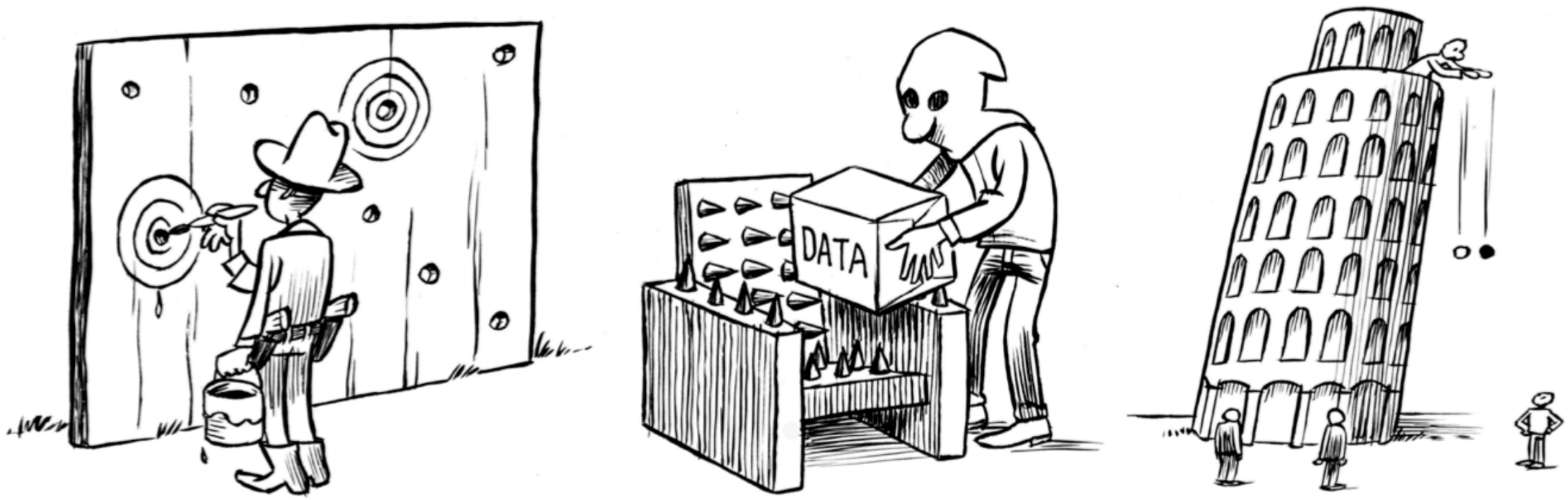
Power analysis



Exploratory Research ⟷ Confirmatory Research

Wonky Stats ⟷ Sound Stats

02/07/2020

Logistics

RMarkdown can be tricky to debug ...

Instructions

This homework is due by **Thursday, February 6th, 8:00pm**.

As per usual, please upload your rendered pdf on Canvas, and note that late submissions will receive 0 points.

Note:

- Some code chunks contain some skeleton code. The code chunk option for these chunks is set to `eval=F` so that knitting the RMarkdown document doesn't throw any errors. Make sure to set these chunks to `eval=T` when you knit your homework, so that your calculations are shown in the pdf.
- Make sure to show the results of your calculations in the knitted pdf, for example, by using the `print()` function at the end of a code chunk.
- Some questions ask for a short written response as indicated by this prompt: **Your answer:**

```
106 ### a) Generate the sampling distributions
107
108 The sample variance $s^2 = \frac{\sum_{i=1}^n(Y_i-\overline Y)^2}{n-1}$ is an
    unbiased estimator of the population variance. Run a simulation to show that
    this is the case. Compare the results of your simulation with another estimator
    $s'^2 = \frac{\sum_{i=1}^n(Y_i-\overline Y)^2}{n}$ and show that $s'^2$ is
    biased. Note that the only difference between $s^2$ and $s'^2$ is in the
    denominator.
109
110 ```{r modeling-data-10, eval=F}
111 # make example reproducible
112 set.seed(1)
```

RMarkdown can be tricky to debug ...

https://evalsp20.classes.andrewheiss.com/reference/rmarkdown/

https://bookdown.org/yihui/rmarkdown/

https://yihui.org/knitr/options/

Midterm

Will most likely be available later today.

