Homework #4

Due: Tuesday, October 19 @ 6pm [35*points*]

Please remember to give R code, as well as answers, for any problems where you used R

Problem 1: [5*points*]

We are interested in estimating the concentration, in parts per billion (ppb) of *E. coli* in Lake Michigan on the basis of measurements of a number of samples. Suppose measurements of such samples will be approximately normally distributed with unknown mean (the true concentration) and known SD $\sigma = 1.5$ ppb. How many samples should we take if we wish our 95% CI for the true concentration to have a width ≤ 1 ppb?

Since we know σ and know that our samples are normally distributed, we can use the CLT. We know that the 95% CI ranges [$\bar{y}$ - $1.96(\sigma/\sqrt{n})$, $\bar{y}$ + $1.96(\sigma/\sqrt{n})$], so it has a total width of $2(1.96(\sigma/\sqrt{n}))$. Plugging in numbers, we get:

$1 \geq$ 95% CI width

$1 \geq 2(1.96(\sigma/\sqrt{n}))$

$1 \geq 2(1.96(1.5/\sqrt{n}))$

$0.255102 \geq 1.5/\sqrt{n}$

$\sqrt{n} \geq 5.880001$

$n \geq 34.57$

Hence, we should take at least n = 35 samples.

Note: qnorm(0.975) would be a more appropriate term than 1.96, either is acceptable.

Problem 2: [8*points*]

Suppose we measure the log_{10} cytokine response of 15 mice following some treatment, and observe the sample mean $\bar{X} = 1.2$ and sample SD $s = 2.3$

a. Suppose that your null hypothesis is that the population mean $\mu = 0$. Under what circumstances could you use a t-test to test this hypothesis? [1*point*]

Answer I am looking for here is if X is normally distributed. (Note: also important that samples are random and independent)

b. Assuming the conditions in part (a) hold, what would the t-statistic be? [1*point*]

```
tStatistic <- (1.2-0)/(2.3/sqrt(15))
tStatistic
```

```
## [1] 2.020687
```

c. If your *alternative* hypothesis is that the cytokine response is *greater* than 0, what would the p-value be? (Use R.) [1*point*]

```
pt(tStatistic, df = 15-1, lower.tail = F)
```

```
## [1] 0.03143395
```

d. In words, how would you interpret/describe the result you got in (c)? [1*point*]

There is a 0.031 chance of having observed a mean greater than or equal to 1.2 in a sample of size 15 if the true population mean is 0. We would thus reject the null hypothesis ($H_0 : \mu = 0$) at a significance (type I error) $\alpha = 0.05$ given our alternative hypothesis.

e. If your *alternative* hypothesis is that the cytokine response is *different from 0*, what would the p-value be? (Use R.) [1*point*]

```
pt(tStatistic, df = 15-1, lower.tail = F) + pt(-1*tStatistic, df = 15-1, lower.tail = T)
```

```
## [1] 0.0628679
```

f. In words, how would you interpret/describe the result you got in (e)? [1*point*]

There is a 0.063 chance of having observed a mean as or more "extreme" as 1.2 (i.e. $\geq$ 1.2 or $\leq$ -1.2) in a sample of size 15 if the true population mean is 0. We could *not* reject the null hypothesis ($H_0 : \mu = 0$) at a significance (type I error) $\alpha = 0.05$ given our alternative hypothesis.

g. What is the smallest sample needed for (e) to be significant at the $\alpha = 0.05$ level assuming that everything else (sample SD, sample mean) remains the same? [17] What about for (c)? [12] (You may solve this algebraically *or* by trial and error in R.) [2*points*]

```
# one way to solve this problem for (e), there might be others:
# first, we need to know what test statistic would give us a p-value (area under the curve)
# of 0.025 (alpha = 0.05 but this is two-tailed so each tail is 0.025)
tvalue <- qt(p = 0.05/2, df = 15 - 1)
tvalue
```

```
## [1] -2.144787
```

```
# second, we can plug this in to the equation for a test statistic and solve for n
# t = (y - mu)/(s/sqrt(n))
# rearrange
# sqrt(n) = (t*s)/(y - mu)
((tvalue*2.3)/(1.2-0))^2
```

```
## [1] 16.89901
```

```
# Hence, we need n = 17
```

Problem 3: [10*points*]

