Assignment 8 solutions

EMATM0061: Statistical Computing and Empirical Methods, TB1, 2022

Introduction

This is the sixth assignment for Statistical Computing and Empirical Methods (Unit EMATM0061) on the MSc in Data Science & MSc in Financial Technology with Data Science. This assignment is mainly based on Lectures 14 and 15 (see the Blackboard).

The submission deadline for this assignment is 23:59, 28 November 2022. Note that this assignment will not count towards your final grade. However, it is recommended that you try to answer the questions to gain a better understanding of the concepts.

Create an R Markdown for the assignment

It is a good practice to use R Markdown to organize your code and results. You can start with the template called `Assignment08_Template.Rmd` which can be downloaded via Blackboard.

If you are considering submitting your solutions, please generate a PDF file. For example, you can choose the "PDF" option when creating the R Markdown file (note that this option may require Tex to be installed on your computer), or use R Markdown to output an HTML and print it as a PDF file in a browser, or use your own way of creating a PDF file that contains your solutions.

Only a PDF file will be accepted in the submission of this assignment. To submit the assignment, please visit the "Assignment" tab on the Blackboard page, where you downloaded the assignment.

Wish to know more about a perticular question?

You may want to ask a question during the computer lab.

Alternatively, we are collecting questions about this assignment that need to be addressed, through the following form. And this can be done either during the labs or outside the lab sessions. So, If you found a question in this assignment interesting but had difficulty in a particular step when trying to develop your answer, please put your remark in the form via the following link. A brief description of the difficulty would be very helpful. Giving your remark is optional, but we aim to know the most common questions that you might want to get some support.

https://forms.office.com/r/xQM4yX45GC

Load packages

Some of the questions in this assignment require the tidyverse package. If it hasn't been installed on your computer, please use `install.packages()` to install them first.

To road the tidyverse package:

```
library(tidyverse)
```

1. Obstacles to valid scientific inference

(Q1)

For each of the following give

(A) an explanation of the concept and
(B) an example of a situation (real or hypothetical) where they create a barrier to drawing scientific conclusions based on data.

You are encouraged to discuss these concepts with your colleagues.

1. Measurement distortions
2. Selection bias
3. Confounding variables

Answer

1. Measurement distortions

Measurement distortions occur whenever there is a mismatch between the quantities recorded within the data and the true variable of interest.

Example: Suppose we are investigating the effect of a new type of feed on chickens. For the purpose of the experiment a large sample of chickens is split into two groups. One group is given feed type A, and the other feed type B. After some time has elapsed, the chicken's weights are all measured. Suppose the measuring equipment used to weight the chickens given feed type A had a downward bias. On the other hand, the measuring device used to weight the chickens given feed type B has no such bias. We could then observe a data set where the recorded weights for chickens receiving feed source B typically exceed those receiving feed type A. However, this difference is purely an artefact of the faulty measurement equipment.

2. Selection bias

Selection bias occurs whenever the sample included in the analysis misrepresents the underlying population of interest.

a) Sample bias: When some members of the population are more likely to be selected than others.
b) Self selection bias: Occurs whenever people decide whether or not they should be assigned to a particular group.
c) Attrition bias: Occurs whenever the sample is distorted by people leaving the study.
d) Post-hoc selection: Occurs when a subset of the data is chosen based on the sample itself.

Example: A classic historical example of sample bias is the Literary Digest poll of 1936. In this people were asked about their voting intentions. The prediction was for a 57% victory for Republican Landon over Roosevelt. In fact Roosevelt won the election. The data was collected based upon surveys. These suveys were carried out based upon lists from telephone books and club membership. At the time both telephone ownership and club membership were indicators of wealth, so the sample was biased towards people with higher levels of wealth. At the time, higher levels of wealth were slightly more likely to vote Republican than Democrat. Hence, the selection bias resulted in a misleading result. Subsequent Gallop polls created more accurate results with smaller sample sizes.

3. Confounding variables

Suppose that we are interested in understanding the causal effect of an independent variable X on a dependent variable Y. A confounding variable is a third causal factor Z which effects both X and Y. This makes it difficult to disentangle causal effects from purely correlative behavior.

Example: As an example consider a scientific study into the causal effect of regular cardio-vascular exercise on longevity. Here a confounding variable could be someones overall interest in a healthy lifestyle. This is likely to effect someones via the causal effect of increased cardio-vascular exercise. However, it is likely that an interest in a healthy lifestyle will also effect the amount of fresh fruit and vegetables someone eats, for example. This may also have a causal effect on longevity. Hence, in light of the confounding variables it is difficult to distinguish a causal effect from a purely correlatative relationship.

2. paired t-test and effect size

The Barley data set gives the yields of two types of barley - Glabron and Velvet across twelve different fields. The data is paired as yields are given for both types of barley across each of the twelve fields.

```
library(PairedData) # you might need to install the package first
data("Barley")
detach('package:PairedData', unload=TRUE)
detach('package:MASS', unload=TRUE)
# unload package because it contains another select() function

head(Barley, 4)
```

```
##   Farm Glabron Velvet
## 1  F01      49     42
## 2  F02      47     47
## 3  F03      39     38
## 4  F04      37     32
```

(Q1)

Carry out a paired t-test to determine whether there is a difference in average yield between the two types of barley. Use a significance level of 0.01. You can use the t.test() function.

Answer

```
t.test(Barley$Glabron, Barley$Velvet, paired = TRUE, conf.level = 0.99)
```

```
##
##  Paired t-test
##
## data:  Barley$Glabron and Barley$Velvet
## t = 3.0133, df = 11, p-value = 0.0118
## alternative hypothesis: true mean difference is not equal to 0
## 99 percent confidence interval:
##  -0.1816612 12.0149945
## sample estimates:
## mean difference
##        5.916667
```

(Q2)

Compute the effect size using Cohen's d statistic.

Answer

```
diffs <- Barley$Glabron - Barley$Velvet
cohens_d<-mean(diffs)/sd(diffs)

cohens_d
```

```
## [1] 0.8698615
```

(Q3)

What assumptions are required for the paired t-test? Are these assumptions justified in this case?

