STAT S400: Statistical Computing with R
Assignment #5 – An Introduction to Hypothesis Testing

Performing any form of (applied) statistical analyses requires data, and so knowing how to get data into R will be helpful. This assignment will introduce you to one way that data can be imported into R. In your last assignment you looked at graphical methods of analyzing data (in the exploratory sense). Here you are going to use R to perform tests on a specific type of data, those that are typically in the form of counts associated with frequencies of observations for categorical variables. As before, a script file, `Assignment5.R`, containing all the code for the following illustrations is posted with this assignment sheet. Use this script file to follow along.

The preliminary exercises begin with Bubba's data (from Assignment 4), this is contained in the data file `Assignment4.RData`. Also posted with these data is a file called `Assignment5.txt`. The data contained in this file (along with some of Bubba's data) will be needed for the problems to be submitted for this assignment.

Relevant Resources for this Assignment

The R Book: Chapters 2, 3, 4, 8, and maybe also 15 and 16

Your favorite elementary statistics text: Relevant topics/chapters.

Statistical Tests for Proportions (Counts Data)

Each subsection begins with a brief review of the relevant basic statistical background (don't forget your favorite elementary statistics text), followed by a general description of the R function(s) applicable to the test in question, and simple examples. For some of the tests, these notes go a bit beyond a typical elementary statistics textbook.

Testing a Single Population Proportion

Let X represent a binomial random variable with parameters n (number of Bernoulli trials) and p (population proportion, or probability of success). The sample proportion for a binomial experiment is defined by $\hat{p} = x/n$, where x represents the number of successes in n Bernoulli trials, and $\hat{q} = 1 - \hat{p}$.

If p_0 (where $0 < p_0 < 1$) is the hypothesized population proportion (with $q_0 = 1 - p_0$), then the goal here is to test any one of the hypotheses

$$\begin{array}{l} H_0 : p \geq p_0 \\ H_1 : p < p_0 \end{array}, \qquad \begin{array}{l} H_0 : p = p_0 \\ H_1 : p \neq p_0 \end{array}, \qquad \begin{array}{l} H_0 : p \leq p_0 \\ H_1 : p > p_0 \end{array}.$$

Three approaches are described here.

The Elementary Statistics Approach – Using a Standard Normal Distribution Approximation

For this approach, the test statistic is computed using

$$z^* = \frac{\hat{p} - p_0}{\sqrt{p_0 q_0 / n}},$$

and if both $np_0 \geq 5$ and $nq_0 \geq 5$ hold true, then the p-value for this test is obtained from the standard normal distribution as follows.

$$\text{p-value} = \begin{cases} P(Z < z^*) & \text{for a left-tailed test;} \\ \\ 2\,P(Z > |z^*|) & \text{for a two-tailed test; and} \\ \\ P(Z > z^*) & \text{for a right-tailed test.} \end{cases}$$

There does not appear to be a built-in function in package `stats` built specifically for *this* approach.

Using a χ^2-distribution Approximation

For this approach, first create a table of *observed* and *expected frequencies*,

	Success	**Failure**
Observed	o_1	o_2
Expected	e_1	e_2

where $o_1 = x$, $o_2 = n - x$, $e_1 = p_0\, n$, and $e_2 = q_0\, n$. The test statistic is computed using

$$\chi^{2*} = \sum_{i=1}^{2} \frac{(o_i - e_i)^2}{e_i},$$

and if $e_1 \geq 5$ and $e_2 \geq 5$ both hold true, then the p-value for a two-tailed test is obtained from the χ^2-distribution with $df = 1$ degree of freedom using

$$\text{p-value} = P(\chi^2 > \chi^{2*}).$$

For a one-tailed test, first compute[1]

$$z^* = \frac{|\hat{p} - p_0|}{(\hat{p} - p_0)} \sqrt{\chi^{2*}},$$

then use the standard normal distribution to compute

$$\text{p-value} = \begin{cases} P(Z < z^*) & \text{for a left-tailed test; and} \\ P(Z > z^*) & \text{for a right-tailed test.} \end{cases}$$

The built-in R function for this approach is `prop.test`, and the general usage is

```
prop.test(x = successes, n = trials, p = hypValue,
     alternative = "testType", conf.level = cLevel, correct = FALSE)
```

where ***hypValue*** is the hypothesized value, "***testType***" is chosen from `"less"`, `"two.sided"`, or `"greater"`, and ***cLevel*** is a number between 0 and 1. The *Yates correction for continuity* (`correct`) is not used,[2] hence the assignment `correct = FALSE`.

Exact Approach – Using the Binomial Distribution

The p-value can be calculated directly from the binomial distribution with parameters n and p_0 as follows: For one-tailed tests, use

$$\text{p-value} = \begin{cases} P(X \leq x) & \text{for left-tailed tests; and} \\ P(X \geq x) & \text{for right-tailed tests.} \end{cases}$$

For two-tailed tests, if x is very *close* to np_0 in value (upto some desired tolerance[3]), then set p-value $= 1$.

[1] Note: This is how the `prop.test` function goes about one-tailed tests for this approach. The rationale might be as follows. Some algebra shows that $(z^*)^2 = \chi^{2*}$; moreover, under the null hypothesis $(z^*)^2$ has a χ^2-distribution with $df = 1$ degree of freedom. So, the z-test and the χ^2-test are equivalent, and one can move from one to the other.

[2] Some suggest it is not necessary, others suggest its use under certain conditions.

[3] The function `binom.test` uses a round-about way that amounts to using a tolerance that seems to vary from *around* 0.1 to 0.5, depending on the value of $P(X = x)$.
The strategy described above produces identical results, except when x is close to (but not equal to) np_0.

