Practical 2.4

CHEN

2024-03-05

Task 1. Calculate a one-sample test power with simulation in R

Let us assume that the heights of the adult men in a country follow a normal distribution with average 175cm and standard derivation 10cm.

```
# Set parameters
population_mean <- 175
population_sd <- 10
milk_effect <- 3
student_sample_size <- 10

# Simulation and t test
p_values <- numeric(100000)

for (i in 1:100000) {
  student <- rnorm(student_sample_size, mean = population_mean + milk_effect, sd = population_sd)
  t_test <- t.test(student, mu = population_mean)
  p_values[i] <- t_test$p.value
}
```

Question 1.1

If you set a cutoff of 0.05, what is the percentage of simulations that you get the p value larger than 0.05? What does this number mean? What is the power?

```
# Calculate percentage of simulations with p-value > 0.05
percentage_above_005 <- length(p_values[p_values > 0.05])/length(p_values)
percentage_above_005
```

```
## [1] 0.86373
```

When the p value of one-sided t test is greater than 0.05, it means that the college simulation cannot be considered to have a differences to the population in height. That is "accept H0". Therefore, the percentage of the p_values > 0.05 refers to the probability of getting a result showing the college students' height is the same as the population's while actually there is a difference. In other words, it is the probability of getting type two error where H0 is accepted but in fact H1 is true.

```r
# Power is the complement of Type II error rate
average_above_005 <- mean(p_values > 0.05)
power <- 1 - average_above_005

power
```

```
## [1] 0.13627
```

Note: Why does the *percentage_above_005* have the same value as *average_above_005*?

percentage_above_005 refers to "how many times, out of 100,000 simulations, the p-value of the t-test was greater than 0.05". *average_above_005* represents the average number of times per simulation that the p-value was greater than 0.05.

In each simulation, you are conducting only one t-test, so each simulation's p-value is either greater than 0.05 or less than or equal to 0.05. Therefore, whether you are calculating how many times the p-value was greater than 0.05 across simulations or computing the average number of times per simulation that the p-value was greater than 0.05, the results are the same. In other words, both calculation methods are statistically equivalent since each simulation yields only one p-value, which is either greater than 0.05 or not. Hence, whether you are counting how many times the p-value exceeds 0.05 across simulations or calculating the average count of times per simulation, the results are identical. This explains why these two values are the same.

Question 1.2

If you measure 50 students instead of 10 students, then does the power change?

```r
# Change number of students to 50
new_student_sample_size <- 50

# Repeat simulations
new_p_values <- numeric(100000)

for (i in 1:100000) {
  new_student <- rnorm(new_student_sample_size, mean = population_mean + milk_effect, sd = population_s
  new_t_test <- t.test(new_student, mu = population_mean)
  new_p_values[i] <- new_t_test$p.value
}

# Calculate power
new_average_above_005 <- mean(new_p_values > 0.05)
new_power <- 1 - new_average_above_005

new_power
```

```
## [1] 0.5464
```

Question 1.3

The cheating way is to use the command power.t.test. Try it out to calculate the power number with different Ns (5, 10, 15, 20. . . .100). Try to plot the power versus N.

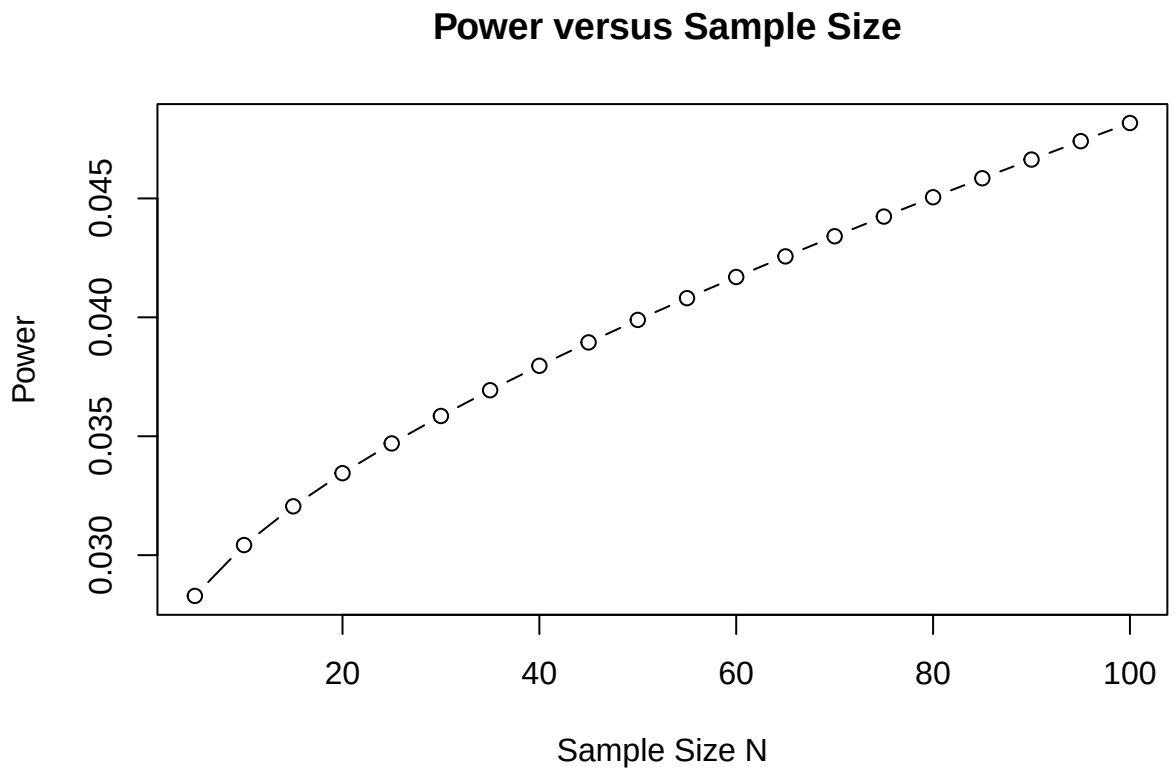
```r
# Define a sequence of sample sizes
size_values <- seq(5, 100, by = 5)

# Initialize vector to store power values
power_values <- numeric(length(size_values))

# Calculate power for different sample sizes
for (i in seq_along(size_values)) { # Using seq_along () is safer than using 1:length (x) because if x
  power_values[i] <- power.t.test(n = size_values[i], delta = milk_effect/population_sd, sd = populatio
}
```