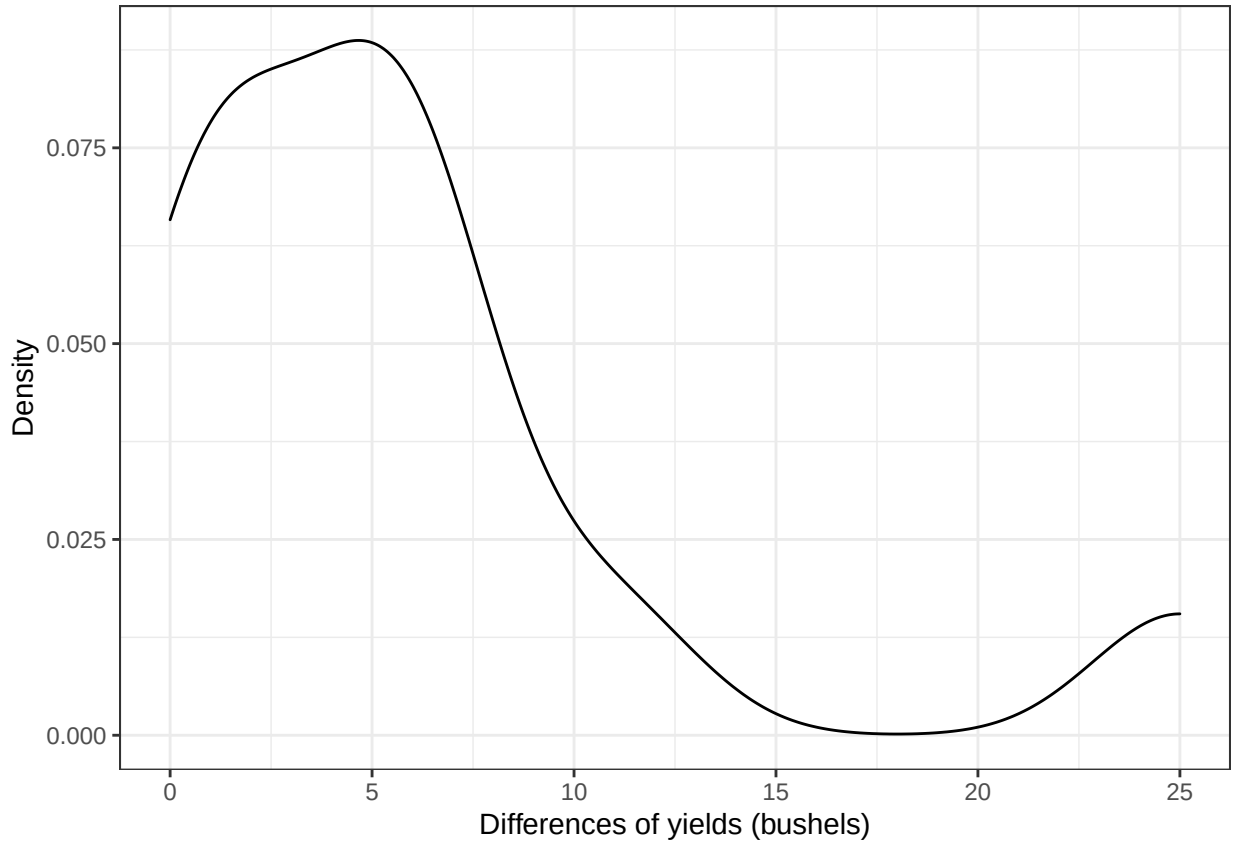
Answer

We assume that the difference of the two sample is either i.i.d. Gaussian or i.i.d. with a large sample size.

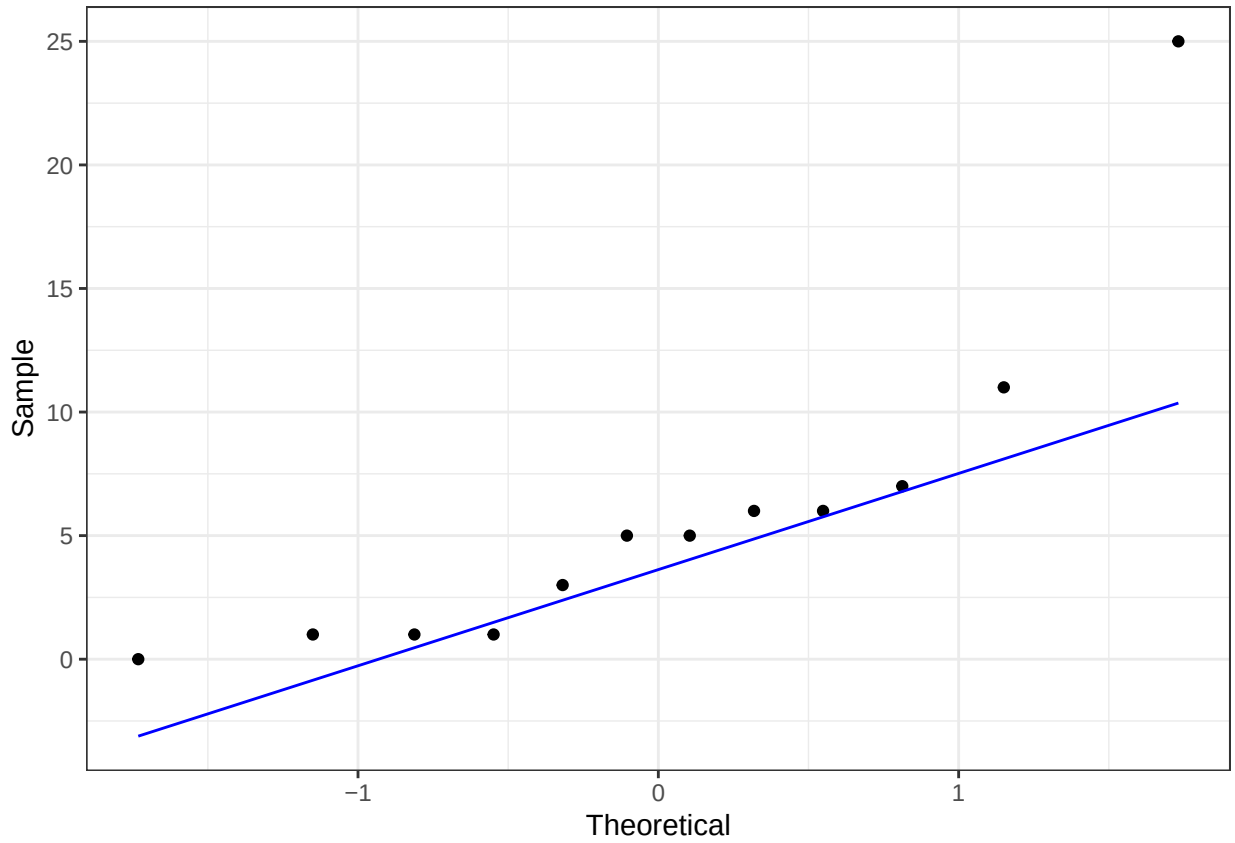
```
Barley %>% nrow()
```

```
## [1] 12
```

```
Barley %>%
  mutate(diff=Glabron-Velvet) %>%
  ggplot(aes(x=diff)) + geom_density() + theme_bw() +
  labs(x="Differences of yields (bushels)",y="Density")
```




```
Barley %>%
  mutate(diff=Glabron-Velvet) %>%
  ggplot(aes(sample=diff))+theme_bw()+
  stat_qq()+stat_qq_line(color="blue")+
  labs(x="Theoretical", y="Sample")
```



It seems that the difference between the two samples is approximately Gaussian (although the sample size itself its relatively small). Since the distribution of the data appears approximately Gaussian we are justified in using this approach.

