Lab 1: Simulations for power calculation

PB HLTH 250C (Spring 2024)

January 25, 2024

> *Power calculations are often a mystery, but they can be easy (and kind of fun) with simulation.*

Today, we will take some time to explore the elegance and fun of power calculations by simulation. Statistical power is the probability $(1-\beta)$ of rejecting a null hypothesis H_0 when it is in fact false. Power calculation requires defining the rejection region and the true value of the parameter of interest (and the distribution it parameterizes). The significance level α is the probability of rejecting the null hypothesis H_0 when it is in fact true. For null hypothesis $H_0 : \theta = \theta_0$, test statistic $\hat{\theta}$, and rejection region $\mathcal{R}$ we have

$$\begin{aligned} &\text{Significance level:} && \alpha &&= \Pr\left[\hat{\theta} \in \mathcal{R} \mid \theta = \theta_0\right] \\ &\text{Power:} && 1-\beta &&= \Pr\left[\hat{\theta} \in \mathcal{R} \mid \theta = \theta_1\right] \end{aligned}$$

where θ is the true value of the parameter of interest.

> *Question.*
>
> If $(1-\beta)$ is statistical power, what is the interpretation of β?

Coin tossing example

Consider the following example. Suppose I toss a coin $n = 10$ times. My null hypothesis is that the coin is a fair coin i.e. the probability of getting heads is $p = 0.5$. I reject this null hypothesis in favor of a two-sided alternative hypothesis if I observe 8 or more heads or 2 or fewer heads. Let X be the number of heads we observe in $n = 10$ tosses.

The significance level of this test of significance is

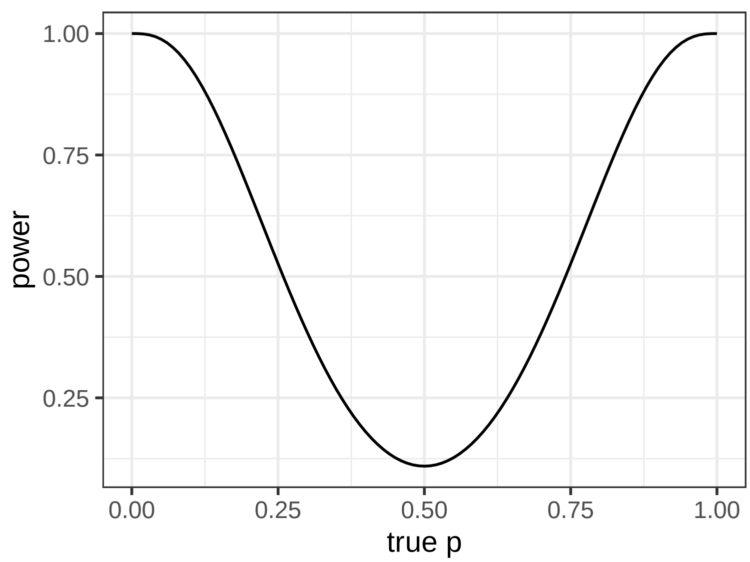
$$\begin{aligned}
\alpha &= \Pr\left[X \in \mathcal{R} \mid p = 0.5\right] \\
&= \Pr\left[X \leq 2 \mid p = 0.5\right] + \Pr\left[X \geq 8 \mid p = 0.5\right] \\
&= \sum_{k=0}^{2} \binom{10}{k} (0.5)^{10} + \sum_{k=8}^{10} \binom{10}{k} (0.5)^{10} \\
&= 0.109
\end{aligned}$$

Now suppose we believe the coin lands on heads with probability p_1. Then, then, we would have statistical power

$$\begin{aligned}
1 - \beta &= \Pr\left[X \in \mathcal{R} \mid p = p_1\right] \\
&= \Pr\left[X \leq 2 \mid p = p_1\right] + \Pr\left[X \geq 8 \mid p = p_1\right] \\
&= \sum_{k=0}^{2} \binom{10}{k} (p_1)^k (1 - p_1)^{10-k} + \sum_{k=8}^{10} \binom{10}{k} (p_1)^k (1 - p_1)^{10-k}
\end{aligned}$$

We can plot statistical power as a function of p_1 to get a "power curve."

```r
ggplot() +
    geom_function(fun = function(x) {
        pbinom(2, 10, x) + 1 - pbinom(7, 10, x)
    }) + labs(x = "true p", y = "power") + theme_bw()
```



With simulation, we could produce this power curve without manipulating equations as we did above.

Exercise.

Try outlining the steps for conducting a power calculation for this problem using simulation. Can you conduct a power calculation by simulation without knowing the explicit form of the sampling distribution?

Risk ratio example

In many realistic settings, no easy closed-form formula for power exists. Consider the following example. Suppose we have $n = 500$ subjects in a closed study population. We randomly allocate binary treatment A and measure subsequent disease status D. Assume $\Pr[A = 1] = 0.5$ and $\Pr[D = 1 \mid A = 0] = 0.4$. We use the following log-binomial model to summarize the hypothetical relationship between treatment and disease

$$\log\left(\Pr[D = 1 \mid A]\right) = \beta_0 + \beta_1 A$$

Goal: Calculate the power to detect a significant RR at the 0.05 significance level for a two-sided test when the true RR $\exp(\beta_1) = \{1,\ 0.8,\ 0.7,\ 0.6,\ 0.5\}$.

This power calculation does not involve a complex estimator (we are not even adjusting for potential confounders). Yet, a closed-form formula is elusive. Nonetheless, we can get an answer by simulating. Here are the general steps:

0. Choose a large number `M` to be the number of simulations.
1. For `i` in `1:M`:
    - Generate data consistent with the assumptions.
    - Apply the estimator.
    - Obtain the p-value (or other measure).
2. Summarize the estimator characteristics over the simulations.

First, we save the constant parameter values that we do not have to re-generate in every iteration of our simulation.

```r
# Set seed for reproducibility
set.seed(876)

# Set the number of simulations
M <- 1e3
# Number of subjects
```

```r
n <- 500
# Pr[D = 1 | X = 0]
b0 <- log(0.40)
# Alternative hypotheses at which to calculate power
b1 <- log(c(1, 0.8, 0.7, 0.6, 0.5))
```

Now, let's initialize an empty object in which we will save our simulated p-values. Note that we set the number of rows to be the number of simulations and the number of columns to be the number of "true" RRs at which we calculate power.

```r
# Your script here
```

We will be generating $M = 1000$ simulated p-values for each of the 5 true RRs. Hence, we need to loop over both the rows and the columns. We can do so by nesting one loop in another.
Note. When nesting loops, be careful when specifying the iterator.

```r
# Your script here
```

Appendix

```r
knitr::opts_chunk$set(
  echo = T,
  fig.align = "center",
  fig.width = 4,
  fig.height = 3
)
library(ggplot2)
ggplot() +
    geom_function(fun = function(x) {
        pbinom(2, 10, x) + 1 - pbinom(7, 10, x)
    }) + labs(x = "true p", y = "power") + theme_bw()

# Set seed for reproducibility
set.seed(876)

# Set the number of simulations
M <- 1e3
# Number of subjects
n <- 500
# Pr[D = 1 | X = 0]
b0 <- log(0.40)
# Alternative hypotheses at which to calculate power
b1 <- log(c(1, 0.8, 0.7, 0.6, 0.5))

# Your script here
# Your script here
```