If not today then tomorrow evening by the latest.

Plan for today

- Quick power analysis review
- Goal: Determine sample size using power analysis via simulation
- Learn about more advanced simulation techniques in R
  - `map()`
  - list columns: `nest()`, `unnest()`

Quick power analysis review

Type I Error

Type II Error

$\mathbf{H_0}$: Not pregnant. $\mathbf{H_1}$: Pregnant.

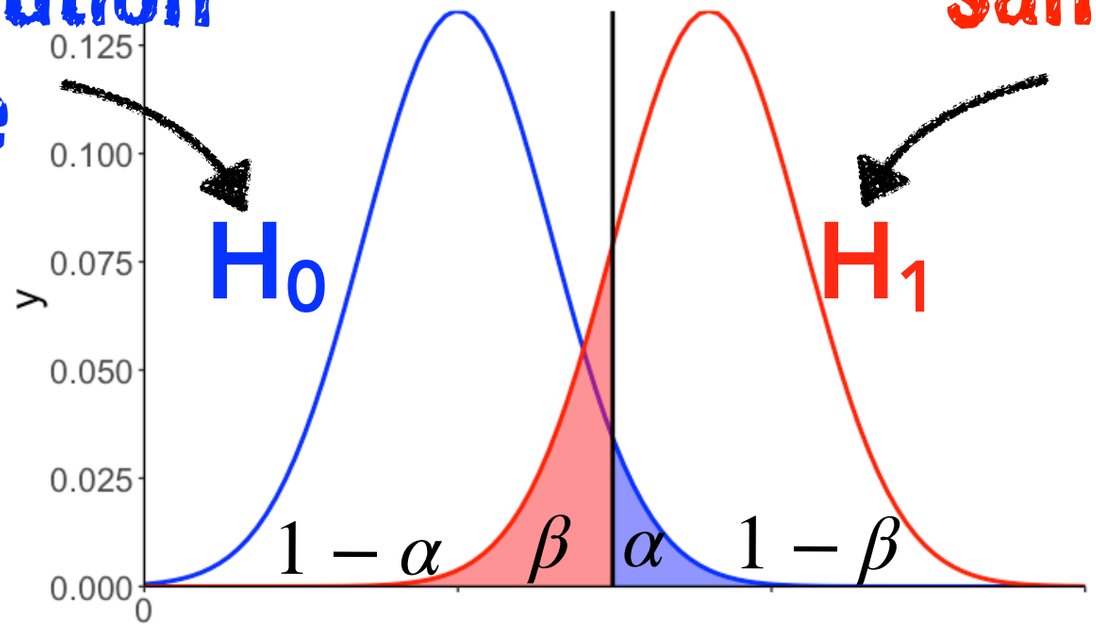
Type I Error: Falsely rejecting the null hypothesis (even though it is true).

Type II Error: Failing to reject the null hypothesis (even though it is false).