3. Implementing unpaired t-test

In this question the goal is to create a function called `t_test_function()` which implements an unpaired Student's t-test, in order to test for a difference of population means between two unpaired samples from two distributions. Your function will play a similar role to the following standard R function:

```
t.test(body_mass_g~species, data=peng_AC,var.equal = TRUE)
```

Begin by creating a data frame called "`peng_AC`" which is a subset of the Palmer penguins data set consisting of all those penguins which belong to either the "`Adelie`" or the "`Chinstrap`" species of penguins.

```
library(palmerpenguins)
peng_AC<-penguins %>%
  drop_na(species,body_mass_g) %>%
  filter(species !="Gentoo")
head(peng_AC %>% select(species, flipper_length_mm, body_mass_g), 5)
```

```
## # A tibble: 5 x 3
##   species flipper_length_mm body_mass_g
##   <fct>               <int>       <int>
## 1 Adelie                181        3750
## 2 Adelie                186        3800
## 3 Adelie                195        3250
```

```
## 4 Adelie                193        3450
## 5 Adelie                190        3650
```

(Q1)

First, try to understand what the following piece of code does.

```
val_col <- "body_mass_g"
group_col <- "species"
data <- peng_AC

data_new <- data %>%
  # rename the columns; note that you can not drop the "!!" (why?)
  rename(group=(!!group_col),val=(!!val_col))%>%
  group_by(group) %>%
  drop_na(val) %>%
  summarise(mn=mean(val))

data_new
```

```
## # A tibble: 2 x 2
##   group        mn
##   <fct>     <dbl>
## 1 Adelie    3701.
## 2 Chinstrap 3733.
```

```
data_new$mn[2]
```

```
## [1] 3733.088
```

Now, let's create the function `t_test_function()`. Your function should take in the following arguments:

1. "`data`" - A data frame argument,
2. "`val_col`"- A string argument. This argument is for the column name for a continuous variable (e.g., the body mass column `body_mass_g`),
3. "`group_col`" - A string argument. This argument is for the column name for a binary variable (e.g., the species column `species`).

The function should carry out an unpaired Student's t test based on the value of the continuous variable with column name "`val_col`":

1. The function should begin by partitioning the sample into two groups based on the value of the binary variable named "`group_col`". For example, suppose that the column "`group_col`" contains entries "Adelie" and "Chinstrap", then you should partition the sample into two groups corresponding to "Adelie" and "Chinstrap" respectively. You can use the `group_by()` function.
2. Your function should then compute the sample mean, sample variance and sample size for each of these two groups, based upon the variable within the column named "`var_col`".
3. Your function should compute a test statistic for the Student's unpaired t-test (Lecture 20). In addition, the function should compute the corresponding p-value. Finally, your function should compute an estimate for the effect size.
4. Your function should return a data frame containing the test statistic, p-value and effect size.

Answer

```
t_test_function <- function(data, val_col, group_col, val_equal=TRUE){
stats <- data %>%
  # rename the columns; note that you can not drop !! (why?)
  rename(group=(!!group_col),val=(!!val_col))%>%
```

```
  group_by(group) %>%
  drop_na(val) %>%
  summarise(mn=mean(val), vr=var(val), n=n())

  pooled_sd <- sqrt(((stats$n[1]-1)*stats$vr[1]+(stats$n[2]-1)*stats$vr[2])/
                      (stats$n[1]+stats$n[2]-2))

  if (val_equal){ # unpaired student t-test
    # test statistic
    t_stat<-(stats$mn[1]-stats$mn[2])/
      (pooled_sd*sqrt(1/stats$n[1]+1/stats$n[2]))
    # degree of freedom
    dof<-stats$n[1]+stats$n[2]-2

  } else { # we can also implement Welch's t-test as follows
    # test statistic
    t_stat=(stats$mn[1]-stats$mn[2])/
      sqrt(stats$vr[1]/stats$n[1]+stats$vr[2]/stats$n[2])
    # degree of freedom
    dof=(stats$vr[1]/stats$n[1]+stats$vr[2]/stats$n[2])^2/(
      (stats$vr[1]/stats$n[1])^2/(stats$n[1]-1)+
        (stats$vr[2]/stats$n[2])^2/(stats$n[2]-1)
    )
  }

  # p-value
  p_val<- 2*(1-pt(abs(t_stat),df=dof))

  # effect size
  effect_size <- (stats$mn[1]-stats$mn[2])/(pooled_sd)

  return(data.frame(t_stat=t_stat, effect_size=effect_size, p_val=p_val))
}
```

```
t_test_function(data=peng_AC,val_col="body_mass_g",group_col="species")
```

```
##       t_stat effect_size     p_val
## 1 -0.5080869 -0.07420226 0.6119085
```

(Q2)

As an additional challenge you can modify your function so that it takes a fourth argument called "`var_equal`" which takes a Boolean value. If the input of "`var_equal`" is the Boolean "TRUE" your function should compute the test statistic and p-value for an unpaired Student's t-test. If, on the other hand, the input of "`var_equal`" is the Boolean "FALSE" your function should compute the test statistic and p-value for Welch's t-test. Your function should have the following output:

Answer

```
t_test_function(data=peng_AC,val_col="body_mass_g",group_col="species", val_equal=FALSE)
```

```
##       t_stat effect_size     p_val
## 1 -0.5430902 -0.07420226 0.5878608
```

You can compare the output of your function with R's inbuilt `t.test()` function.

Answer

```
t.test(body_mass_g~species, data=peng_AC,var.equal = FALSE)
```

```
##
##  Welch Two Sample t-test
##
## data:  body_mass_g by species
## t = -0.54309, df = 152.45, p-value = 0.5879
## alternative hypothesis: true difference in means between group Adelie and group Chinstrap is not equa
## 95 percent confidence interval:
##  -150.38481   85.53284
## sample estimates:
##    mean in group Adelie mean in group Chinstrap
##                3700.662                3733.088
```

4. Useful concepts in statistical hypothesis testing

This question is about the basic concepts in statistical hypothesis testing.