Otherwise, first compute $m = \lfloor np_0 \rfloor$ (that is, round np_0 *down* to the nearest non-negative integer),[4] then assuming $x \neq m$, use

$$\text{p-value} = \begin{cases} P(X \leq 2m - x) + P(X \geq x) & \text{if } x > np_0\text{; and} \\ \\ P(X \leq x) + P(X \geq 2m - x) & \text{if } x < np_0. \end{cases}$$

The built-in R function for this approach is `binom.test`, and the general usage is

```
binom.test(x = successes, n = trials, p = hypValue,
     alternative = "testType", conf.level = cLevel)
```

where the arguments mean the same as for the `prop.test` function.

Examples

On a recent visit to the Pack Creek Bear Viewing Area near Juneau, a tourist was informed by the accompanying (newly hired) forest ranger that adult male brown bears make up less than 30% of the bears that are seen there. Unknown to the ranger, recent randomly gathered field data from the viewing area found that of the 35 distinct bears seen, there were 13 adult, male, brown bears.

To test the ranger's claim, the hypotheses

$$H_0 : p \geq 0.30 \qquad \text{vs.} \qquad H_1 : p < 0.30 \quad \text{(Claim)}$$

are tested using each of the above described approaches. First,

***z*-test for one population proportion:** The following code performs the task, and produces reasonably nice output. First set hypothesized proportion and input the needed data.

```
> p0 <- 0.30; q0 <- 1 - p0; n = 35; x = 13
```

Then compute the sample proportion, the test statistic, and the p-value.

```
> pHat <- x/n
> z <- (pHat - p0)/sqrt(p0*q0/n)
> pValue <- round(pnorm(q = z, lower.tail = TRUE), 4)
```

Finally, and just for kicks, use the cat function to format and output results in a pretty way.

```
> cat(paste("\n", "z-test for a Single Population Proportion\n\n",
+      "Data:  Successes = ", x, ", Trials = ", n, "\n",
+      "Alternative:  True proportion is less than ", p0, "\n",
+      "z = ", z, ", p-value = ", pValue, "\n",
+      "Sample Estimate = ", round(pHat, 7), "\n", sep = ""))
```

The result,

```
z-test for a Single Population Proportion


Data:  Successes = 13, Trials = 35
Alternative:  True proportion is less than 0.3
z = 0.922139, p-value = 0.8218
Sample Estimate = 0.3714286
```

Conclusion: There is insufficient evidence to support the park ranger's claim that adult male brown bears make up less than 30% of the bears that are seen at the Pack Creek Bear Viewing Area at any typical level of significance ($z = 0.922139$, p-value = 0.8218).

In Assignment 6 you will work on an example of turning such code into user-defined functions.

[4] The notation $\lfloor \cdot \rfloor$ represents the *floor function.*

χ^2-test for one population proportion: An identical p-value comes from running

```
> prop.test(x = 13, n = 35, p = 0.30,
+     alternative = "less", conf.level = .95, correct = FALSE)
```

which produces the output

```
        1-sample proportions test without continuity correction


data:  13 out of 35, null probability 0.3
X-squared = 0.8503, df = 1, p-value = 0.8218
alternative hypothesis:  true p is less than 0.3
95 percent confidence interval:
 0.0000000 0.5104139
sample estimates:
        p
0.3714286
```

Conclusion: Once again, there is insufficient evidence to support the park ranger's claim that adult male brown bears make up less than 30% of the bears that are seen at the Pack Creek Bear Viewing Area at any typical level of significance ($\chi^2 = 0.8503$, $df = 1$, p-value $= 0.8218$).

Notice that computing `z`^{2} from the previous example shows `z`^{2} `= X-squared = 0.8503`.

Exact binomial test for one population proportion: This test is conservative (less likely to result in a rejection of the null hypothesis) and, as should be expected, produces a different p-value. Running

```
> binom.test(x = 13, n = 35, p = 0.30, alternative = "less", conf.level = .95)
```

produces the following output

```
         Exact binomial test


data:  13 and 35
number of successes = 13, number of trials = 35, p-value = 0.865
alternative hypothesis:  true probability of success is less than 0.3
95 percent confidence interval:
 0.0000000 0.5244042
sample estimates:
probability of success
             0.3714286
```

As a check, use the `pbinom` function to compute the p-value directly.

```
> pbinom(q = 13, size = 35, prob = 0.30, lower.tail = TRUE)
[1] 0.8649532
```

With appropriate rounding, this agrees with `p-value = 0.865`, in the output for `binom.test`.

Conclusion: Yet again, there is insufficient evidence to support the park ranger's claim that adult male brown bears make up less than 30% of the bears that are seen at the Pack Creek Bear Viewing Area at any typical level of significance ($\hat{p} = 0.8503$, p-value $= 0.865$).

Testing Two Population Proportions

Let x_1 be the number of successes in n_1 Bernoulli trials on one population, and x_2 the number of successes in n_2 Bernoulli trials on a second population. Then the sample proportions are given by $\hat{p}_1 = x_1/n_1$ and $\hat{p}_2 = x_2/n_2$ (with $\hat{q}_1 = 1 - \hat{p}_1$ and $\hat{q}_2 = 1 - \hat{p}_2$).