The knobs we can turn to affect power

sampling distribution if H_0 is true

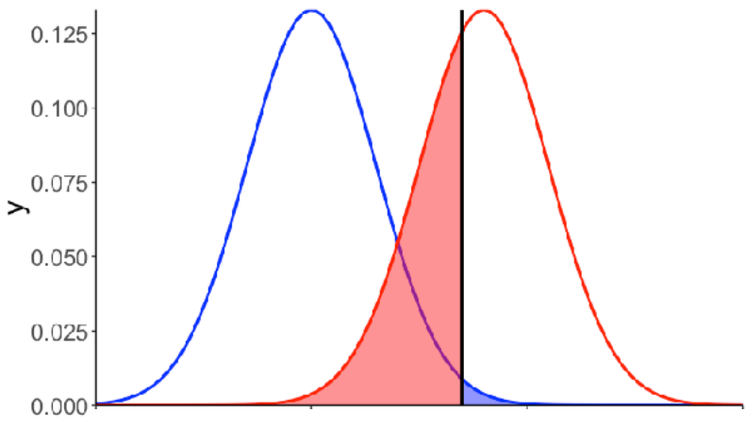
sampling distribution if H_1 is true

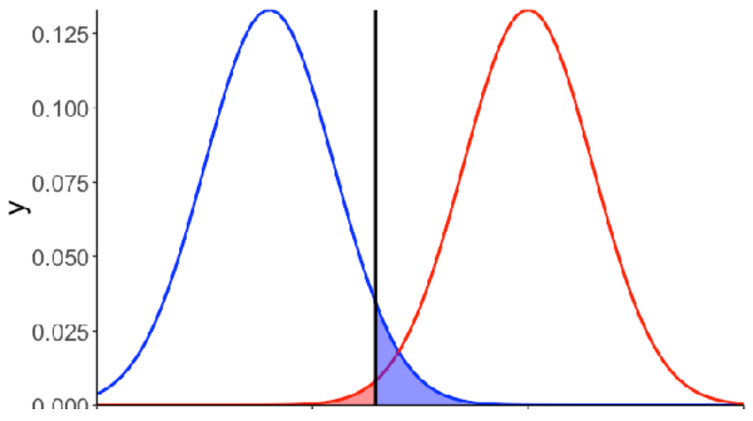


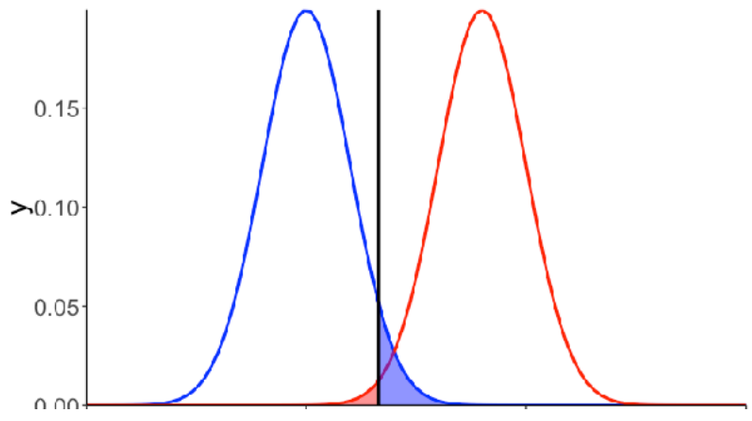
α

effect size

sample size







Determining sample size

How many participants do I need to run to have a good chance of detecting a true effect?

Power simulation recipe

- assume:
  - α, n, effect size
- simulate a large number of data sets of size n with the specified effect size
- for each data set, run a statistical test to calculate the p-value
- determine the probability of rejecting the H_0 (given that H_1 is true)

