739e-12    ← Test statistic is -9.9765.

                                           Degrees of freedom = 35.6

                                           The p-value=3.739 x 10-12

alternative hypothesis: true difference in means is less than 0    ← Indicates lower-tailed test.

95 percent confidence interval:              ← Tells you that you can be 95% confident that the
-Inf    -4.311453                            mean male life expectancy is at least 4.311453 years
                                           less than the mean female life expectancy.

sample estimates:
mean of x    mean of y
  74.105      79.295                                ← Calculated sample means.
```

The output, as usual, does not include the t-critical value. You can obtain this in the usual way, but remember that all of $\alpha = .05$ belongs in the lower tail since this was a lower-tailed test. Use $df = 35.6$ from the previous output.

```
> LCV = qt (.05, 35.6)
> LCV
```

R returns a critical value for a lower tail test.

```
-1.688799
```

The test statistic exceeds (negatively) the lower critical value, and the p-value is less than α . Both of these indicate that you should reject the null hypothesis.

INTERPRETATION: There is sufficient evidence to indicate that the mean male life expectancy is less than the mean female life expectancy in Missouri.

Section 19: How to Test a Claim about a Single Population Variance

(Uses data file: Life Exp by Gender – nonpaired.txt)

Sometimes you will want to test a claim about a single population variance. For example, the National Bureau of Economic Research estimates the standard deviation in adult life spans in the United States as being around 15 years. Equivalently, the variance is approximately $15^2 = 225$.

Lifespans in the state of Missouri might be more variable, less variable, or about the same as lifespans nationally. You can use the data set “Life Exp by Gender – unpaired” to test this. Assume you want a two-sided test, with level of significance $\alpha = .05$. Since $1 - .05 = .95$, this corresponds to a 95% confidence level for the confidence interval.

As of this writing, the single population variance test is not included in base R, so you will first have to install and load the package that contains the appropriate test. The package you want is: EnvStats. See Section 3: “How to Find, Install and Load R Packages.”

Once that is done, you proceed as usual to read in and attach the data set. In this test, you will be using the entire column Years without differentiating by gender. You are simply going to compare the variance in the overall Missouri life expectancy to the national value of 225.

```
> Data = read.table ("E:/Data Files/Life Exp by Gender - nonpaired.txt", header = TRUE)
> attach (Data)
> Data
```

A portion of the data is shown below.

	Gender	Years
1	M	75.4
2	M	75.1
3	M	75.5
4	M	73.3
:		
:		
38	F	81.3
39	F	80.3
40	F	79.6

Assuming you have installed and loaded the EnvStats package (see section 3 of this Manual), you are ready to run the variance test. In the general case, this looks like:

```
> varTest (Variable Name, form of alternative, confidence level, hypothesized variance from the null)
```

Notice that the command requires the confidence level, not the level of significance. To be clear, you should plan your test.

Your hypotheses are: H_0 : Variance of MO life expectancies by county equals 225.

H_1 : Variance of MO life expectancy by county does not equal 225.

Pre-select your level of significance; this example uses $\alpha = .05$, so confidence level is .95.

Specifically then, for this example with a two-tailed test, the command you want would therefore be:

```
> varTest (Years, alternative = "two.sided", conf.level = .95, sigma.squared = 225)
```

The output is shown below. Actual output appears on the left; explanatory comments are on the right.

Results of Hypothesis Test

Null Hypothesis:	Variance = 225	← Hypothesized population variance.
Alternative Hypothesis:	True variance is not equal to 225	← Indicates two-sided test.
Test Name:	Chi-Squared Test on Variance	
Estimated Parameter(s):	variance = 9.54359	← This is the sample variance.
Data:	Years	← Tells you the variable name.
Test Statistic:	Chi-Squared = 1.654222	
Test Statistic Parameter:	df = 39	← Chi-squared distribution uses degrees of freedom n-1.
P-value:	4.161694e-20	← Given in scientific notation, this means 4.2×10^{-20} .

Note that the p-value is much less than α , so the null hypothesis is rejected.

You also get the following two lines at the end of the output.

95% Confidence Interval:	LCL = 6.403985
	UCL = 15.734966

These last two lines of output represent the endpoints of a 95% confidence interval for the standard deviation of the population lifespan, based on the Missouri data. Note that it does include the hypothesized standard deviation of $\sigma=15$ that corresponds to the hypothesized value of the variance, although just barely. This would seem to contradict the conclusion based on the p-value. It is probably an indication that the sample data is not a representative random sample.

There are several issues with the data set that may be contributing to this discrepancy. For one thing, the data was obtained by county and not weighted at all according to the size of the county. Further, this particular data set is borderline in terms of normality, and the variance test has normality as an assumption. And finally, the actual test is for the variance but the confidence interval is for the standard deviation.

Section 20: How to Perform an F-Test to Compare Two Variances

(Uses data file: Life Exp by Gender – nonpaired.txt)

Sometimes you will want to test whether or not two samples come from populations with a common variance. Usually this is because you ultimately want to use a two independent sample t-test to compare the means, and one of the assumptions for that test is that the populations have a common variance.

The variance test creates a ratio of one variance over the other. If the ratio is “close to” one, the variances are considered to be equal; otherwise they are not. The distribution is an F distribution with degrees of freedom given by:

$df(\text{numerator}) = \text{first sample size} - 1$

$df(\text{denominator}) = \text{second sample size} - 1.$

The example here compares the variances of male and female life expectancies by county in Missouri. An earlier example used a t-test for two independent samples to compare their means. (See section 16.)

Your hypotheses are: H_0 : Variances of MO life expectancies by county are the same for both genders.

H_1 : Variances are not the same for both genders.

Pre-select your level of significance; this example uses alpha (α) = .05.

As usual, read in and attach the data set. You can display it if you wish.

```
> LifebyGender = read.table ("E:/Data Files/Life Exp by Gender - nonpaired.txt", header = TRUE)
> attach (LifebyGender)
> LifebyGender
```

A portion of the data set is shown below.

	Gender	Years
1	M	75.4
2	M	75.1
3	M	75.5
4	M	73.3
:		
:		
38	F	81.3
39	F	80.3
40	F	79.6

Now you are ready to run the test to compare the two variances.

```
> var.test (Years ~ Gender)
```

The resulting output is on the left; explanatory comments have been added on the right.

F test to compare two variances

data: Years by Gender

F = 0.5878, num df = 19, denom df = 19, p-value = 0.2557

← Test statistic is 0.5878.

Both samples have n=20 so both df = 19.

alternative hypothesis: true ratio of variances is not equal to 1

← Alternative says variances unequal.

95 percent confidence interval:

0.2326461 1.4849694

← This is a confidence interval for the ratio of the variances

sample estimates:

ratio of variances 0.5877689

← Calculated ratio of sample variances.

The p-value of 0.2557 is greater than α . This tells you to fail to reject the null hypothesis. You do not have evidence to believe that the variances are unequal.

You can also have R find the .025 and .975 quantiles of the F(19,19) distribution, as follows.

```
> qf (.025, 19, 19)
```

R returns the following.

```
0.3958122
```

Now for the other quantile.

```
> qf (.975, 19, 19)
```

This time R returns the following.

```
2.526451
```

These are lower and upper critical values for a confidence interval for the ratio of the population variances. The test statistic falls between these two quantiles, which confirms your “fail to reject” conclusion.

Section 21: How to Do One-Way ANOVA

Compare Means of Multiple Populations

(Uses data file: Anxiety.txt)

You use one-way ANOVA when you want to compare the means of more than two populations, where the populations are distinguished by a single characteristic. It is similar to a t-test but uses sample data from three or more populations.

This example uses a fictional data set of “anxiety scores” of 156 college students in four types of institutions (coded as LPR = large private university, SPR = small private university, STA = state university and COM = community college). The goal is to determine whether or not the mean score is the same or different by type of institution. The null hypothesis is that all four types have the same mean anxiety score; the alternative is that at least one mean is different. Use level of significance $\alpha = .05$.

First read in the data table giving the scores and name it “Anxiety.” Be sure to “attach” it and display it if you wish.

```
> Anxiety = read.table ("E:/Data Files/Anxiety.txt", header = TRUE)
> attach (Anxiety)
> Anxiety
```

A partial display of the output is as follows.

	Type	AnxScore
1	LPR	25
2	SPR	11
3	STA	8
4	COM	17
	:	
153	SPR	33
154	SPR	30
155	COM	21
156	COM	19

Before trying to run the test, you should plan it.

Your hypotheses are: H_0 : Mean anxiety scores are the same for all four types of institutions.

H_1 : At least one type has a different mean anxiety score.

Pre-select your level of significance; this example uses $\alpha = .05$.

You then have to create a linear model of the score as a function of the institution type. This is required because ANOVA can only be performed on a model, not on the data table itself. In the following command, “lm” stands for “linear model.”

```
> AnxModel = lm (AnxScore ~ Type)
```

Then you can run the “anova” command on the model.

```
> anova (AnxModel)
```

The output is below. For space reasons, the explanatory comments are in a paragraph below the output (instead of on the right as usual).

Analysis of Variance Table

Response: AnxScore

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Type	3	261.2	87.076	0.8578	0.4645
Residuals	152	15429.9	101.513		

Here is what the output tells you. The F-value is the test statistic. The p-value is the value in the column on the far right, labelled "Pr(>F)." The other values in the output are used in calculating F. Since the p-value is larger than α , you fail to reject the null hypothesis.

INTERPRETATION: The evidence is not sufficient to conclude that any institution type has a different mean anxiety score.

Notice, however, that the output does not give you the individual institution type sample means. If you want those, you have a couple of choices.

- (1) Create another data file in which the scores for different types are listed in separate columns, and then find the mean for each column.

OR

- (2) Apply a logical function to the original data set to create four vectors, each containing the data for a specific type of institution.

Here is how to do option (2). The double-equal sign (==) is a logical operator that tells R to compare each type listed in the data set to the item on the right, to see whether they match. The results are stored in the variables LPRScores, SPRScores, STAScores and COMScores, respectively.

```
> LPRScores = AnxScore*(Type=="LPR")
> SPRScores = AnxScore*(Type=="SPR")
> STAScores = AnxScore*(Type=="STA")
> COMScores = AnxScore*(Type=="COM")
```

Now total the values in LPRScores.

```
> sum (LPRScores)
```

The sum of the large private institution scores returned is:

```
764
```

Do the others similarly:

> sum (SPRScores)	← R returns 921.
> sum (STAScores)	← R returns 825.
> sum (COMScores)	← R returns 801.

Now get the counts by institution type. The “ftable” command will get the counts.

```
> ftable (Type)
```

The resulting output follows; note that it is listed alphabetically.

```
COM LPR SPR STA
39   39  40  38
```

You can now use the score sums and counts per institution type to obtain the four sample means. For instance, you know that the sum of the scores for all the LPR students is 764, and you know there are 39 students from LPR's. So the LPR sample mean is $764/39 = 19.589$ (to three decimal places).

The following commands do this for all four institution types.

> meanLPR = sum (LPRScores)/39; meanLPR	← R returns 19.58974
> meanSPR = sum (SPRScores)/40; meanSPR	← R returns 23.025
> meanSTA = sum (STAScores)/38; meanSTA	← R returns 21.71053
> meanCOM = sum (COMScores)/39; meanCOM	← R returns 20.53846

Now for a bit of assumption checking. ANOVA is based on an underlying assumption that the variance is the same across all categories, in this case, for all four institution types. There are several standard tests for homogeneity of variance, one of them due to Bartlett. The null hypothesis is that all of the variances are equal. The alternative is that at least one variance is different from the others. Here alpha (α) = .05 is used as the level of significance.

```
> bartlett.test (AnxScore ~ Type)
```

The resulting output is as follows.

```
Bartlett test of homogeneity of variances

data: AnxScore by Type
Bartlett's K-squared = 0.55801, df = 3, p-value = 0.906
```

Since the p-value is greater than α , you fail to reject the null hypothesis about the variances. Therefore, it appears that the variances are the same across the four institution types, and consequently the use of ANOVA as the main test is acceptable. If you had been forced to reject the common variance assumption, you would need to find an alternative approach to your main test comparing the means. One option would be the Kruskal-Wallis test that is presented in Section 33 of this Manual.

Section 22: How to Run a Repeated Measures ANOVA

Compare Multiple Population Means in the Case of Repeated Measures

(Uses data file: AnxietyRepeat.txt)

The example uses the data set AnxietyRepeat.txt. This data set contains scores on an Anxiety Test, repeated three times on thirty-six (fictional) students. The test is first given during their first term in college (labelled Fall1), repeated second term (labelled Spr1) and then again in their third term (labelled Fall2). The same students are tested all three times. The goal of the testing is to see whether or not, on average, students' anxiety levels are consistent or change over time as they adjust to college.

First read in data table giving the anxiety scores and attach it. You can then display the data set if you wish.

```
> Data = read.table ("E:/Data Files/AnxietyRepeat.txt", header = TRUE)
> attach (Data)
> Data
```

A partial display of the output is as follows.

	ID	Test.Session	Anx.Score
1	S1	Fall1	25
2	S1	Spr1	22
3	S1	Fall2	33
4	S2	Fall1	11
5	S2	Spr1	5
6	S2	Fall2	20
:			
:			
103	S35	Fall1	22
104	S35	Spr1	27
105	S35	Fall2	17
106	S36	Fall1	24
107	S36	Spr1	30
108	S36	Fall2	37

As a first look, you may want to obtain the sample means for each test session. You therefore need to “split out” each session from the whole data set. To see how to do this, refer to Section 6: “How to Extract Particular Data Items or Sequences of Them.” If you follow example D in that section, you will obtain the summary information for each test session.

Test Session	Sample Mean Anxiety Score	Sample Variance in Anxiety Score
Fall1	19.17	90.77
Spr1	23.78	142.92
Fall2	21.36	90.24

The summary shows that the sample means are different but you don't know whether or not the population means are different. If you choose to run a two-way ANOVA to test whether or not the population means are the same or different, you have two options. You can run it as:

- (1) Anx.Score as a function of person (ID) and Test.Session, OR
- (2) Anx.Score as a function of person(ID) and Test.Session, plus an interaction term (like a practice effect).

First set up your hypotheses; assume your chosen level of significance is alpha (α) = .05.

H_0 : The means of the anxiety scores are the same for all test sessions.

H_1 : The mean anxiety score for at least one test session is different.

As shown below, this example has no interaction term, because there is no reason to think that anxiety levels are affected by repetition of the testing process. The first command creates a linear model with Anx.Score as a function of Test.Session and ID (individual person). The second command runs an analysis of variance on the model.

```
> Repeat.Model = lm (Anx.Score ~ Test.Session + ID)
> anova (Repeat.Model)
```

The resulting output is below. For space reasons, the explanatory comments are in the paragraph below the box, rather than inside it as usual.

Analysis of Variance Table
Response: Anx.Score

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Test.Session	2	383.0	191.509	2.1971	0.11874
ID	35	5235.9	149.597	1.7162	0.02781 *
Residuals	70	6101.6	87.166		

INTERPRETATION:

The F-values are in the next-to-last column. They are actually the ratios:

$$\begin{aligned} &(\text{Mean Squares of Test.Session})/(\text{Mean Squares of Residuals}) = 191.509/87.166 = 2.1971 \\ &\text{and} \\ &(\text{Mean Squares of ID})/(\text{Mean Squares of Residuals}) = 149.597/87.166 = 1.7162 \end{aligned}$$

These F values are the test statistics, and the entries in the last column are their corresponding p-values. Here, the p-value for the test session is 0.119 (rounded), which is larger than α . So there is no reason to believe that mean anxiety scores change over time. (What this might mean: adjustment to college is not a major source of anxiety. Other factors, such as course load in a particular semester or personal issues, may produce anxiety more randomly over time.)

You might notice, however, that the p-value for ID (i.e., individual student) is about 0.028. That is less than α , and therefore is small enough to be statistically significant. While you did not have a hypothesis

about it, it tells you that the data suggest that mean anxiety level varies by student. This should not be a surprise, in this case, since some people have more anxiety than others.

Note 1: See also section 24 for a test to check the data set for sphericity. This is an underlying assumption that should be satisfied if you want to use repeated measures ANOVA.

Note 2: The model above has no interaction term. If the test had been a test of some kind of skill, rather than anxiety, there might very well have been a practice effect. This could have been tested by including an interaction term in the model. The command would then be as follows, with the interaction term showing up as a “product” expression after the last “+” in the command, highlighted below.

```
> Repeat.Model = lm (Anx.Score ~ Test.Session + ID + Test.Session*ID)
> anova (Repeat.Model)
```

This kind of model, with an interaction term, is discussed in more detail in Section 23.

Section 23: How to Run a Two-Way ANOVA

Compare Multiple Population Means with Two Factors and Potential Interaction

(Uses data file: Anxiety By Major and Term.txt)

This data contains scores on an Anxiety Test. The test is given to a randomly selected group of twelve statistics majors, twelve engineering majors and twelve physics majors during fall of their first year in college (Fall1). The test is then given to another randomly selected group of twelve students from each major during their second term (Spr1). Finally, this is done again with a third randomly selected group of twelve students from each major in their third term (labelled Fall2). The goal of the testing is to see whether or not, on average, students' anxiety levels are consistent or change over time as they adjust to college. Also, because some semesters have a heavier course load than other semesters in the same major, it is believed that there may be an interaction effect between Major and Test.Session. Note that this is NOT a repeated measurement process because different students are involved at each session.

First read in data table giving the anxiety scores, attach it and display it if you wish.