The goal here is to test any one of the hypotheses

$$\begin{matrix} H_0 : p_1 \geq p_2 \\ H_1 : p_1 < p_2 \end{matrix}, \qquad \begin{matrix} H_0 : p_1 = p_2 \\ H_1 : p_1 \neq p_2 \end{matrix}, \qquad \begin{matrix} H_0 : p_1 \leq p_2 \\ H_1 : p_1 > p_2 \end{matrix}.$$

Two approaches are demonstrated.

The Elementary Statistics Approach – Using a Standard Normal Distribution Approximation

For this approach, the test statistic is computed using

$$z^* = \frac{\hat{p}_1 - \hat{p}_2}{\sqrt{\bar{p}\,\bar{q}\left(\dfrac{1}{n_1} + \dfrac{1}{n_2}\right)}},$$

where

$$\bar{p} = \frac{X_1 + X_2}{n_1 + n_2}, \text{ and } \bar{q} = 1 - \bar{p}.$$

If the samples are independent, and if $n_1\hat{p}_1 \geq 5$, $n\hat{q}_1 \geq 5$, $n_2\hat{p}_2 \geq 5$, and $n_2\hat{q}_2 \geq 5$ all hold true, then the p-value for this test is obtained from the standard normal distribution using the same approach as before. That is, use

$$\text{p-value} = \begin{cases} P(Z < z^*) & \text{for a left-tailed test;} \\ 2\,P(Z > |z^*|) & \text{for a two-tailed test; and} \\ P(Z > z^*) & \text{for a right-tailed test.} \end{cases}$$

There does not appear to be a built-in function in package `stats` for this approach.

By Looking at Goodness-of-Fit – Using a χ^2-distribution Approximation

For this approach, first create tables of *observed* and *expected frequencies*. Letting $c_1 = n_1$, $c_2 = n_2$, denote the total number of trials for both samples by

$$n = \sum_{j=1}^{2} c_j.$$

Then the observed frequencies are given in

	1	**2**	**Row Totals**
Success	o_{11}	o_{12}	r_1
Failure	o_{21}	o_{22}	r_2
Column Totals	c_1	c_2	n

To find the expected frequencies (e_{ij}) for each cell, observe that under the null hypothesis ($p_1 = p_2$),[5]

$$\frac{e_{ij}}{c_j} = k \quad \text{(a constant).}$$

[5]Note that while the null hypothesis is often written in one of three forms, that is, $p_1 \geq p_2$, $p_1 = p_2$, or $p_1 \leq p_2$, the form $p_1 = p_2$ is used in developing the test.

So it follows that $e_{ij} = k\, c_j$, and

$$r_i = \sum_{j=1}^{2} k\, c_j = k \sum_{j=1}^{2} c_j = k\, n.$$

Therefore,

$$\frac{r_i}{n} = \frac{k\, n}{n} = \frac{e_{ij}}{c_j},$$

and $e_{ij} = r_i\, c_j \,/\, n$, giving the expected frequencies table.

	1	**2**	**Row Totals**
Success	e_{11}	e_{12}	r_1
Failure	e_{21}	e_{22}	r_2
Column Totals	c_1	c_2	n

Then, the test statistic is computed using

$$\chi^{2*} = \sum_{i=1}^{2} \sum_{j=1}^{2} \frac{(o_{ij} - e_{ij})^2}{e_{ij}}.$$

If the samples are independent, and if all the expected frequencies satisfy $e_{ij} \geq 5$, then the p-value for this test is obtained from the χ^2-distribution with $df = 1$ degree of freedom. As previously, for two-tailed tests, use

$$\text{p-value} = P(\chi^2 > \chi^{2*})$$

and for a one-tailed test, first compute

$$z^* = \frac{|\hat{p}_1 - \hat{p}_2|}{(\hat{p}_1 - \hat{p}_2)} \sqrt{\chi^{2*}},$$

then use the standard normal distribution to compute

$$\text{p-value} = \begin{cases} P(Z < z^*) & \text{for a left-tailed test; and} \\ P(Z > z^*) & \text{for a right-tailed test.} \end{cases}$$

The built-in R function for this approach is `prop.test`, and the general usage is

```
prop.test(x = successes, n = trials,
     alternative = "testType", conf.level = cLevel, correct = FALSE)
```

where, in this case, *`successes = c(x1, x2)`*, *`trials = c(n1, n2)`*, and in this two-sample case the hypothesized value is left out – the default value being 0. As before, `"`*`testType`*`"` is chosen from `"less"`, `"two.sided"`, or `"greater"`, and *`cLevel`* is a number between 0 and 1. The *Yates correction for continuity* is not used at all in this case.

As for the one sample case, it can be shown (with a bit more work) that the z-test and the χ^2-test for two population proportions are equivalent So, once again, the `prop.test` function suffices.

Examples

Consider Bubba's data (see Assignment 3 for the story), to determine whether the true proportion of students who will secure at least 70% on the final in the project-based course differs from the true proportion of students who will secure at least 70% in the traditional course.

The first task is to extract the data, and one way to do this is as follows. First get the number of students from each course who scored at least 70%.

```
> (successes <- with(bubbaStudy[bubbaStudy$score >= 70, ],
+     expr = tapply(X = score, INDEX = course, FUN = length)))
proj trad
  59   25
```

and then find the respective course specific sample sizes

```
> (trials <- with(bubbaStudy,
+     expr = tapply(X = score, INDEX = course, FUN = length)))
proj trad
  83   92
```

Then, the hypotheses to be tested are

$$H_0 : p_{\text{proj}} = p_{\text{trad}}, \qquad \text{vs.} \qquad H_1 : p_{\text{proj}} \neq p_{\text{trad}}.$$

Here are examples of the two approaches.