Learn about more advanced simulation techniques in R

Let's simulate

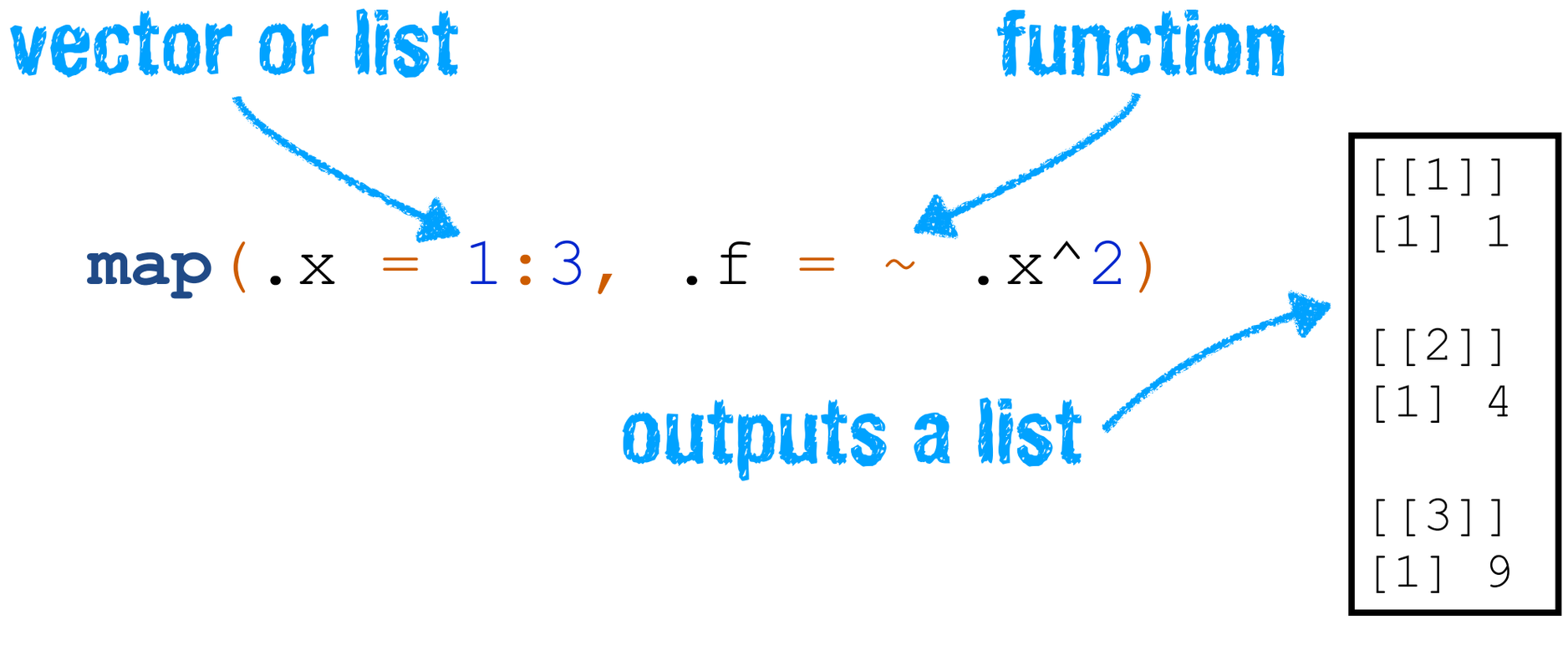
```
library("purrr")
```

automatically loaded with

```
library("tidyverse")
```

`map()`

`map()`



- **`map`**`(list, function)` applies a function to each element of the list
- it's a unified version of the many different **`apply`**`()` functions in base R
- you already know a cousin of **`map`**`()`: **`replicate`**`()`
- use **`map`**`()`, don't write `for(){}` loops!
- it's extremely powerful in combination with data frames

map()

vector or list

function

```
map(.x = 1:3, .f = ~ .x^2)
```

outputs a list

```
[[1]]
[1] 1

[[2]]
[1] 4

[[3]]
[1] 9
```

same same but different

```
map(.x = 1:3, .f = function(.x) .x^2)
```

anonymous function

map()

same same but different

```
map(.x = 1:3, .f = ~ .x^2)

map(1:3, ~ .x^2)

map(1:3, ~ .^2)
```

```
map(.x = 1:3, .f = function(.x) .x^2)

# using a function
square = function(x){x^2}
map(1:3, square)
```

time

05:00

blue

pink

blue

pink

CAUTION

The practice examples are more advanced than what we've done in the past.

Try your best, play around with the previous code chunks, don't worry if you don't manage to do the practice exercises straight away.

I'll go through the solutions slowly, step by step in class.

Power analysis example

Let's simulate ...

```
1 # number of simulations
2 n_simulations = 1000
3
4 # run simulation
5 df.power = crossing(n = seq(10, 50, 1),
6                      simulation = 1:n_simulations,
7                      p = c(0.75, 0.8, 0.85)) %>%
```

set up simulation grid

n	simulation	p
10	1	0.75
10	1	0.80
10	1	0.85
10	2	0.75
10	2	0.80
10	2	0.85
10	3	0.75
10	3	0.80
10	3	0.85
10	4	0.75

Let's simulate ...

```
1 # number of simulations
2 n_simulations = 1000
3
4 # run simulation
5 df.power = crossing(n = seq(10, 50, 1),
6                     simulation = 1:n_simulations,
7                     p = c(0.75, 0.8, 0.85)) %>%
8   mutate(index = 1:n()) %>%
9   mutate(response = rbinom(n = n(), size = n, prob = p)) %>%
```

generate random data

n	simulation	p	index	response
10	1	0.75	1	6
10	2	0.75	1	10
10	3	0.75	1	9
10	4	0.75	1	8
10	5	0.75	1	7
10	6	0.75	1	7
10	7	0.75	1	10