(Q1)

Explain the following concepts:

1. Null hypothesis
2. Alternative hypothesis
3. Test statistic
4. Type 1 error
5. Type 2 error
6. The size of a test
7. The power of a test
8. The significance level
9. The p-value
10. Effect size

Answer

1. The null hypothesis is our default position in a statistical hypothesis which typically declares the absence of some interesting phenomenon for example the equality of two statistical parameters.
2. The alternative hypothesis is a statistical hypothesis which contradicts the null hypothesis and typically declares the presence of some interesting phenomenon, often consistent with the research hypothesis a scientist is attempting to prove. For example, a difference in the values of two statistical parameters.
3. A test statistic is some function of the data used within a statistical hypothesis test. The test statistic must have a known distribution under the null hypothesis. In addition, the test statistic should emphasize differences between null and alternative hypothesis.
4. A type 1 error is a rejection of the null hypothesis in favor of the alternative hypothesis when the null hypothesis is true.
5. A type 2 error is a failure to reject the null hypothesis in favor of the alternative hypothesis when the alternative hypothesis holds.
6. The size of the test is the probability of a type 1 error under the null hypothesis.
7. The power of a test is one minus the probability of a type 2 error under an alternative hypothesis. That is, the probability of rejecting the null hypothesis under an alternative hypothesis. Typically this depends upon the particular alternative hypothesis.
8. The significance level is an upper bound on the size of the test. This should be chosen in advance of seeing the data. A typical value is $\alpha = 0.05$.

9. The p-value is probability under the null hypothesis that the test statistic will achieve a value as extreme or more extreme than the value which is actually observed. A very small p-value indicates that the observed data is sufficiently inconsistent with the null hypothesis that the null hypothesis can be reasonably rejected.
10. The effect size is a measure of the magnitude of the observed phenomenon which reflects the extent to which the null hypothesis is false.

(Q2)

(1). Is the p-value the probability that the null hypothesis is true?

(2). If I conduct a statistical hypothesis test, and my p-value exceeds the significance level, do I have good evidence that the null hypothesis is true?

Answer

(1). No. The p-value is the probability, under the null hypothesis, of a test-statistic as extreme or more extreme than the observed numerical value.

(2). No. For example, it might be the case that the null hypothesis is false, but we have either a small sample size or a relatively small effect size.

5. Investigating test size for an unpaired Student's t-test

In this question, we shall investigate the performance of an unpaired Student's t-test from the perspective of test size. Recall a Type 1 error occurs when we reject the null hypothesis when the null hypothesis is true. The size of a test is the probability of a Type 1 error. A key property of valid statistical hypothesis tests with a given significance level is that the size of the test does not exceed the significance level.

Note that we can apply unpaired Student's t-test with significance level alpha on a samples `sample_0`, `sample_1`:

```
t.test(sample_0,sample_1,var.equal = TRUE, conf.level = 1-alpha)
```

We can apply an unpaired Student's t-test and extract the p-value as follows:

```
t.test(sample_0,sample_1,var.equal = TRUE)$p.value
```

Notice that the significance level wasn't supplied as an argument. Is this a problem?

The following code checks the size of an unpaired Student's t-test with a significance level of $\alpha = 0.05$.

```
num_trials<-10000
sample_size<-30
mu_0<-1
mu_1<-1
sigma_0<-3
sigma_1<-3
alpha<-0.05
set.seed(0) # set random seed for reproducibility

single_alpha_test_size_simulation_df <- data.frame(trial=seq(num_trials)) %>%
  # generate random Gaussian samples
  mutate(sample_0=map(.x=trial,.f=~rnorm(n=sample_size,mean=mu_0,sd=sigma_0)),
         sample_1=map(.x=trial,.f=~rnorm(n=sample_size,mean=mu_1,sd=sigma_1))) %>%
  # generate p values
  mutate(p_value=pmap(.l=list(trial,sample_0,sample_1),
                      .f=~t.test(..2,..3,var.equal = TRUE)$p.value))%>%
```

```
  # type I error
  mutate(type_1_error=p_value<alpha)

single_alpha_test_size_simulation_df %>%
  pull(type_1_error) %>%
  mean() # estimate of coverage probability
```

```
## [1] 0.0502
```

Check that you understand the above code.

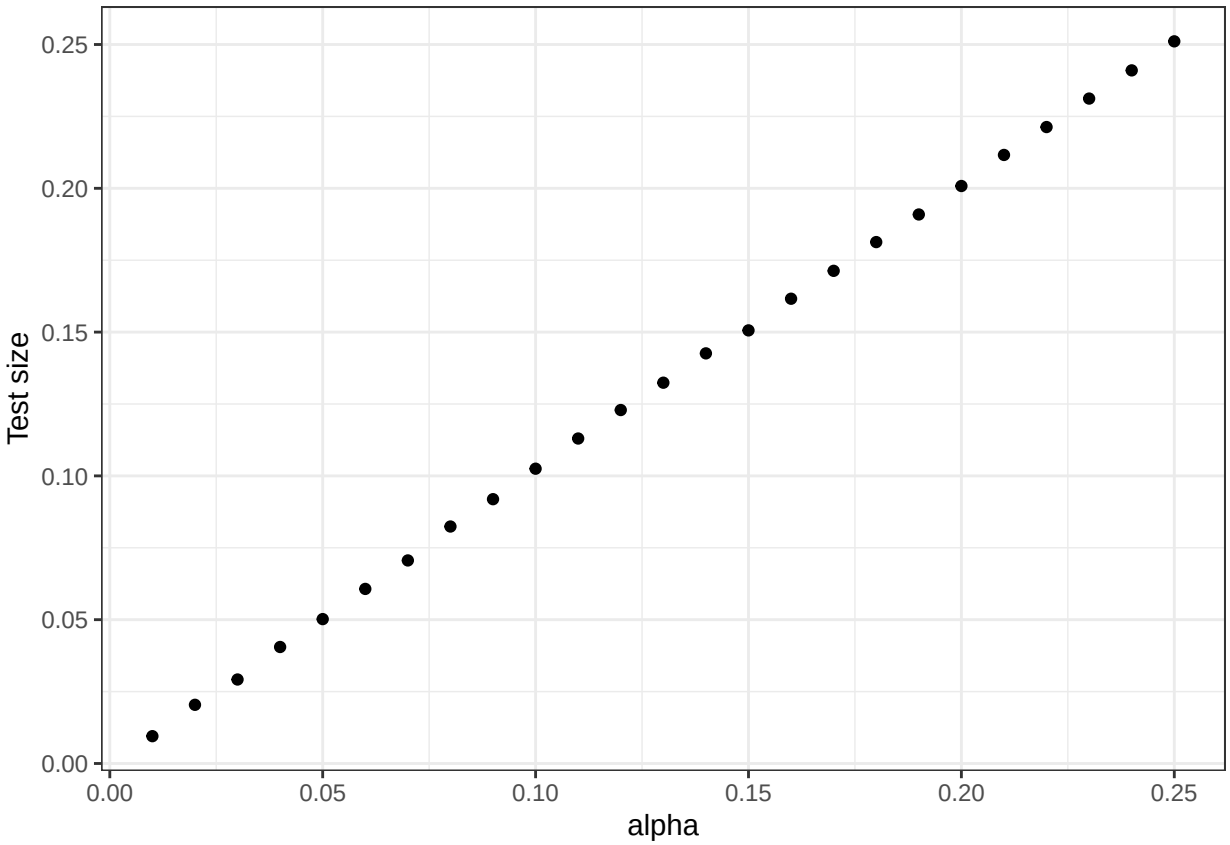
(Q1)

Modify the above code to explore how the size of the test varies as a function of the significance level α. You might want to use visualization.