***z*-test for two population proportions:** Code to perform the computations and output the results are as follows. First gather the relevant statistics,

```
> pHats <- successes/trials
> pBar <- sum(successes)/sum(trials); qBar <- 1 - pBar
```

Then calculate the test statistic and p-value; note use of `[[ ]]` instead of `[ ]`

```
> z <- (pHats[[1]] - pHats[[2]])/sqrt(pBar*qBar*sum(1/trials))
> pValue <- 2*round(pnorm(q = abs(z), lower.tail = FALSE), 4)
```

Now prepare things to produce pretty output. First create output strings

```
> suc <- paste(names(successes), " = ", successes, sep = "")
> trl <- paste(names(trials), " = ", trials, sep = "")
> props <- paste(names(pHats), " = ", round(pHats,4), sep = "")
```

then use the `cat` function to format and output results

```
> cat(paste("\n", "z-test for Two Population Proportions\n\n",
+     "Data and Sample Estimates:\n",
+     " Successes:\t", suc[1], ",\t ", suc[2], "\n",
+     " Trials:\t", trl[1], ",\t ", trl[2], "\n",
+     " Estimates:\t", props[1], ",\t ", props[2],"\n\n",
+     "Alternative:  True proportions are not equal\n",
+     "z = ", round(z, 6), ", p-value = ", pValue, "\n", sep = ""))
```

and the output is

```
z-test for Two Population Proportions


Data and Sample Estimates:
 Successes:     proj = 59,         trad = 25
 Trials:        proj = 83,         trad = 92
 Estimates:     proj = 0.7108,     trad = 0.2717


Alternative:  True proportions are not equal
z = 5.805763, p-value = 0
```

Conclusion: At any level of significance, there is sufficient evidence to conclude that the true proportion of students who will secure at least 70% in the project-based course differs from the true proportion of students who will secure at least 70% in the traditional course ($z = 5.805763$, p-value = 0).

There are two points to notice in the above code. The first is the use of `[[ ]]` as opposed to `[ ]`. While not necessary here, this (use of `[[ ]]` as opposed to `[ ]`) is important for some situations (see `?"["` for more). Second, in the `cat` function call, notice the use of the (output) string formatting codes `\n` (newline), and `\t` (tab). See `?Quotes` for more.

χ^2-test for two population proportions: An equivalent result is obtained using

```
> prop.test(x = successes, n = trials,
+     alternative = "two.sided", conf.level = .95, correct = FALSE)
```

The output in this case is

```
 2-sample test for equality of proportions without continuity correction


data:     successes out of trials
X-squared = 33.7069, df = 1, p-value = 6.407e-09
alternative hypothesis:  two.sided
95 percent confidence interval:
 0.3057763 0.5724322
sample estimates:
     prop 1    prop 2
  0.7108434 0.2717391
```

Conclusion: Again, at any level of significance there is sufficient evidence to conclude that the true proportion of students who will secure at least 70% in the project-based course differs from the true proportion of students who will secure at least 70% in the traditional course ($\chi^2 = 33.7069$, $df = 1$, p-value = 0).

As for the single population proportion, allowing for rounding, it can be seen that, $(z^*)^2 = \chi^{2*}$

Testing More than Two Population Proportions

In some elementary statistics texts this is referred to as the χ^2-*homogeneity of population proportions test*. Suppose $p_1, p_2, \ldots, p_J$ represent the true population proportions of successes across $J > 2$ populations, then the procedure to test the hypotheses

$$H_0 : p_1 = p_2 = \cdots = p_J \quad \text{vs.} \quad H_1 : p_j \neq p_k \text{ for at least one pair } j \neq k,\ j, k = 1, 2, \ldots, J$$

is analogous to the previous two-sided χ^2-tests for one and two population proportions.

Data from J random samples are placed in a two-dimensional contingency table with two rows, as shown below.

	1	**2**	$\cdots$	**J**	**Row Totals**
Success	o_{11}	o_{12}	$\cdots$	o_{1J}	r_1
Failure	o_{21}	o_{22}	$\cdots$	o_{2J}	r_2
Column Totals	c_1	c_2	$\cdots$	c_J	n

Entries in the first row (o_{1j}) represent observed frequencies for "Successes," and entries in the second row (o_{2j}) represent the corresponding observed frequencies for "Failures." The columns represent the J populations of interest.

Then, under the null hypothisis ($p_1 = p_2 = \cdots = p_J$), for $i = 1, 2$ (rows) and $j = 1, 2, \ldots, J$ (columns), the expected frequencies are obtained as before using $e_{ij} = r_i\, c_j / n$,

	1	**2**	$\cdots$	**J**	**Row Totals**
Success	e_{11}	e_{12}	$\cdots$	e_{1J}	r_1
Failure	e_{21}	e_{22}	$\cdots$	e_{2J}	r_2
Column Totals	c_1	c_2	$\cdots$	c_J	n

and the test statistic is computed using

$$\chi^{2*} = \sum_{i=1}^{2} \sum_{j=1}^{J} \frac{(o_{ij} - e_{ij})^2}{e_{ij}},$$

Once again, if the samples are independent, and every expected frequency satisfies $e_{ij} \geq 5$, then the p-value for this test is obtained from the χ^2-distribution with *df* $= J - 1$ degrees of freedom[6] using

$$\text{p-value} = P(\chi^2 > \chi^{2*}).$$

The `prop.test` function works as before, but since only a two-sided alternative is involved, less arguments are included. One can use

```
prop.test(x = successes, n = trials, correct = FALSE)
```

where *`successes = c(x1, x2, ..., xJ)`* and *`trials = c(n1, n2, ..., nJ)`*. Or, one can also use

```
prop.test(x = obs, correct = FALSE)
```

where *`obs`* is a $J \times 2$ matrix that contains the *observed successes* in the first column, and *observed failures* in the second column.

