Lab 9: Power analysis

STAT218

The objective of this lab is to explore statistical power. Your goals are to:

1. develop an understanding of population and procedural factors that influence power
2. learn to conduct post-hoc power analyses to estimate power for a test performed
3. learn to conduct study design power analyses for sample size determination

We won’t use much data for this lab, but the cloud seeding and body temperature data will be referenced in passing. Run these lines to load the datasets:

library(Sleuth3)  
library(tidyverse)  
cloud <- case0301  
load('data/temps.RData')

In the first part of the lab, we’ll aim to simply understand type II errors a little better. This will be achieved through simulation. In the second part of the lab, we’ll translate some of the insights from the first part into performing basic power analyses.

## Simulating type II errors

type2sim <- function(delta, n, sd, alpha, nsim = 1000){  
 # simulate nsim tests by...  
 sim.pvals <- sapply(1:nsim, function(i){  
 # draw sample with true group difference of delta  
 samp <- data.frame(variable = c(rnorm(n, mean = 0, sd = sd), rnorm(n, mean = delta, sd = sd)),  
 group = rep(1:2, each = n))  
 # perform test and compute p value  
 pval <- t.test(variable ~ group, data = samp, mu = 0, alternative = 'two.sided')$p.value  
 return(pval)  
 })  
 # compute proportion of tests that failed to reject (made a type ii error)  
 err.rate <- mean(sim.pvals > alpha)  
 return(err.rate)  
}  
  
save(type2sim, file = 'type2sim.RData')

Your first task in this lab is to explore the impact of several factors on type II error rates for two-sample inference on means using simulation. These factors are:

* true difference in means ()
* population variability ()
* sample size per group ()
* significance level ()

We will base our simulations on the two-sided test for the cloud dataset.

# two-sided test  
cloud.test.out <- t.test(Rainfall ~ Treatment, data = cloud, mu = 0, alternative = 'two.sided')  
  
# estimated difference in means  
diff(cloud.test.out$estimate)

mean in group Unseeded   
 -277.3962

# population SDs (don't worry about syntax here)  
cloud |> group\_by(Treatment) |> summarize(sd(Rainfall))

# A tibble: 2 × 2  
 Treatment `sd(Rainfall)`  
 <fct> <dbl>  
1 Seeded 651.  
2 Unseeded 278.

The following function will simulate 1000 datasets assuming:

* a true difference in means of delta
* a sample size per group of n
* a population standard deviation per group of sd

For each dataset, a two-sided test is computed at level alpha. The function returns the proportion of tests that failed to reject.

# load simulation function  
load('type2sim.RData')  
  
# simulation-based estimate of type 2 error  
type2sim(delta = 277, n = 26, sd = 650, alpha = 0.05)

[1] 0.676

This approximates the type II error rate for the test.

|  |
| --- |
| Your turn |
| Your objective is to explore the impact of population and test conditions on type II error rates. You’ll do this by changing the arguments of the type2sim() function.  Carry out each change in turn and record the results. Repeat your runs a few times, and try a few different magnitudes each time.  # store 'baseline' type ii error rate estimated.error <- type2sim(delta = 277, n = 26, sd = 650, alpha = 0.05)  # increase delta type2sim(delta = ..., n = 26, sd = 650, alpha = 0.05) - estimated.error  # decrease delta type2sim(delta = ..., n = 26, sd = 650, alpha = 0.05) - estimated.error  # increase sd type2sim(delta = 277, n = 26, sd = ..., alpha = 0.05) - estimated.error  # decrease sd type2sim(delta = 277, n = 26, sd = ..., alpha = 0.05) - estimated.error  # increase sample size type2sim(delta = 277, n = ..., sd = 650, alpha = 0.05) - estimated.error  # decrease sample size type2sim(delta = 277, n = ..., sd = 650, alpha = 0.05) - estimated.error  # increase significance level type2sim(delta = 277, n = 26, sd = 650, alpha = ...) - estimated.error  # decrease significance level type2sim(delta = 277, n = 26, sd = 650, alpha = ...) - estimated.error |

## Power analyses

The statistical power of a test is the “complement” of a type II error rate:

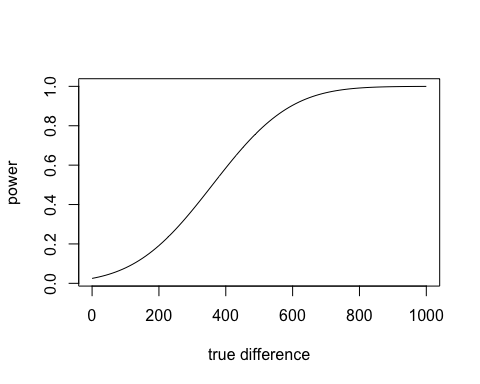
It depends on all of the same factors you explored above that affect type II error rates. Perhaps most importantly, the power of a test is specific to an alternative. In the case of two-sample inference, we can think of power as a function of the true difference.

