No one GLM should have all that power

Power analysis of multilevel/hierarchical generalized linear models

Mikhail Popov

01 May 2019

Wikimedia Foundation

Overview

1. Linear regression modeling
2. Generalized linear models
3. Multilevel/hierarchical regression modeling
4. Null hypothesis significance testing
5. Simple power analysis via equations
6. Complex power analysis via simulations

Linear regression modeling

$$y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \epsilon_i,\ i = 1, \ldots, n$$

- i indexes the individual n observations
- y is the response variable
- x_1 and x_2 are the predictor variables
- β are the regression coefficients
  - β_0 is the intercept term
  - β_1 and β_2 are the slopes for x_1 and x_2, respectively
- $\epsilon_i \sim \mathcal{N}(0, \sigma)$ is the random error term
  - independently and identically distributed (iid)
  - from a $\mathcal{N}$ormal distribution with mean 0 and standard deviation σ

In general, when there are p predictor variables:

$$y_i = \beta_0 + \beta_1 x_{1i} + \cdots + \beta_p x_{pi} + \epsilon_i,$$
$$\epsilon_i \sim \mathcal{N}(0, \sigma),\ i = 1, \ldots, n$$

which we write consisely (in matrix multiplication form) as

$$\mathbf{y} = \mathbf{X}\beta + \epsilon,\ \epsilon \overset{iid}{\sim} \mathcal{N}(0, \sigma)$$

which is sometimes written as

$$y \sim \mathcal{N}(\mathbf{X}\beta, \sigma)$$

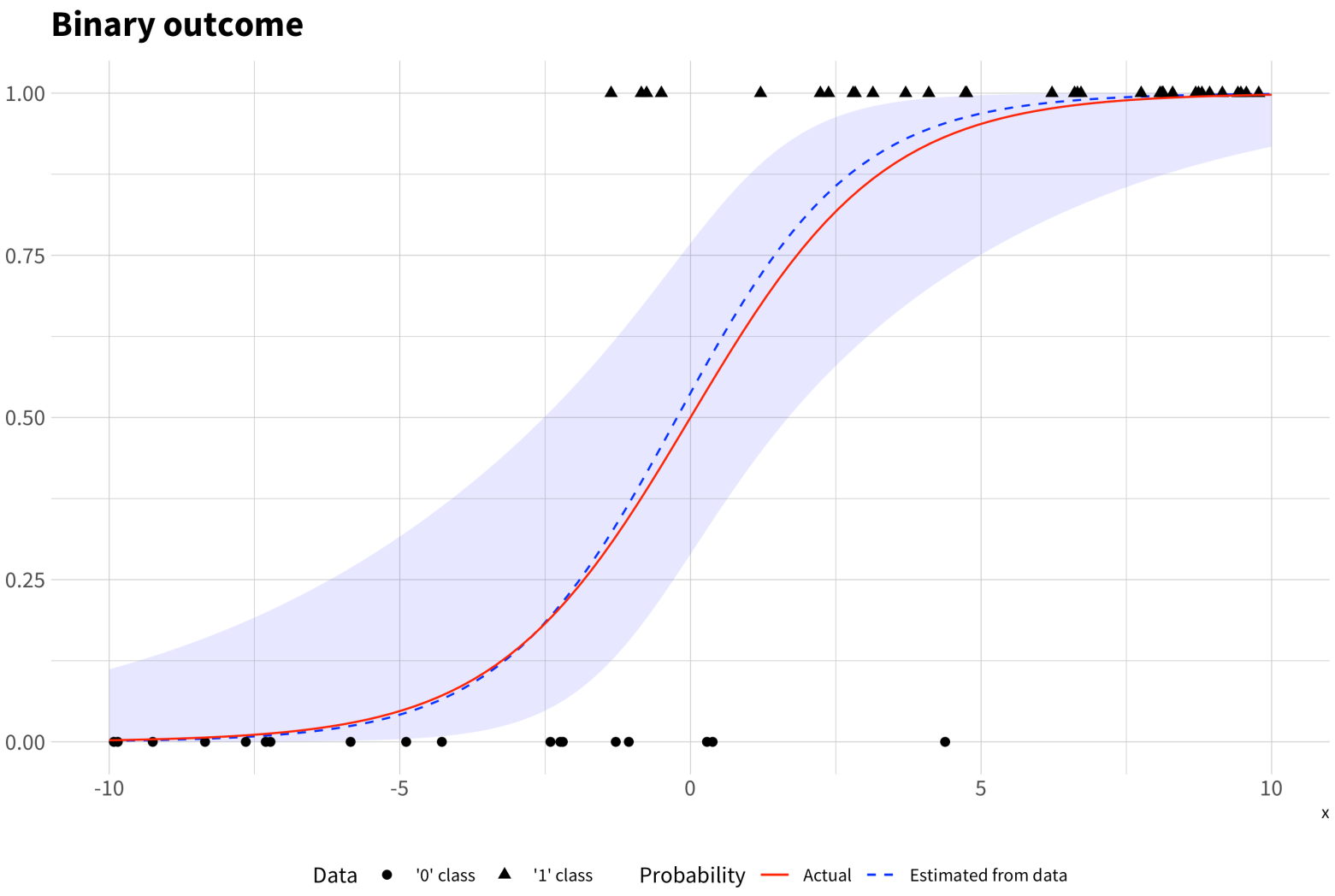
Generalized Linear Models (GLMs)