Alternatively, the function `chisq.test` function can be used to perform the same task as follows.

```
chisq.test(x = obs, correct = FALSE)
```

Here *`obs`* is a $2 \times J$ matrix that contains the observed successes in the first row and the observed failures in the second row.

Examples

Using Bubba's data, consider testing the claim that the true proportions of students who are likely to get at least 70% in the project-based course final is the same across fields. That is, if p_1, p_2, p_3, and p_4 represent the true proportions of students (from `mathsci`, `natsci`, `other`, and `socsci`, respectively) who are likely to get at least 70% in the project-based course final, then the hypotheses to be tested are

$$H_0 : p_1 = p_2 = p_3 = p_4 \quad \text{(Claim) vs.} \quad H_1 : p_j \neq p_k \text{ for at least one pair } j \neq k, \;\in \{1, 2, 3, 4\}.$$

As before, first extract the observed data. Here's one way to do this. First extract the observed frequencies and sample sizes, and then bind the successes and failures by rows in a 2×4 matrix.

[6] Some suggest that this strict requirement can be relaxed a bit. A rule of thumb suggested in some texts is that the computed p-value is acceptable (approximately accurate) if none of the expected frequencies are zero, and at least 80% of the expected frequencies are greater than or equal to 5.

```
> obs <- rbind(
+     with(bubbaStudy[bubbaStudy$score >= 70, ],
+         expr = tapply(X = score, INDEX = field, FUN = length)),
+     with(bubbaStudy,
+         expr = tapply(X = score, INDEX = field, FUN = length)))
```

Next, give the rows names and take a look at the results.

```
> rownames(obs) <- c("successes", "trials"); obs
          mathsci natsci other socsci
successes      19     33     8     24
trials         36     69    21     49
```

Using the `prop.test` function,

```
> prop.test(x = obs["successes", ],
+     n = obs["trials", ], correct = FALSE)
```

the results are

```
     4-sample test for equality of proportions without continuity correction


data:  obs["successes", ] out of obs["trials", ]
X-squared = 1.1743, df = 3, p-value = 0.7592
alternative hypothesis:  two.sided
sample estimates:
   prop 1    prop 2    prop 3    prop 4
0.5277778 0.4782609 0.3809524 0.4897959
```

To use the `chisq.test` function to perform the same test, the observed frequencies table needs to be changed so that the second row contains the number of failures. One can do this using a slick trick from matrix multiplication (see script file), or more simply, as follows. First, create a new observed frequency table

```
> obsNew <- obs
```

and then replace second row with failures.

```
> obsNew[2, ] <- obsNew[2, ] - obsNew[1, ]
```

Next, rename the rows and take a look.

```
> rownames(obsNew) <- c("successes", "failures"); obsNew
          mathsci natsci other socsci
successes      19     33     8     24
failures       17     36    13     25
```

Finally, the `chisq.test` function produces the following output.

```
> chisq.test(x = obsNew, correct = FALSE)


        Pearson's Chi-squared test


data:  obsNew
X-squared = 1.1743, df = 3, p-value = 0.7592
```

Conclusion: At any traditional level of significance there is insufficient efidence to reject Bubba's claim that the true proportions of students who are likely to get at least 70% in the project-based course final is the same across fields ($\chi^2 = 1.1743$, $df = 3$, p-value $= 0.7592$).

See script file for the alternative approach to using the `prop.test` function code.

Exploratory Exercises

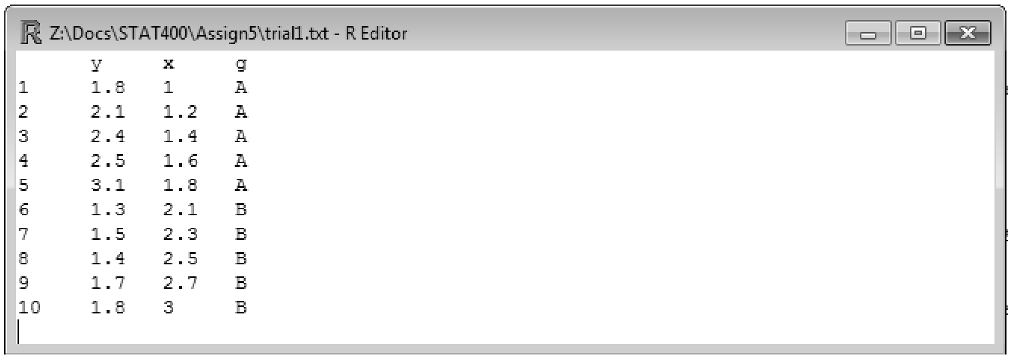
Spreadsheet programs, such as Excel, can be used to prepare and save data in *text* or *csv* (*comma-separated-values*) format. To do this, for example, open a new Excel workbook and first remove all but one worksheet by right-clicking the mouse on each extra sheet (for example, Sheet2) and selecting **Delete**. Next, place the variable names in the first row of the remaining worksheet (for example, Sheet1) and corresponding data below each variable name. Missing data can be identified, for example, by an empty space (for csv format), or some other character such as an asterisk (for text format). Finally, use the **File→Save As** option in Excel to save the file as a "`*.txt`" or "`*.csv`" type file.