```
> Data = read.table ("E:/Data Files/Anxiety By Major and Term.txt", header = TRUE)
> attach (Data)
> Data
```

A partial display of the data is as follows.

	ID	Major	Test.Session	Anx.Score
1	S1	Statistics	Fall1	25
2	S2	Statistics	Fall1	11
3	S3	Statistics	Fall1	8
:				
:				
13	S13	Engineering	Fall1	12
14	S14	Engineering	Fall1	45
15	S15	Engineering	Fall1	6
:				
:				
25	S25	Physics	Fall1	27
26	S26	Physics	Fall1	27
27	S27	Physics	Fall1	18
:				
:				

(and then similar for all of Spr1 followed by all of Fall2)

Then set up your hypotheses; there are actually three sets of hypotheses here.

Set 1: H_0 : The mean anxiety score is the same for all majors.
 H_1 : The mean anxiety score for at least one major is different.

Set 2: H_0 : The mean anxiety score is the same for all test sessions.
 H_1 : The mean anxiety score for at least one test session is different.

Set 3: H_0 : There is no interaction between a student's major and the test session.

H_1 : There is an interaction, i.e., students in at least one major exhibit higher or lower anxiety at a particular test session.

Assume your level of significance is $\alpha = .05$.

You want to create a linear model that has the anxiety scores as a linear function of major, test session and an interaction term. You need two lines of code. The first line creates the linear model. Notice that the interaction term appears as a "product" expression after the last "+" in the line creating the model; this term is highlighted below. The second line runs an analysis of variance on the model.

```
> Model = lm (Anx.Score ~ Major + Test.Session + Major*Test.Session)
> anova (Model)
```

The output of the anova is below.

Analysis of Variance Table

Response: Anx.Score

	Df	Sum Sq	Mean Sq	F value	Pr(>F)	
Major	2	129.5	64.731	0.5762	0.5639	← p-value for Hypotheses Set 1.
Test.Session	2	383.0	191.509	1.7048	0.1871	← p-value for Hypotheses Set 2.
Major:Test.Session	4	87.0	21.745	0.1936	0.9412	← p-value for Hypotheses Set 3.
Residuals	99	11121.1	112.334			

INTERPRETATION: The p-values are in the rightmost column. As you can see, none of them are smaller than α . Therefore, neither major, test session, nor interaction appears to have any significant effect on anxiety scores.

Section 24: How to Run Mauchly's Test for Sphericity

(Uses data file: AnxietyRepeat2.txt)

In Section 22: “How to Run a Repeated Measures ANOVA,” the example uses the data set AnxietyRepeat.txt. This data set contains scores on an Anxiety Test, repeated three times on thirty-six (fictional) students. The test is first given during their first term in college (labelled Fall1), repeated second term (labelled Spr1) and then again in their third term (labelled Fall2). The goal of the testing is to see whether or not, on average, students' anxiety levels are consistent from one term to the next, or whether they change over time as students adjust to college. In Section 22: “How to Run a Repeated Measures ANOVA,” you will find the following hypothesis test.

H_0 : The means of the anxiety scores are the same for all test sessions.

H_1 : The mean anxiety score for at least one test session is different.

<pre>> Repeat.Model = lm (Anx.Score ~ Test.Session + ID)</pre>	<p>← Creates a linear model with Anx.Score as a function of Test.Session and ID (individual person)</p>
<pre>> anova (Repeat.Model)</pre>	

The resulting output is repeated below for reference; see Section 22 for the detailed comments on it.

Analysis of Variance Table					
Response: Anx.Score					
	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Test.Session	2	383.0	191.509	2.1971	0.11874
ID	35	5235.9	149.597	1.7162	0.02781 *
Residuals	70	6101.6	87.166		

One key assumption for repeated measures ANOVA to be valid is sphericity. Sphericity means that the variances of the differences between all possible pairs of “within subject” conditions are equal. In this example, it means that you would need to have equal variances for the three differences: Fall1 - Spring1, Fall1 - Fall2, and Spring1 - Fall2.

H_0 : Data appears to satisfy the sphericity property.

H_1 : Data does not appear to satisfy the sphericity property.

Preset your level of significance; for this example, assume that $\alpha = .05$

You can test this sphericity property using a test called Mauchly's Test. In R, it is included in the package: car. Install this package if necessary. You can find an explanation of how to install a package in Section 3 of this Manual. Now you are ready to set up for the test.

Preliminary Step. Rearrange the data set so that it is in rows by student, columns by test session. But leave out the headings. Here that has been done for you and the rearranged data set is called: AnxietyRepeat2.txt. But if you are working on other data, you may want to put it into a spreadsheet to reconfigure it. Then proceed as follows.

Here is an overview of the steps you will follow. Details are supplied, step-by-step, after the overview.

Overview:

1. Read in this new data set.
2. Convert it from a table to a matrix.
3. Tell R the names to use in the model to indicate the repeated measurements.
4. Load the package: car, if not already done.
5. Create the linear model using the matrix.
6. Run the repeated measures analysis on the resulting model and save the results.

Step 1. Read in the rearranged data set, attach it, and have a look at it to be sure it is in the right form.

```
> Data = read.table ("E:/Data Files/AnxietyRepeat2.txt", header = FALSE)
> attach (Data)
> Data
```

The first few lines of data are now arranged as shown below. Notice that where you would ordinarily have headings in a table, R has written the labels V1, V2 and V3. This is OK, because you do not want the actual test session labels to go into the matrix. V1 is the first set of scores from Fall1, V2 is the set from Spring1, and V3 is the last set from Fall3.

The numbers on the far left are not part of the data; R is just numbering the lines as usual.

	V1	V2	V3
1	25	22	33
2	11	5	20
3	8	31	13
4	17	13	18
5	7	20	13
:	:	:	:
:	:	:	:

Step 2. Convert this new table to a matrix. The commands you need are as follows.

```
> AnxietyMatrix = as.matrix (Data, rownames.force = NA)
> AnxietyMatrix
```

The “as.matrix” command changes the table to a matrix. Since you called the table “Data” when you read it into R, that name is the first item in the parentheses. The second item, “rownames.force” is set to NA because you are going to supply your own names in the next step.

Step 3. Make a list that tells R what names you want to use for the repeated measurements (also called levels of the factor). There is a catch though; R will try to use the names for the “levels of the factor” in alphabetical order. Therefore, YOU DO NOT want to use Fall1, Spring1, Fall2 as the names in your matrix. That would result in Spring1 being put last, and it should not be. Instead use something like Term1, Term2 and Term3 that will stay in the right order when alphabetized.

You make the list by using the command `c("Term1", "Term2", "Term3")` inside the `"factor"` command, as shown below. In order to be able to refer to this later, you need to give it a name; here it is called `"design"` but you can use a different name if you want.

```
> design = factor(c("Term1", "Term2", "Term3"))  
> design
```

The resulting output that shows what `"design"` is as follows:

```
[1]      Term1      Term2      Term3  
Levels: Term1      Term2      Term3
```

Step 4. Load the package: `car`. See Section 3: `"How to Find, Install and Load R Packages."`

Step 5. Create the linear model using the matrix. In step 2 of this example, you called it `AnxietyMatrix`. So that is what goes into the command. The last part of the command (that says `~1`) is telling R what kind of matrix to compare your matrix against. Also, note that you won't see any output until Step 6.

```
> AnxietyModel = lm (AnxietyMatrix ~ 1)
```

Step 6. Run an ANOVA on this model. Note several things.

When you are working in the package `"car"`, the first letter of the command to do an analysis of variance has to be capitalized. This is different from when you run the same test in the basic R program, where it is not capitalized.

The command is applied to the model you just created, which you named `AnxietyModel`.

The rest of the command (highlighted below) is telling R to use the names that you specified in `"design"` as the labels for the repeated measurements/levels of the factor. If you always use the name `"design"` for your labels, you can just copy this part of the command as written.

```
> results = Anova (AnxietyModel, idata = data.frame(design), idesign = ~design, type = "III")  
> summary (results, multivariate = FALSE)
```

Here is the output from these two lines of code. In the package `"car"`, the output will not include the effect of the individual student by ID. It will only show the effect of the repeated test sessions. This is another difference from the version of the ANOVA command in basic R. It is fine because you are really running this analysis to get the results of Mauchly's sphericity test anyway.

The output is relatively long because it has several parts. The first part of it is the analysis of variance repeated all over again by the `"car"` package. The second part of it is the results of Mauchly's sphericity test.

Explanatory comments have been added on the right.

Univariate Type III Repeated-Measures ANOVA Assuming Sphericity ← This is just the anova all over again in “car.”

	SS	num Df	Error SS	den Df	F	Pr(>F)
(Intercept)	49622	1	5235.9	35	331.7085	<2e-16 ***
design	383	2	6101.6	70	2.1971	0.1187 ← Same p-value as before.

Mauchly Tests for Sphericity ← This is the start of the output about the sphericity test.

	Test statistic	p-value
design	0.96142	0.5123 ← p-value > α , so the data appears to satisfy sphericity.

Greenhouse-Geisser and Huynh-Feldt Corrections for Departure from Sphericity ← Correction factors that can be used if sphericity fails; beyond the scope of this discussion.

	GG eps	Pr(>F[GG])
design	0.96285	0.1209

	HF eps	Pr(>F[HF])
design	1.017791	0.1187361

Section 25: How to Check Pairs of Data Values for Linear Correlation (Pearson's r)

(Uses data file: HeightWeight.txt)

Frequently, you will have a set of data consisting of pairs of measurements on the same individual, such as a person's height and weight. You may want to determine whether or not these measurement pairs fall approximately on a line. To determine this, the standard method to use is linear correlation.

Here is an example, using the heights (in inches) and weights (in pounds) of 25 fictional army recruits. The question is whether or not they are linearly related. First read in the table of data, make it available to R by "attaching" it and then display it.

```
> HWTable = read.table("E:/Data Files/HeightWeight.txt", header = TRUE)
> attach(HWTable)
> HWTable
```

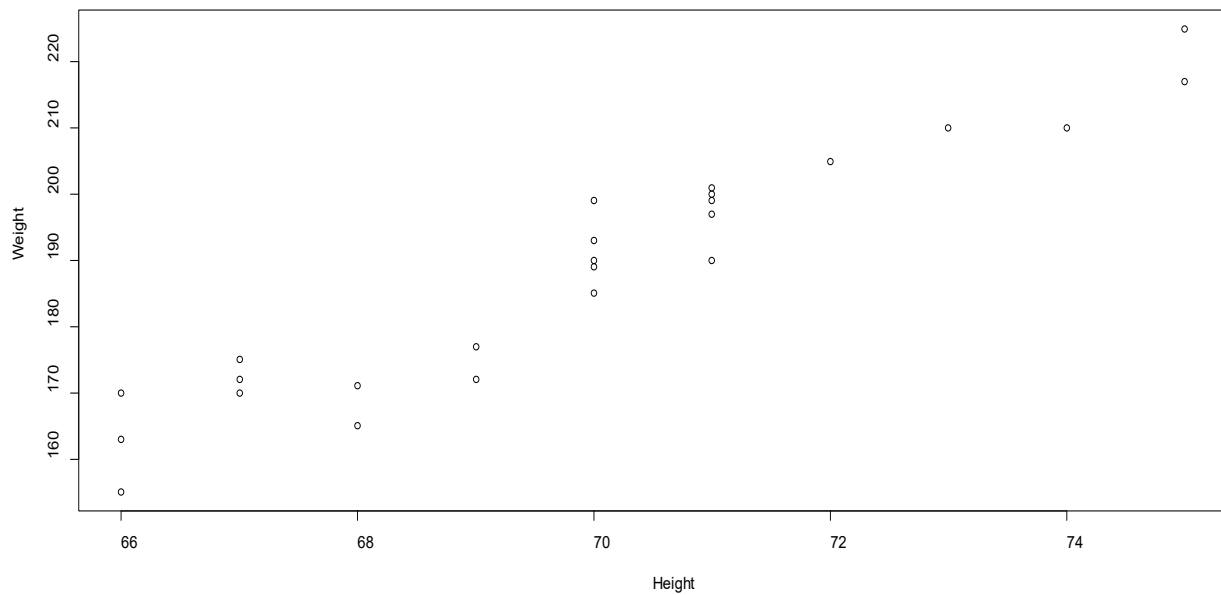
The resulting output is the following.

	Height	Weight
1	67	175
2	73	210
3	70	189
4	75	225
5	70	193
6	71	197
7	71	200
8	75	217
9	68	165
10	74	210
11	67	170
12	71	201
13	66	163
14	68	171
15	71	190
16	66	170
17	67	172
18	69	172
19	70	199
20	69	177
21	71	199
22	66	155
23	72	205
24	70	190
25	70	185

Now create the scatterplot. This is a good idea to do first, as it will help you visualize your data. The first variable that you list will go on the horizontal axis; the second one will go on the vertical axis.

```
> plot(Height, Weight)
```


The resulting graph is as shown below.



This graph certainly looks as if a rising line would fit fairly well through the data set. To get confirmation for this, you should now find the correlation coefficient. The command to do this is “cor” with the names of the variables in parentheses.

```
> cor (Height, Weight)
```

R returns the sample correlation coefficient, called Pearson’s r .

```
0.9540717
```

In this example, the value is positive and very close to one. It supports the belief that a rising line is a good way to represent the relationship between height and weight of army recruits.

Section 26: How to Run a Simple Linear Regression

(Uses data file: HeightWeight.txt)

Suppose you have already made a scatter plot and run a correlation for a set of (X,Y) data pairs, and both indicate that a straight line will be a good fit to the data. Usually you then want to know the specific line that fits the data best, called the regression line.

Refer back to the example of army recruits' height-weight data. You have already seen the scatterplot and the correlation in Section 25 of this Manual. To get the regression line, you continue from your previous work. Here is a repetition of the code from the earlier work in case you need it.

```
> HWTable = read.table ("E:/Data Files/HeightWeight.txt", header = TRUE)
> attach (HWTable)
> HWTable
> plot (Height, Weight)
> cor (Height, Weight)
```

As displayed in the previous section, the scatterplot looked fairly linear and the sample correlation coefficient was 0.9540717, indicating a strong positive linear relationship.

Now you can move on to actually testing the hypothesis that the correlation is significant. That is, you will test the hypotheses:

H_0 : The true population correlation is zero (no linear relationship).

H_1 : The true population correlation is not zero (there is a linear relationship).

Preset your level of significance. This example uses alpha (α) = .05.

```
> cor.test (Height, Weight)
```

The output from this test follows, with some interpretation added on the right-hand side. Note that it was not really necessary to run the correlation before, as you did in Section 25, because it is included as part of the output of the correlation hypothesis test.

Pearson's product-moment correlation

data: Height and Weight

t = 15.2734, df = 23, p-value = 1.568e-13

← p-value in scientific notation: 1.568×10^{-13} ;
this is much less than α .

alternative hypothesis: true correlation is not equal to 0

95 percent confidence interval:

0.8971521 0.9798250

← You can be 95% sure that the true
population correlation coefficient lies between
about 0.897 and 0.980.

sample estimates:

cor 0.9540717

← Calculated sample correlation coefficient.

INTERPRETATION: Since the p-value is less than α , that indicates that you should reject the null hypothesis. There is sufficient evidence to conclude that a linear relationship exists between height and weight.

Finally, fit the regression line. You use "lm" to create the linear model. The expression (Weight ~ Height) tells R that you want it to consider Weight the response, based on predictor Height. Name the model HWLine. Output includes the slope and the y-intercept.

```
> HWLine = lm (Weight ~Height)
>HWLine
```

The following output results; one explanatory comment has been added on the right.

Call:

```
lm(formula = Weight ~ Height)
```

Coefficients:

(Intercept)	Height	
-274.856	6.624	←This tells you that the "y-intercept" is -274.856 and the slope is 6.624. Slope is the coefficient of the predictor, in this case "Height."

This means that the resulting regression line is:

$$\text{Weight} = -274.856 + 6.624 * \text{Height}$$

For practical purposes, since you probably only want weight to the nearest pound anyway, you could round the terms to use the equation:

$$\text{Weight} = -274.9 + 6.6 * \text{Height}$$

Section 27: How to Obtain Residuals and Fits from a Regression Line And Check the Assumptions (Uses data file: HeightWeight.txt)

This example is a continuation of what you did in Section 26: “How to Run a Simple Linear Regression.” First run all of the instructions in that section. The result is a regression line with equation:

$$\text{Weight} = -274.856 + 6.624 * \text{Height}$$

More information can be obtained about the model with the "summary" command, applied to the object HWLine – the name that you gave to the linear model in R if you worked through the previous section of this Manual.