$$\mathrm{E}(y) = \mu = g^{-1}(\mathbf{X}\beta)$$
$$g(\mu) = \mathbf{X}\beta$$

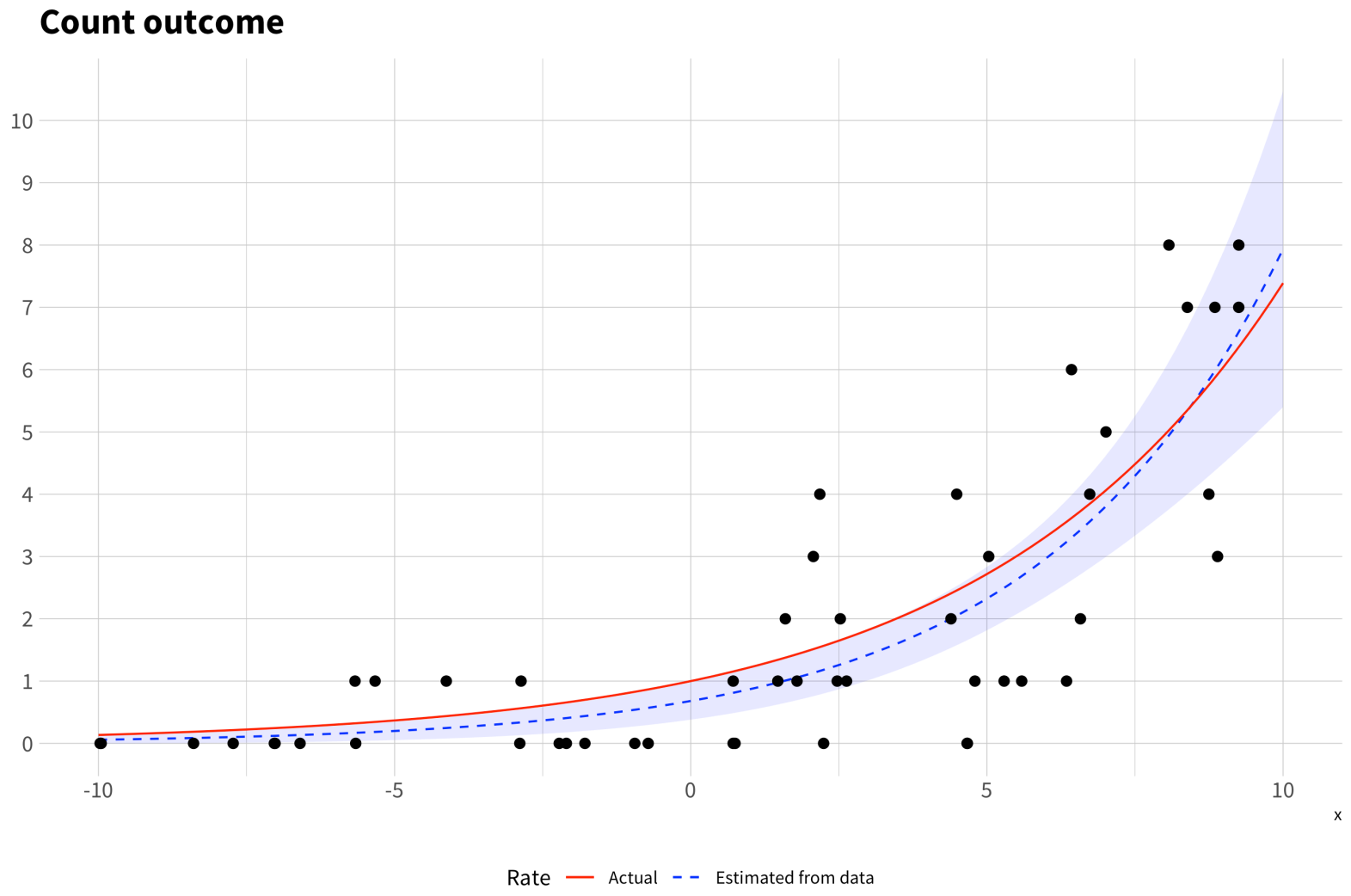
- Model expected value μ; notice no individual-level error term ϵ
- Link function g transforms μ to all real numbers $\mathbb{R}$
  - $g = \log$[1] is most common for Poisson regression for count outcomes, in which case the inverse is the $\exp$ function
  - logit is most common for logistic regression for binary outcomes; logit^{-1} is the inverse function which maps $[-\infty, +\infty] \to [0, 1]$