In R, theory-based power calculations for common tests are implemented in a user-friendly way. An estimate of the power for the cloud seeding test can be calculated as follows:

# minimal arguments  
power.t.test(delta = 277, n = 26, sd = 650)  
  
# verbose  
power.t.test(delta = 277, n = 26, sd = 650, sig.level = 0.05, type = 'two.sample', alternative = 'two.sided')

It is common to visualize *power curves* that show statistical power as a function of the alternative value of the parameter of interest, keeping all other population and test attributes fixed. For example, again using conditions similar to the cloud seeding test:

# plot power curve for cloud seeding test   
curve(power.t.test(delta = x, n = 26, sd = 650)$power,   
 from = 0, to = 1000, xlab = 'true difference', ylab = 'power')



|  |
| --- |
| Your turn |
| Experiment with the test conditions and examine their impact on the power curve. Try:   1. increasing/decreasing sample size 2. increasing/decreasing population standard deviation 3. increasing/decreasing significance level   # plot power curve for cloud seeding test  curve(power.t.test(delta = x, n = 26, sd = 650, sig.level = 0.05)$power,   from = 0, to = 1000, xlab = 'true difference', ylab = 'power') |

### Post hoc power analysis

What we’ve done above in estimating the power of a test we’ve already performed by assuming population conditions based on estimates and fixing test conditions is an example of a *post-hoc* power analysis. The goal of a post-hoc analysis is to estimate the power of a test already performed. This is usually only done if the test produces a negative result (*i.e.*, fails to reject ).

Let’s consider a different example and have you carry out your own post-hoc analysis. The following is a test of whether body temperature differs by sex:

# summary stats (don't worry about syntax here)  
temps |> group\_by(sex) |> summarize(mean = mean(body.temp), sd = sd(body.temp))

# A tibble: 2 × 3  
 sex mean sd  
 <fct> <dbl> <dbl>  
1 female 98.7 0.993  
2 male 98.2 0.788

# two-sided test  
t.test(body.temp ~ sex, data = temps, mu = 0, alternative = 'two.sided')

Welch Two Sample t-test  
  
data: body.temp by sex  
t = 1.7118, df = 34.329, p-value = 0.09595  
alternative hypothesis: true difference in means between group female and group male is not equal to 0  
95 percent confidence interval:  
 -0.09204497 1.07783444  
sample estimates:  
mean in group female mean in group male   
 98.65789 98.16500

The hypotheses for this test are:

Notice the moderate value: the test fails to reject , but just barely. This type of result is usually what prompts a post-hoc analysis. The goal is to answer the question: *how strongly does the result rule out the alternative?*

|  |
| --- |
| Your turn |
| Use power.t.test with conditions based on the sample estimates. Assume the true difference is actually 0.5 °F. Determine the power of the test above when:   1. Population SD is the smaller of the two groups 2. Population SD is the larger of the two groups 3. A one-sided test is used instead   # power with smaller sd power.t.test(n = ..., delta = ..., sd = ..., sig.level = ..., alternative = ...)  # power with larger sd power.t.test(n = ..., delta = ..., sd = ..., sig.level = ..., alternative = ...)  # power for one-sided test power.t.test(n = ..., delta = ..., sd = ..., sig.level = ..., alternative = ...)  Based on your answers, discuss with your group/neighbor whether you think the negative test result rules out the alternative hypothesis of a difference by sex. |

### Power analysis for study design

The other kind of power analysis relates to study design, and seeks to answer the question: *how much data do I need to collect to achieve a specified power?*

The same function power.t.test() can be used *with a power argument instead of an n argument* to calculate the sample size that achieves a particular power for a particular difference, under assumptions about population variability and test conditions.

For the cloud seeding test, if we wished to detect a difference of at least 250 with power 0.9 using the same test and supposing that the population variability is about what we observed in the previous study, the power calculation for sample size determination would look as follows:

power.t.test(power = 0.9, delta = 250, sd = 650, sig.level = 0.05, alternative = 'two.sided')

Two-sample t test power calculation   
  
 n = 143.0276  
 delta = 250  
 sd = 650  
 sig.level = 0.05  
 power = 0.9  
 alternative = two.sided  
  
NOTE: n is number in \*each\* group

Thus, we would need 144 observations in each group.

|  |
| --- |
| Your turn |
| Continuing with the example of body temperatures, suppose you are designing a follow-up study and wish for the study to be sensitive to temperature differences of 0.4 °F.  Suppose also that you have a $2K budget and data acquisition costs are estimated at $10 per study subject. Your task is to determine the best power achievable within budget and assess the feasibility of the study objectives.  Do a sample size power calculation to determine how much data you would need to collect to achieve power 0.7 if…   1. You use a two-sided test and assume population standard deviation is as estimated above. 2. You use a two-sided test and assume population standard deviation is 1.2 times larger than estimated above. 3. You use a one-sided test and assume population standard deviation is as estimated above. 4. You use a one-sided test and assume population standard deviation is 1.2 times larger than estimated above.   # 1. two sided test, population sd = 1 power.t.test(power = ..., delta = ..., sd = ..., sig.level = ..., alternative = ...)  # 2. two sided test, population sd = 1.2 power.t.test(power = ..., delta = ..., sd = ..., sig.level = ..., alternative = ...)  # 3. one sided test, population sd = 1 power.t.test(power = ..., delta = ..., sd = ..., sig.level = ..., alternative = ...)  # 4. one sided test, population sd = 1.2 power.t.test(power = ..., delta = ..., sd = ..., sig.level = ..., alternative = ...) |