If I have a goal to acheive an uncertain value, how hard should I look?

Drew Tyre

2020-07-10

A colleague wrote:

I am working on a restoration monitoring protocol for [a grassland park] with one of my staff. I am struggling with determining an appropriate sample size, but more importantly, an appropriate statistical approach. What I think would be an appropriate approach is take enough samples that we can say with some level of confidence that our sample’s confidence interval includes our target. Does that sound reasonable to you?

For example, say the target for achieving restoration is <38% bare ground visible. We measure this attribute at 40 plots in a restoration field, and then we can calculate the mean, SD, SE, and 90% confidence intervals (testing for normality here and transforming if necessary). The result is mean 43% (LCI 37%, UCI 49%). Target met.

I have data from previous monitoring that I could plug into a power analysis to help with sample size, but I can’t seem to figure that out–in Program R, with the “pwr” package, I have tried the pwr.t.test with one-sample, but the Cohen’s D description doesn’t seem to make sense for a one-sample test.

I can think of two ways to go about doing this.

1. Use a one sided hypothesis – reject the hypothesis that bare ground is greater than 38%.
2. Use a Bayesian calculation of the probability that percent bare ground is less than 38%

## Power of a one sided hypothesis

It is so common to use a null hypothesis of with a two sided alternative that it is easy to forget there are other possibilities! In this case I think the appropriate null hypothesis is . This can be calculated with pwr.t.test() using the options for a one sample test against the alternative of . The somewhat unusual bit is figuring out what effect size to specify.

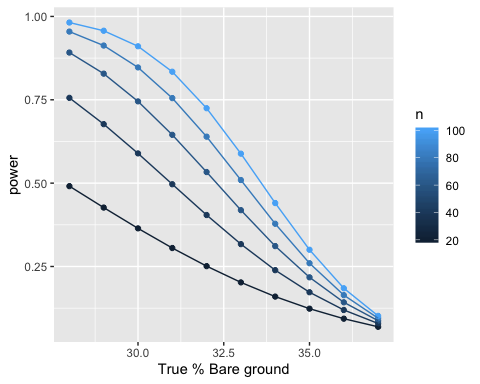
In a “normal” power calculation, you specify Cohen’s d as the minimum difference between means. It is a minimum because larger differences will have greater power – the specified effect size will have the desired power for the specified sample size. In the current case, the effect size is the minimum value below the target that has the specified power. So if the effect size is 1, power is 0.8, then the function will tell me how big a sample is needed to reject the hypothesis . An effect size of 1 means that the calculated power will apply if . We get the value of from pilot data.

library("tidyverse")  
bareground <- read\_csv(here::here("content/post/data/bareground.csv"))  
(sd\_bg <- sd(bareground$bareground))

## [1] 26.58041

To calculate the power of a given sample size, we need to choose two of sample size, effect size, and power. Of the three, I think it is simpler to specify sample size, because we can calculate the effort needed to measure 40 plots for example. In ecological work, the limitation is usually the budget, and we want to know how well a given budgeted effort will work. Given the sample size I want to calculate the power of rejecting the null for many different effect sizes.

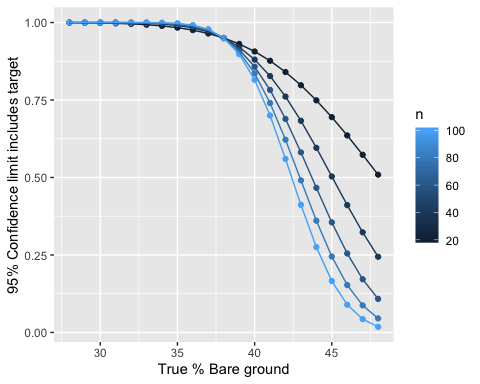
library("pwr")  
results <- crossing(n = c(20, 40, 60, 80, 100),   
 true\_diff = -(1:10)) %>%   
 mutate(bg = true\_diff + 38,  
 d = true\_diff / sd\_bg,  
 power = map2\_dbl(n, d, ~pwr.t.test(n = .x,   
 d = .y,   
 power = NULL,   
 type = "one.sample",   
 alternative = "less")$power))  
ggplot(data = results,  
 mapping = aes(x = bg, y = power, color = n, group = n)) +   
 geom\_point() +   
 geom\_line() +   
 xlab("True % Bare ground")



This graph gives the probability that an upper 95% one sided confidence limit will exclude the target, given a true value of average bare ground.

Now that I see this, and re-read the question, I think my correspondent is actually satisfied if the *lower* confidence limit includes the target. In other words, the goal is acheived if the test *fails* to reject a null of . This makes the question a bit tricky, because the best way to acheive the target is to take a *small* sample size, which will give large confidence limits! Also, if then the power is just , the value of a type I error.

results <- crossing(n = c(20, 40, 60, 80, 100),   
 true\_diff = -10:10) %>%   
 mutate(bg = true\_diff + 38,  
 d = true\_diff / sd\_bg,  
 power = map2\_dbl(n, d, ~pwr.t.test(n = .x,   
 d = .y,   
 power = NULL,   
 type = "one.sample",   
 alternative = "greater")$power),  
 failure = 1 - power)  
ggplot(data = results,  
 mapping = aes(x = bg, y = failure, color = n, group = n)) +   
 geom\_point() +   
 geom\_line() +   
 xlab("True % Bare ground") +  
 ylab("95% Confidence limit includes target")

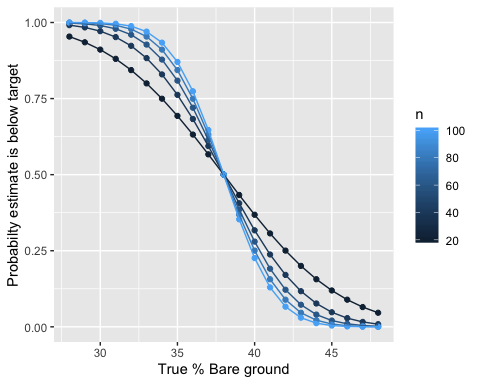


This is very non-intuitive! By associating acheivement of the target with the *failure* to reject a given hypothesis, the probability of success goes *down* with increasing sample size, if the true amount of bare ground is bigger than the target. I’ve often blithely advised my wildlife management students to use the lower confidence limit to evaluate whether a target has been acheived or not, but this is actually a bad idea.

