```
################################################################################

# Brian Weinstein - bmw2148
# STAT S4201 001
# Homework 2
# 2016-02-10

# set working directory
setwd("~/Documents/advanced-data-analysis/homework_02")

# load packages
library(dplyr)
library(Sleuth3) # Data sets from Ramsey and Schafer's "Statistical Sleuth
(3rd ed)"
library(ggplot2); theme_set(theme_bw())
library(tidyr)



# Problem 1
########################################################################

# Define a fn to calcuate the conf interval for the sample var, given the
sample var, n, and conf level
MakeVarConfidenceInterval <- function(sampleVar, n, confLevel){

  sigLevel <- 1 - confLevel

  lowerBound <- (n-1) * sampleVar / qchisq(p=(1-(sigLevel/2)), df=(n-1),
lower.tail=TRUE)
  upperBound <- (n-1) * sampleVar / qchisq(p=(sigLevel/2), df=(n-1),
lower.tail=TRUE)

  confidenceInterval <- c(lowerBound, upperBound)

  return(confidenceInterval)

}

# define variables for this problem
variance <- 1
mean <- 0
sample_size <- 10

# Initialize a counter at 0
numTrialsCapturingTrueVar=0

# Run 1000 trials, checking if the 95% CI captures the true varaince
set.seed(1)
for(trial in 1:1000){

  # Generate a random sample from N(mu=0, var=1)
  random_sample <- rnorm(n=sample_size, mean=mean, sd=sqrt(variance))
```

```
  # Compute the sample var of the random sample
  sample_var <- var(random_sample)

  # Compute a 95% CI for the sample variance
  confInt <- MakeVarConfidenceInterval(sampleVar=sample_var, n=sample_size,
confLevel=0.95)

  # Check if CI captured true sample variance and increment counter
  numTrialsCapturingTrueVar = numTrialsCapturingTrueVar + ifelse(confInt[1] <
variance & confInt[2] > variance, 1, 0)

}

c(numTrialsCapturingTrueVar, trial)
numTrialsCapturingTrueVar/trial

rm(list = ls()) # clear working environment



# Problem 2: Ramsey 3.22
######################################################################

# Data input
time26 <- c(5.79, 1579.52, 2323.70)
time28 <- c(68.8, 108.29, 110.29, 426.07, 1067.60)

# Part a ### ### ### ### ### ### ### ### ### ### ### ### ### ### ### ### ###
###

Y1 <- log(time26) ; Y1
Y2 <- log(time28) ; Y2

# Part b ### ### ### ### ### ### ### ### ### ### ### ### ### ### ### ### ###
###

mean(Y1)
mean(Y2)

mean(Y1) - mean(Y2)

# Part c ### ### ### ### ### ### ### ### ### ### ### ### ### ### ### ### ###
###

exp(mean(Y1) - mean(Y2))

# Part d ### ### ### ### ### ### ### ### ### ### ### ### ### ### ### ### ###
###

# Define variables to calculate confidence intervals
logDiff <- mean(Y1) - mean(Y2)
pooledSD <- sqrt( ((length(Y1) - 1)*var(Y1) + (length(Y2) - 1)*var(Y2)) /
(length(Y1) + length(Y2) -2) )
se <- pooledSD * sqrt((1/3) + (1/5))
```

```
quantile <- qt(1-(0.05/2), df=(3+5-2))

# Compute confidence interval
logDiff - (quantile * se)
logDiff + (quantile * se)

# Compute anilogarithm of confidence interval
exp(logDiff - (quantile * se))
exp(logDiff + (quantile * se))

rm(list = ls()) # clear working environment



# Problem 3: Ramsey 3.25
######################################################################

# load data
agentOrangeData <- Sleuth3::case0302

# compute group difference, 1-sided p-value, and 95% CI; with all observations
t1 <- t.test(formula=Dioxin~Veteran, data=agentOrangeData,
             var.equal=TRUE, conf.level=0.95, alternative="less")
-diff(t1$estimate)[[1]]
t1$p.value
t.test(formula=Dioxin~Veteran, data=agentOrangeData,
       var.equal=TRUE, conf.level=0.95)$conf.int

# compute group difference, 1-sided p-value, and 95% CI; excluding observation
646
t2 <- t.test(formula=Dioxin~Veteran, data=agentOrangeData[-646, ],
             var.equal=TRUE, conf.level=0.95, alternative="less")
-diff(t2$estimate)[[1]]
t2$p.value
t.test(formula=Dioxin~Veteran, data=agentOrangeData,
       var.equal=TRUE, conf.level=0.95)$conf.int

# compute group difference, 1-sided p-value, and 95% CI; excluding
observations 646 and 645