power.t.test(n = NULL, delta = NULL, sd = 1, sig.level = 0.05, power = NULL, type = c("two.sample", "one.sample", "paired"), alternative = c("two.sided", "one.sided"), strict = FALSE, tol = .Machine$double.eps^0.25) n: number of observations (per group) delta: true difference in means sd: standard deviation sig.level: significance level (Type I error probability) power: power of test (1 minus Type II error probability) type: string specifying the type of t test. Can be abbreviated. alternative: one- or two-sided test. Can be abbreviated. strict: use strict interpretation in two-sided case tol:numerical tolerance used in root finding, the default providing (at least) four significant digits.

```r
# Plot power versus sample size
plot(size_values, power_values, type = "b", xlab = "Sample Size N", ylab = "Power", main = " Power versu
```



Task 2. Choose the right sample size

A company is developing a diet pill, which may help people lose weight. In the animal model, the drug could lead to 10% of weight loss. Now the company recruits 20 volunteers and separate them into two groups: placebo group and drug group to perform a trial. Let us assume that the weight in the normal population is 130 pound with standard derivation of 30.

```
# Set parameters
population_mean_weight <- 130
population_sd_weight <- 30
drug_effect_percentage <- 0.1
drug_effect <- drug_effect_percentage * population_mean_weight

# Function to simulate weight data for placebo and drug groups
weight_simulation <- function(size, effect) {
  placebo_group_weight <- rnorm(size, mean = population_mean_weight, sd = population_sd_weight)
  drug_group_weight <- rnorm(size, mean = population_mean_weight*(1-effect), sd = population_sd_weight)
  return(list(placebo = placebo_group_weight,
              drug = drug_group_weight))
}

# Function to perform t-test and calculate p-value
perform_t_test <- function(placebo, drug) {
  t_result <- t.test(placebo, drug)
  return(t_result$p.value)
}
```

Question 2.1

If the drug is indeed effective as showed on the animal model, then what is the probability that they do not see a significant effect of the drug (p-value cutoff = 0.05)? Please perform a simulation as you did before to give an answer.

```
replication <- 100000
significant_results <- replicate(replication, {
  weights <- weight_simulation(10, drug_effect_percentage)
  p_value <- perform_t_test(weights$placebo, weights$drug)
  p_value >= 0.05
}) # Stored as boolean variables, which can be computed as 0 or 1

probability_not_significant <- mean(significant_results)
```

The answer to question 2.1:

```
## Probability of not seeing a significant effect: 0.85447
```

Question 2.2

Do you think the power is good enough? If the company truly believes the effect of the drug and want to be sure that they will not largely miss the effect in the trial (type II error rate < 0.2), then how many volunteers do they need to recruit?

```r
desired_power <- 1 - 0.2
required_sample_size1 <- 1

# Use while-loop to get the sample size
while (TRUE) {
  drug_power <- power.t.test(n = required_sample_size1, delta = drug_effect, sd = population_sd_weight,
  # Use "one.sided" because the effect of drug is useful only when it can improve the symptom.
  if (drug_power >= desired_power) {
    break
  }
  required_sample_size1 <- required_sample_size1 + 1
}
```

Another method to get the sample size:

```r
required_sample_size2 <- power.t.test(n = NULL, delta = drug_effect, sd = population_sd_weight, sig.leve

required_sample_size2 <- ceiling(required_sample_size2)
```

This two method will receive the same value.

```r
required_sample_size1 == required_sample_size2
```

```
## [1] TRUE
```