Preparing and Saving Text Files in Excel

Open an Excel spreadsheet, remove the extra worksheets, and then enter the data exactly as shown below. Note that the first column contains case numbers, and the first entry in the first row is left blank.

```
     y   x g
1  1.8 1.0 A
2  2.1 1.2 A
3  2.4 1.4 A
4  2.5 1.6 A
5  3.1 1.8 A
6  1.3 2.1 B
7  1.5 2.3 B
8  1.4 2.5 B
9  1.7 2.7 B
10 1.8 3.0 B
```

Now use the **File→Save As** option in Excel to save the file (in your STAT S400 Assignment 5 folder) as a "`*.txt`" file, call this file `trial1.txt`. Check the folder to make sure it was saved. Now close the Excel worksheet. If the following steps do not work, it *may* be because your Excel worksheet is still open, or because the format in which the files is saved is not correct.



Z:\Docs\STAT400\Assign5\trial1.txt - R Editor

```
      y     x     g
1     1.8   1     A
2     2.1   1.2   A
3     2.4   1.4   A
4     2.5   1.6   A
5     3.1   1.8   A
6     1.3   2.1   B
7     1.5   2.3   B
8     1.4   2.5   B
9     1.7   2.7   B
10    1.8   3     B
```

Figure 1: Viewing the data text file `trial1.txt` in the R Editor window. Note, this suggests you could very easily prepare a text file of data in the R Editor!

Reading Text Files into R

Run the code

```
> read.table(file = file.choose())
```

and, when prompted, go to your STAT S400 Assignment 5 folder and select the file `trial1.txt`. The console should show the following output

```
     y   x g
1  1.8 1.0 A
2  2.1 1.2 A
3  2.4 1.4 A
4  2.5 1.6 A
5  3.1 1.8 A
6  1.3 2.1 B
7  1.5 2.3 B
8  1.4 2.5 B
9  1.7 2.7 B
10 1.8 3.0 B
```

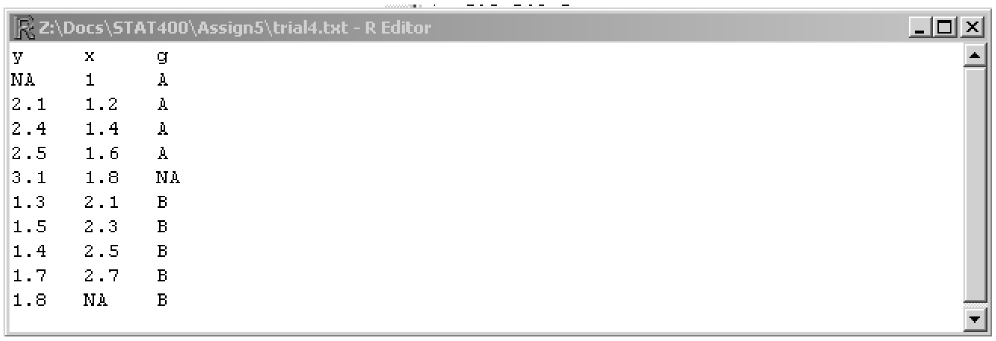
Now, use the script editor to open the file `trial1.txt` – You will have to select "Files of type:" "All files (*.*)" to be able to see the file. What you see in the new R Editor window should be exactly what you see on the console (See Figure 1). Close the R Editor window for `trial1.txt`, and read on.

There are some points to keep in mind when preparing a data text file for importing into R using the `read.table` function.

Variations in File Format

First consider a nicely filled out text file, such as the file `trial1.txt`. The first row in this file contains the variable names, and there is one less entry in this row as compared to the rest. Next, each row entry is separated from its next neighbor by a blank space (or a tab), and the first column contains the row names (or case numbers). Finally, there are no missing entries in any of the rows. In this scenario, since there is one less entry in the first row, R identifies the first row as the header-row, and the first column as the row-names column. In summary, the above simple `read.table` function call suffices.

Now consider two variations of the file `trial1.txt`.



```
y     x     g
NA    1     A
2.1   1.2   A
2.4   1.4   A
2.5   1.6   A
3.1   1.8   NA
1.3   2.1   B
1.5   2.3   B
1.4   2.5   B
1.7   2.7   B
1.8   NA    B
```

Figure 2: A data file with some missing entries.

Variation 1

In the first scenario, use Excel to open the file `trial1.txt`, and then remove the first column (the row-names column), Save this as the file `trial2.txt`, now close Excel.

Notice that here the first row, containing the variable names, has the same number of entries as the remaining rows. You have to alert R to this by including the argument assignment `header = TRUE`; run

```
> read.table(file = file.choose(), header = TRUE)
```

and choose the file `trial2.txt` as described earlier. You should see exactly the same output as before.

Variation 2

In the second scenario, use Excel to open the file `trial1.txt`, and then remove the first column (the row-names column) *and* the first row (column-names, or header row). Save this as the file `trial3.txt`, now close Excel, and then run

```
> read.table(file = file.choose(), header = FALSE,
+     col.names = c("y","x","g"))
```

and choose the file `trial3.txt`. It should be the case that all data frames produced so far are identical.

Placing Data from a Text File into an R Object

Any one of the above `read.table` function calls can be used to directly store the data in a named object (which will always be a data frame) through the use of the *assignment operator*. For example, running

```
> trialData <- read.table(file = file.choose(), header = TRUE)
```

and then selecting the file `trial2.txt` results in the data frame `trialData` being created. To be sure, run the function `ls()`, and also see what the contents of `trialData` look like.

Handling Missing Data

When it comes to missing data, the default setting for the `read.table` function in identifying such cases is to read and flag all entries containing an `NA` as being missing. To illustrate this, remove some data entries from the file `trial2.txt` and identify these with an `NA` (see Figure 2), and save this as `trial4.txt`.

