MTH 3220 Lab 2

Due Wed., Sept. 19

1 Part A: Power of the One-Sample t Test

Recall that the ***power*** of a test (when H_a is true) is:

$$\text{Power} \;=\; 1 - \beta(\mu)$$

where

$$\beta(\mu) \;=\; P(\text{Type II error}) \,.$$

To compute the **power** of a t **test** in R, we use `power.t.test()`. Among its arguments are:

`n`	number of observations (per group)
`delta`	(absolute value of) true difference in means
`sd`	standard deviation σ
`sig.level`	significance level (Type I error probability), with default 0.05
`power`	power of test (1 minus Type II error probability) for computing sample size
`type`	string specifying the type of t test. One of `"two.sample"`, `"one.sample"`, or `"paired"`.
`alternative`	one- or two-sided test. Must be one of `"one.sided"` or `"two.sided"`.

For example, consider a **one-sample t test** of the hypotheses:

$$\begin{aligned} H_0 : \mu &= 40 \\ H_a : \mu &> 40 \end{aligned}$$

The **power** of a t **test** (when H_a is true) will depend on:

- The level of significance of the test $\boldsymbol{\alpha}$ (a larger α leads to more power but also to higher Type I error probability).
- The population mean $\boldsymbol{\mu}$ (a μ farther above 40 leads to more power).
- The population standard deviation $\boldsymbol{\sigma}$ (a larger σ leads to less power).
- The sample size $\boldsymbol{n}$ (a larger n leads to more power).

We'll find the **power** of the test (using $\boldsymbol{\alpha = 0.05}$) when the population mean is $\boldsymbol{\mu = 42}$ and the sample size is $\boldsymbol{n = 10}$.

We need to specify (guess) a value for $\boldsymbol{\sigma}$. We'll suppose $\boldsymbol{\sigma = 7}$.

To find the **power** under these circumstances, we'd type:

```r
power.t.test(n = 10, delta = 2, sd = 7, sig.level = 0.05,
             type = "one.sample",
             alternative = "one.sided")
```

```
##
##      One-sample t test power calculation
##
##               n = 10
##           delta = 2
##              sd = 7
##       sig.level = 0.05
##           power = 0.209448
##     alternative = one.sided
```

The argument `delta` is used to specified $|\mu - \mu_0|$, in our case 42 - 40 = 2.

From the output, under the specified circumstances the **power** is only about **0.209**.

1. Use `power.t.test()` to find the **power** of the test described above under each of these three scenarios: 1) $\boldsymbol{\mu = 42}$; 2) $\boldsymbol{\mu = 45}$; 3) $\boldsymbol{\mu = 50}$. Assume $\boldsymbol{\sigma = 7}$ and $\boldsymbol{n = 10}$ in each case.

2. Now use `power.t.test()` to find the **power** of the test described above under each of these three scenarios: 1) $\boldsymbol{n = 10}$; 2) $\boldsymbol{n = 25}$; 3) $\boldsymbol{n = 100}$. Assume $\boldsymbol{\mu = 42}$ and $\boldsymbol{\sigma = 7}$ and in each case.

3. **Larger sample sizes** result in **more power**.

   To find the **necessary sample size** for attaining a **desired power**, we specify the **desired power** via the `power` argument in `power.t.test()`, and leave `n` unspecified.

   For example, to find the **necessary sample size** for attaining a **power** of **0.80** when $\boldsymbol{\mu = 42}$ and $\boldsymbol{\sigma = 7}$ (using $\boldsymbol{\alpha = 0.05}$), we'd type:

```r
power.t.test(delta = 2, sd = 7, sig.level = 0.05,
             power = 0.8,
             type = "one.sample",
             alternative = "one.sided")
```

```
##
##      One-sample t test power calculation
##
##               n = 77.10731
##           delta = 2
##              sd = 7
##       sig.level = 0.05
##           power = 0.8
##     alternative = one.sided
```

From the output, we'd require a sample size of *at least* $\boldsymbol{n = 78}$ (we always round up).

Use `power.t.test()` to find the **necessary sample size** for attaining a **power** of **0.90** when $\boldsymbol{\mu = 42}$ and $\boldsymbol{\sigma = 7}$ (using $\boldsymbol{\alpha = 0.05}$).

```
power.t.test(delta = 2, sd = 7, sig.level = 0.05,
             power = 0.9,
             type = "one.sample",
             alternative = "one.sided")
```

2 Part B: Paired t Test and Confidence Interval

2.1 Books versus DVDs Data Set

A study was conducted to decide if college students retain information better by reading the material in a **book** or by watching it on a **DVD**.

A sample of $\mathbf{n} = \mathbf{7}$ students was each assigned a 19th-century novel to read and a different one to watch. Afterward, each student was given a quiz on each novel. The test results (as percents) are below.

Student	Book Test Result	DVD Test Result	Difference
1	90	85	5
2	80	72	8
3	90	80	10
4	75	80	-5
5	80	70	10
6	90	75	15
7	84	80	4

1. Use `c()` and `<-` to create one vector containing the **book** test results and another the **DVD** results. Here are the data in a more convenient format:

Book	90,	80,	90,	75,	80,	90,	84
DVD	85,	72,	80,	80,	70,	75,	80

2. The function `t.test()` will carry out a ***paired t test*** and compute a ***95% paired t CI***. Among its argument are:

`x`	a data vector.
`y`	another optional data vector.
`alternative`	the direction for the alternative hypothesis, one of `"two.sided"`, `"less"`, or `"greater"`.
`mu`	the null hypothesized value for the unknown population mean (or difference in means if you are performing a two-sample test), with default value 0.
`paired`	a logical indicating whether you want a paired t-test.
`conf.level`	the confidence level for a confidence interval for the unknown population mean (or difference in means for a two-sample problem), with default value 0.95.