```
> summary(HWLine)
```

The resulting output is at left, with explanatory comments added at right. Recall that the level of significance was alpha (α) = .05.

Call:					
lm(formula = Weight ~ Height)					
Residuals:					
Min	1Q	Median	3Q	Max	← Summarizes distribution of the residuals (errors).
-10.548	-4.913	1.205	3.582	10.205	
Coefficients:					
	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	-274.8556	30.3257	-9.063	4.73e-09 ***	← p-value for intercept is less than α ; the Intercept is significant.
Height	6.6236	0.4337	15.273	1.57e-13 ***	← p-value for coefficient of Height is less than α ; slope is significant.
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1					
Residual standard error: 5.632 on 23 degrees of freedom					← Standard deviation of distribution of the residuals is 5.632.
Multiple R-squared: 0.9103, Adjusted R-squared: 0.9064					← See discussion below this box.
F-statistic: 233.3 on 1 and 23 DF, p-value: 1.566e-13					← Overall p-value is less than α ; thus the regression line is a good representation of the Height and Weight relationship.

The term “R-squared” is the square of the correlation coefficient. When you did the original regression, you found the correlation was 0.9540717. If you square that, you get 0.9103 as shown in the box above. This number tells you that about 91% of the variability in Weight is explained by the regression line model. This is a very large effect size, and so the model is very useful.

You can get the list of residuals (errors) with the “resid” command applied to the model (called HWLine above).

```
> resid (HWLine)
```

The output of residuals is as follows.

1	2	3	4	5	6	25
6.0759013	1.3344402	0.2051708	3.0872865	4.2051708	1.5815939	-3.7948292

Similarly, you can also obtain the fits (predicted values) of Weight that result when the Heights are substituted into the regression equation.

```
> fitted (HWLine)
```

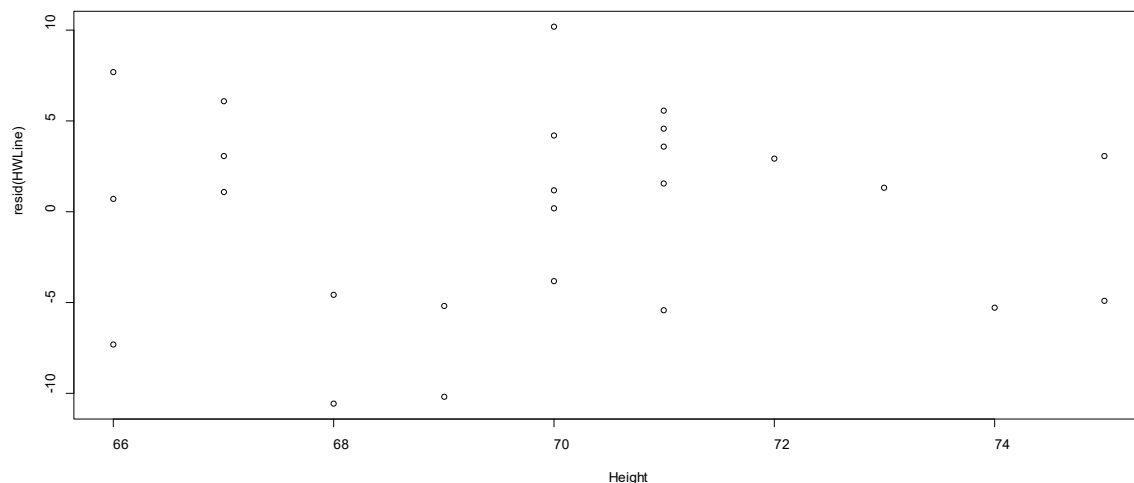
The output of predicted values is as follows.

1	2	3	4	5	6	25
168.9241	208.6656	188.7948	221.9127	188.7948	195.4184	188.7948

This means that if you substitute the first data value for Height (67 inches) into the regression equation, it predicts the weight should be 168.9 lbs. This is the first “fit” in the output of predicted values. The residual, or error term, is the difference between the observed Weight and the predicted Weight. You could calculate this yourself, using the first Weight in the data set and the first predicted value. However, it appears as the first value in the list of residuals, i.e., 6.08 lbs. The others work similarly.

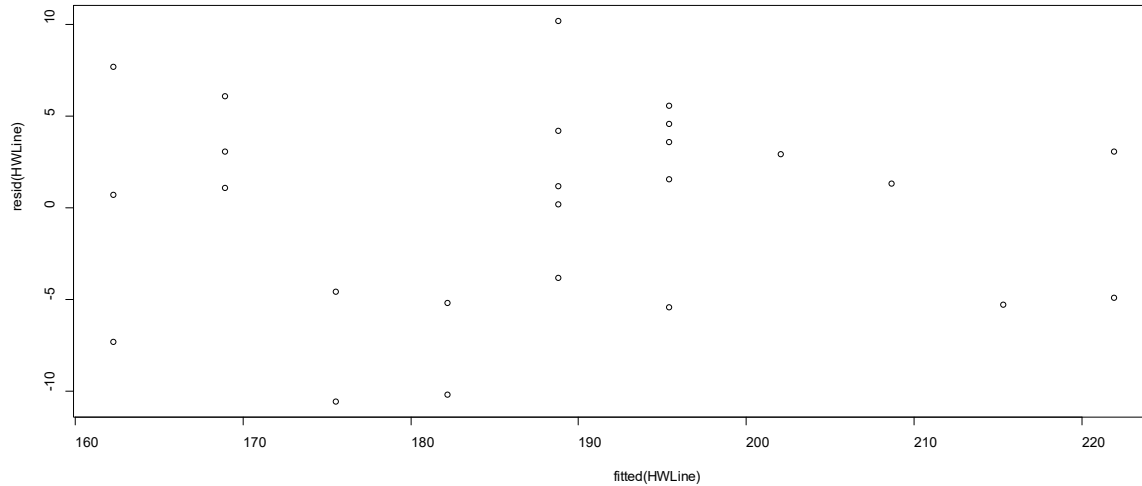
Now for assumption checking. You need to check for independence, common variance and normality. The necessary residual plots and the normality-test graph can be generated. If you want to keep the plots, enter one command at a time, get the graph and copy/paste it into Word. Then enter the next command to get the next graph, and so on. The first graph plots the residuals versus the predictor variable.

```
>plot (Height, resid (HWLine))
```



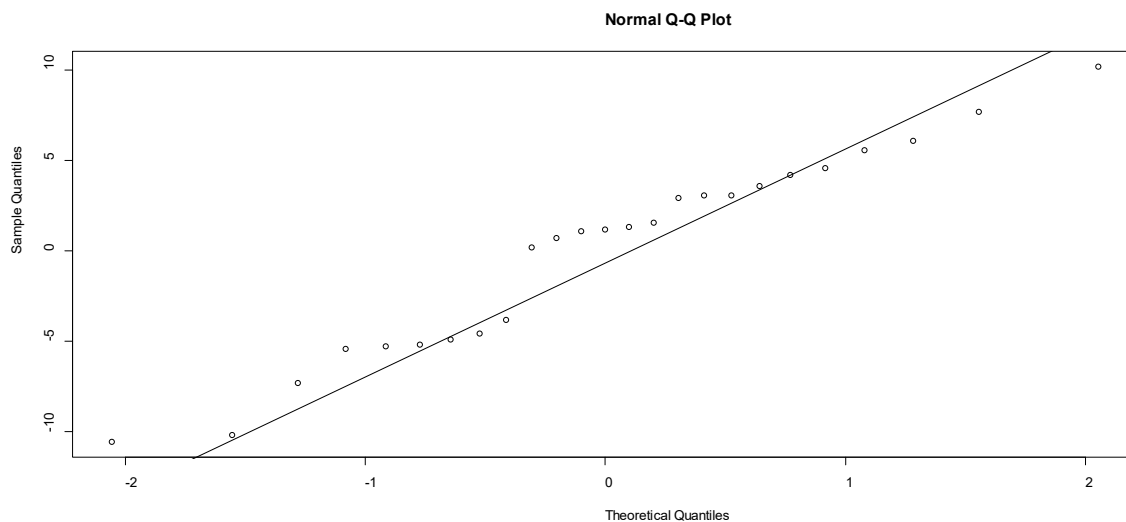
The second graph plots the residuals versus the fitted values.

```
> plot (fitted (HWLine), resid (HWLine))
```



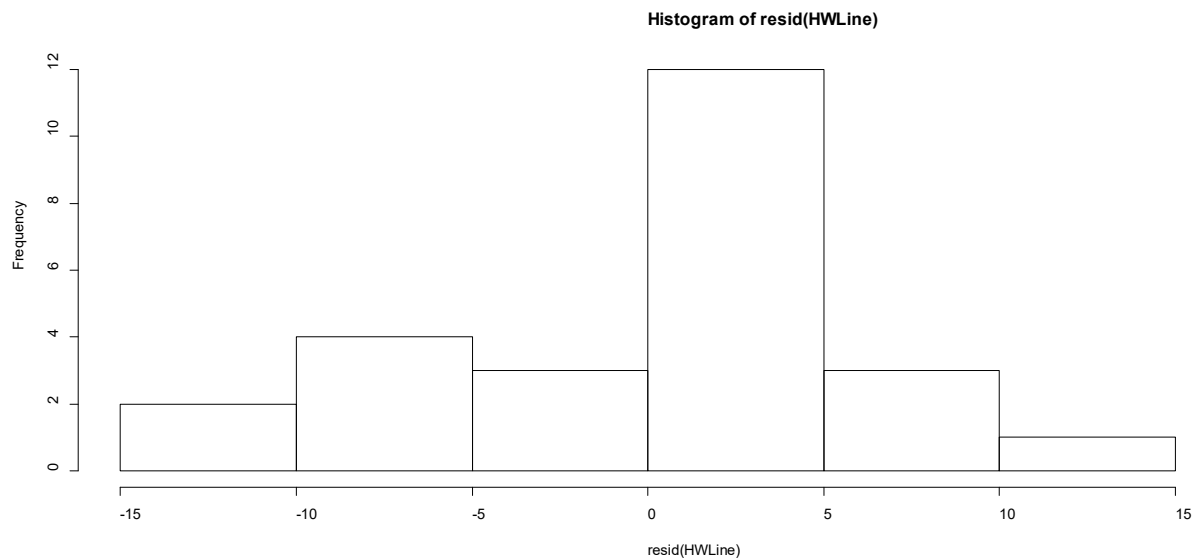
Neither plot shows any pattern, so you can generally believe that the common variance assumption and independence assumption hold for the data set. Now check for normality by creating a normal probability plot of the residuals. These are discussed in Section 10: “How to Check for Normality Using a Normal Probability Plot.”

```
> qqnorm (resid (HWLine))  
> qqline (resid (HWLine))
```



You may also want to look at a histogram of the residuals.

```
> hist (resid (HWLine))
```



The histogram of residuals looks approximately normal. In the “qq-plot,” the points representing the residuals fall fairly well along the normal quantile line. Both of these plots therefore suggest that you can safely accept the normality assumption. All of the assumptions required in order to use regression have been met, so you can use your regression line with confidence.

Section 28: How to Run a Basic Multiple Regression

(Uses data file: FluData1.txt)

This example uses a data set called FluData1.txt. It contains information on various people who have influenza. The variables and the meaning of the values are:

Age (in years)
Gender (0 = Female, 1 = Male)
Vaccine (0 = No flu shot, 1 = Had flu shot)
Treatment (0 = Not treated, 1 = Treated)
Smoking (0 = Non-smoker, 1 = Smoker)
Temperature (body temperature in degrees Fahrenheit)
Cough.Severity (rated on a scale from 0 to 8, higher rating means more severe).

The goal is to find a linear equation that describes Cough.Severity as a function of the variables that make a difference, and omit those that do not. Preset the level of significance; in this example, you will use alpha (α)= 0.05.

First read in the data table, and attach it so that R can work with it. Display it if you choose.

```
> Data = read.table ("E:/Data Files/FluData1.txt", header = TRUE)
> attach (Data)
> Data
```

A portion of the data set is as shown.

	Age	Gender	Vaccine	Treatment	Smoking	Temperature	Cough.Severity
1	55	1	0	1	1	98.0	1.5
2	17	0	0	1	0	98.6	0.8
3	36	1	1	0	0	97.9	1.1
4	30	0	0	0	0	97.9	0.0
5	63	0	0	0	0	98.2	2.8
:							
:							
475	14	1	0	1	0	102.8	6.6
476	5	0	1	0	0	103.2	4.2
477	8	1	0	1	0	101.5	4.4
478	5	1	0	0	0	100.6	7.7

If you have no idea which of the variables play a role in making a cough more or less severe, you could create a regression model for the response Cough.Severity with all five (except Temperature) other variables as predictors initially.

Comment: The rationale for leaving out Temperature is that it is generally also a response to having influenza, not a factor influencing the severity of other symptoms. However, if you think temperature might affect Cough.Severity, you could certainly include it as a predictor in the model too.

The first line of code creates the model; “lm” stands for linear model. Cough.Severity is on the left side of the ~ symbol as the response variable. The five predictors, separated by plus signs, are on the right. The “summary” command displays details about the resulting model.

```
> CoughModel1 = lm (Cough.Severity ~ Age + Gender + Vaccine + Treatment + Smoking)
> summary (CoughModel1)
```

The resulting output is as follows, with explanatory comments added on the right.

```
lm(formula = Cough.Severity ~ Age + Gender + Vaccine + Treatment + Smoking)

Residuals:
    Min       1Q   Median       3Q      Max
-5.8603  -1.0291   0.1955   1.3049   2.8370

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  5.750420   0.156116  36.834 < 2e-16 ***
Age         -0.013480   0.004662   2.892  0.00401 **
Gender       -0.032635   0.158750  -0.206  0.83721
Vaccine       0.005707   0.212573   0.027  0.97859
Treatment    0.258138   0.168351   1.533  0.12586
Smoking       0.490297   0.246478   1.989  0.04725 *

---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.72 on 472 degrees of freedom
Multiple R-squared:  0.02432, Adjusted R-squared:  0.01398
F-statistic: 2.353 on 5 and 472 DF, p-value: 0.03979
```

← The last column contains the p-values.
 ← p-value is less than α .
 ← p-value is less than α .
 ← p-value is less than α .

← Model only explains about 1-2% of the variability in Cough.Severity.

← Overall p-value is less than α .

At this point, you might decide to abandon this approach, since the R-squared values indicate that the model explains so little. But for purposes of the example, suppose you decide you want to continue with building a multiple regression model, eliminating the variables from the model that did not show up as statistically significant (their p-values are not less than α). Therefore, you keep only Age and Smoking as predictors.

```
> CoughModel2 = lm (Cough.Severity ~ Age + Smoking)
> summary (CoughModel2)
```

This time, the output appears as follows, except that the highlighting has been added. Overall, the second model is simpler in that it contains fewer predictor variables. Also, the same terms continue to show up as being statistically significant. However, the overall performance of the model is still about the same, accounting for only about 1-2% of the variability in Cough.Severity.

Call:

```
lm(formula = Cough.Severity ~ Age + Smoking)
```

Residuals:

Min	1Q	Median	3Q	Max
-5.6874	-1.0199	0.2798	1.2971	2.7338

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	5.813861	0.121409	47.886	< 2e-16 ***
Age	-0.013217	0.004639	-2.849	0.00457 **
Smoking	0.495108	0.245090	2.020	0.04393 *

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.719 on 475 degrees of freedom

Multiple R-squared: 0.0194, Adjusted R-squared: 0.01527

F-statistic: 4.698 on 2 and 475 DF, p-value: 0.009538

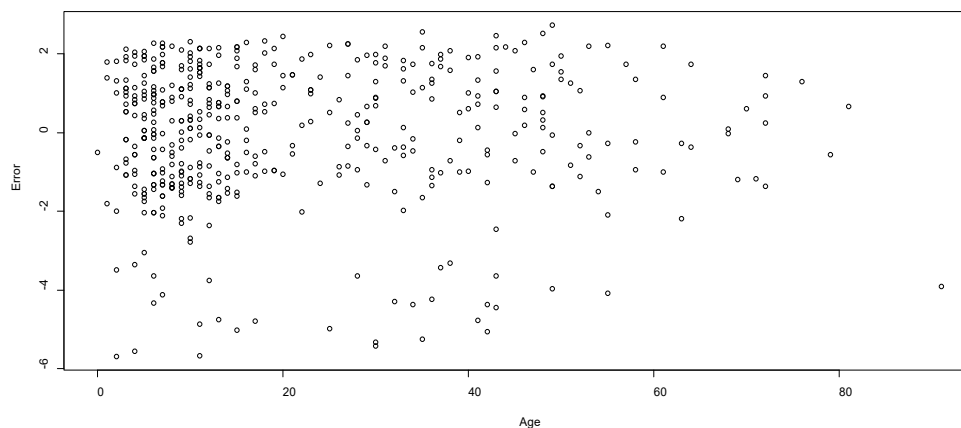
Using the estimates that are highlighted, the new, simpler equation is:

$$\text{Cough.Severity} = 5.8 - .013 \text{ Age} + .495 \text{ Smoking}$$

You may want to plot the residuals (error terms) vs. the predictors, and check for patterns. The first graph shows the residuals versus the predictor Age.