t3 <- t.test(formula=Dioxin~Veteran, data=agentOrangeData[-(645:646), ],
             var.equal=TRUE, conf.level=0.95, alternative="less")
-diff(t3$estimate)[[1]]
t3$p.value
t.test(formula=Dioxin~Veteran, data=agentOrangeData,
       var.equal=TRUE, conf.level=0.95)$conf.int

rm(list = ls()) # clear working environment



# Problem 4: Ramsey 3.28
######################################################################

# load data
sparrowData <- Sleuth3::ex0221
```

```
# create boxplots
ggplot(sparrowData, aes(x=Status, y=Humerus)) +
  geom_violin(alpha=0.15) +
  geom_boxplot() +
  labs(y="Humerus Length (inches)", title="Humerus Length: Perished vs
Survived")
ggsave(filename="writeup/4_full.png", width=6.125, height=3.5, units="in")

# compute group difference, 2-sided p-value, and 95% CI; with all observations
t.test(formula=Humerus~Status, data=sparrowData,
       var.equal=TRUE, conf.level=0.95)

# compute group difference, 2-sided p-value, and 95% CI; excluding the
smallest length in the perished group
t.test(formula=Humerus~Status, data=sparrowData,
       subset=(!(Humerus==min(sparrowData$Humerus) & Status=="Perished")),
       var.equal=TRUE, conf.level=0.95)

rm(list = ls()) # clear working environment



# Problem 5: Ramsey 3.32
######################################################################

# load data
tuitionData <- Sleuth3::ex0332

# Part a: Public school tuition (out-of-state > in-state) ### ### ### ### ###
### ### ### ### ### ### ### ### ### ### ### ### ### ###

# Subset the data for part a and convert to long format
tuitionData.a <- tuitionData %>%
  filter(Type=="Public") %>%
  gather(key="tuitionType", value="tuition", c(OutOfState, InState),
factor_key=TRUE) %>%
  arrange(College, tuitionType)

# create boxplots
ggplot(tuitionData.a, aes(x=tuitionType, y=tuition)) +
  geom_violin(alpha=0.15) +
  geom_boxplot() +
  labs(y="Tuition (USD)", title="(5a) Public School Tuition: In-State vs Out-
of-State")
ggsave(filename="writeup/5a_twoSample.png", width=6.125, height=3.5,
units="in")

# check group standard errors
tuitionData.a %>%
  group_by(tuitionType) %>%
  summarize(sd=sd(tuition))

# group standard errors are sginificantly different, so two-sample t-test is
unreliable here
```

```
# since the measurements are each paired, we can try the one-sample paired t-
test

# Create a new dataframe with the tuition differences
tuitionData.a2 <- tuitionData %>%
  filter(Type=="Public") %>%
  mutate(diff=OutOfState-InState)

# create boxplot
ggplot(tuitionData.a2, aes(x=Type, y=diff)) +
  geom_violin(alpha=0.15) +
  geom_boxplot() +
  labs(y="Tuition (USD)", title="(5a) Public School Tuition: (Out-of-State) -
(In-State)")
ggsave(filename="writeup/5a_paired.png", width=6.125, height=3.5, units="in")

# Perform paired t-test
t.test(x=tuitionData.a2$OutOfState, y=tuitionData.a2$InState, paired=TRUE,
       conf.level=0.95, alternative="greater")
# Perform a two sided paired t-test for the confidence interval
t.test(x=tuitionData.a2$OutOfState, y=tuitionData.a2$InState, paired=TRUE,
       conf.level=0.95, alternative="two.sided")$conf.int

# Part b: In-state tutition (private > public) ### ### ### ### ### ### ### ###
### ### ### ### ### ### ### ### ### ###

# Create dataset for part b
tuitionData.b <- tuitionData %>%
  mutate(OutOfState=NULL)

# create boxplots
ggplot(tuitionData.b, aes(x=Type, y=InState)) +
  geom_violin(alpha=0.15) +
  geom_boxplot() +
  labs(y="In-State Tuition (USD)", title="(5b) In-State Tuition: Private vs
Public")
ggsave(filename="writeup/5b_original.png", width=6.125, height=3.5,
units="in")

# Create a log-transformed InState tuition column
tuitionData.b <- tuitionData.b %>%
  mutate(LogInState=log(InState))

# create boxplots of the log-transformed data
ggplot(tuitionData.b, aes(x=Type, y=LogInState)) +
  geom_violin(alpha=0.15) +
  geom_boxplot() +
  labs(y="Log(In-State Tuition (USD))", title="(5b) Log(In-State Tuition):