The answer to question 2.2:

```
## Required sample size for power > 0.8: 67
```

Question 2.3

If the company changes their strategy, asking all the volunteers to take the pills and measuring their weights before and afterward, how many volunteers do they need?

```r
new_required_sample_size1 <- 1

while (TRUE) {
  new_drug_power <- power.t.test(n = new_required_sample_size1, delta = drug_effect, sd = population_sd
  # type = "paired" because the different conditions are conducted on the same sample
  if (new_drug_power >= desired_power) {
    break
  }
  new_required_sample_size1 <- new_required_sample_size1 + 1
}
```

Another method to get the sample size:

```r
new_required_sample_size2 <- power.t.test(n = NULL, delta = drug_effect, sd = population_sd_weight, sig

new_required_sample_size2 <- ceiling(new_required_sample_size2)
```

The answer to question 2.3:

```
## Number of volunteers needed for a paired design: 35
```

Question 2.4

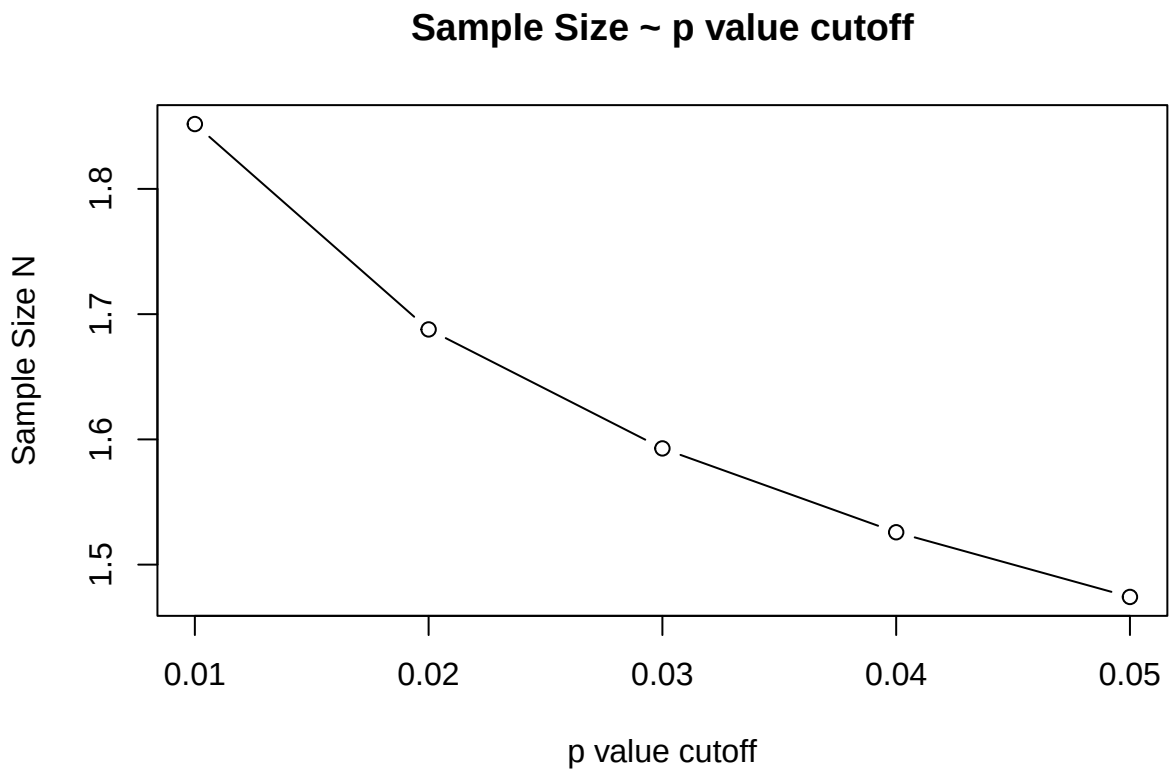
Manipulate the p-value cutoff or power to see the change of the needed sample size. You could also plot them out. The output should be similar to these.

```
# Set up independent variables and vectors to store the changed sample size.
p_value_cutoff <- seq(0.01, 0.05, 0.01)
value_of_power <- seq(0.5, 0.9, 0.1) # According to the figure in the instruction
sample_size_N1 <- numeric(length(p_value_cutoff))
sample_size_N2 <- numeric(length(value_of_power))

# Fill the vectors with sample size values.
for (i in seq_along(p_value_cutoff)) {
  sample_size_N1[i] <- power.t.test(n = NULL, delta = drug_effect*population_mean_weight, sd = populati
}

for (j in seq_along(value_of_power)) {
  sample_size_N2[j] <- power.t.test(n = NULL, delta = drug_effect*population_mean_weight, sd = populati
}
```

```
# Plot the results of needed sample size.
plot(p_value_cutoff, sample_size_N1, type = "b",
     xlab = "p value cutoff", ylab = "Sample Size N", main = "Sample Size ~ p value cutoff")
```



```r
plot(value_of_power, sample_size_N2, type = "b",
     xlab = "Power", ylab = "Sample Size N", main = "Sample Size ~ Power")
```