```
> Error = resid (CoughModel2)
```

```
> plot (Age, Error)
```



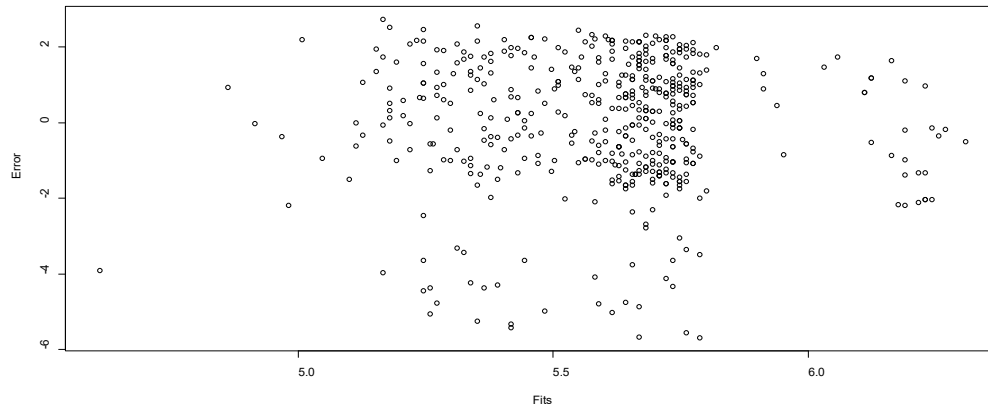
No particular pattern is detectable here, except that more young people get the flu than older people. You might speculate that this is probably because the older people have some immunity from prior exposures. But you cannot conclude that from the plot; that is only speculation.

The other residual plot, for Error vs. Smoking, could be done the same way, but it is not very productive to examine because Smoking was a yes/no variable.

You probably also want to examine a plot of the residuals versus the predicted values. First store the fitted values in a variable, such as Fits (used here). Plotting Error vs. Fits from the model can then be done, as follows.

```
> Fits = fitted(CoughModel2)
> plot(Fits, Error)
```

The resulting graph is shown below; again there is no particular pattern (except for the fact that the fitted Cough.Severity scores cluster around the center, showing that the average is around 5.7).



Non-Parametric Methods: Sections 29 - 38

29. How to Test a Hypothesis about a Proportion or Comparing Two Proportions

30. How to Do a One Sample Mann-Whitney Wilcoxon Test

31. How to Run a Mann-Whitney-Wilcoxon Test for Two Independent Samples

32. How to Run a Repeated Measures Mann-Whitney-Wilcoxon Test

33. How to Run a Kruskal-Wallis Test

34. How to Run Friedman's ANOVA

35. How to Create a Contingency Table and Run a Chi-Square Test

36. How to Test for Normality: Beyond Graphical Methods

37. How to Check Pairs of Values for Correlation Non-parametrically

38. How to Run a Binary Logistic Regression

Section 29: How to Test a Hypothesis about a Proportion or Comparing Two Proportions (Uses no data files)

Sometimes you need to test a hypothesis about proportions. This can be a test concerning the proportion of one population that has a certain property, or it can be a test to compare the proportions of two populations that have a certain property. For the single sample case, use:

```
> binom.test (number in sample with property, sample size, hypothesized proportion, hypothesis type)
```

OR

```
> prop.test (number in sample with property, sample size, hypothesized proportion, hypothesis type)
```

In the two proportion case, there are some additional commands that must be used first to set up the test. There is also a slight modification to the command itself in that you do not specify any hypothesized proportion. (See EXAMPLE B below.)

EXAMPLE A (Single proportion): Suppose you want to test the hypothesis that a particular biased coin will come up with heads more than 65% of the time. Set up your hypotheses. Assume that you have flipped the coin 50 times and gotten 35 heads in your sample.

H_0 : $\text{prop} \leq .65$

H_1 : $\text{prop} > .65$.

Set level of significance; this example uses alpha (α) = .05.

You have a choice to make. You can use either the binomial test or the proportion test. The binomial test is exact, but often not included in textbooks because the computations are more involved than those in the approximate proportion test. With software, this is not an issue. Here is how to do it both ways.

```
> binom.test (35, 50, 0.65, alternative = "greater")
```

OR

```
> prop.test (35, 50, 0.65, alternative = "greater")
```

The output from the binomial test is as shown below, with comments on the right.

Exact binomial test

data: 35 and 50

number of successes = 35, number of trials = 50, p-value = 0.2801 ← p-value is greater than α .

alternative hypothesis: true probability of success is greater than 0.65 ← Indicates upper tail test.

95 percent confidence interval: ← You can be 95% confident that the actual proportion of
0.576267 1.000000 heads produced by this coin is at least 57.6%.

sample estimates:

probability of success ← Calculated sample percentage of heads is 70%.

0.7

Note: For the other method, the (approximate) proportion test, most textbooks use a normal approximation with a correction factor. This is because the binomial is a discrete distribution and the normal is a continuous distribution. R uses a modification of the latter, in which it squares the test statistic that is used with the normal approximation. The square of a normal random variable has a chi-square distribution with one degree of freedom. While you really don't need to know all this to run the test, it explains why there is a "df" value in the output and the test statistic is a square.

The resulting output from the proportion test is on the left as follows; explanatory comments are on the right.

1-sample proportions test with continuity correction	
data: 35 out of 50, null probability 0.65	← "Null probability" is the 65% as in $H_0: \text{prop} \leq .65$.
X-squared = 0.3516, df = 1, p-value = 0.2766	← X-squared is the test statistic. p-value is greater than α .
alternative hypothesis: true p is greater than 0.65	← Indicates upper tail test.
95 percent confidence interval: percentage of 0.5750075 1.0000000	← You can be 95% confident that the actual proportion of heads produced by this coin is at least 57.5%.
sample estimates: p 0.7	← Calculated sample percentage of heads is 70%.

INTERPRETATION: Whichever, method you choose, the p-value is greater than α . That means there is not enough evidence to reject the null hypothesis. That is, there is insufficient evidence to conclude that the coin comes up heads more than 65% of the time.

EXAMPLE B (Compare two proportions): Suppose you have two coins, and you believe the first coin is more biased than the second one. You want to test hypotheses to compare their proportions of heads. Set up your hypotheses.

H_0 : proportion (coin 1) \leq proportion (coin 2)

H_1 : proportion (coin1) $>$ proportion (coin2)

Pre-set your level of significance; this example uses alpha (α) = .05.

For the sample, you flip each coin 40 times. The first coin comes up with heads 28 times. The second coin comes up with heads 26 times.

First, some preliminaries. You have to create two vectors, one for the numbers of heads and one for the sample sizes. In R, the "c" command puts the values you supply into a single vector. Be sure to keep the entries in the same order: c (first coin, second coin).

```
> heads = c (28, 26)
> samplesize = c (40, 40)
```

Here you use a proportion test. You use the vectors instead of single values in the command where the numbers of items with the property of interest (in this case, heads) and the sample size (here, number of flips) are specified. You do not have to specify the hypothesized difference as zero, since that is the default. You do have to indicate the form of the alternative hypothesis, however, so that R runs an upper tail test. So you use:

```
> prop.test (heads, samplesize, alternative = "greater")
```

The output for the proportion test is shown below.

```
2-sample test for equality of proportions with continuity correction
data:  heads out of samplesize
X-squared = 0.057, df = 1, p-value = 0.4057  ← X-squared is the test statistic; p-value is greater than  $\alpha$ 
alternative hypothesis: greater              ← Indicates upper tail test.
95 percent confidence interval:              ← You can be 95% sure that the proportions of heads
-0.1470229  1.0000000                      for the two coins differ by between -0.15 and 1
                                             (not a very useful confidence interval in this example).
sample estimates:
prop 1  prop 2                               ← Calculated sample proportions.
 0.70   0.65
```

INTERPRETATION: The p-value is greater than α , so you fail to reject the null hypothesis. There is insufficient evidence to indicate that coin 1 is more biased than coin 2.

Section 30: How to Run a One-Sample Mann-Whitney-Wilcoxon Test To Test a Single Population Median (Uses data file: MO Life Expectancy.txt)

This example will test the hypothesis that the median overall U.S. life expectancy is 76 years. You will use the Missouri life expectancy by county as your sample. This may not be random enough to provide a representative sample for the whole country, but proceed as if it is for this example.

First read in the data, attach it and display it if you wish.

```
> Data = read.table ("E:/Data Files/MO Life Expectancy.txt", header = TRUE)
> attach (Data)
> Data
```

Here is a portion of the data set.

	County	Total	Male	Female
1	Adair	77.7	75.4	79.9
2	Andrew	77.8	75.1	80.3
3	Atchison	78.3	75.5	81.2
:				
114	Worth	78.7	75.3	81.8
115	Wright	75.5	72.7	78.2

For purposes of this example, the hypothesis are:

H_0 : Median is 76 years.

H_1 : Median is not 76 years.

Pre-select your level of significance: α = .05

Run a Mann-Whitney-Wilcoxon test. The syntax for this in R is “wilcox.test” – the other two names are dropped. Specify the variable name and the hypothesized median. The variable you need from this data set is the column called Total and the hypothesized value is 76. The last item in the command, where it says “correct=FALSE,” tells R not to bother with a continuity correction factor because the data (life span totals combining men and women) is already continuous data.

```
> wilcox.test (Total, mu = 76, correct = FALSE)
```

The resulting output is as shown at left. Explanations are on the right.

Wilcoxon signed rank test

data: Total

V = 4586, p-value = 3.638e-05 ← Small p-value in scientific notation, meaning .00003638

alternative hypothesis: true location is not equal to 76

INTERPRETATION: Based on the p-value, you would reject the null hypothesis. The data suggests that the median is not 76 years.

Section 31: How to Run a Mann-Whitney-Wilcoxon Test for Two Independent Samples Compare the Medians of Two Independent Populations – Unpaired Data (Uses data file: Life Exp by Gender - nonpaired.txt)

Sometimes you may want to compare the medians of two populations, using two unpaired samples. The samples may or may not be the same size. This is similar to the t-test to compare the means of two populations using two unpaired samples, although it requires fewer assumptions. Since it is similar, the same data set is used here in the example.

```
>Data = read.table ("E:/Data Files/Life Exp by Gender - nonpaired.txt", header = TRUE)
>attach (Data)
> Data
```

A portion of the data set is shown below.

	Gender	Years
1	M	75.4
2	M	75.1
3	M	75.5
:		
38	F	81.3
39	F	80.3
40	F	79.6

While it is not part of the test, you may want to split the data into two separate lists. In the full data set, the first 20 entries are for males (M) and the remaining entries are for females (F). You can split the data into two lists, and then display both lists, as follows.

```
> M.Years = Years [1:20]
> F.Years = Years [21:40]
> M.Years; F.Years
```

This gives you the following two lists, the first for males and the second for females.

```
75.4 75.1 75.5 73.3 73.1 75.5 74.4 73.4 73.3 77.5 73.6 71.1 73.0 74.6 75.8 75.9 73.6 69.5 75.9 72.6
79.6 81.4 78.5 80.7 78.9 80.5 79.4 78.4 78.8 79.6 78.8 78.2 78.3 80.4 74.9 79.3 79.0 81.3 80.3 79.6
```

You can now find their sample medians as shown below.

```
> median (M.Years)
```

R returns the following as the sample median for males.

```
74
```

```
> median (F.Years)
```

R returns the following as the sample median for females.

```
79.35
```

There is about a five-year difference between the median male and female life expectancies in the sample. To test whether or not there is a difference in the medians of male and female populations, proceed as follows. Assume you want a two-tailed test with level of significance = .05

H_0 : Median male life expectancy = Median female life expectancy.

H_1 : Median male and female life expectancies are different.

Set alpha (α) = .05

```
> wilcox.test (Years ~ Gender,mu=0,alternative="two.sided", correct=FALSE, conf.int=TRUE)
```

Note that you tell R to test $\mu=0$; that is because, if the medians are equal, then the hypothesized difference should be zero. You also specify not to use the correction factor ("correct = FALSE") because the variable Years is continuous. Finally, "conf.int=TRUE" tells R to find a confidence interval for the difference in medians.

The following output results. Actual output is on the left; explanations are on the right.

Wilcoxon rank sum test

data: Years by Gender

W = 392, p-value = 2.038e-07

← Small p-value in scientific notation; means .0000002038.
This p-value is less than α .

alternative hypothesis: true location shift is not equal to 0

95 percent confidence interval:

4.100038 6.000015 ← Thus you can be 95% confident the actual difference in population medians is between about 4.1 and 6.0.

sample estimates:

difference in location

5.200001 ← This is a calculated difference that, in theory, should match the difference in sample medians (79.35-74 = 5.35 from above). It does not quite match because the test works with a quantity called the "pseudomedian." If the distribution is symmetric, it will match; otherwise, it may not.

Warning messages:

1: In wilcox.test.default(x = c(79.6, 81.4, 78.5, 80.7, 78.9, 80.5, : ← There were ties in the data, that is, cannot compute exact p-value with ties at least one of the data values occurred more than once.
2: In wilcox.test.default(x = c(79.6, 81.4, 78.5, 80.7, 78.9, 80.5, : So an approximation was used.
cannot compute exact confidence intervals with ties

Note: Another way to test the same hypothesis is the following. It used the two separate samples you created: M.Years, F.Years in a “wilcox test” command.

```
>wilcox.test (M.Years, F.Years, alternative="two.sided", correct = FALSE, conf.int=TRUE)
```

Section 32: How to Run a Repeated Measures Mann-Whitney-Wilcoxon Test Comparing Two Population Medians Using Paired Data

(Uses data file: Life Exp by Gender - paired.txt)

This is an example of a Mann-Whitney-Wilcoxon test to compare the medians of paired data. In this particular example, the data consists of Missouri life expectancies for men and women, paired by county. A sample of $n=24$ counties is used.

The null hypothesis is that the median life expectancy is the same for both genders; the alternative is that it is not the same. Set up your test.

H_0 : The median life expectancies of males and females are equal.

H_1 : The median life expectancies are not equal.

Preset your level of significance; this example uses alpha (α) = .05.

```
> Data = read.table ("E:/Data Files/Life Exp by Gender - paired.txt", header = TRUE)
> attach (Data)
> Data
```

A portion of the data set is shown below. The “Difference” column is not relevant for this example and should be ignored.

	County	Male	Female	Difference
1	Adair	75.4	79.9	4.5
2	Andrew	75.1	80.3	5.2
3	Cole	76.1	80.5	4.4
:				
23	Worth	75.3	81.8	6.4
24	Wright	72.7	78.2	5.5

Now run a “paired Wilcoxon” test as follows. Note that you set “paired=TRUE” because the data is paired by county. Also, since ages are continuous data, you do not need a correction factor. Therefore you set “correct = FALSE”).

```
wilcox.test (Male, Female, alternative = "two.sided", paired = TRUE, correct = FALSE)
```

The resulting output is shown below. Output is on the left; explanatory comments are on the right.

```
Wilcoxon signed rank test
data: Male and Female
V = 0, p-value = 1.804e-05          ← Small p-value in scientific notation; meaning .00001804
alternative hypothesis: true location shift is not equal to 0
```

Since the p-value is less than α , reject the hypothesis of equal medians.

Sometimes there is additional output when you run this test. In this example, your output also shows the following message at the end.

Warning message:
In wilcox.test.default (Male, Female, alternative = "two.sided", :
cannot compute exact p-value with ties

This is no cause for alarm. R is telling you that it used an approximation because there were ties, that is, there were some duplicate values in the data set.

Section 33: How to Run a Kruskal-Wallis Test Compare Medians of Multiple Populations (Uses data file: Anxiety.txt)

You use the Kruskal-Wallis test when you want to compare the medians of more than two populations and the populations are defined by a single characteristic. It is the non-parametric analogue of one-way ANOVA, and R calls it "kruskal.test."

This example uses a fictional data set of "anxiety scores" of 156 college students in four types of institutions (coded as LPR = large private university, SPR = small private university, STA = state university and COM = community college). The goal is to determine whether or not the median score is the same or different by type of institution. The null hypothesis is that all four types have the same median anxiety score; the alternative is that at least one median is different. Use level of significance $\alpha = .05$.

Set up your test formally.

H_0 : The median scores for all types of institutions are equal.

H_1 : At least one type has a different median score.

Preset your level of significance; this example uses $\alpha = .05$.

First read in data table giving the scores, attach it and display it if you wish.

```
> Data = read.table ("E:/Data Files/Anxiety.txt", header = TRUE)
> attach (Data)
> Data
```

A partial display of the data is as follows.

	Type	AnxScore
1	LPR	25
2	SPR	11
3	STA	8
4	COM	17
	:	
153	SPR	33
154	SPR	30
155	COM	21
156	COM	19

A single line of code is sufficient to run the test, using the AnxScore as a function of Type.