[1] base e is assumed for logarithms in statistics literature, base 10 is always made explicit as $\log_{10}$

GLM example 1: logistic regression



GLM example 2: Poisson regression



Multilevel/hierarchical regression modeling

Also called “multilevel modeling” and “mixed-effects modeling”, which refers to model having *constant* (fixed) effects and *varying* (random) effects:

- β coefficients (slopes and intercept) are *constant* effects
- ϵ is a *varying* effect which allows each observation to vary from the expected value

Hierarchical regression models enable:

- accounting for individual- and group-level variation
- pooling of information; you can get decent estimates even with small sample sizes

Software

- Python: StatsModels (look for “mixed”), PyMC3
- R: lme4, brms, RStanArm

Examples

sameAs A/B test

- Model of search engine-referred visits per wiki
- Language treated as random effect (269 groups)

Cohort analysis

- Multiple observations across time per user (“repeated measures”)
- Multiple cohorts of users
- Multiple wikis

Brand awareness survey

- Multiple survey responses per household
- Random households per neighborhood
- Random neighborhoods in area

Varying intercept model notation

$$y_i \sim \mathcal{N}(\alpha_{j[i]} + \beta_1 x_1 + \cdots + \beta_p x_p, \sigma_y),\ i = 1, \ldots, n$$
$$\alpha_j \sim \mathcal{N}(\mu, \sigma_\alpha),\ j = 1, \ldots, J$$

- i indexes individual observations in dataset
- j indexes groups present in data
- $j[i]$ is the j-th group that i-th observation belongs to
- $\alpha_1, \ldots, \alpha_J$ are varying intercepts (random effects)
- μ is the overall intercept around which $\alpha_1, \ldots, \alpha_J$ are distributed
- σ_y is individual variability, σ_α is group variability

Hierarchy (multiple levels) of varying intercepts

$$y_i \sim \mathcal{N}(\alpha_{j[i]} + \beta x_i, \sigma_y),\ i = 1, \dots, n$$
$$\alpha_j \sim \mathcal{N}(\gamma_{k[j]}, \sigma_\alpha),\ j = 1, \dots, J$$
$$\gamma_k \sim \mathcal{N}(\mu, \sigma_\gamma),\ k = 1, \dots, K$$