Review the following functions to describe the confidence interval of the mean and perform a one-sided t-test in R:

```r
# define confidence interval of the mean
mean.conf.int <- function(x, CI = 0.95) {
    xbar <- mean(x)
    n <- length(x) # number of samples
    t.quantile <- qt(1-(1-CI)/2, df = n-1)
    std.error <- sd(x)/sqrt(n)
    conf.int <- c(xbar - t.quantile*std.error, xbar+t.quantile*std.error)
    return(conf.int)
}

# Perform a one-sided t-test
# when lower.tail = T, tests if mean of x is significantly LESS than mu
# when lower.tail = F, tests if mean of x is significantly GREATER than mu
# by default, mu = 0 and lower.tail = T
one.sided.t.test <- function(x, mu = 0, lower.tail = TRUE) {
    xbar <- mean(x)
    n <- length(x)
    sampSD <- sd(x)
    tStatistic <- (xbar - mu)/(sampSD/sqrt(n))
    p.value <- pt(tStatistic, df = n - 1, lower.tail = lower.tail)
    return(p.value)
}
```

a. What would happen if you applied these functions to a vector `x` containing NA's? [1*point*]

They would return NA

b. How would you modify these functions to accept data which has NA's in it? [2*points*]

There are a few options. One would be to insert as the first line of the function (immediately before computing xbar) x <- x[!is.na(x)] to only work with non-NA x values. Another option would be to use the na.rm = T options for mean() and sd(), being careful to change the sample size as well, for instance using n <- sum(!is.na(x)) to count only the non-NA samples

c. Write a function to test if the mean is significantly *different* from μ in *either* direction (greater than or less than). Note: your answer in this part will not count against your grade for this problem, as you will have the opportunity to revise it for part (g) below. [0*point*]

Note: what follows is an example of INCORRECT code that will be debugged below

```r
# INCORRECT VERSION
two.sided.t.test <- function(x, mu = 0) {
    xbar <- mean(x)
    n <- length(x)
    sampSD <- sd(x)
    tStatistic <- (xbar-mu)/(sampSD/sqrt(n)) ### THIS LINE INCORRECT!!!
    p.value <- pt(tStatistic, df = n-1, lower.tail = F) + pt(-1*tStatistic, df = n-1, lower.tail = T)
    return(p.value)
}
```

d. Create a vector `testData` (below). Without actually calculating it, would you expect the mean to be significantly different from 0? [1*point*]

```
testData <- c(-3, rep(-2, 5), rep(-1,10), rep(0,10), rep(1,5),2)
testData
```

```
##  [1] -3 -2 -2 -2 -2 -2 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  0  0  0  0  0  0  0  0  0
## [26]  0  1  1  1  1  1  2
```

Yes, there are many more negative values than positive values.

e. Using your function from (d), test $H_0 : \mu = 0$ against the alternative that $\mu \neq 0$. What do you get? Is this what you expected? If not, why not? [2*points*]

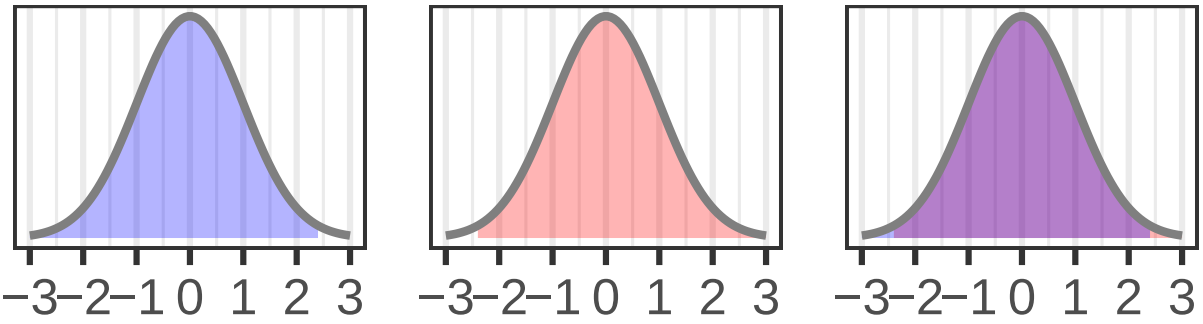
```
two.sided.t.test(testData)
```

```
## [1] 1.981655
```

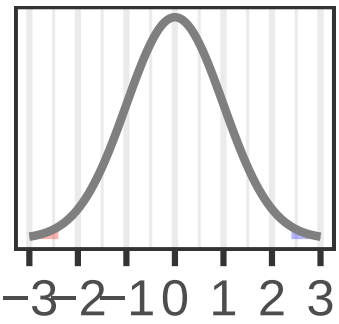
```
# Uh oh... p-values are probabilities, and should never be > 1!!!
```

We are obviously getting the wrong p-value (> 1). This is because our two-tailed test is accidentally going the "wrong way", up from our negative value rather than down, as shown below.