```
> kruskal.test (AnxScore ~ Type)
```

The resulting output is as shown below.

Kruskal-Wallis rank sum test

data: AnxScore by Type

Kruskal-Wallis chi-squared = 2.5423, df = 3, p-value = 0.4677

The p-value= 0.4677 > α , so the null hypothesis is accepted based on this sample. This means that the sample data does not indicate a difference in population median anxiety scores for the four types of institutions.

Section 34: How to Run Friedman's ANOVA

Comparing Multiple Medians in the Case of Repeated Measures

(Uses data file: AnxietyRepeat.txt)

The example uses the data set AnxietyRepeat.txt. This data set contains scores on an Anxiety Test, repeated three times on thirty-six (fictional) students. The test is originally given during their first term in college (labelled Fall1), repeated during their second term (labelled Spr1) and then again in their third term (labelled Fall2). The supposed goal of the testing is to see whether or not, on average, students' median anxiety level is generally consistent or changes over time as they adjust to college.

First read in data table giving the anxiety scores and attach it. Display it if you wish.

```
> Data = read.table ("E:/Data Files/AnxietyRepeat.txt", header = TRUE)
> attach (Data)
> Data
```

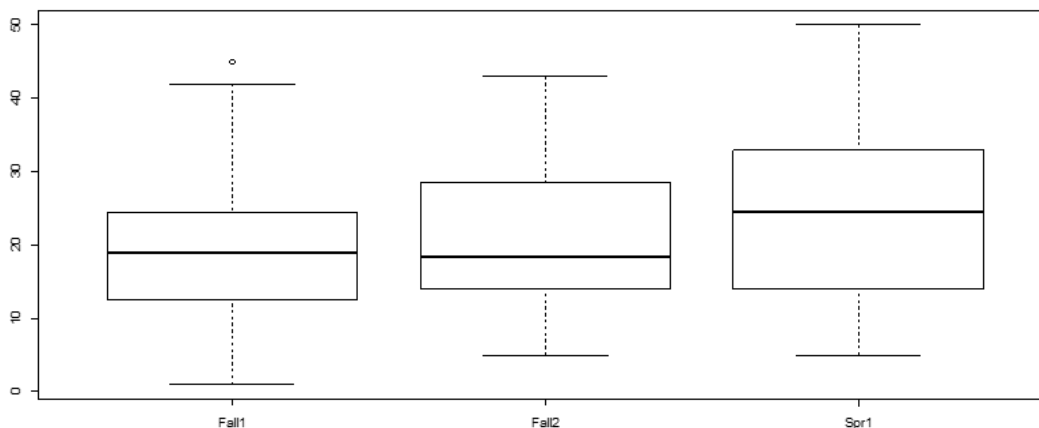
A partial display of the output is as follows.

	ID	Test.Session	Anx.Score
1	S1	Fall1	25
2	S1	Spr1	22
3	S1	Fall2	33
:			
106	S36	Fall1	24
107	S36	Spr1	30
108	S36	Fall2	37

As a first look, you may want to obtain the boxplots for each test session and see how the medians compare. One line of code will get you this.

```
> boxplot (Anx.Score ~ Test.Session)
```

The following graph results.



The boxplot indicates that the median of the Spr1 sample is different but you don't know if the same statement applies to the population medians. An appropriate hypothesis test for this purpose is Friedman's (non-parametric) ANOVA.

First set up your hypotheses; assume your chosen level of significance is alpha (α) = .05.

H₀: The medians of the anxiety scores are the same for all test sessions.

H₁: The median anxiety score for at least one test session is different.

One line of code will do this. It applies the Friedman test to the anxiety scores, treating them as a function of test session with each individual serving as his/her own "control" from one session to the next. (That is the meaning of the vertical bar followed by "ID" at the end of the line, right before the close-parenthesis.)

```
> friedman.test (Anx.Score ~ Test.Session | ID)
```

You get the following output.

Friedman rank sum test

data: Anx.Score and Test.Session and ID

Friedman chi-squared = 2.3803, df = 2, p-value = 0.3042

As indicated by the p-value being greater than α , the evidence in the sample does not support rejecting the null hypothesis.

Section 35: How to Create a Contingency Table and Run a Chi-Square Test on It (Uses no data files)

The null hypothesis in this example is that the scores on a test are independent of gender. The alternative is that there is a difference in the distribution of test results that is dependent on gender.

The results are presented in a contingency table, with the first row being men's results and the second row being women's results. The result counts are in the following order: Pass, Fail, Retest.

The data to be entered are:

	Pass	Fail	Retest
Men	42	29	10
Women	50	23	17

To create a contingency table: Enter it as a matrix, in order row by row, and specify dimensions (number of rows, not counting the labels). The example here has two rows of values, so specify that "nrow = 2." The "byrow = TRUE" subcommand will cause R to split the list of values into the number of rows you specified. That means, in this example, you will end up with your six data values in two rows and three columns, as you wanted them. The name you are giving the matrix is ExamResult.

```
> ExamResult = matrix(c(42,29,10,50,23,17), nrow = 2, byrow = TRUE)
> ExamResult
```

The resulting output is at left; explanatory comments are added at right.

<pre> [,1] [,2] [,3] [1,] 42 29 10 [2,] 50 23 17</pre>	<p>← R-default labels are column and row numbers. As presented above, the rows are men and women. Columns are Pass, Fail, Retest.</p>
---	---

Now set up your hypothesis test.

H_0 : Test results are independent of gender.

H_1 : Test results and gender are dependent.

Preselect your level of significance; this example uses alpha (α) = .05.

You are now ready to run the test. It will be a two-tail test by default.

```
> chisq.test(ExamResult)
```

Output is as follows:

```
Pearson's Chi-squared test
data: ExamResult
X-squared = 2.7367, df = 2, p-value = 0.2545
```

If you want, you can also have R obtain the .025 and .975 quantiles of the chi-square distribution with degrees of freedom = (rows-1)*(columns-1) = (2-1)*(3-1) = 2. This process is discussed in more detail in Section 13 of this Manual.

To get the lower critical value, enter the following.

```
> qchisq (.025, 2)
```

R returns the lower critical value.

```
0.05063562
```

Now for the upper critical value.

```
> qchisq (.975, 2)
```

R returns the upper critical value.

```
7.377759
```

The p-value is larger than α , so the null hypothesis is accepted. There is insufficient evidence to believe that test results are dependent upon gender. The test statistic 2.7367 is within the interval between the critical values, and confirms your “no difference by gender” conclusion.

There are a couple of related items that may be of interest. First, you may want to know the expected values for each cell of the contingency table if the null hypothesis is true. You can calculate these by hand without much trouble, or there is a single-word command in R that will do it for you. You just have to add it to the command that you used to run the test.

Modify the command you used before to say the following; the new piece is highlighted for you.

```
> chisq.test (ExamResult)$expected
```

Now the output is not the test result, but the expected values in each cell as follows:

	[,1]	[,2]	[,3]	
[1,]	43.57895	24.63158	12.78947	← R-default labels are column and row numbers Rows are men/women. Columns are Pass, Fail, Retest.
[2,]	48.42105	27.36842	14.21053	

Also, to get a measure of strength of association between variables (gender and exam outcome), you may want to use Cramer’s V. You can have R calculate this. If necessary, install the package: DescTools. Then load it. If you need a reference on how to install and load packages, see Section 3.

Then type the command below.

```
> CramerV (ExamResult)
```

For this example, R produces a value $V = 0.1265066$, or for practical purposes, 0.127. This is small, since Cramer's V always falls in the range between zero and one, with zero meaning "no association" and one meaning "perfect association." Therefore, in this example, there is little association between gender and exam outcome. This is what you should expect, since the hypothesis test indicated that there was no significant difference by gender.

Section 36: How to Test for Normality beyond Graphical Methods

(Uses data file: Mortgage Rates2.txt)

You can check your data set for normality using graphical methods in two ways: (1) by creating a histogram and making a judgment about whether or not it indicates that the data could reasonably come from a normal distribution, or (2) by creating a “normal Q-Q plot.” These were discussed in Section 10: “How to Check for Normality with a Normal Probability Plot.” This section will revisit them and then extend your options to include analytic methods.

The following example uses mortgage rate data by month for the years 2003-2017. (Data came from <http://www.freddiemac.com/pmms/pmms30.html>.)

First, read the data into R and attach it. Display it if you choose.

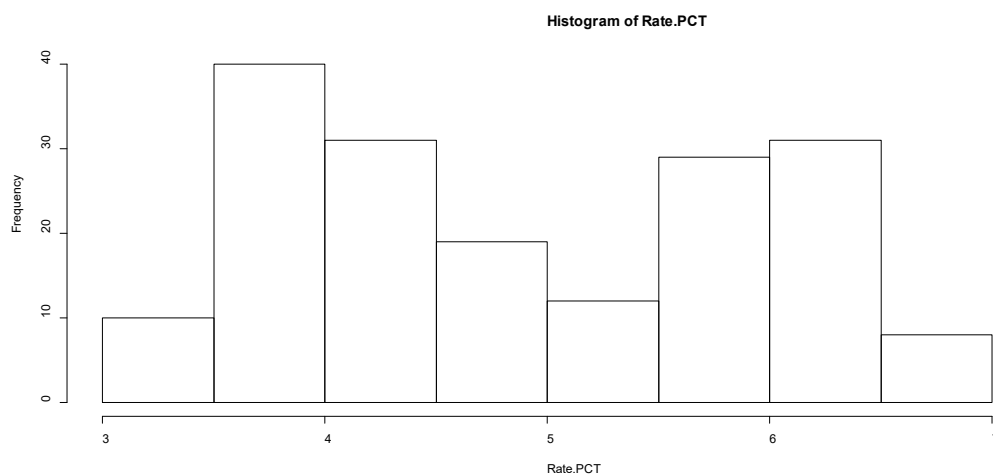
```
> Data = read.table("E:/Data Files/Mortgage Rates2.txt", header = TRUE)
> attach(Data)
> Data
```

The beginning of data set is shown below. The example will use the column called “Rate.PCT.”

	Time	Rate.PCT
1	Jan17	4.15
2	Feb17	4.17
3	Mar17	4.20
4	Apr17	4.05
:		
:	(etc)	

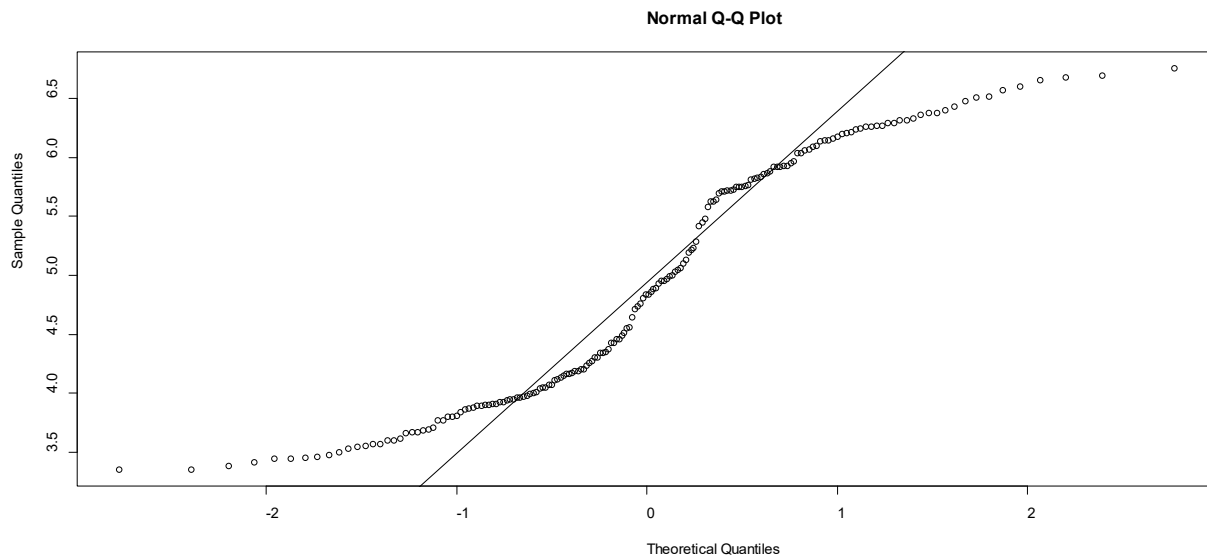
First, examine the data graphically. Make R create the histogram of the column called “Rate.PCT.”

```
> hist(Rate.PCT)
```



Also, make R create the normal probability plot of the column called “Rate.PCT.”

```
> qqnorm (Rate.PCT)
> qqline (Rate.PCT)
```



Both graphs indicate non-normality. The histogram appears to have two local maximum values, which would not occur in a normal distribution. The normal probability plot shows long “tails” on both ends that move farther and farther away from the line that corresponds to normally distributed data. Thus the data does not appear to come from a normally distributed population.

For a more analytical approach, you also have the option of running a hypothesis test for normality. Here is the set-up.

H_0 : The sample is compatible with a normally-distributed population.

H_1 : The sample is not compatible with a normally-distributed population.

Preset your level of significance; this example uses alpha (α) = .05.

There are two tests that are commonly used. The first is the Shapiro Test, which only requires you to enter the name of the variable. For this example, assume alpha (α) = .05.

```
> shapiro.test (Rate.PCT)
```

The output is equally brief.

Shapiro-Wilk normality test

data: Rate.PCT

W = 0.91644, p-value = 1.313e-08

← p-value is less than α ; reject normality hypothesis.

Alternatively, you can use the Kolmogorov-Smirnov Test, which is called “ks.test” in R. If you use this test, you also have to enter (in order):

1. The variable name
2. The name of the hypothesized distribution (here it is pnorm; the test can be used for other types of distributions as well)
3. The parameters of the hypothesized distribution (to illustrate here, hypothesize that the mean is about 4.9 and the standard deviation is about 1.2).

Therefore the appropriate R command is:

```
> ks.test (Rate.PCT, pnorm, 4.9, 1.2)
```

You will get the following output, shown on the left. Comments are on the right.

One-sample Kolmogorov-Smirnov test

data: Rate.PCT

D = 0.10306, p-value = 0.04368 ← p-value is less than α ; reject the normality hypothesis.

alternative hypothesis: two-sided

A further comment about the Kolmogorov-Smirnov test: if you make a minor change in the mean and standard deviation, the test may result in accepting the normality hypothesis. See the following.

```
> ks.test (Rate.PCT, pnorm, 4.91, 1.21)
```

The output is:

One-sample Kolmogorov-Smirnov test

data: Rate.PCT

D = 0.098655, p-value = 0.06017 ← p-value is greater than α ; accept the normality hypothesis.

alternative hypothesis: two-sided

Given that all of the previous evidence supports a conclusion of non-normality, this last result suggests that the Kolmogorov-Smirnov Test is not a good choice for use on this data. The output, both times, also showed a warning message.

Warning message:

In ks.test, ties should not be present for the Kolmogorov-Smirnov test.

If you go back and sort the values of Rate.PCT, you will find several values that are repeated. These are “ties” and may very well be the reason why the Kolmogorov-Smirnov test yields poor results here. Therefore, it is wise to run either the graphical methods or the Shapiro test as well, and not rely solely on the results of the Kolmogorov-Smirnov test.

Section 37: How to Check Pairs of Data Values for Correlation Non-Parametrically Spearman's rho and Kendall's tau

(Uses data files: HeightWeight.txt, Head Circumference.txt)

Here is an example, using the heights (in inches) and weights (in pounds) of twenty-five fictional army recruits. The data set is the same one used in Section 25 dealing with Pearson's r . The question here is whether or not Height and Weight are correlated in some fashion, but not necessarily linearly as with Pearson's r .

EXAMPLE A The first example uses a data set where a linear relationship is actually appropriate.

First read in the table of data, attach it and then display it if you wish.

```
> HWTable = read.table("E:/Data Files/HeightWeight.txt", header = TRUE)
> attach(HWTable)
> HWTable
```

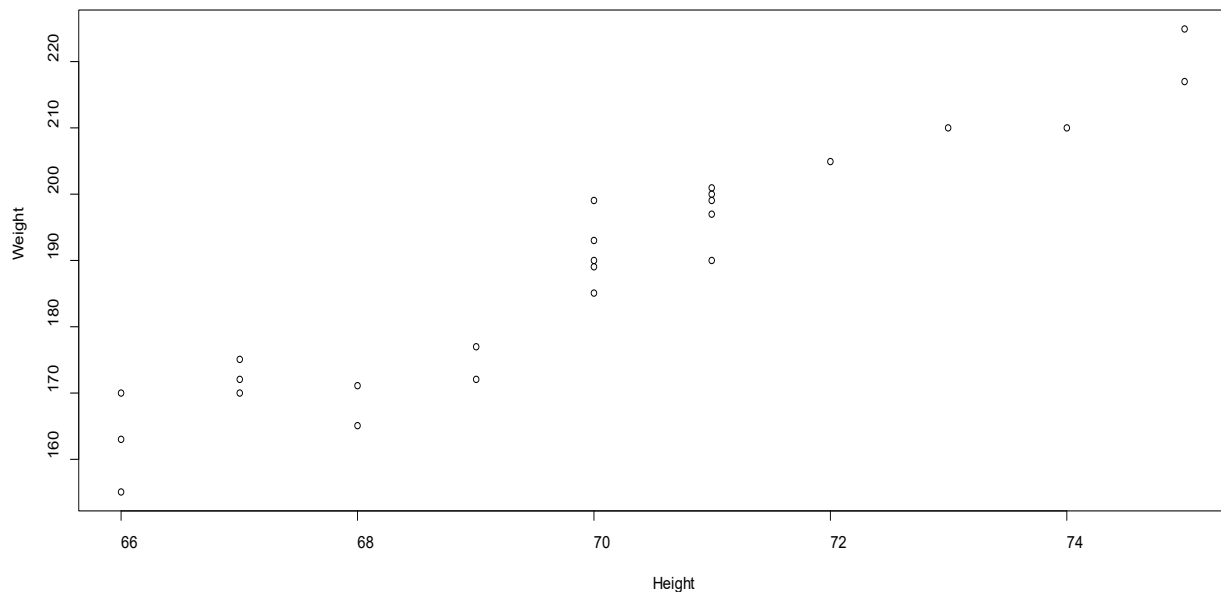
Resulting data output is the following.

	Height	Weight
1	67	175
2	73	210
3	70	189
4	75	225
5	70	193
6	71	197
7	71	200
8	75	217
9	68	165
10	74	210
11	67	170
12	71	201
13	66	163
14	68	171
15	71	190
16	66	170
17	67	172
18	69	172
19	70	199
20	69	177
21	71	199
22	66	155
23	72	205
24	70	190
25	70	185

Now create the scatterplot. This is a good idea to do first, as it will help you visualize your data. The first variable you list will go on the horizontal axis; the second one will go on the vertical axis.