Answer

```
num_trials<-10000
sample_size<-30
mu_0<-1
mu_1<-1
sigma_0<-3
sigma_1<-3
set.seed(0) # set random seed for reproducibility

single_alpha_test_size_simulation_df <- data.frame(trial=seq(num_trials)) %>%
  # generate random Gaussian samples
  mutate(sample_0=map(.x=trial,.f=~rnorm(n=sample_size,mean=mu_0,sd=sigma_0)),
         sample_1=map(.x=trial,.f=~rnorm(n=sample_size,mean=mu_1,sd=sigma_1))) %>%
  # generate p values
  mutate(p_value=pmap(.l=list(trial,sample_0,sample_1),
                      .f=~t.test(..2,..3,var.equal = TRUE)$p.value))


alpha_list = seq(0.01, 0.25, 0.01)

compute_test_size <- function(alpha){
  # type I error
  type_1_error = single_alpha_test_size_simulation_df$p_value<alpha
  return (mean(type_1_error)) # estimate of coverage probability
}
```

```
multiple_alpha_test_size_simulation_df <- data.frame(alpha=alpha_list) %>%
  mutate(test_size = map_dbl(alpha, compute_test_size))

multiple_alpha_test_size_simulation_df %>% ggplot(aes(x=alpha, y=test_size)) +
  geom_point() + ylab('Test size') + theme_bw()
```



In this case, the test size is equal to the significance level.

6. The statistical power of an unpaired t-test

In this question, we shall investigate the performance of an unpaired Student's t-test from the perspective of statistical power. Recall that the statistical power of a test is the probability of correctly rejecting the null hypothesis when an alternative hypothesis holds.

Consider a setting in which we have two samples i.i.d with Gaussian distribution. The first sample consists of n_0 observations with a population mean μ_0 and population variance σ_0^2. The second sample consists of n_1 observations with a population mean μ_1 and population variance σ_1^2.

The following code checks the statistical power of an unpaired Student's t-test in sample sizes $n_0 = n_1 = 30$, $\mu_0 = 3$, $\mu_1 = 4$, $\sigma_0 = \sigma_1 = 1$ and with a significance level of $\alpha = 0.05$.

```
num_trials<-10000

n_0<-30
n_1<-30
mu_0<-3
mu_1<-4
sigma_0<-2
sigma_1<-2

alpha<-0.05
set.seed(0) # set random seed for reproducibility
```

```
data.frame(trial=seq(num_trials)) %>%
  # generate random Gaussian samples
  mutate(sample_0 = map(.x=trial,.f =~ rnorm(n=n_0,mean=mu_0,sd=sigma_0)),
         sample_1 = map(.x=trial,.f =~ rnorm(n=n_1,mean=mu_1,sd=sigma_1))) %>%
  # for each sample, generate p value; check examples of pmap() with ?map
  mutate(p_value=pmap(.l = list(trial,sample_0,sample_1),
                      .f =~ t.test(..2, ..3, var.equal = TRUE)$p.value)) %>%
  # estimate of coverage probability
  mutate(reject_null = p_value<alpha ) %>%
  # extract a column
  pull(reject_null) %>%
  # compute probability
  mean()
```

```
## [1] 0.4862
```

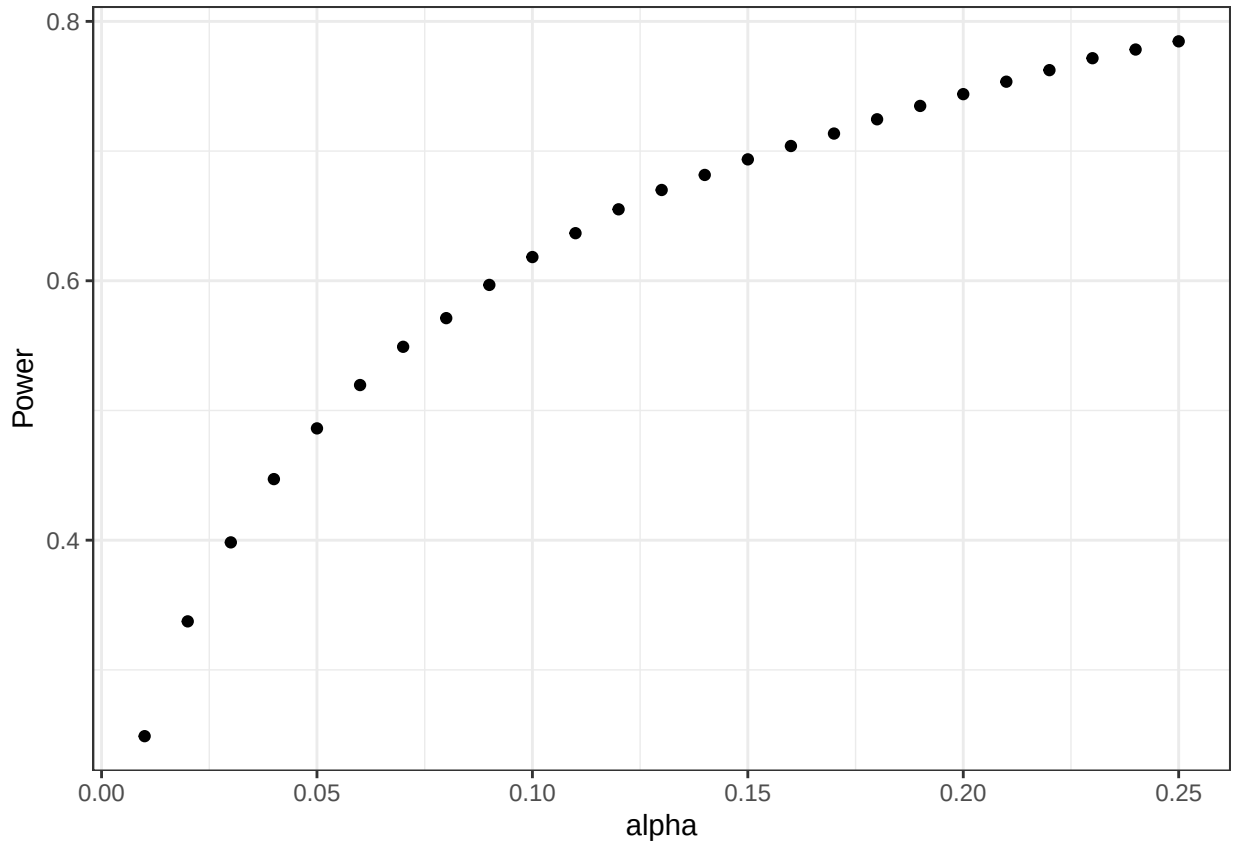
(Q1)

Conduct a simulation study to explore how the statistical power varies as a function of the significance level.