Let's simulate ...

```
# number of simulations
n_simulations = 1000

# run simulation
df.power = crossing(n = seq(10, 50, 1),
                    simulation = 1:n_simulations,
                    p = c(0.75, 0.8, 0.85)) %>%
  mutate(index = 1:n()) %>%
  mutate(response = rbinom(n = n(), size = n, prob = p)) %>%
  group_by(index, simulation, p) %>%
  nest() %>%
```

put data in a list column

	index	simulation	p	data
1	1	1	0.75	list(n = 10, response = 7)
2	2	1	0.80	list(n = 10, response = 8)
3	3	1	0.85	list(n = 10, response = 9)
4	4	2	0.75	list(n = 10, response = 9)
5	5	2	0.80	list(n = 10, response = 9)
6	6	2	0.85	list(n = 10, response = 10)
7	7	3	0.75	list(n = 10, response = 9)
8	8	3	0.80	list(n = 10, response = 5)
9	9	3	0.85	list(n = 10, response = 6)
10	10	4	0.75	list(n = 10, response = 7)

Let's simulate ...

	index	simulation	p	data	fit
1	1	1	0.75	list(n = 10, response = 7)	list(statistic = c(`number of successes` = 7), parame...
2	2	1	0.80	list(n = 10, response = 9)	list(statistic = c(`number of successes` = 9), parame...
3	3	1	0.85	list(n = 10, response = 9)	list(statistic = c(`number of successes` = 9), parame...
4	4	2	0.75	list(n = 10, response = 8)	list(statistic = c(`number of successes` = 8), parame...
5	5	2	0.80	list(n = 10, response = 8)	list(statistic = c(`number of successes` = 8), parame...
6	6	2	0.85	list(n = 10, response = 10)	list(statistic = c(`number of successes` = 10), param...
7	7	3	0.75	list(n = 10, response = 7)	list(statistic = c(`number of successes` = 7), parame...
8	8	3	0.80	list(n = 10, response = 8)	list(statistic = c(`number of successes` = 8), parame...
9	9	3	0.85	list(n = 10, response = 9)	list(statistic = c(`number of successes` = 9), parame...
10	10	4	0.75	list(n = 10, response = 6)	list(statistic = c(`number of successes` = 6), parame...

```
12    mutate(fit = map(data,
13                     ~ binom.test(x = .$response,
14                                  n = .$n,
15                                  p = 0.5,
16                                  alternative = "two.sided")),
```

run binomial test for each cell in the data column

map(list, function) applies function to each element of the list

Let's simulate ...

	index	simulation	p	p.value	n	response
1	1	1	0.75	1.093750e-01	10	8
2	2	1	0.80	1.953125e-03	10	10
3	3	1	0.85	3.437500e-01	10	7
4	4	2	0.75	2.148438e-02	10	9
5	5	2	0.80	2.148438e-02	10	9
6	6	2	0.85	3.437500e-01	10	7
7	7	3	0.75	3.437500e-01	10	7
8	8	3	0.80	1.093750e-01	10	8
9	9	3	0.85	2.148438e-02	10	9
10	10	4	0.75	2.148438e-02	10	9

```
13    mutate(fit = map(data,
14                     ~ binom.test(x = .$response,
15                                  n = .$n,
16                                  p = 0.5,
17                                  alternative = "two.sided")),
18           p.value = map_dbl(fit, ~ .$p.value)) %>%
19    unnest(data) %>%
20    select(-fit)
```

extract p-values

Let's simulate ...