Now run

```
> incompData <- read.table(file = file.choose(),
+     row.names = as.character(1:10), header = TRUE)
```

and choose the file `trial4.txt`. On looking at the contents of the data frame `incompData`, it will be noticed that the missing data are identified by an `NA`.

One can also check each entry of a dataset for the presence of an `NA` "value" using

```
> is.na(incompData)
       y     x     g
1   TRUE FALSE FALSE
2  FALSE FALSE FALSE
3  FALSE FALSE FALSE
4  FALSE FALSE FALSE
5  FALSE FALSE  TRUE
6  FALSE FALSE FALSE
7  FALSE FALSE FALSE
8  FALSE FALSE FALSE
9  FALSE FALSE FALSE
10 FALSE  TRUE FALSE
```

Entries that are not available (have a value of `NA`) are flagged by a `TRUE`. Alternatively, one could find the row (or case) and column numbers containing missing data using

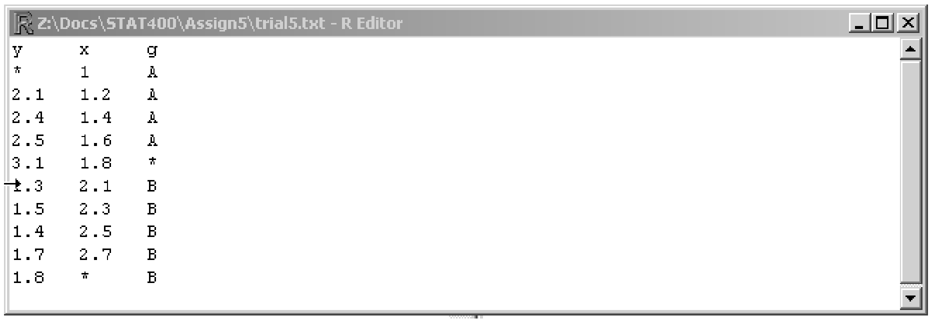
```
> which(is.na(incompData), arr.ind = TRUE)
   row col
1    1   1
10  10   2
5    5   3
```

Any letter, symbol, and even a blank space can be used to denote a missing value in a text file, and R can be instructed to identify such cases as being `NA`. For example, if the `NA` identifiers in `trial4.txt` are replaced by asterisks, and the result is saved as `trial5.txt` (See Figure 3), then run

```
> read.table(file = file.choose(), header = TRUE,
+     row.names = as.character(1:10), na.strings = "*")
```

and select the file `trial5.txt`. The resulting output is exactly the same as the data frame `incompData`.

Having determined that there are cases with missing values present in a dataset, the data can be left as is and missing cases can be handled later, if and when the necessity arises.



```
y     x     g
*     1     A
2.1   1.2   A
2.4   1.4   A
2.5   1.6   A
3.1   1.8   *
1.3   2.1   B
1.5   2.3   B
1.4   2.5   B
1.7   2.7   B
1.8   *     B
```

Figure 3: Alternative ways of flagging missing data in text files.

Alternatively, all cases having missing data can be omitted using

```
> na.omit(incompData)
```

Many functions in R are equipped to handle missing cases by a default setting, typically through an argument named `na.rm`. Assigning this argument a value of `TRUE` removes all `NA` cases, and a `FALSE` leaves things as is. Default settings vary from function to function.

Reading CSV Files

Excel can also be used to prepare CSV files in pretty much the same way as text files, just save the file as a "`*.csv`" file. Files containing tables of data in *comma-separated-values* format (CSV), have values separated by commas. Missing data are represented by the absence of data. Row and column names are handled in the same manner as for the `read.table` function. In such cases data can be read using

```
read.csv(file = file.choose(), na.strings = "")
```

or

```
read.csv(file = file.choose(), na.strings = "", header = TRUE)
```

as appropriate.

If the argument assignment `na.strings = ""` is left out of the `read.csv` function call, missing data in the last column will not be flagged with an `NA`. In fact, this entry is read as an "empty character" (as `""`), and is defined as one of the levels of a factor. As with the `read.table`, other symbols can be used to represent missing entries.

Saving and Exporting Data Revised in R

For the following illustrations, consider a "cleaned" version of the previously obtained data frame `incompData`. Remove all rows containing an `NA`, name the resulting data frame `cleanData`, and display the contents by enclosing the assignment statement within parentheses.

```
> (cleanData <- na.omit(incompData))
    y   x g
2 2.1 1.2 A
3 2.4 1.4 A
4 2.5 1.6 A
6 1.3 2.1 B
7 1.5 2.3 B
8 1.4 2.5 B
9 1.7 2.7 B
```

It can be observed that there are seven rows, however, for larger data frames the number of rows can be found more conveniently using

```
> nrow(cleanData)
[1] 7
```

The number of columns in a data frame can be found using `ncol`. Alternatively, the dimensions of a data frame can be found using the function `dim`.

Suppose it is preferably to rename the rows with new case numbers, 1 through 7. Then,

```
> row.names(cleanData) <- as.character(c(1:7)); cleanData
    y   x g
1 2.1 1.2 A
2 2.4 1.4 A
3 2.5 1.6 A
4 1.3 2.1 B
5 1.5 2.3 B
6 1.4 2.5 B
7 1.7 2.7 B
```

does the job.

Two alternatives to saving data are illustrated below, the choice of which approach to use being dependent on individual preferences or need.