Answer

```
num_trials<-10000

n_0<-30
n_1<-30
mu_0<-3
mu_1<-4
sigma_0<-2
sigma_1<-2

alpha<-0.05
set.seed(0) # set random seed for reproducibility

single_alpha_power_df <- data.frame(trial=seq(num_trials)) %>%
  # generate random Gaussian samples
  mutate(sample_0 = map(.x=trial,.f =~ rnorm(n=n_0,mean=mu_0,sd=sigma_0)),
         sample_1 = map(.x=trial,.f =~ rnorm(n=n_1,mean=mu_1,sd=sigma_1))) %>%
  # for each sample, generate p value; check examples of pmap() with ?map
  mutate(p_value=pmap(.l = list(trial,sample_0,sample_1),
                      .f =~ t.test(..2, ..3, var.equal = TRUE)$p.value))

compute_power <- function(alpha){
  reject_null <- single_alpha_power_df$p_value < alpha
  return (mean(reject_null))
}
```

```
multiple_alpha_power_df <- data.frame(alpha=seq(0.01, 0.25, 0.01)) %>%
  mutate(power= map_dbl(alpha, compute_power))

multiple_alpha_power_df %>% ggplot(aes(x=alpha, y=power)) +
  geom_point() + ylab('Power') + theme_bw()
```



Note that the power depends on the parameters μ_0 and μ_1.

(Q2)

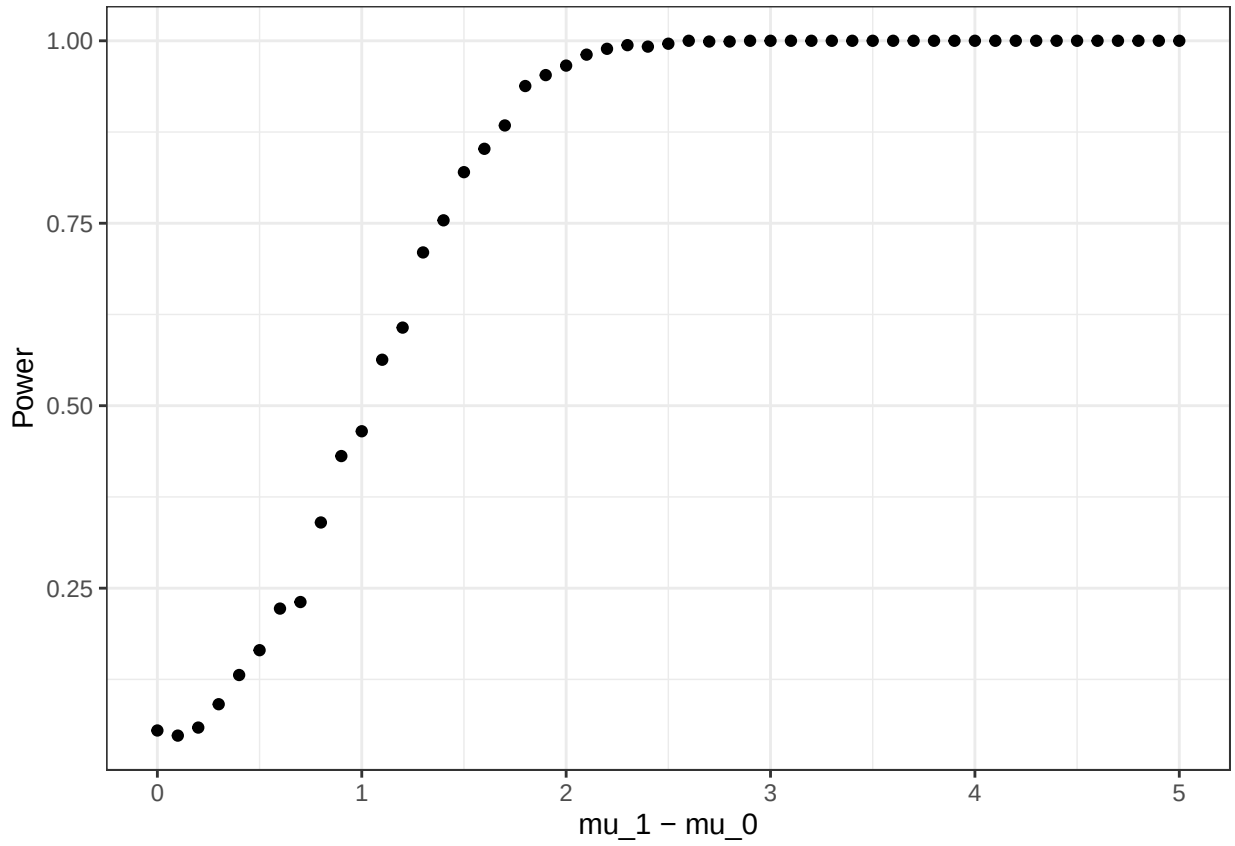
Conduct a simulation study to explore how the statistical power varies as a function of the difference in means $\mu_1 - \mu_0$.

Answer

```
num_trials<-1000

n_0<-30
n_1<-30
mu_0<-3
# mu_1<-4
sigma_0<-2
sigma_1<-2

alpha<-0.05
set.seed(0) # set random seed for reproducibility


compute_power_from_mu1 <- function(mu_1){

  single_mu1_power_df <- data.frame(trial=seq(num_trials)) %>%
    # generate random Gaussian samples
    mutate(sample_0 = map(.x=trial,.f =~ rnorm(n=n_0,mean=mu_0,sd=sigma_0)),
```

```
          sample_1 = map(.x=trial,.f =~ rnorm(n=n_1,mean=mu_1,sd=sigma_1))) %>%
    # for each sample, generate p value; check examples of pmap() with ?map
    mutate(p_value=pmap(.l = list(trial,sample_0,sample_1),
                  .f =~ t.test(..2, ..3, var.equal = TRUE)$p.value))

  reject_null <- single_mu1_power_df$p_value < alpha
  return (mean(reject_null))
}

multiple_mu1_power_df <- data.frame(mu_1=seq(3.0, 8.0, 0.1)) %>%
  mutate(power= map_dbl(mu_1, compute_power_from_mu1))
```

```
multiple_mu1_power_df %>% ggplot(aes(x=mu_1-mu_0, y=power)) +
  geom_point() + ylab('Power') + theme_bw()
```



(Q3)