Scenario: local awareness of Wikipedia

- y is an awareness score, x is age at survey time (or other covariate)
- each household may have 1-10 people in it
- n total survey responses from J households
    - responses from household likely to correlate
    - α_j is average awareness of household
- γ_k is average awareness across households in neighborhood k
- μ is average awareness across neighborhoods in surveyed area

Next step: letting effect of age on awareness vary by neighborhood

Varying intercept and varying slope

$$y_i \sim \mathcal{N}(\alpha_{j[i]} + \beta_{j[i]} x, \sigma_y), \; i = 1, \dots, n$$

$$\begin{pmatrix} \alpha_j \\ \beta_j \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \mu_\alpha \\ \mu_\beta \end{pmatrix}, \begin{pmatrix} \sigma_\alpha^2 & \rho\sigma_\alpha\sigma_\beta \\ \rho\sigma_\alpha\sigma_\beta & \sigma_\beta^2 \end{pmatrix} \right), \; j = 1, \dots, J$$

In the way that $\alpha_{j[i]}$ is the **intercept** of the j-th group that observation i belongs to, $\beta_{j[i]}$ is the **slope** of that j-th group.

The model has the α_j, β_j pairs distributed according to a multivariate Normal: $\mathcal{N}(\mu, \Sigma)$ where:

- $\mu = \begin{pmatrix} \mu_\alpha \\ \mu_\beta \end{pmatrix}$ is the overall intercept and overall slope
- Σ is the covariance matrix with the between-group correlation parameter ρ

Null hypothesis signficance testing

Suppose we have a varying-intercept model with one predictor variable x which may be continuous or a binary indicator:

$$
\begin{aligned}
y_i &= \alpha_{j[i]} + \beta x_i + \epsilon_i \\
\alpha_j &\sim \mathcal{N}(\mu, \sigma_\alpha),\ j = 1, \ldots, J \\
\epsilon_i &\sim \mathcal{N}(0, \sigma_\epsilon),\ i = 1, \ldots, n
\end{aligned}
$$

We are interested in whether β (effect of unit increase in x on expected value of y) is statistically significant:

$$
\begin{aligned}
\mathrm{H}_0 &: \beta = 0 \\
\mathrm{H}_a &: \beta \neq 0
\end{aligned}
$$

which is tested through a two-tailed *t*-test with $n - J - 1$ degrees of freedom.

NHST continued

Decision about H_0	Null hypothesis H_0	
	True	**False**
Fail to reject	TN: $1 - \alpha$	FP (Type I error): α
Reject	FN (Type II error): β	TP: $1 - \beta$

- Type I error (false positive) is also the **significance level**. α is the probability of (incorrectly) rejecting H_0 – when it is actually true and should not be rejected
- **Power** $(1 - \beta)$ is the probability of (correctly) rejecting H_0 – when it is actually false and should be rejected

Simple power analysis via equations

May be used to perform power analyses for *t* tests, *F* tests, χ^2 tests, and others:

- G*Power application for Mac and Windows
- pwr package for R
- power module in StatsModels library for Python

Anything more complex requires fake data simulation.

Complex power analysis via simulations

Inputs: effect sizes, standard errors, per-group sample sizes, number of simulations, significance level

Outputs: proportion of simulations that yielded statistically significant results

For each simulation:

1. Generate model parameters β, μ, α_j, etc. based on effect sizes and standard errors
2. Generate predictor variables $\mathbf{X}$ at random
3. Generate response variable y at random from model and predictors
4. Fit model to generated dataset; e.g. using `lme4::lmer`
5. Record significance of β at 0.05 significance level – that is, if the lower bound of the 95% CI is to the right of 0

Scenario: welcome email A/B test

User creates a new account and provides us with an email address. They are randomly selected (with 50%/50% probability) to be in

- **control group**: no email is sent
- **treatment group**: they receive a welcome email with helpful information about contributing to Wikipedia

Purpose of the welcome email is to get more users to edit at least once in the first 48 hours after registration. We will refer to this as *activation*.

We hypothesize that the welcome email will increase the probability of activation by at most 4%.