Incorrect:



Correct:



f. If the answer to (f) was what you expected, great! Rewrite your answer from (d) here for credit. If not, figure out how to modify your answer to part (d) such that it gives you the result you expect in (f), and write that below. Hint: `?abs` may be helpful. [4*points*]

We can correct the mistake above and ensure that we are going "up" from the higher number by taking the absolute value of the difference between xbar and mu. This makes sense, since when we say "different from mu in either direction" we don't care about the sign (i.e. about *which* direction it is), only the magnitude

```r
# INCORRECT VERSION
two.sided.t.test <- function(x, mu = 0) {
    xbar <- mean(x)
    n <- length(x)
    sampSD <- sd(x)
    # different in either direction --> need abs. value!
    tStatistic <- abs(xbar-mu)/(sampSD/sqrt(n)) ### THIS LINE INCORRECT!!!
    p.value <- pt(tStatistic, df = n-1, lower.tail = F) + pt(-1*tStatistic, df = n-1, lower.tail = T)
    return(p.value)
}

# and testing the real result
two.sided.t.test(testData)
```

```
## [1] 0.01834461
```

g. Reflect on this problem and write down any observations/lessons learned. [1*point*]

This problem demonstrates the importance of testing one's code against test data for which one has a intuition about what the result should be, and thinking about the output one gets to identify and debug errors.

Problem 4: [12*points*]

In problem 3, we wrote our own function to carry out a t-test. However, it happens that R already has a function built-in to do that! For this problem, you should first read the help page for the R function `t.test`. At this point, some of its options will be beyond what we've covered, so here we'll focus on the simplest usage.

a. Look at the usage for the "`## Default S3 method`". This is the usage it will default to when you call `t.test` on your data. What arguments *must* be specified (i.e., do not have default values)? [1*point*]

x, the data. y may optionally be given for a two-sample test, and all other arguments have default values.

b. For a one-sample test, like we've been doing so far in this homework, we can ignore the `y`, `paired`, and `var.equal` arguments. Suppose you had a sample called `myData`. How would you test: [3*points*]

- If the mean of `myData` were significantly different from 0?

t.test(myData)

- If the mean of `myData` were significantly less than 5?

t.test(myData, mu = 5, alternative = "less")

- $H_0 : \mu = 0$ vs. $H_A : \mu > 0$

t.test(myData, alternative = "greater")

c. Let `myData` be a sample of size 20 drawn from N(3,2). Create this data in R. Using YOUR functions from problem #3 (*not* `t.test`), carry out the tests described in part (b) above. [3*points*]

```r
# create a random distribution for myData - note, your data may differ!
set.seed(48)
myData <- rnorm(n = 20, mean = 3, sd = sqrt(2))

# HA: mu != 0
two.sided.t.test(myData)
```

```
## [1] 3.435697e-07
```

```
# HA: mu < 5
one.sided.t.test(myData, mu = 5, lower.tail = T)
```

```
## [1] 0.0002353483
```

```
# HA: mu > 0
one.sided.t.test(myData, lower.tail = F)
```

```
## [1] 1.717849e-07
```

d. Now, without changing `myData`, carry out these same tests using `t.test` as specified in your answer to (b). Do the p-values agree with yours? For the first test, describe each element of its output. What is it telling you? [5*points*]

The output shows the t-statistic and the df of the test, along with the p-value and the alternative being tested. It also gives the 95% CI for the mean

```
# HA: mu != 0
t.test(myData)
```

```
##
##  One Sample t-test
##
## data:  myData
## t = 7.6186, df = 19, p-value = 3.436e-07
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  2.335004 4.103960
## sample estimates:
## mean of x
##  3.219482
```

```
# HA: mu < 5
t.test(myData, mu = 5, alternative = "less")
```

```
##
##  One Sample t-test
##
## data:  myData
## t = -4.2134, df = 19, p-value = 0.0002353
## alternative hypothesis: true mean is less than 5
## 95 percent confidence interval:
##      -Inf 3.950185
## sample estimates:
## mean of x
##  3.219482
```

```
# HA: mu > 0
t.test(myData, alternative = "greater")
```

```
##
##  One Sample t-test
##
## data:  myData
## t = 7.6186, df = 19, p-value = 1.718e-07
## alternative hypothesis: true mean is greater than 0
## 95 percent confidence interval:
##  2.488778      Inf
## sample estimates:
## mean of x
##  3.219482
```