```
> plot (Height, Weight)
```

The resulting graph is as shown below.



This graph certainly looks as if a rising line would fit fairly well through the data set. You actually got prior confirmation for this in Section 25 where you found the linear correlation coefficient Pearson's r . That result is repeated here for reference.

```
> cor (Height, Weight)
```

R returned the following value for Pearson's r :

```
.9540717
```

This value is positive and very close to one. It supports the belief that a rising line is a good way to represent the relationship between height and weight of army recruits.

However, suppose you were only interested in whether or not height and weight generally increased together, or whether they increased according to some non-specified (and not necessarily linear) pattern. Then you might choose to use Spearman's rho or Kendall's tau.

Here is the "cor" command three ways, adapted to specify the method and the output in each case. The first one is equivalent to the one immediately above, because Pearson's r is the default when no method is specified. Thus the output will be the value you already got immediately above.

```
> cor (Height, Weight, method = "pearson")
```

Here is the command adapted for Spearman's rho, with its output.

```
> cor (Height, Weight, method = "spearman")
```

The value returned for Spearman's rho is:

```
0.9547427
```

And here it is adapted for Kendall's tau.

```
> cor (Height, Weight, method = "kendall")
```

The value returned for Kendall's tau is:

```
0.8700405
```

As you can see, all indicate strong positive relationships.

EXAMPLE B Now suppose you try it on a data set where the two variables change together, but not linearly. An example uses the data set "Head Circumference.txt", which give the 50th percentile head circumferences for male infants from birth to three years of age (0 to 36 months). The data is excerpted from: https://www.cdc.gov/growthcharts/html_charts/hcageinf.htm#males

As usual, first read in, attach and examine the data set.

```
> Data = read.table ("E:/Data Files/Head Circumference.txt", header = TRUE)
> attach (Data)
> Data
```

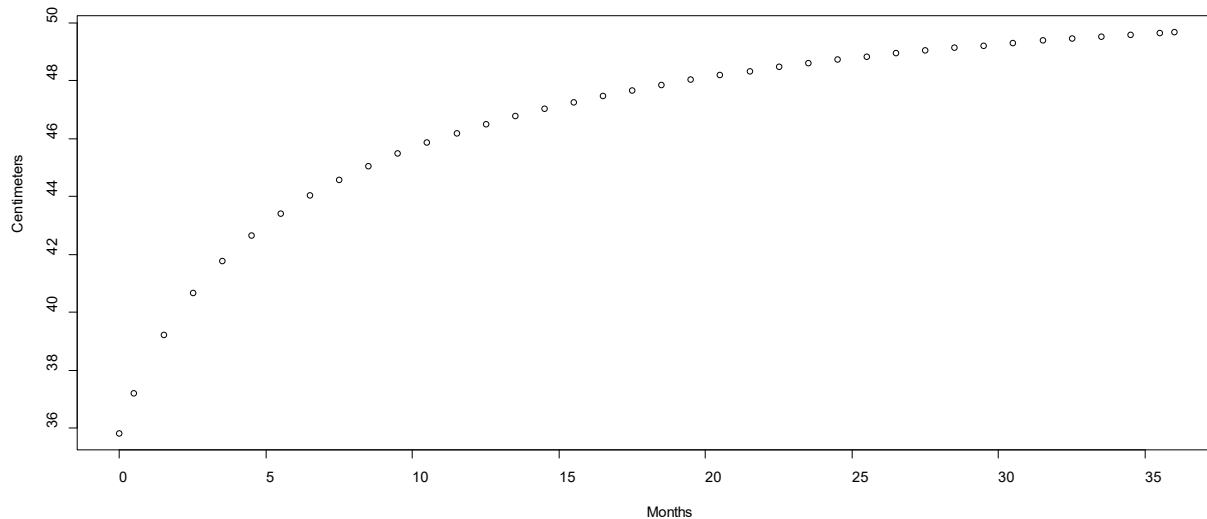
A portion of the data set is shown below.

	Months	Centimeters
1	0.0	35.814
2	0.5	37.194
3	1.5	39.207
4	2.5	40.652
5	3.5	41.765
:		
35	33.5	49.526
36	34.5	49.592
37	35.5	49.654
38	36.0	49.684

You should check the scatter plot.

```
> plot (Months, Centimeters)
```

The graph is shown below; you can see that the variables increase together but not linearly.



Since the relationship is not linear, you would not choose to use Pearson's r , but the following shows all three methods for comparison purposes.

```
> cor (Months, Centimeters, method = "pearson") ← R returns 0.8849858
```

```
> cor (Months, Centimeters, method = "spearman") ← R returns 1
```

```
> cor (Months, Centimeters, method = "kendall") ← R returns 1
```

This example is a little too “perfect” because the data did not come from a random sample but a known growth rate relationship. However you can see clearly that, since the increasing relationship of the two variables is “perfect”, Spearman's rho and Kendall's tau both result in a value of 1. But since the relationship is not linear, Pearson's r indicates a strong relationship but not a “perfect” linear one.

Section 38: How to Run a Binary Logistic Regression

(Uses data file: AdultFluData.txt)

This example uses a data set called AdultFluData.txt. It is a modification of the file used in the section on multiple regression. It contains information on various patients, age 20 or older, who have influenza. For convenience, it has been sorted into those without severe cough and those with severe cough. Within each group, it has also been sorted by smoking status and then by age. (The sorting was not necessary for the analysis to work.)

The variables and coding information are:

Age (in years)

Gender (0 = Female, 1 = Male)

Vaccine (0 = No flu shot, 1 = Had flu shot)

Treatment (0 = Not treated, 1 = Treated)

Smoking (0 = Non-smoker, 1 = Smoker)

Temperature (body temperature in Fahrenheit)

Severe.Cough (0 = Cough absent or not severe)
1 = Severe cough present)

The goal is to find a linear equation that describes the odds of having a severe cough as a function of the variables that make a difference, and to omit those that do not.

First read in the data table, and attach it so that R can work with it. You probably do not want to display all of it because there are 182 lines of data.

```
> Data = read.table ("E:/Data Files/AdultFluData.txt", header = TRUE)
> attach (Data)
```

A portion of the data set is as shown. The line in the middle has been added here to help you to spot where the “no severe cough” data ends and the “severe cough” data starts; it is not part of the actual data file.

	Age	Gender	Vaccine	Treatment	Smoking	Temperature	Severe.Cough
1	20	0	0	1	0	101.8	0
2	22	1	1	0	0	103.3	0
3	24	0	0	0	0	101.7	0
:							
60	79	1	1	1	1	100.5	0
61	91	0	1	0	0	99.9	0

62	20	0	0	0	0	102.8	1
63	20	1	0	1	0	103.7	1
64	20	0	0	0	0	103.6	1
:							
181	76	0	1	1	1	101.3	1
182	81	1	1	1	1	100.7	1

If you have no idea which of the variables play a role in determining whether a severe cough occurs, you could create a binary logistic regression model for response Severe.Cough with all five other variables as

predictors initially (excluding temperature, which is more likely another response than a predictor). The “summary” command displays details about the resulting model. Note the following:

1. The ~ symbol means that the variable on its left is being considered as a function of the variables on its right,
2. the code uses “glm” for “generalized linear model” (instead of the “lm” for “linear model”) that is used in simple and multiple regression, and
3. since the cough is either not severe (0) or severe (1), the binomial distribution must be specified at the end (where the command says “family=binomial”).

```
> CoughModel = glm (Severe.Cough ~ Age+Gender+Vaccine+Treatment+Smoking, family = binomial)
> summary (CoughModel)
```

Here is the resulting output, with explanatory comments added on the right.

```
glm(formula = Severe.Cough ~ Age + Gender + Vaccine + Treatment + Smoking, family = binomial)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.8256  -1.2829   0.7657   0.9164   1.3714

Coefficients:
              Estimate      Std. Error    z value    Pr(>|z|)
(Intercept)   2.048118     0.565917     3.619     0.000296 ***
Age           -0.034563     0.013842    -2.497     0.012527 *  ← Significant if α = .05
Gender        -0.434944     0.335940    -1.295     0.195421
Vaccine        0.312389     0.467650     0.668     0.504136
Treatment     -0.004883     0.354819    -0.014     0.989019
Smoking        0.966185     0.494432     1.954     0.050686 .  ← Significant if α = .10
                                                    and very close at .05.

---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 232.15  on 181  degrees of freedom    ← Deviance and AIC will be discussed
                                                    briefly at the end of this section.
Residual deviance: 222.65  on 176  degrees of freedom
AIC: 234.65

Number of Fisher Scoring iterations: 4
```

In the results above, the intercept also shows up as significant, but your purpose here is to determine which predictor variables are significant. As noted in the comments on the right, only Age and Smoking show up as being significant predictor variables. Therefore, you might choose to run another model, using only Age and Smoking as input variables.

```
> CoughModel2 = glm (Severe.Cough ~ Age+Smoking, family = binomial)
> summary (CoughModel2)
```

The resulting output is as follows.

```
glm(formula = Severe.Cough ~ Age + Smoking, family = binomial)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.7251  -1.3634   0.7816   0.9187   1.4290

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)   1.92695    0.54996   3.504  0.000459 ***
Age          -0.03474    0.01371  -2.534  0.011275 *
Smoking       0.94958    0.48707   1.950  0.051228 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 232.15 on 181 degrees of freedom
Residual deviance: 224.62 on 179 degrees of freedom
AIC: 230.62

Number of Fisher Scoring iterations: 4
```

You can see that both Age and Smoking are still significant, at approximately the same levels as before. The deviances are virtually unchanged. The AIC, which stands for Aikeke's Information Criterion, has not changed much, and in fact, has decreased slightly. Generally speaking, the smaller the AIC is, the better. The same holds for the magnitude of the deviances. Finally, the second model is considerably simpler. Therefore the second model is preferable to the first.

Now a word or two about interpreting and using the result. The equation that comes from the second model is:

$$\ln \left(\frac{p}{1-p} \right) = 1.93 - (0.035 \text{ Age}) + (0.950 \text{ Smoking}), \text{ where } p = \text{probability severe cough is present.}$$

The right-hand side was obtained from the output, using the estimates of the intercept and coefficients of Age and Smoking. The left side is not just the response itself (Severe.Cough) as it would be in a simple or multiple regression. In binary logistic regression, the left-hand side is the natural logarithm of the odds of the response. If you want the value of p , you need to solve the logarithmic equation for p .

Here is one example. Suppose you have a new patient who is age 45 and a non-smoker. Then the equation, applied to the data for this specific new patient, becomes:

$$\begin{aligned}
 \ln \left(\frac{p}{1-p} \right) &= 1.93 - (0.035 \text{ Age}) + (0.950 \text{ Smoking}) \\
 &= 1.93 - (0.035 * 45) + (0.950 * 0) \\
 &= 0.355
 \end{aligned}$$

Using the fact that the exponential function (base e) and the natural logarithm function are inverses, you obtain:

$$\frac{p}{1-p} = e^{0.355} = 1.426$$

Solve this algebraically for p : $p = 1.426 - 1.426*p$, or equivalently $p = 0.589$. Therefore, the prediction is that there is about a 59% chance that this person will develop a severe cough.

Follow-Up Comparisons: Sections 39-40

39. How to Do Pairwise Comparisons of Multiple Means or Proportions

40. How to Do Pairwise Comparisons of Multiple Medians

Section 39: How to Do Pairwise Comparisons of Multiple Means or Proportions (Uses data files: Absences.txt)

Suppose you have samples from more than two populations, and have determined (such as by using ANOVA) that at least one population mean is different. The ANOVA does not tell you which one(s) is/are significantly different.

Or perhaps you have proportions based on data from several populations, and you believe that at least one of them is different. Again, you want some justification for deciding which one(s) is/are different from the others.

In these situations, it is helpful to be able to do pairwise comparisons. However, since you want to do multiple comparisons on the same data set, a correction must be made to the p-values to control the overall probability of making a Type 1 error (rejecting a true null hypothesis). There are different correction procedures available, but the most common one is the Bonferroni correction. R software allows you to make this correction automatically.

Both of the data sets used in the following examples are fictitious. The first example uses one called Absences.txt. The second example simply has data entered as part of the R code. While they both deal with the topic of measuring absences from school, they are not related.

EXAMPLE A This example runs an ANOVA to test whether or not the mean number of absences is essentially the same for the months November through February. The ANOVA output leads to a rejection of the hypothesis that the means are the same. Hence you would run a subsequent pairwise t.test to try to identify which mean(s) is/are different. It includes the Bonferroni correction.

The data set contains counts of the numbers of students absent at least three days in the month, for ten classrooms. Different ten-classroom samples are randomly selected each month. The classes are all the same size, so there is no need to weight the counts. The actual data set is as follows.

Month Absent			Month Absent			Month Absent			Month Absent		
1	Nov	3	11	Dec	6	21	Jan	6	31	Feb	1
2	Nov	1	12	Dec	4	22	Jan	0	32	Feb	0
3	Nov	2	13	Dec	1	23	Jan	4	33	Feb	4
4	Nov	4	14	Dec	7	24	Jan	3	34	Feb	3
5	Nov	3	15	Dec	5	25	Jan	6	35	Feb	2
6	Nov	1	16	Dec	3	26	Jan	4	36	Feb	4
7	Nov	5	17	Dec	6	27	Jan	2	37	Feb	2
8	Nov	2	18	Dec	3	28	Jan	7	38	Feb	3
9	Nov	1	19	Dec	8	29	Jan	6	39	Feb	6
10	Nov	0	20	Dec	8	30	Jan	5	40	Feb	3

As usual, first read in the data set and attach it.

```
> Data = read.table("E:/Data Files/Absences.txt", header = TRUE)
> attach(Data)
```

It was stated above that the preliminary ANOVA indicates at least one of the monthly means is significantly different. That test is repeated here for reference.

First you create a linear model with the response variable Absent as a function of the variable Month. Run an ANOVA on the resulting model. Assume your level of significance is $\alpha = .05$.

```
> Model = lm (Absent ~ Month)
> anova (Model)
```

The results of the ANOVA are on the left as follows; comments are on the right.

Analysis of Variance Table

Response: Absent

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Month	3	53.4	17.8000	4.6368	0.007667 **
Residuals	36	138.2	3.8389		

← p-value is less than α ; reject the equal mean hypothesis.

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

In this example, the ANOVA leads you to conclude that the mean number of absences is different in at least one month. It does not tell you which one(s) is/are different. Now you probably want to compare the means pairwise to figure that out; this requires a pairwise t-test with the Bonferroni correction. Assume you are still using level of significance $\alpha = .05$, with a built-in Bonferroni correction.

A few words about the subcommands are needed here.

- (1) Note that the subcommand "p.adjust.method" as shown below tells R to make the Bonferroni correction.
- (2) Note the subcommand that says: "paired=FALSE"; it does NOT refer to the fact that the means are being compared pairwise. Instead, it is used to specify whether the data does or does not consist of repeated measurements; here it does not. If the same ten classrooms had been used every month, then you would say: "paired=TRUE."

```
> pairwise.t.test (Absent, Month, p.adjust.method = "bonf", paired = FALSE)
```

The resulting output is the following table.

Pairwise comparisons using t tests with pooled SD

data: Absent and Month

	Dec	Feb	Jan
Feb	0.076	-	-
Jan	1.000	0.573	-
Nov	0.013	1.000	0.131

P value adjustment method: bonferroni

Only one p-value in the table is less than α . That p-value is 0.013, which came from comparing the means for November and December. Hence these two months are the only pair for which the difference in the mean number of absences is statistically significant.

EXAMPLE B This example compares proportions (absentee rates) among school students, pairwise by month, from November through February. A sample of 150 students in the first grade of a particular school district remains the same each month because no transfers occur. You want to see which months, if any, have significantly different absentee rates (counted as a student being absent more than three days a month). Suppose the counts of students absent more than three days per month are: 13 students in November, 9 in December, 35 in January, and 27 in February.

First enter the data, in order by month: Nov, Dec, Jan, Feb.