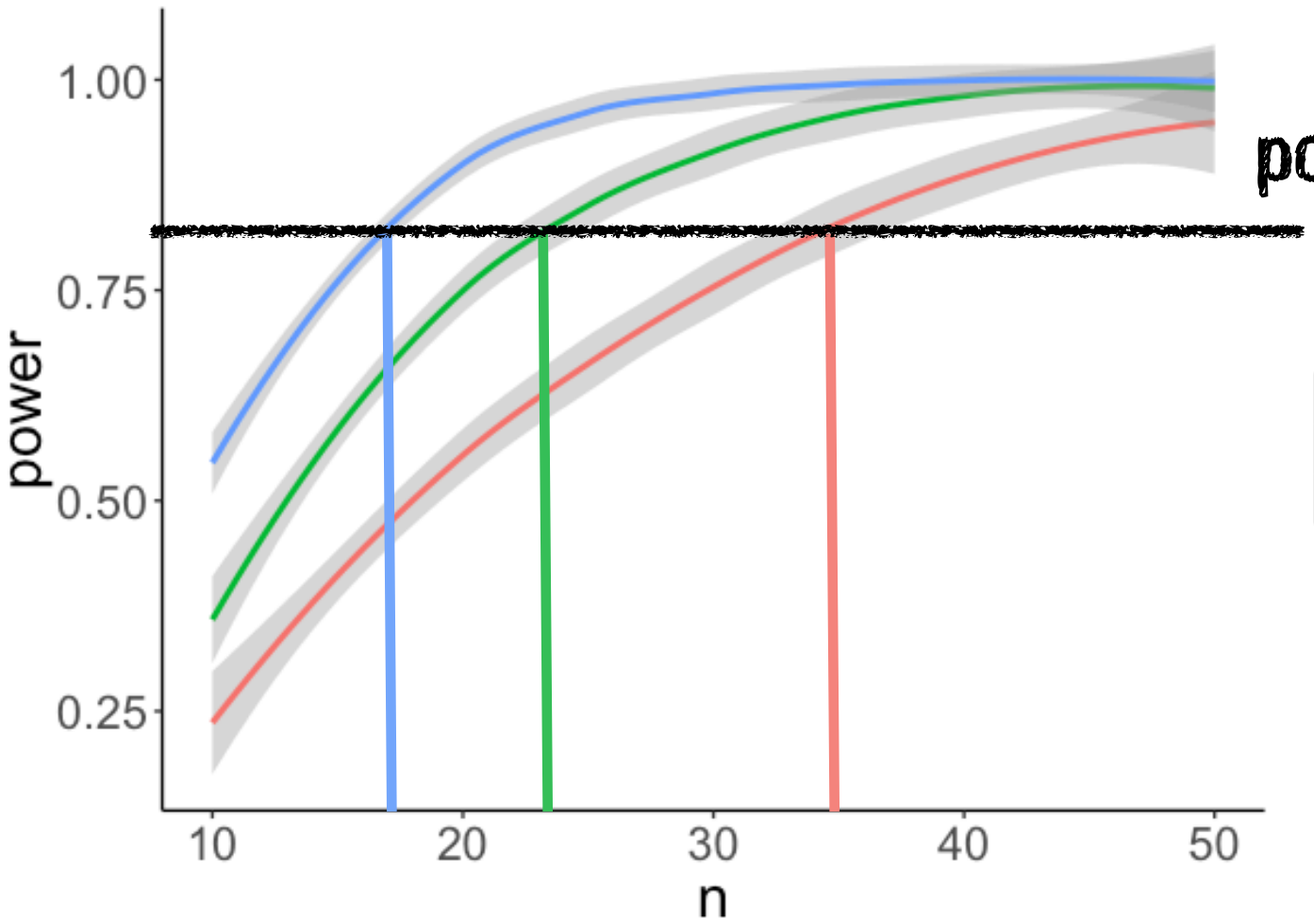
```
1 # data frame for plot
2 df.plot = df.power %>%
3   group_by(n, p) %>%
4   summarize(power = sum(p.value < 0.05) / n())
```

probability of rejecting H_0 when H_1 is true

Power

$$1 - \beta$$

$p(\text{reject } H_0 | H_1 \text{ is true})$



p = probability of coin coming up heads

n	p	power
10	0.75	0.24
10	0.8	0.38
10	0.85	0.58
11	0.75	0.24
11	0.8	0.40
11	0.85	0.50
12	0.75	0.45
12	0.8	0.55
12	0.85	0.82
13	0.75	0.26