Private vs Public")
ggsave(filename="writeup/5b_log.png", width=6.125, height=3.5, units="in")

# check group standard errors
tuitionData.b %>%
  group_by(Type) %>%
  summarize(sd=sd(LogInState))
```

```
# Perform a two-sample t-test
tt.b <- t.test(formula=LogInState~Type, data=tuitionData.b,
               var.equal=TRUE, conf.level=0.95, alternative="greater")
tt.b

# take antilog of the estimate
exp(-diff(tt.b$estimate)[[1]])

# Perform a two sided t-test for the confidence interval and take antilog
exp(t.test(formula=LogInState~Type, data=tuitionData.b,
           var.equal=TRUE, conf.level=0.95, alternative="two.sided")$conf.int)

# Part c: Out-of-state tutition (private > public) ### ### ### ### ### ### ###
### ### ### ### ### ### ### ### ### ### ###

# Create dataset for part c
tuitionData.c <- tuitionData %>%
  mutate(InState=NULL)

# create boxplots
ggplot(tuitionData.c, aes(x=Type, y=OutOfState)) +
  geom_violin(alpha=0.15) +
  geom_boxplot() +
  labs(y="Out-of-State Tuition (USD)", title="(5c) Out-of-State Tuition:
Private vs Public")
ggsave(filename="writeup/5c_original.png", width=6.125, height=3.5,
units="in")

# Create a log-transformed OutOfState tuition column
tuitionData.c <- tuitionData.c %>%
  mutate(LogOutOfState=log(OutOfState))

# create boxplots of the log-transformed data
ggplot(tuitionData.c, aes(x=Type, y=LogOutOfState)) +
  geom_violin(alpha=0.15) +
  geom_boxplot() +
  labs(y="Log(Out-ofn-State Tuition (USD))", title="(5c) Log(Out-of-State
Tuition): Private vs Public")
ggsave(filename="writeup/5c_log.png", width=6.125, height=3.5, units="in")

# check group standard errors
tuitionData.c %>%
  group_by(Type) %>%
  summarize(sd=sd(LogOutOfState))

# Perform a two-sample t-test
tt.c <- t.test(formula=LogOutOfState~Type, data=tuitionData.c,
               var.equal=TRUE, conf.level=0.95, alternative="greater")
tt.c

# take antilog of the estimate
exp(-diff(tt.c$estimate)[[1]])

# Perform a two sided t-test for the confidence interval and take antilog
```

```
exp(t.test(formula=LogOutOfState~Type, data=tuitionData.c,
           var.equal=TRUE, conf.level=0.95, alternative="two.sided")$conf.int)

# Check if outliers impace results in part c ## ## ## ## ## ## ## ## ## ## ##
## ## ## ##

# Create a dataset for part c excluding outliers
tuitionData.c2 <- tuitionData.c %>%
  filter(Type=="Private" & LogOutOfState < 11 |
           Type=="Public" & LogOutOfState < 10 & LogOutOfState > 9)

# Perform a two-sample t-test
tt.c2 <- t.test(formula=LogOutOfState~Type, data=tuitionData.c,
                var.equal=TRUE, conf.level=0.95, alternative="greater")
tt.c2

# take antilog of the estimate
exp(-diff(tt.c2$estimate)[[1]])

# Perform a two sided t-test for the confidence interval and take antilog
exp(t.test(formula=LogOutOfState~Type, data=tuitionData.c2,
           var.equal=TRUE, conf.level=0.95, alternative="two.sided")$conf.int)

# outliers don't impact the conclusion, so we only report the results that
included outliers

rm(list = ls()) # clear working environment



# Problem 6: Ramsey 4.19
#########################################################################

# load data
sparrowData <- Sleuth3::ex0221

# perform rank-sum test
wilcox.test(formula=Humerus~Status, data=sparrowData,
            paired=FALSE, # perform rank-sum test
            correct=TRUE, # apply continuity correction in the normal approx
for the p-value
            conf.int=TRUE, conf.level=0.95)

# check if group sd are nearly equal
sparrowData %>%
  group_by(Status) %>%
  summarize(count=n(),
            mean=mean(Humerus),
            sd=sd(Humerus))

# check if group sd are nearly equal when excluding the outlier
sparrowData %>%
  filter(!(Humerus==min(Humerus) & Status=="Perished")) %>%
  group_by(Status) %>%
  summarize(count=n(),
```

```
            mean=mean(Humerus),
            sd=sd(Humerus))

rm(list = ls()) # clear working environment
```