```
> Absent = c(13, 9, 35, 27)
> Students = c(150, 150, 150, 150)
```

Calculate the sample proportions.

```
> Absent.Rate = Absent/Students
> Absent.Rate
```

You will get the following proportions; the month labels have been added here for clarity.

November	December	January	February
0.08666667	0.06000000	0.23333333	0.18000000

These rates, ranging from 6% to 23.3%, certainly do not appear to be all the same. However, if they are viewed as a sample of a larger population (such as all elementary classrooms in the district), they need to be tested to determine which rates are significantly different over the larger population and which are not. Again, assume your criterion for significance is alpha (α) = .05, with a built-in Bonferroni correction.

The “pairwise.prop.test” command will do this. You have to give it:

- the counts of the item you are interested in (Absent)
- the total sample size each time (Students)
- the instruction to use the Bonferroni correction (p.adjust="bonf").

```
> pairwise.prop.test(Absent, Students, p.adjust = "bonf")
```

Here is the output; explanatory comments are included on the right. The number labels are in order by month: 1 = November, 2 = December, 3 = January, 4 = February. The decimal values in the grid are the p-values for the corresponding comparisons of proportions.

As you can see from the output and the explanatory comments, the proportions of first grade students absent more than three days per month in the period November through February are significantly different:

- between November and January
- between December and January
- between December and February.

The other three pairs show no statistically significant difference in proportions.

Pairwise comparisons using Pairwise comparison of proportions

data: Absent out of Students

	1	2	3
2	1.00000	-	-
3	0.00565	0.00027	-
4	0.16349	0.01515	1.00000

← p-value comparing Nov and Dec is 1; no difference

← p-value comparing Nov and Jan is .00565; p-value

← comparing Dec and Jan is .00027. Both p-values are less than α so differences in proportions (Absent.Rate) are both significant.

---The only p-value that is less than α is .01515; indicates the difference in proportions (Absent.Rate) between Dec and Feb is significant.

P value adjustment method: Bonferroni

Section 40: How to Do Pairwise Comparisons of Multiple Medians

(Uses data files: Anxiety.txt, AnxietyRepeat.txt)

Suppose you have samples from more than two populations, and have determined (by using the Kruskal-Wallis Test or Friedman's ANOVA) that at least one population median is different. Recall that these tests do not tell you which one(s) is/are different, only that at least one is.

In this situation, you need to be able to do pairwise comparisons of medians. However, since you want to do multiple comparisons on the same data set, a correction must be made to the p-values to control the overall probability of making a Type 1 error (rejecting a true null hypothesis). There are different correction procedures available, but the most common one is the Bonferroni correction. R software allows you to make this correction automatically.

Both of the data sets used in the following examples are fictitious. The first example uses one called Anxiety.txt. The second example uses one called AnxietyRepeat.txt.

EXAMPLE A Refer to Section 33: "How to Run a Kruskal-Wallis Test," which shows the process of using a Kruskal-Wallis test to determine whether or not the median anxiety score of students varies by type of institution. The output there leads to not rejecting the null hypothesis, which suggests that the median score is the same for students from all types of institutions in the study.

However, for purposes of this example, suppose that the test had shown that at least one median was different. You would probably then run a `pairwise.wilcox.test` to see which one (or ones) differ. Assume the level of significance is $\alpha = .05$, with a built-in Bonferroni correction.

The general syntax for this type of situation is:

```
> pairwise.wilcox.test (Response variable, grouping variable, paired =_____, p.adjust.method="bonf")
```

In particular, in this example, the variable being measured is `AnxScore`, and the grouping is done by `Type`. You do not have repeated measures on the same students, so you set: `"paired=FALSE."` Since you are doing multiple tests using the same data, you need to include the Bonferroni correction factor. Thus, for this example, the command is as follows.

```
> pairwise.wilcox.test (AnxScore, Type, paired=FALSE, p.adjust.method="bonf")
```

Here is the output.

Pairwise comparisons using Wilcoxon rank sum test

data: AnxScore and Type

	COM	LPR	SPR
LPR	1.00	-	-
SPR	1.00	0.69	-
STA	1.00	1.00	1.00

This table shows the Bonferroni adjusted p-values for the pairwise comparisons. For example, the adjusted p-value for the comparison of the LPR median and the SPR median is 0.69. It is greater than α , so you can conclude there is no statistically significant difference in the median scores of students from the two types of institutions. In this example, all cases have large p-values; this was expected because the original Kruskal-Wallis test did not indicate any differences.

EXAMPLE B Refer to Section 34: “How to Run a Friedman’s ANOVA,” which shows the process of using a Friedman test to determine whether or not the median anxiety score of students varies from one semester to the next. The same students are tested three times each; it is a repeated measurement situation. The output leads to not rejecting the null hypothesis, which suggests that the median score remains the same over time.

However, again suppose the result had been different. That is, suppose that the Friedman test had indicated that at least one median was different. Then you would probably want to run a “pairwise.wilcox.test” on this data. The data now has “paired = TRUE” because you are repeating the anxiety test on the same students several times. Following the general syntax above, you would adapt it for this example as:

```
> pairwise.wilcox.test (Anx.Score, Test.Session, paired=TRUE, p.adjust.method = "bonf")
```

The output is as follows.

Pairwise comparisons using Wilcoxon signed rank test		
data: Anx.Score and Test.Session		
	Fall1	Fall2
Fall2	0.54	-
Spr1	0.19	1.00

Again, the p-values here are all greater than α , indicating no statistically significant pairwise differences in this example. That was expected, based on the Friedman test results.

(Page intentionally left blank)

Effect Sizes: Sections 41-45

41. How to Calculate Cohen's d

42. How to Calculate Cliff's δ

43. How to Calculate Risk, Odds and the Odds Ratio

44. How to Calculate Eta-Squared and Omega-Squared

45. How to Calculate Effect Sizes Relating to Correlation and Regression

Section 41: How to Calculate Cohen's d Effect Size for Difference between Two Means

(Uses data files: Life Exp by Gender – nonpaired.txt, Life Exp by Gender – paired.txt)

Cohen's d is a commonly used value for measuring the effect size when you are dealing with the difference between two means.

Independent Sample Case – Use the “nonpaired” data file listed above.

This example is one that deals with two independent samples. The null hypothesis is that the mean male life expectancy and the mean female life expectancy are equal. You can find the details of the initial test in Section 16: “How to Run Two Sample T-tests: Independent Samples.” Only the output is repeated here for reference. The level of significance was alpha (α) = .05.

Welch Two Sample t-test

data: Years by Gender

t = 9.9765, df = 35.6, p-value = 7.479e-12

← Test statistic is 9.9765

df is calculated by a more complicated formula than in most textbooks

p-value is 7.5×10^{-12} , which is less than α

alternative hypothesis: true difference in means is not equal to 0

← Indicates a two-tailed test: one tail would specify “greater than” or “less than”

Based on the small p-value, you would reject the hypothesis of equal means. In that case, you believe the mean life expectancies are different, but you would like a measure of the effect size. That is, is the difference small, medium or large? Therefore, you probably want to calculate a measure called “Cohen's d” that is used for this purpose, and then “classify” your effect size.

To calculate Cohen's d, you should install (if necessary) and load the R package: effsize. This process is explained in Section 3: “How to Find, Install and Load R Packages.” Once you have loaded the package, you need the following command:

```
> cohen.d (Years, Gender, pooled = TRUE, paired = FALSE)
```

The first variable, Years, is the one whose difference in means is being measured. Gender is the grouping factor. You want to tell R to pool the standard deviations of the two groups; that is done by the subcommand “pooled = TRUE.” Finally, since the data were not paired, and you specify that in the last subcommand: “paired = FALSE.” The resulting output is below.

Cohen's d

d estimate: 3.154861 (large)

95 percent confidence interval:

inf	sup
4.113865	2.195857

As you can see, the output is giving you an estimate for Cohen's d around 3.15. This is classified (by its magnitude, regardless of sign) as a large effect size.

IMPORTANT NOTE: You should treat the size classification carefully. What is large, medium or small depends heavily on the context of the situation. A large effect in one situation may be relatively minor and unimportant in another. The classification is only a rule-of-thumb and should not be taken as absolute.

Dependent Sample Case (Paired Data) – Use the “paired” data file listed above.

This example is one that deals with paired data. In this example, the male and female life expectancies are matched pairs by county of residence. The null hypothesis is again that both genders have the same mean life expectancy, or equivalently, that the mean difference in their life expectancies is zero. The details of the test can be found in Section 17: “How to Run Two Sample T-Tests with Paired Data.” Only the output is repeated here for reference. The level of significance used was alpha (α) = .05.

One Sample t-test	
data: Difference t = 24.9683, df = 23, p-value < 2.2e-16	← Test statistic is 24.9683. df=number of pairs – 1 = 23 The p-value = 2.2×10^{-16} , which is less than α .
alternative hypothesis: true mean is not equal to 0	← Indicates that this is a two-tail test.

This p-value is less than α , leading you to reject the null hypothesis. Therefore, in this example also, you have reason to believe that the mean life expectancy differs by gender. You probably want to calculate Cohen's d. If you have not done so, install and load the package: effsize. Note that the subcommand indicating paired data is now set as TRUE.

```
> cohen.d (Male, Female, pooled = TRUE, paired = TRUE)
```

Here is the output.

Cohen's d	
d estimate: -2.372378 (large)	
95 percent confidence interval:	
inf	sup
-2.742595	-2.002161

As you can see, the effect size is classified (by its magnitude, ignoring the sign) as “large.”

IMPORTANT NOTE: Again, you should treat the size classification carefully. What is large, medium or small depends heavily on the context of the situation. A large effect in one situation may be relatively minor and unimportant in another. The classification is only a rule-of-thumb and should not be taken as absolute.

Section 42: How to Calculate Cliff's Delta Effect Size for Difference between Two Medians

(Uses data files: Life Exp by Gender – nonpaired.txt, Life Exp by Gender – paired.txt)

Cliff's Δ is a commonly used value for measuring the effect size when you are dealing with the difference between two medians.

Independent Sample Case – Use the “nonpaired” data file listed above.

This example is one that deals with two independent samples. The null hypothesis is that the median male life expectancy and the median female life expectancy are equal. You can find the details of the initial test in Section 31: “How to Run a Mann-Whitney-Wilcoxon Test for Two Independent Samples.” Only the output is repeated here for reference. The level of significance was alpha (α) = .05.

```
Wilcoxon rank sum test
data: Years by Gender
W = 392, p-value = 2.038e-07      ← Small p-value in scientific notation; means .0000002038
alternative hypothesis: true location shift is not equal to 0
95 percent confidence interval:  4.100038  6.000015
sample estimates:
difference in location          5.200001
```

Based on the p-value being less than α , you would reject the hypothesis of equal medians. That means that you believe the median life expectancies are different, but you would probably like a measure of the effect size. That is, is the difference small, medium or large? Therefore, you want to calculate a measure called “Cliff's Δ ” that is used for this purpose, and then “classify” your effect size.

To calculate Cliff's Δ , you should install (if not already installed) and load the R package: effsize. This process is explained in Section 3: “How to Find, Install and Load R Packages.” Once you have loaded the package, you need the following command:

```
> cliff.delta (Years~Gender, use.unbiased = TRUE)
```

The first variable, Years, is the one whose difference in medians is being measured. Gender is the grouping factor. The last part of the command, “use.unbiased = TRUE,” refers to the method for calculating the variance of Cliff's Δ ; an extensive discussion of this is beyond the scope of this Manual.

The resulting output gives an estimate for Cliff's Δ around 0.96, which is classified as a large effect size.

```
Cliff's Delta
delta estimate: 0.96 (large)
95 percent confidence interval:
  inf      sup
0.7784777 0.9933347
```

IMPORTANT NOTE: You should treat the size classification carefully. What is large, medium or small depends heavily on the context of the situation. A large effect in one situation may be relatively minor or unimportant in another. The classification is only a rule-of-thumb and should not be taken as absolute.

Dependent Samples Case (Paired Data) – Use the “paired” data file listed above.

This example is one that deals with paired data. The null hypothesis was that the median male life expectancy and the median female life expectancy are equal. In this example, the male and female life expectancies are paired by county.

You can find the details of the initial test in Section 32: “How to Run a Repeated Measures Mann-Whitney-Wilcoxon Test.” Only the output is repeated here for reference. The level of significance was set at alpha (α) = .05.

```
Wilcoxon signed rank test
data: Male and Female
V = 0, p-value = 1.804e-05          ← Small p-value in scientific notation; meaning .00001804
alternative hypothesis: true location shift is not equal to 0
Warning message:
In wilcox.test.default (Male, Female, alternative = "two.sided", :
cannot compute exact p-value with ties
```

Since the p-value is less than α , you conclude that you should reject the hypothesis of equal medians. Since you believe the evidence suggests that the life expectancy medians differ by gender, you would probably want to know whether the difference is large, medium or small. You can use Cliff’s Δ to classify the effect size.

You need to install (if not already present) and load the package: `effsize`. The details of this process are explained in Section 3: “How to Find, Install and Load R Packages.” Then the necessary command is modified somewhat from the version in the first example. Here is what you want; note that you have specified that the data is “paired” in this version of the command.

```
> cliff.delta (Female, Male, paired = TRUE, use.unbiased = TRUE)
```

The resulting output is as follows; it gives an estimate for Cliff’s Δ around 0.93, which is classified as a large effect size. Again, treat the classification carefully. See the “Important Note” at the end of the first example.

```
Cliff's Delta

delta estimate: 0.9340278 (large)
95 percent confidence interval:
  inf      sup 
0.7711221 0.9821525
```

Section 43: How to Calculate Risk, Odds and the Odds Ratio

Effect Size for Proportions

(Uses no data files)

Risk and odds are commonly used values for measuring the effect size when you are dealing with single population proportions.

EXAMPLE A (Single proportion) Refer to Example A in Section 29: “How to Test a Hypothesis about a Proportion or Comparing Two Proportions.” The example dealt with a biased coin that came up heads 35 out of 50 times in a sample. The null hypothesis was that the coin produces heads 65% of the time; the alternative was the it produces heads more than 65% of the time. The example used level of significance $\alpha = .05$.

The output is reproduced below for reference.

Exact binomial test

data: 35 and 50

number of successes = 35, number of trials = 50, p-value = 0.2801 ← p-value is greater than α

alternative hypothesis: true probability of success is greater than 0.65 ← Indicates upper tail test

95 percent confidence interval: ← You can be 95% confident that the actual proportion of heads produced by this coin is at least 57.5%.
0.576267 1.000000

sample estimates:

probability of success ← Calculated sample percentage of heads is 70%.
0.7

Based on the large p-value, you would probably not go any further with this. But suppose that you want to “quantify” the amount of bias of the coin anyhow. From the output, you can see that the coin produced a sample proportion of 70% heads; in other words, the sample probability of getting heads is $P(\text{heads}) = 0.70$.

This probability is also known as the “risk” of getting heads. The terminology sounds a bit peculiar here, but it seems more appropriate when you are dealing medical questions, such as the probability of getting a particular disease. Therefore, using the “risk” terminology, the result is that:

The “risk” of getting heads = $P(\text{heads}) = 0.70$.

The “risk” of getting tails = $P(\text{tails}) = 0.30$.

Alternatively, you may want to express the “odds” of getting heads. In general:

$$\text{Odds of an event occurring} = \frac{\text{Risk or probability of event occurring}}{\text{Risk or probability of event not occurring}}$$

So for this example, the odds of getting heads = $0.70/0.30 = 2.33$. This means that, using that particular coin, the sample indicates that you are 2.33 times as likely to get heads as you are to get tails on a toss.

Now consider the situation when you are dealing with two samples. The odds ratio is frequently used when dealing with two proportions. Essentially, you compute the risk and then the odds for the sample from each population. Then calculate:

$$\text{Odds ratio of event} = \frac{\text{Odds of event in sample from first population}}{\text{Odds of event in sample from second population}}$$

EXAMPLE B (Proportions from two populations) Refer to Example B in Section 29: “How to Test a Hypothesis about a Proportion or Comparing Two Proportions.” This example deals with two biased coins. Coin One produced heads on 28 out of 40 tosses. Coin Two produced heads on 26 out of 40 tosses. A proportion test was run to test the claim that Coin One is “more biased” than Coin Two. The level of significance was $\alpha = .05$.

The output is reproduced here for reference.

2-sample test for equality of proportions with continuity correction	
data: heads out of samplesize	
X-squared = 0.057, df = 1, p-value = 0.4057	← X-squared is the test statistic p-value is greater than α
alternative hypothesis: greater	← Indicates upper tail test
95 percent confidence interval: -0.1470229 1.0000000	← You can be 95% sure that the proportions of heads for the two coins differs by between 0 and 1 (not a particularly useful confidence interval in this case)
sample estimates: prop 1 prop 2 0.70 0.65	← Calculated sample proportions

Using the sample proportions, $P(\text{heads for Coin One}) = 0.70$ and $P(\text{heads for Coin Two}) = 0.65$.

The calculations then proceed as follows.

- As shown above, the odds of heads for Coin One = 2.33.
- Do similar calculations for Coin Two.
“Risk” of heads for Coin Two = $P(\text{heads for Coin Two}) = 0.65$.
“Risk” of tails for Coin Two = $P(\text{tails for Coin Two}) = 1 - 0.65 = 0.35$.
Odds of heads for Coin Two = $0.65/0.35 = 1.86$.
- So the odds ratio of heads for these coins (in order) is: $2.33/1.86 = 1.25$.

Section 44: How to Calculate Eta-Squared and Omega-Squared Effect Size for Differences among Several Means (Uses data files: Anxiety.txt, AnxietyRepeat.txt)

Eta-squared is a commonly used value for measuring the effect size when you are dealing with the differences among multiple means. It is generally used following an ANOVA when a difference has been detected.

Note: Eta-squared applies only to the sample and SHOULD NOT BE USED for inferences about effect size in the populations. See additional comments concerning omega-squared after the two examples below.

Eta-Squared for a One-Way ANOVA

This example deals with four independent samples. It was used before in Section 21: “How to Do One-Way ANOVA,” where you can look at the data set if you wish. It deals with Anxiety Scores for students from four different types of educational institutions. The null hypothesis was that the mean scores are the same for students from all four types. The alternative was that at least one type has a different mean. The test used alpha (α) = .05. The test and its output are repeated here for reference.