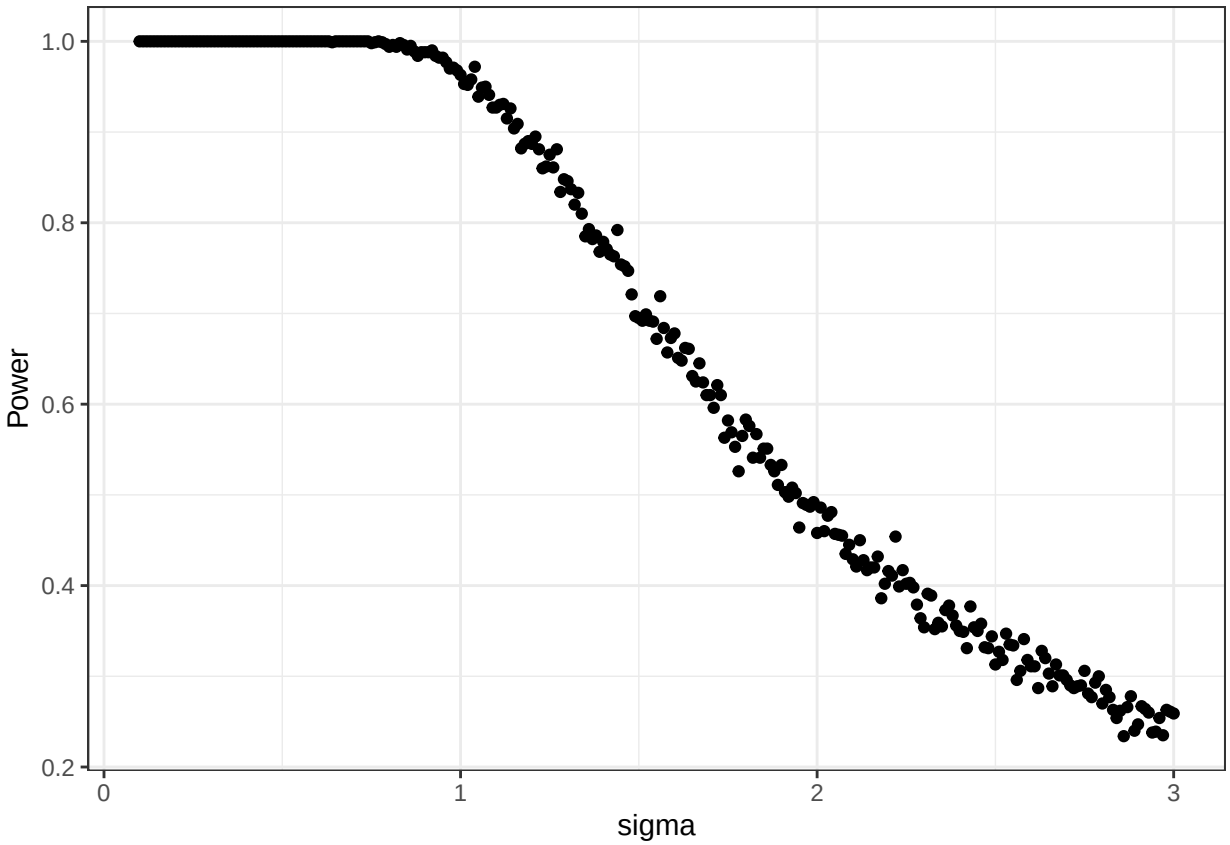
Conduct a simulation study to explore how the statistical power varies as a function of the population standard deviation $\sigma = \sigma_0 = \sigma_1$.

Answer

```
num_trials<-1000

n_0<-30
n_1<-30
mu_0<-3
```

```r
mu_1<-4
#sigma_0<-2
#sigma_1<-2

alpha<-0.05
set.seed(0) # set random seed for reproducibility


compute_power_from_sigma <- function(sigma){

  sigma_0 <- sigma
  sigma_1 <- sigma

  single_sigma_power_df <- data.frame(trial=seq(num_trials)) %>%
    # generate random Gaussian samples
    mutate(sample_0 = map(.x=trial,.f =~ rnorm(n=n_0,mean=mu_0,sd=sigma_0)),
           sample_1 = map(.x=trial,.f =~ rnorm(n=n_1,mean=mu_1,sd=sigma_1))) %>%
    # for each sample, generate p value; check examples of pmap() with ?map
    mutate(p_value=pmap(.l = list(trial,sample_0,sample_1),
                  .f =~ t.test(..2, ..3, var.equal = TRUE)$p.value))

  reject_null <- single_sigma_power_df$p_value < alpha
  return (mean(reject_null))
}

multiple_sigma_power_df <- data.frame(sigma=seq(0.1, 3.0, 0.01)) %>%
  mutate(power= map_dbl(sigma, compute_power_from_sigma))
```

```r
multiple_sigma_power_df %>% ggplot(aes(x=sigma, y=power)) +
  geom_point() + ylab('Power') + theme_bw()
```



(Q4)

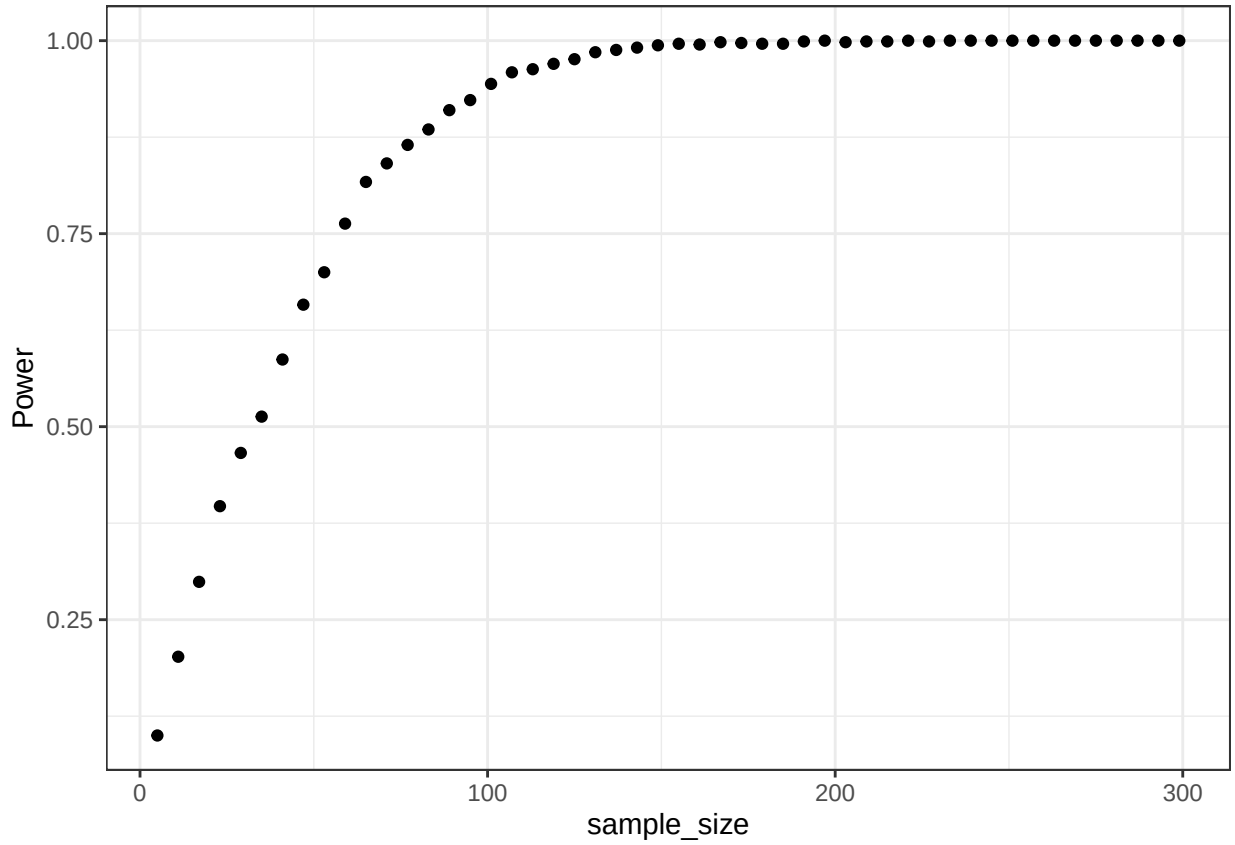
Conduct a simulation study to explore how the statistical power varies as a function of the sample size $n = n_0 = n_1$