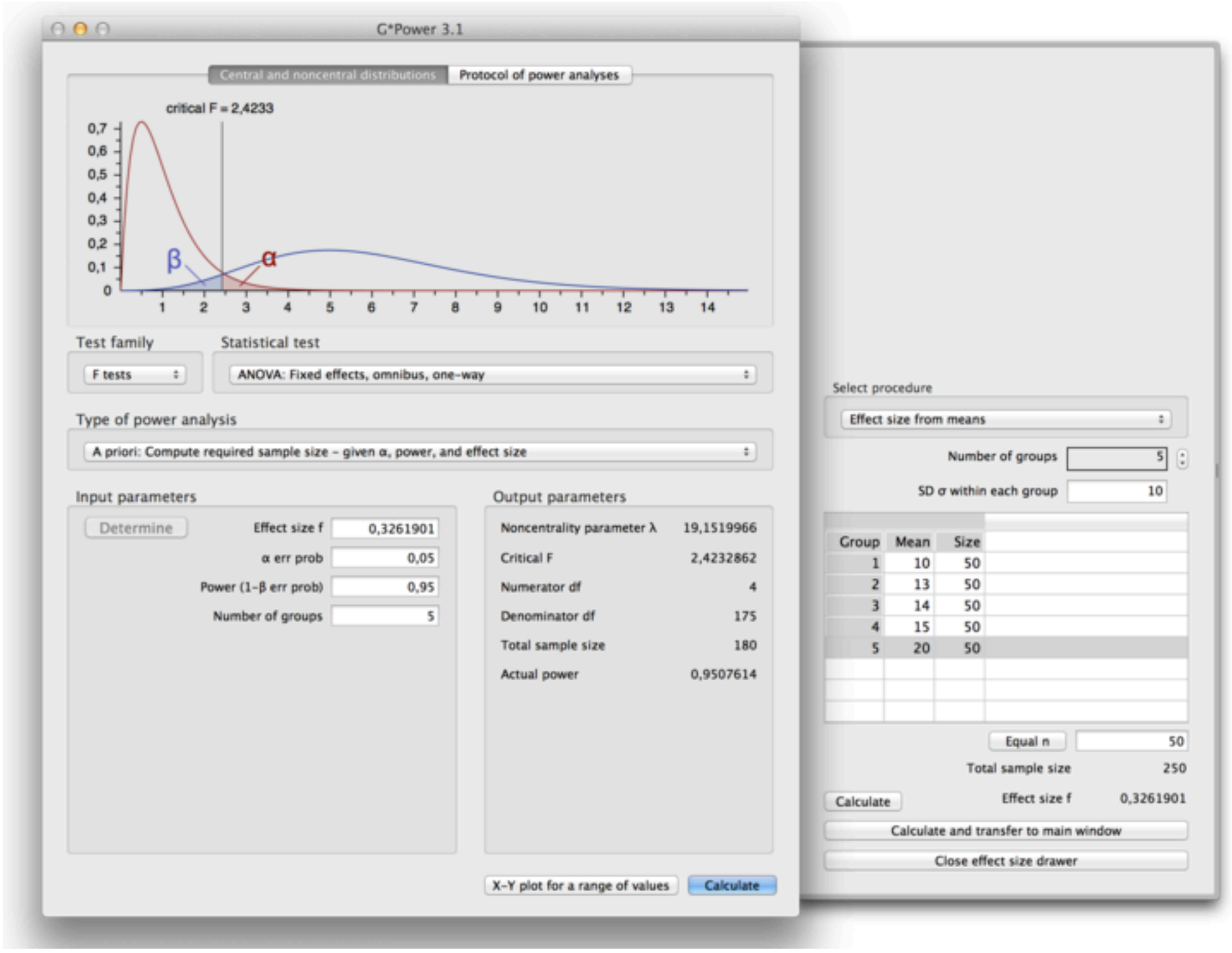
Let's simulate ...