```
> AnxModel = lm (AnxScore ~ Type)
> anova (AnxModel)
```

← Note the model has been named AnxModel. That is needed below in the code for finding eta-squared.

The output from the analysis of variance follows.

Analysis of Variance Table

Response: AnxScore

	Df	Sum Sq	Mean Sq	F value	Pr(>F)	
Type	3	261.2	87.076	0.8578	0.4645	← Test statistic is the F value.
Residuals	152	15429.9	101.513			← p-value is 0.4645.

Since the p-value is larger than α , you fail to reject the null hypothesis. The interpretation is that the evidence is not sufficient to conclude that any type of institution has a different mean anxiety score.

However, for purposes of illustration, suppose that the p-value had been small and the null hypothesis was rejected. Then you would believe that at least one mean was different from the rest, but you would not have a measure of how large the difference was. That is, you would need a measure of effect size. The usual one to use is eta-squared.

To calculate eta-squared, you should install (if not already installed) and load the R package: lsr. Refer to Section 3: “How to Find, Install and Load R Packages.”

Once you have installed and loaded the package, you need the following command:

```
> etaSquared (AnxModel, type=2, anova=TRUE)
```

In general, the items you have to supply in the command are:

- The first item is the name of the model that you created when you ran the original ANOVA.
- The “type” refers to how a particular sum of squares is calculated; you can leave it as “2” as long as you are not dealing with interactions.
- The last item tells R to display the ANOVA test as well as the effect size. You do not have to do this since you have already run an ANOVA, but it doesn’t hurt to leave that option as “TRUE.”

Here is the output.

	eta.sq	eta.sq.part	SS	df	MS	F	p
Type	0.01664814	0.01664814	261.2284	3	87.07615	0.8577864	0.4645137
Residuals	0.98335186	NA	15429.9190	152	101.51262	NA	NA

As you can see, the output matches that from the earlier ANOVA, except that two columns called “etq.sq” and “eta.sq.part” have been added. The second one is irrelevant, since the ANOVA was only a one-way model. Thus, the effect size for Type’s contribution to the Anxiety Scores is about .0166.

The R output does not attempt to classify the effect size as small, medium or large. A set of “rules of thumb” for classifying eta-squared is the following:

0.01 = small effect 0.06 = medium effect 0.14 = large effect

(Source: <http://imaging.mrc-cbu.cam.ac.uk/statswiki/FAQ/effectSize>)

Using these as guidelines, you would probably classify the effect size here as being “small.” That is what you would expect, since the original ANOVA indicated there was no significant difference in the means.

IMPORTANT NOTE: You should treat the size classification carefully. What is large, medium or small depends heavily of the context of the situation. A large effect in one situation may be relatively minor and unimportant in another. The classification is only a rule-of-thumb and should not be taken as absolute.

Eta-Squared for a Two-Way ANOVA (Repeated Measures Type)

This example also deals with Anxiety Scores, but the State variable is no longer used. Instead, each student has been tested three times in three different test sessions. Therefore, this is a repeated measures model.

This example was in Section 22: “How to Run a Repeated Measures ANOVA,” where you can examine the data set if you wish. The null hypothesis was that the mean scores are the same for all test sessions. The code for the test and its output are repeated here for reference. The test used alpha (α) = .05.

```
> Repeat.Model = lm (Anx.Score ~ Test.Session + ID) ← Creates a linear model with Anx.Score as a  
function of Test.Session and ID (individual))
```

The following output results from the analysis of variance.

Analysis of Variance Table

Response: Anx.Score

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Test.Session	2	383.0	191.509	2.1971	0.11874
ID	35	5235.9	149.597	1.7162	0.02781 *
Residuals	70	6101.6	87.166		

Now, as it turns out, you see that the p-value for Test.Session is greater than α , so test session shows no significant effect on the mean score. But if you had found significant results, you would probably want to measure the effect size(s).

As in the first example, if not already done, you would install and load the package: lsr. Then type the following.

```
> etaSquared (Repeat.Model, type=2, anova=TRUE)
```

The output is as follows.

	eta.sq	eta.sq.part	SS	df	MS	F	p
Test.Session	0.03267924	0.05906526	383.0185	2	191.5093	2.197054	0.11873606
ID	0.44672659	0.46181846	5235.8796	35	149.5966	1.716218	0.02780888
Residuals	0.52059418	NA	6101.6481	70	87.1664	NA	NA

Since you have two independent variables (Test.Session and ID), you want to look at the partial eta-squared column. The column labelled eta.sq.part gives these values. The partial eta-squared value for Test.Session is 0.059, which would round to a medium effect size based on the “rules of thumb” given above.

IMPORTANT NOTE: You should treat the size classification carefully. What is large, medium or small depends heavily on the context of the situation. A large effect in one situation may be relatively minor and unimportant in another. The classification is only a rule-of-thumb and should not be taken as absolute.

Section 45: How to Calculate Effect Sizes Related to Correlation and Regression

This is largely handled by R automatically when the regression analysis is run. See the sections:

- How to Run a Simple Linear Regression (Section 26)
- How to Check Pairs of Values for Correlation Non-Parametrically (Section 37)
- How to Run a Multiple Regression (Section 28)
- How to Run a Binary Logistic Regression (Section 38)

Without repeating the details of each application, the results are summarized again here for reference.

Simple Linear Regression: The example involves finding a linear relationship between height and weight. Before starting on the regression, the Pearson's correlation coefficient was obtained.

```
> cor (Height, Weight)
```

R returned the following sample correlation coefficient.

```
0.9540717
```

This is a strong correlation. Also, a hypothesis test in Section 26 tests this value for significance. Another measure used is "R-squared," which is straightforward to calculate once you know the value of Pearson's correlation coefficient – just square it. Here the value is approximately $0.954^2 = 0.91$. This means, roughly speaking, that 91% of the variability in Weight can be explained by its linear relationship with Height.

Multiple Regression: The example involves trying to find a linear equation that describes the severity of a cough among influenza patients, based on age and smoking status. The results are shown again here for reference; the details are in Section 28.

```
lm(formula = Cough.Severity ~ Age + Smoking)
```

Residuals:

Min	1Q	Median	3Q	Max
-5.6874	-1.0199	0.2798	1.2971	2.7338

Coefficients:

← Based on the following, the equation is:

Cough.Severity = 5.8 - .013 Age + .495 Smoking

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	5.813861	0.121409	47.886	< 2e-16 ***
Age	-0.013217	0.004639	-2.849	0.00457 **
Smoking	0.495108	0.245090	2.020	0.04393 *

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.719 on 475 degrees of freedom

Multiple R-squared: 0.0194, Adjusted R-squared: 0.01527

F-statistic: 4.698 on 2 and 475 DF, p-value: 0.009538

This time, the multiple R-squared and adjusted R-squared are the relevant measures. They are both small (only between 1 % and 2%), indicating that the model does little to explain cough severity.

Binary Logistic Regression: This example also deals with cough, but does not try to “rate” its severity. The only objective is to state the binary result: “yes, the cough is severe” (denoted 1) or “no, the cough is not severe” (denoted 0). Two models were run in Section 38 on binary regression. The first used all available input variables; the second used only age and smoking status. Part of the output is a quantity labelled AIC. This stands for Aikeke’s Information Criterion.

If you look back at the output in the Section 38, you will see that:

First Model AIC = 234.65

Second Model AIC = 230.62.

Generally, the smaller value for the AIC, the better. More details about the AIC are beyond the scope of this Manual.

(Page intentionally left blank)

Reference Tables

Table 1: Methods, Their Purposes, Typical Assumptions, Alternatives, Cross-Listed References

Note: (1) All methods assume that the samples are **simple random samples**, so that assumption is not listed individually.

(2) Alternatives are only listed if they are discussed in this Manual.

(3) Confidence intervals are not listed separately because they can be generated using the hypothesis test methods.

Method (with Section Number)	Purpose	Assumptions (with Section Numbers)	Alternatives (with Section Numbers)
1-Sample t-Test (15)	Tests single population mean	Underlying population is normally distributed (8, 10, 36) OR sample size is “large”	1-Sample Mann-Whitney-Wilcoxon Test (30)
2-Independent-Sample t-Test (16)	Compares means of two independent populations	Underlying populations are normally distributed (8, 10, 36) OR sample sizes are both “large”	2-Sample Mann-Whitney-Wilcoxon Test: Independent Samples (31)
Paired Data t-Test (17)	Compares two population means when data is paired or dependent	Underlying population of differences is normally distributed (8,10,36) OR sample size is “large”	2-Sample Mann-Whitney-Wilcoxon Test: Repeated Measures (32)
1-Sample Variance Test (19)	Tests single population variance	Underlying population is normally distributed (8,10, 36)	

Method (with Section Number)	Purpose	Assumptions (with Section Numbers)	Alternatives (with Section Numbers)
F-test (20)	Compares two population variances	Underlying populations are normally distributed (8, 10, 36) Populations are independent of one another	
ANOVA: One-way (21)	Compares multiple population means when one factor is used	Underlying populations are normally distributed (8, 10, 36) OR sample sizes are “large” Samples are independent of one another Common variance (20)	Kruskal-Wallis Test (33)
ANOVA: Repeated Measures (22)	Compares multiple means when measurements are repeated on the same individuals	Underlying populations are normally distributed (8, 10, 36) OR sample sizes are “large” Samples are independent of one another Common variance (20) Sphericity holds (24)	Friedman’s ANOVA (34)

Method (with Section Number)	Purpose	Assumptions (with Section Numbers)	Alternatives (with Section Numbers)
ANOVA: Two-Way (23)	Compares multiple population means when two or more factors are used	Underlying populations are normally distributed (8, 10, 36) OR sample sizes are “large” Samples are independent of one another Common variance (20)	
Mauchly’s Sphericity Test (24)	Checks repeated measures for sphericity	Differences between pairs of repeated measurements are normally distributed (8, 10, 36)	
Linear Correlation (25)	Checks for strength of linear relationship between two variables	Pairs come from a population that is bivariate normal	Non-parametric Correlation (37)
Simple Linear Regression (26)	Finds least-squares line relating two variables	“Strong” linear correlation (25) Bivariate normality; no pattern to residuals in scatterplot (27)	

Method (with Section Number)	Purpose	Assumptions (with Section Numbers)	Alternatives (with Section Numbers)
Multiple Regression (28)	Finds least-squares line that describes one response variable using multiple predictor variables	<p>Relationship is linear; check scatter plots of response vs. each predictor</p> <p>Uncorrelated predictors (25)</p> <p>Common variance of residuals across predictors; check scatterplots of residuals vs. predictors (27)</p> <p>Underlying normal distribution for response variable (8, 10, 36)</p> <p>No significant outliers in response; check its boxplot</p>	
Binomial or Proportion Test (29)	Tests one population proportion or compares two population proportions	<p>If two samples are being compared, they must be independent of one another</p> <p>Underlying population(s) has/have binomial distributions</p> <p>Expected number(s) of successes and failures under the null hypothesis are greater than or equal to 5</p>	

Method (with Section Number)	Purpose	Assumptions (with Section Numbers)	Alternatives (with Section Numbers)
1-Sample Mann-Whitney-Wilcoxon Test (30)	Tests single population median	Must be possible to rank data	
2-Sample Mann-Whitney-Wilcoxon Test: Independent Samples (31)	Compares medians of two independent populations	Independence both between and within samples Must be possible to rank data Populations have the same distribution OR difference is in location	
2-Sample Mann-Whitney-Wilcoxon Test: Repeated Measures/Paired Data (32)	Compares two population medians when data consists of repeated measurements	Must be possible to rank data Populations have the same distribution OR difference is in location	
Kruskal-Wallis Test (33)	Compares medians of multiple populations	Samples are independent of one another Must be possible to rank data Populations all have the same distribution OR difference is in location	

Method (with Section Number)	Purpose	Assumptions (with Section Numbers)	Alternatives (with Section Numbers)
Friedman's ANOVA (34)	Compares multiple medians when data consists of repeated measurements in blocks	Block variables are mutually independent Must be possible to rank data within blocks	
Chi-Square Test (35)	Tests contingency table data for independence of rows and columns	Data consists of frequency counts for the different categories in the table In each category, the expected frequency is greater than or equal to 5	
Shapiro Normality Test (36)	Tests for whether sample data is consistent with a normally distributed population	None	Graphical methods: Inspect Histogram (8) Normal Probability Plot (10)
Non-parametric Correlation (37)	Checks for more general correlation; need not be linear	For Spearman's rho: Ranking of the data must be possible For Kendall's tau: data must be either interval or ratio data so that differences can be computed	

Method (with Section Number)	Purpose	Assumptions (with Section Numbers)	Alternatives (with Section Numbers)
Binary Logistic Regression (38)	Classifies binary response variable based on values of independent variable(s)	<p>Independent observations (no repeated measurements)</p> <p>No correlation in pairs of predictors (25)</p> <p>Linear relationship between the log-odds of the response variable and the predictor variables</p>	
Multiple Comparisons: Means or Proportions (39)	Compares multiple means or multiple proportions and makes correction for multiple comparisons	Used following an ANOVA that indicates at least one mean is different from the rest	Multiple Comparisons: Medians (40)
Multiple Comparisons: Medians (40)	Compares multiple medians and makes correction for multiple comparisons	Use following a Kruskal-Wallis that indicates at least one median is different from the rest	

Bibliography

References

- Conover, W. *Practical Nonparametric Statistics*. New York, NY: John Wiley and Sons, 1999.
- Dancey, C., Reidy, J., Rowe, R. *Statistics for the Health Sciences*. Thousand Oaks, CA: SAGE, 2012.
- Faraway, J. *Linear Models with R*. Boca Raton, FL: CRC Press, 2015.
- Gibbons, J., Chakraborti, S. *Nonparametric Statistics*. Boca Raton, FL: Chapman and Hall CRC, 2010.
- Hollander, M., Wolfe, D. *Nonparametric Statistical Methods*. New York, NY: John Wiley and Sons, 1973.
- Hothorn, T., Everitt, B. *A Handbook of Statistical Analyses Using R*. Boca Raton, FL: CRC Press, 2014.
- Kloke, J., McKean, J. *Nonparametric Statistical Methods Using R*. Boca Raton, FL: CRC Press, 2015.
- Raykov, T. , Marcoulides, G. *Basic Statistics: An Introduction with R*. Rowman and Littlefield: Lanham, MD, 2012.
- Saville, D., Wood, G. *Statistical Methods: The Geometric Approach*. New York, NY: Springer-Verlag, 1991.
- Sprent, P., Smeeton, N. *Applied Nonparametric Statistical Methods*. Boca Raton, FL: Chapman and Hall CRC, 2007.
- <http://imaging.mrc-cbu.cam.ac.uk/statswiki/FAQ/effectSize>

R – basic program

R Core Team (2019). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL <https://www.R-project.org/>.

R packages

Torchiano M (2019). *_effsize: Efficient Effect Size Computation_*. doi: 10.5281/zenodo.1480624 (URL: <https://doi.org/10.5281/zenodo.1480624>), R package version 0.7.6, <URL: <https://CRAN.R-project.org/package=effsize>>.

Navarro, D. J. (2015) *lsr: Learning statistics with R: A tutorial for psychology students and other beginners*. (Version 0.5) University of Adelaide. Adelaide, Australia

John Fox and Sanford Weisberg (2019). *car: An {R} Companion to Applied Regression, Third Edition*. Thousand Oaks CA: Sage. URL: <https://socialsciences.mcmaster.ca/jfox/Books/Companion/>

Millard SP (2013). *_EnvStats: An R Package for Environmental Statistics_*. Springer, New York. ISBN 978-1-4614-8455-4, <URL: <http://www.springer.com>>.

Andri Signorell et mult. al. (2019). *DescTools: Tools for descriptive statistics*. R package version 0.99.28