We want to estimate sample size that has enough power to reliable detect a 4% difference in activation probability.

Data generating process

We assume there is not an excess of non-activations, but in practice:

- Some users provide fake email addresses
- Some users don't check their emails frequently enough
- Some users may get discouraged by wikitext, editing UX, policies, and/or communities

These result in *multiple* data generating processes, one of which yields excessive non-activations (zeros), which is handled through ***zero-inflated models***.

For this example we assume a single data generating process.

The “divide by 4 rule” for logistic regression

Gelman and Hill (2006)[2] introduce the helpful “divide by 4 rule”:

> *We can take logistic regression coefficients (other than the constant term) and divide them by 4 to get an upper bound of the predictive difference corresponding to a unit difference in x.*

Going backwards from the upper bound of predictive difference of 4%, we get $\beta = 0.04 \times 4 = 0.16$. For the standard error:

- 0.01 yields a 95% CI of (3.5%, 4.5%)
- 0.05 yields a 95% CI of (1.6%, 6.4%)
- 0.1 yields a 95% CI of (-0.9%, 8.9%), which contains 0

[2] Gelman, A., & Hill, J. (2006). Data Analysis Using Regression and Multilevel/Hierarchical Models. Cambridge University Press.

Model and parameters

$$y_i \sim \text{Bernoulli}(p_i),\ i = 1, \dots, n$$
$$\text{logit}(p_i) = \alpha_{j[i]} + \beta x_i$$
$$\alpha_j \sim \mathcal{N}(\mu, \sigma_\alpha),\ j = 1, \dots, J$$

- From previous slide, we have $\beta = 0.16, \sigma_\beta = 0.05$
- Set $\mu = -2.5, \sigma_\mu = 0.3$
  - $\text{logit}^{-1}(-2.5)$ is 7.5%
  - $-2.5 \pm 2 \times 0.3$ yields a 95% CI of (4.3%, 13.0%)
  - wide range for overall, baseline activation probabilities
- Set each wiki's baseline activation rate α_j as deviating from overall baseline activation rate μ
  - Since we assume Normality, the 68-95-99.7 rule states 99.7% of wikis would be within $3\sigma_\alpha$ of the mean
  - With $\sigma_\alpha = 0.2$ and $\mu = -2.5$, we'd get a 95% CI of (4.3%, 13.0%)

R packages

Essentials

```
library(magrittr) # for %>% piping
library(arm) # for invlogit & se.fixef
library(purrr) # for functional programming
```

Parallelization

```
library(furrr) # future + purrr
options(mc.cores = 4)
plan(multiprocess)
```

Quick intro to purrr

`map(x, f)` applies function `f` to each element in `x`:

```
named_list <- list(A = 1:3, B = 4:9, C = 1:9)
map(named_list, length) # output is always a list
```

```
## $A
## [1] 3
##
## $B
## [1] 6
##
## $C
## [1] 9
```

Specifying output type

```
map_int(named_list, length) # integer vector


## A B C
## 3 6 9

map_lgl(named_list, ~ length(.x) > 3) # logical vector


##     A     B     C
## FALSE  TRUE  TRUE

try(map_lgl(named_list, length)) # error


## Error : Can't coerce element 1 from a integer to a logical
```

Stitching `data.frame` output

Suppose we have a sample of size n drawn from distribution $\mathcal{N}(\mu = 3, \sigma = 2)$. To do inference on μ, we estimate via sample mean and quantify uncertainty via standard error. To see the difference that n makes:

```
sample_mean <- function(n) {
  x <- rnorm(n, mean = 3, sd = 2)
  return(list(n = n, est = mean(x), se = sd(x)/sqrt(n)))
}
map_dfr(c(10, 50, 100, 500), sample_mean)
```