- here, I've used a simple example (Binomial test)
- but: we can use the same recipe for any statistical test that we are planning on running

Power simulation recipe

- assume:
  - α, n, effect size
- simulate a large number of data sets of size n with the specified effect size
- for each data set, run a statistical test to calculate the p-value for a given α
- determine the probability of rejecting the H_0 (given that H_1 is true)

G*Power 3.1: Alternative software for power calculations



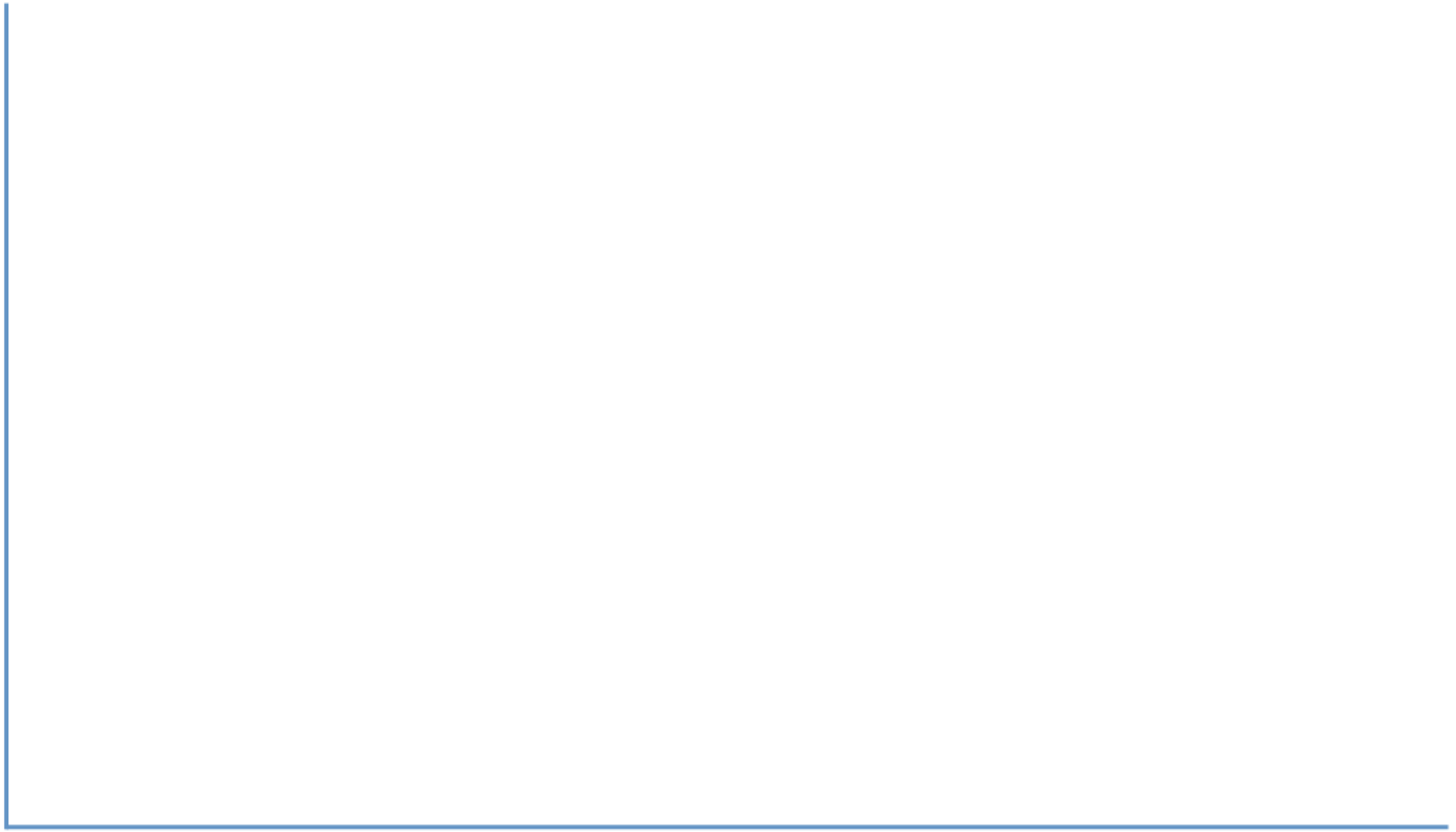
http://www.gpower.hhu.de/

Plan for today

- Quick power analysis review
- Goal: Determine sample size using power analysis via simulation
- Learn about more advanced simulation techniques in R
  - `map()`
  - list columns: `nest()`, `unnest()`

Feedback

How was the pace of today's class?



How happy were you with today's class overall?

What did you like about today's class? What could be improved next time?

Thank you!