Answer

```
num_trials<-1000

#n_0<-30
#n_1<-30
mu_0<-3
mu_1<-4
sigma_0<-2
sigma_1<-2

alpha<-0.05
set.seed(0) # set random seed for reproducibility


compute_power_from_sz <- function(sz){

  n_0 <- sz
  n_1 <- sz

  single_sigma_power_df <- data.frame(trial=seq(num_trials)) %>%
    # generate random Gaussian samples
```

```
    mutate(sample_0 = map(.x=trial,.f =~ rnorm(n=n_0,mean=mu_0,sd=sigma_0)),
           sample_1 = map(.x=trial,.f =~ rnorm(n=n_1,mean=mu_1,sd=sigma_1))) %>%
    # for each sample, generate p value; check examples of pmap() with ?map
    mutate(p_value=pmap(.l = list(trial,sample_0,sample_1),
                  .f =~ t.test(..2, ..3, var.equal = TRUE)$p.value))

  reject_null <- single_sigma_power_df$p_value < alpha
  return (mean(reject_null))
}

multiple_sz_power_df <- data.frame(sample_size=seq(5, 300, 6)) %>%
  mutate(power= map_dbl(sample_size, compute_power_from_sz))
```

```
multiple_sz_power_df %>% ggplot(aes(x=sample_size, y=power)) +
  geom_point() + ylab('Power') + theme_bw()
```



7. (*Optional) Comparing the paired and unpaired t-tests on paired data

The aim of this question is to explore the benefits of using a paired test when a natural pairing is available. Consider a situation in which we have two i.i.d. samples $X_1, \cdots, X_n$ and $Y_1, \cdots, Y_n$.

Suppose that $X_1, \cdots, X_n \sim \mathcal{N}(\mu_X, \sigma_X^2)$ and for each $i = 1, \cdots, n$, we have $Y_i = X_i + Z_i$ where $Z_1, \cdots, Z_n \sim \mathcal{N}(\mu_Z, \sigma_Z^2)$ are independent and identically distributed random variables. It follows that $Y_1, \cdots, Y_n \sim \mathcal{N}(\mu_Y, \sigma_Y^2)$ are independent and identically distributed with $\mu_Y = \mu_X + \mu_Z$ and $\sigma_Y^2 = \sigma_X^2 + \sigma_Z^2$.

In this situation we only observe the two samples $X_1, \cdots, X_n$ and $Y_1, \cdots, Y_n$. We are interested in performing a statistical hypothesis test to see if $\mu_X \neq \mu_Y$. We have two options here. We could either (1) use the pairing and apply a paired test or (2) ignore the pairing and use an unpaired test. In the console run `?t.test()` to see how to carry out an unpaired and a pared test within R.

(Q1)

Conduct a simulation study to explore the statistical power of these two approaches. You may want to consider a setting in which $n = 30, \mu_X = 10, \sigma_X = 5, \mu_Z = 1 and \sigma_Z = 1$. Consider a range of different significance levels α

Answer

```
num_trials<-10000

n_0<-30
n_1<-30
mu_0<-3
mu_Z<-1 # mean of Z
sigma_0<-2
sigma_Z<-2

alpha<-0.05
set.seed(0) # set random seed for reproducibility

single_alpha_power_df <- data.frame(trial=seq(num_trials)) %>%
  # generate random Gaussian samples
  mutate(sample_0 = map(.x=trial,.f =~ rnorm(n=n_0,mean=mu_0,sd=sigma_0)) ) %>%
  mutate(sample_Z = map(.x=trial,.f =~ rnorm(n=n_0,mean=mu_Z,sd=sigma_Z)) ) %>%
  mutate(sample_1 = pmap(.l=list(sample_0, sample_Z),.f =~ (..1+..2))) %>%
  # for each sample, generate p value; check examples of pmap() with ?map
  mutate(p_value_paired=pmap(.l = list(trial,sample_0,sample_1),
                      .f =~ t.test(..2, ..3, paired=TRUE)$p.value)) %>%
  mutate(p_value_unpaired=pmap(.l = list(trial,sample_0,sample_1),
                      .f =~ t.test(..2, ..3, paired=FALSE)$p.value))

compute_power_paired <- function(alpha){
  reject_null <- single_alpha_power_df$p_value_paired < alpha
  return (mean(reject_null))
}

compute_power_unpaired <- function(alpha){
  reject_null <- single_alpha_power_df$p_value_unpaired < alpha
  return (mean(reject_null))
}
```

```
multiple_alpha_power_df <- data.frame(alpha=seq(0.01, 0.25, 0.01)) %>%
  mutate(power_paired= map_dbl(alpha, compute_power_paired)) %>%
  mutate(power_unpaired= map_dbl(alpha, compute_power_unpaired))

colors <- c('Paired'='red', 'Unpaired'='blue')

multiple_alpha_power_df %>% ggplot() +
  geom_point(aes(x=alpha, y=power_paired, color='Paired')) +
  geom_point(aes(x=alpha, y=power_unpaired, color='Unpaired')) + ylab('Power') +
```

```
theme_bw() + scale_color_manual(name = "method", values=colors)
```