n	est	se
10	4.09	0.528
50	2.72	0.333
100	2.90	0.180
500	2.96	0.088

Simulating fake data

```r
simulate_dataset <- function(N) {
  mu <- rnorm(1, -2.5, 0.3)
  alphas <- replicate(length(N), rnorm(1, mu, 0.2))
  beta <- rnorm(1, 0.16, 0.05)
  fake_data <- imap_dfr(N, ~ data.frame(
    wiki = .y,
    treatment = rbinom(.x, 1, 0.5)
  )) %>% dplyr::mutate(
    p = invlogit(alphas[wiki] + beta * treatment),
    y = map_int(p, ~ rbinom(1, 1, prob = .x)),
  )
  return(fake_data)
}
```

```
simulate_dataset(c(3, 3, 3))
```

wiki	treatment	p	y
1	0	0.100	0
1	1	0.117	1
1	1	0.117	0
2	0	0.118	1
2	0	0.118	1
2	1	0.138	0
3	1	0.144	0
3	0	0.123	0
3	0	0.123	1

Calculating power

```
estimate_power <- function(N, n_sims = 1000) {
  significant <- future_map_lgl(1:n_sims, function(i) {
    fit <- lme4::glmer(
      y ~ treatment + (1 | wiki),
      data = simulate_dataset(N), family = binomial()
    )
    beta_est <- fixef(fit)["treatment"]
    beta_se <- se.fixef(fit)["treatment"]
    return(beta_est - 2 * beta_se > 0) # H0: beta <= 0
  })
  return(mean(significant)) # Pr(reject H0 | H0 is false)
}
```

If want to calculate the probability of correctly rejecting

$$\mathrm{H}_0 : \beta \leq 0$$

in favor of

$$\mathrm{H}_a : \beta > 0$$

after running an experiment on 2 wikis on 2.5K and 1.7K users, respectively:

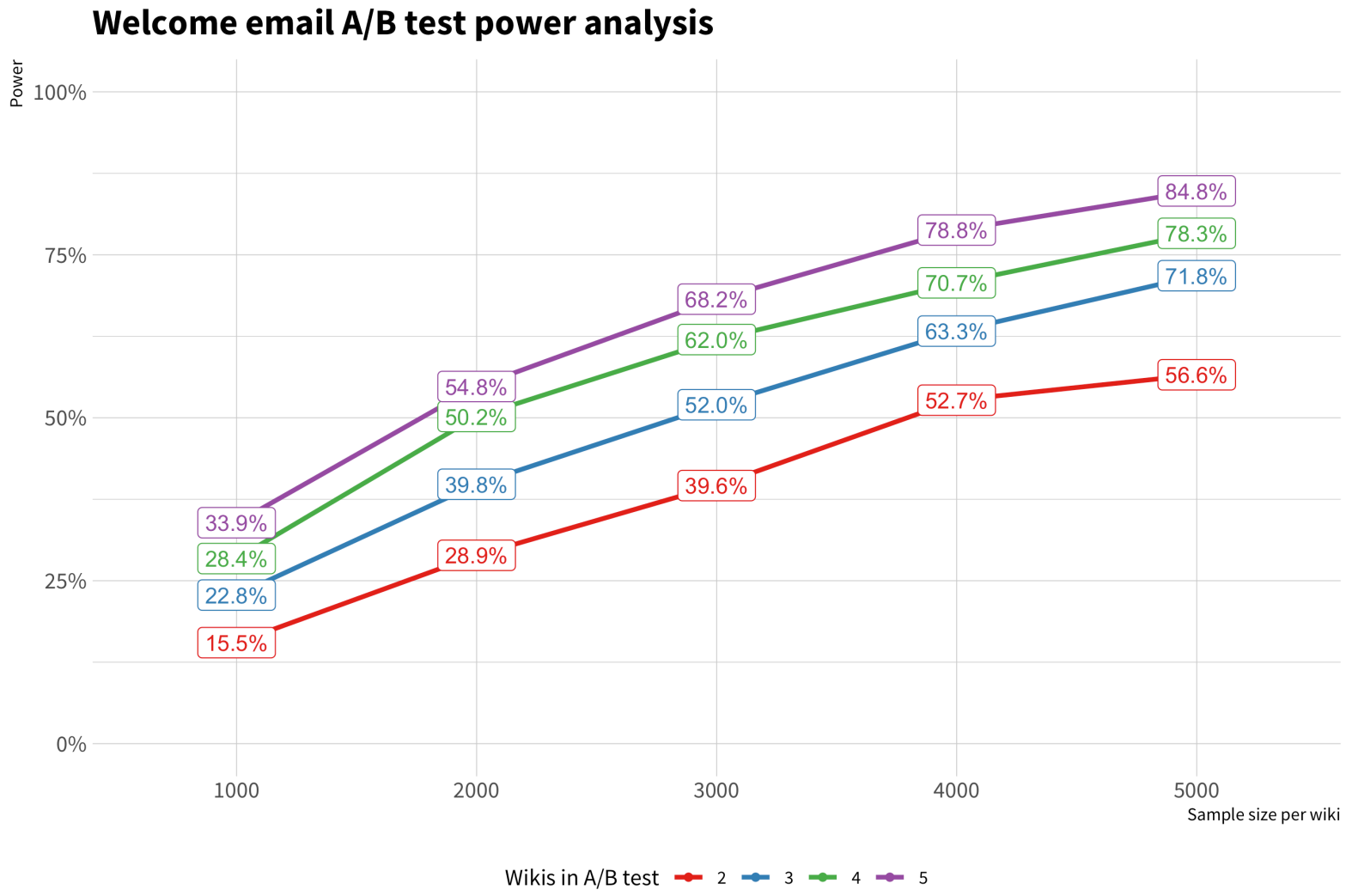
```
estimate_power(c(2500, 1700))
```

```
## [1] 0.317
```