As *.RData Files

To write an "external" representation of the data frame `cleanData` to a new file named, for example, `clean.RData`, run

```
> save("cleanData", file = file.choose())
```

After moving to an appropriate directory and entering the file name `clean.RData` in the "Select file" window, and then pressing the "Open" button a popup window appears asking whether to create the file. Click on "Yes" to complete the process. The file should appear in the directory chosen.

If, in the future, this dataset is needed, then the `load` function can be used as follows. Run

```
> load(file = file.choose())
```

and at the prompt in the "Select file" window, find and choose the file `clean.RData`. This task can also be accomplished using the R Console menu option sequence **File→Load Workspace...**.

As Text Files

To write a data table to a text file, run, for example,

```
> write.table(x = cleanData, file = file.choose())
```

then, in the "Select file" window, go to an appropriate directory and enter the new file name, say, `cleanData.txt`. The file will appear in this directory, and can be opened and inspected using the R Editor. Note that you may have to identify "Files of type:" as "All files (*.*)" in the "Open script" window in order to see the filename.

As with earlier created text files, the data in this file can be imported using the `read.table` function as described previously.

To be Submitted

Two datasets are to be used for the following exercises.

The Data

The first, Bubba's data, is as described in Assignment 4 (in the file `Assignment4.RData`). You may want to clear the workspace and reload this dataset.

The second dataset is contained in a text file, called `Assignment5.txt`, the contents of which are described below. Use the `read.table` function to load the data in a data frame called `montana` (you might want to first look inside `Assignment5.txt` to get a feel for what to expect). Make sure to take a look at the contents of the data frame to check that `NA` entries are correctly identified.

Montana Outlook Poll Data

These data were obtained from the *DASL* (Data and Story Library) website, located at

```
http://lib.stat.cmu.edu/DASL/
```

Here is the description of the data.

The Montana poll asked a random sample of Montana residents whether their personal financial status was the worse, the same, or better than a year ago, and whether they thought the state economic outlook was better over the next year. This file contains these items and accompanying demographics about the respondents. The file contains results for every *other* person ($n = 209$) included in the poll (i.e., first, third, fifth, etc.).

The variable names, and possible values are

`age`	1 = under 35, 2 = 35-54, 3 = 55 and over
`sex`	0 = male, 1 = female
`inc`	Yearly income: 1 = under \$20K, 2 = 20-35\$K, 3 = over \$35K
`pol`	Political Affiliations: 1 = Democrat, 2 = Independent, 3 = Republican
`area`	Region of Montana: 1 = Western, 2 = Northeastern, 3 = Southeastern Montana
`fin`	Personal financial status: 1 = worse, 2 = same, 3 = better than a year ago
`stat`	State economic outlook: 0 = better, 1 = not better than a year ago

Observe that even though all of the variables are categorical, the values assigned are integers. Also, be aware that missing data are identified by an asterisk. You may pretend that these data are very recent.

After loading the data into the data frame `montana` you will need to define all variables within `montana` as *factors*. Run, for example,

```
str(montana)
```

You can do this one at a time using code of the form, for example

```
montana$age <- factor(montana$age)
```

or, you can do this all at once using code of the form

```
montana <- with(data = montana,
    expr = data.frame(age = factor(age), sex = factor(sex),
                      inc = factor(inc), pol = factor(pol), ...))
```

Recall, `as.factor` can also be used. A quick way to check whether the conversion has succeeded is to run any of the following: `summary(montana)`, `attributes(montana)`, or `str(montana)`.

Finally, create a new data frame in which all rows with missing entries are removed. You may call the new data frame, for example, `montanaClean`.

The Problem Statements

Please read the directions in the problem statements carefully. If anything is unclear, ask.

1. So, Bubba claims that a higher proportion of (all possible) students who take a project-based elementary statistics will secure at least 70% on the final, as compared to students who take a traditional course. The task here is to use his data to test his claim. That is, let p_{proj} represent the true proportion of all students who, on taking the project-based course, will score at least 70% on the final, and let and p_{trad} represent the true proportion of all students who, on taking the traditional course, will score at least 70% on the final. Then, test the hypotheses

   $$H_0 : p_{\text{proj}} \leq p_{\text{trad}} \quad \text{vs.} \quad H_1 : p_{\text{proj}} > p_{\text{trad}} \quad \text{(Claim)}$$

   You may do the computations for this "from scratch" *or* you may use an appropriate built-in function.

2. Use the `montanaClean` data to answer the question: "Is there a difference in the true proportions *of women* across political affiliations?" This can be restated more formally in the following manner. Let p_1, p_2 and p_3 represent the true proportion *of women* in Montana who are Democrats, Independents and Republicans, respectively. The task, then, is to test hypotheses

   $$H_0 : p_1 = p_2 = p_3 \quad \text{vs.} \quad H_1 : p_j \neq p_k \text{ for at least one pair } j \neq k, \ \in \{1, 2, 3\}\,.$$

   You may do the computations for this "from scratch" *or* you may use an appropriate built-in function.

3. An incumbant Montana politician claims that the majority of Montana residents believe they (personally) are better off than they were a year ago. Use the `montanaClean` data to test this claim. As with the previous exercises, it is a good idea to set the stage. Let $p_{\text{better off}}$ represent the true proportion of Montana residents who believe they (personally) are better off than they were a year ago. Then the hypotheses to be tested are

   $$H_1 : p_{\text{better off}} \leq 0.5, \qquad H_1 : p_{\text{better off}} > 0.5 \quad \text{(Claim)}$$

   Again, you may do the computations for this "from scratch" *or* you may use an appropriate built-in function.