We want to carry out a **paired t** test of

$$\begin{aligned} H_0 : \mu_1 - \mu_2 &= 0 \\ H_a : \mu_1 - \mu_2 &\neq 0 \end{aligned} \qquad \text{or equivalently} \qquad \begin{aligned} H_0 : \mu_d &= 0 \\ H_a : \mu_d &\neq 0 \end{aligned}$$

where $\boldsymbol{\mu_1}$ is the (unknown) population mean score after reading a **book**, $\boldsymbol{\mu_2}$ is the population mean after watching a **DVD**, and $\boldsymbol{\mu_d}$ is the true mean **difference** (**book** score minus **DVD** score).

Specifying `paired = TRUE` in `t.test()` tells R the samples are **paired**. Now carry out the **paired t test**, for example by typing:

```
t.test(x = Book, y = DVD, mu = 0, paired = TRUE, alternative = "two.sided")
```

(assuming you named your vectors `Book` and `DVD`).

3. The paired t test rests on an assumption that either the **differences** can be considered as a sample from a **normal population** or $\boldsymbol{n}$ **is large**.

   Compute the **differences** (**book** minus **DVD**):

```
Diffs <- Book - DVD
```

   Check the normality assumption using a **histogram** and a **normal probability plot** of the **differences**:

```
hist(x = Diffs, col = "blue", main = "Histogram of Differences",
     xlab = "Difference")
```

```
qqnorm(y = Diffs, pch = 19)
qqline(y = Diffs, col = "blue", lwd = 2)
```

3 Part C: One Sample Z Test for a Population Proportion p

3.1 Political Poll Results

If a random sample is from a population of *successes* and *failures*, then the **test statistic** in a ***one-sample z test*** of

$$\begin{aligned} H_0 : p &= p_0 \\ H_a : p &\neq p_0 \end{aligned}$$

where $\boldsymbol{p}$ is the population **proportion** of successes, is

$$Z = \frac{\hat{P} - p_0}{\sqrt{\frac{p_0(1-p_0)}{n}}}.$$

A November 18, 2015, online CNN article had the headline "Poll: Clinton trails GOP rivals in Colorado."

The headline referred to a Quinnipiac poll which found that "Dr. Ben Carson tops Democratic front-runner Hillary Clinton **52 - 38** percent in a general election matchup."

Referring to its survey methodology, Quinnipiac states "This RDD (Random Digital Dialing) telephone survey was conducted from November 11 - 15, 2015 throughout the state of Colorado. Responses are reported for **1,262** self-identified registered voters."

1. The function `prop.test()` takes arguments

`x`	the number of successes in the sample.
`n`	the sample size.
`p`	the null hypothesized value for the unknown population proportion.
`alternative`	the direction for the alternative hypothesis, one of `"two.sided"`, `"less"`, or `"greater"`.
`conf.level`	the confidence level for a confidence interval for the unknown population proportion, with default value 0.95.

It returns the ***sample proportion*** $\hat{P}$ along with the results of a ***one-sample z test*** (and ***95% CI***) for p.

Assume that **656** (52%) of the **1,262** respondents supported Carson.

Use `prop.test()` to carry out a **one-sample *z* test** of of

$$\begin{aligned} H_0 : p &= 0.5 \\ H_a : p &\neq 0.5 \end{aligned}$$

where p is the (unknown) population **proportion** that supported Carson, for example:

```
prop.test(x = 656, n = 1262, p = 0.5, alternative = "two.sided")
```

Note: The test statistic returned by `prop.test()` is the **square** of Z, and is denoted `X-squared` in the output.

4 Part D: The Two Sample *Z* Test (for Two Population Proportions)

4.1 Secondhand Stores Data Set

Shopping at secondhand stores is becoming more popular and has even attracted the attention of some business schools. A study of customers' attitudes toward secondhand stores interviewed samples of shoppers at two secondhand stores of the same chain in **two cities**. The breakdown of the respondents **by gender** is as follows:

		Total Sample Size	Gender: Women	Gender: Men
City	City 1	$m = 241$	203	38
	City 2	$n = 218$	150	68

1. Compute the two ***sample proportions*** of women shoppers $\hat{P}_1$ and $\hat{P}_2$ for the two cities.
2. We want to carry out a **two-sample z test** of

$$\begin{aligned} H_0 : p_1 - p_2 &= 0 \\ H_a : p_1 - p_2 &\neq 0 \end{aligned}$$

where p_1 and p_2 are the (unknown) population proportions of shoppers who are women in the two cities. We also want to compute a ***95% CI*** for the difference $p_1 - p_2$.

The `prop.test()` function will carry out a ***two-sample z test*** and compute a ***95% two-sample z confidence interval for $p_1 - p_2$***. In this case, we pass a *two-element vector* of success counts for the two samples via `x`, and a *two-element vector* of sample sizes via `n`. No value is passed for `p`. For example:

```
prop.test(x = c(203, 150), n = c(241, 218), alternative = "two.sided")
```

Note: The test statistic returned by `prop.test()` is the **square of z**, and is denoted `X-squared` in the output.

Now carry out the test and compute the confidence interval using `prop.test()`.