The point where all the lines cross is just the probability of a Type I error, here. That happens at the point where the true amount of bare ground is equal to the target. This makes sense; a type I error is rejecting the null (confidence limits excluding the target) when it is in fact true. If the true amount of bare ground is less than the target (the desired condition), the situation flips around and increasing sample size increases the probability of success. I think that happens because a larger sample size decreases the probability that the estimated mean is above the target. The differences between sample sizes are very small in that case.

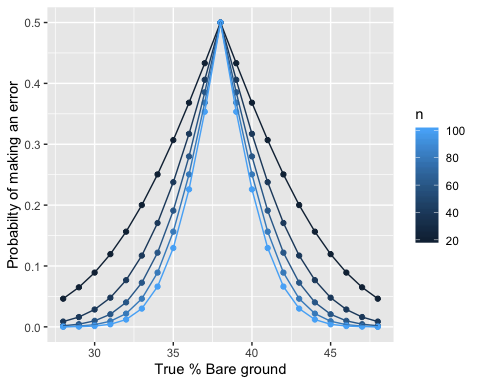
What if we say the estimated mean has to be below the target? If the true mean is equal to the target, this strategy has a 50% chance of success *independent of the sample size*! The probability of success rises if the true mean is less than the target, and that does depend on the sample size because the distribution of estimated means has .

results <- crossing(n = c(20, 40, 60, 80, 100),   
 true\_diff = -10:10) %>%   
 mutate(bg = true\_diff + 38,  
 p = pnorm(true\_diff, sd = sd\_bg/sqrt(n), lower.tail = FALSE))   
ggplot(data = results,  
 mapping = aes(x = bg, y = p, color = n, group = n)) +   
 geom\_point() +   
 geom\_line() +   
 xlab("True % Bare ground") +  
 ylab("Probabilty estimate is below target")



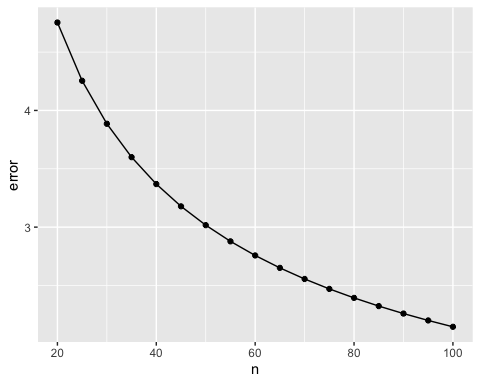
That makes sense to me – if the truth is above the target, having an estimate below the target is an error, and more samples reduces the probability of that error. If the truth is below the target, then an estimate above the target is an error (1-probability on y axis). So I should redraw that like this:

results <- results %>%   
 mutate(error = if\_else(true\_diff < 0, 1-p, p))  
ggplot(data = results,  
 mapping = aes(x = bg, y = error, color = n, group = n)) +   
 geom\_point() +   
 geom\_line() +   
 xlab("True % Bare ground") +  
 ylab("Probabilty of making an error")



Now we’re getting somewhere! More samples always reduces the probability of making an error. The probability of an error is highest when the true amount of bare ground is at the target, and at that point the error is independent of sample size. Additional sampling makes the biggest difference at intermediate deviations of truth from the target. So really, I ought to integrate over all possible values of the truth.

results <- crossing(n = seq(20,100, 5),   
 true\_bg = 0:100) %>%   
 mutate(true\_diff = true\_bg - 38,  
 p = pnorm(true\_diff, sd = sd\_bg/sqrt(n), lower.tail = FALSE),  
 error = if\_else(true\_diff < 0, 1-p, p)) %>%   
 group\_by(n) %>%   
 mutate(tpz = (lag(error,1) + error)/2) %>% # this step is wrong somehow  
 summarize(error = sum(tpz, na.rm = TRUE),  
 sump = sum(p))  
  
ggplot(data = results,   
 mapping = aes(x = n, y = error)) +   
 geom\_line() + geom\_point()



So this is nearly there, but still not quite right. I’m missing a normalizing constant somewhere because these are all bigger than 1.

test <- matrix(rnorm(20000, mean = 0, sd = sd\_bg), nrow = 20, ncol = 1000) %>%   
 as.data.frame() %>%   
 summarize\_all(mean) %>%   
 pivot\_longer(cols = 1:1000)  
  
ggplot(data = test) +   
 geom\_histogram(mapping = aes(x = value, y = ..density..)) +   
 stat\_function(data = data.frame(x = seq(-3,3,0.1)),   
 mapping = aes(x = x\*sd\_bg),   
 fun = dt, args = list(df = 20)) +   
 stat\_function(data = data.frame(x = seq(-10,10,0.1)),   
 mapping = aes(x = x),   
 fun = dnorm, args = list(sd = sd\_bg / sqrt(20)))

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.