Power analysis

```
sample_sizes <- expand.grid(
  users_per_wiki = seq(1e3, 5e3, 1e3),
  wikis = 2:5
)

power_analysis <- sample_sizes %>%
  dplyr::mutate(
    power = map2_dbl(users_per_wiki, wikis, function(N, J) {
      Ns <- rep(N, J) # e.g. rep(10, 2) creates c(10, 10)
      return(estimate_power(Ns))
    })
  )
```


Welcome email A/B test power analysis
Power
100%
75%
50%
25%
0%
1000
2000
3000
4000
5000
Sample size per wiki
84.8%
78.8%
68.2%
54.8%
33.9%
78.3%
70.7%
62.0%
50.2%
28.4%
71.8%
63.3%
52.0%
39.8%
22.8%
56.6%
52.7%
39.6%
28.9%
15.5%
Wikis in A/B test
2
3
4
5

Total sample sizes and power

Users per wiki	Wikis	Total users	Estimated power
5000	5	25000	84.8%
4000	5	20000	78.8%
5000	4	20000	78.3%
4000	4	16000	70.7%
5000	3	15000	71.8%
3000	5	15000	68.2%
4000	3	12000	63.3%
3000	4	12000	62.0%
5000	2	10000	56.6%
2000	5	10000	54.8%

Key takeaways

- Multilevel/hierarchical models when data has a nested structure
    - Repeated measures and pre/post-intervention measurements are nested within subject
    - You may have multiple levels of nesting
- Estimating power through simulation is actually relatively easy
    - The hard part is figuring out what effect sizes and standard deviations/errors to use
    - R packages like `purrr` (and `furrr`) make computations easier
- You get about the same power from 5K users from 2 wikis as you get from 2K users from 5 wikis
    - So if a large sample size on a per-wiki basis is difficult or expensive, testing on more wikis but with smaller sample sizes yields similar reliability

Resources on multilevel/hierarchical modeling

Python

- A Primer on Bayesian Methods for Multilevel Modeling (in PyMC3)
- Hierarchical GLM in PyMC3
- A Primer on Bayesian Multilevel Modeling using PyStan
- Comparing R lmer to Statsmodels MixedLM

R

- Advanced Bayesian Multilevel Modeling with the R Package brms (PDF)
- Bayesian Linear Mixed Models using Stan: A tutorial for psychologists, linguists, and cognitive scientists
- Introduction to multilevel modeling using rstanarm: A tutorial for education researchers
- Longitudinal mixed-effects models with lme4 (PDF, slides)