Hypothesis Tests Under Separation

# Reproduction tl;dr

The file install-packages.R (written dynamically below) will install all the packages I use for this project in the appropriate version. Please run if you want to use the same package versions as I did.

The file do-all.R reproduces all the tables and figures (though Table 3 requires a little editing to get the formatting right). The simulations run by R/02b-sims-do-random.R and R/02c-sims-summarize.R and take 1-2 days to complete. All other scripts run very quickly (i.e., seconds).

# Overview

Separation commonly occurs in political science, usually when a binary ex- planatory variable perfectly predicts a binary outcome. In these situations, methodologists often recommend penalized maximum likelihood or Bayesian estimation. But researchers might struggle to identify an appropriate penalty or prior distribution. Fortunately, I show that researchers can easily test hy- potheses about the model coefficients with standard frequentist tools. While the popular Wald test produces misleading (even nonsensical) p-values under separation, I show that likelihood ratio tests and score tests behave in the usual manner. Therefore, researchers can produce meaningful p-values with standard frequentist tools under separation without the use of penalties or prior information.

The latest draft is [here](doc/wilks.pdf) (conditionally accepted at *Political Analysis* pending their team’s successful reproduction of my results).

# Directory Structure

All the data and scripts necessary to reproduce these results are included in this repository.

I named files and directory so that their purpose can (hopefully) be understood from the name.

# Raw Data

The project uses one data set from previous research. This data set is included in this repository as data/politics\_and\_need\_rescale.csv.

* politics\_and\_need\_rescaled.csv comes from [Barrilleaux and Rainey (2014)](http://www.carlislerainey.com/papers/need.pdf) and their [replication files on Dataverse](https://dataverse.unc.edu/dataset.xhtml?persistentId=doi:10.15139/S3/12130).

# R Scripts

There are three categories of code

* 01-trinity-intuition.R reproduces Figure 1.
* 02\*-sims-\*.R together reproduce Table 1 and Figures 2-6. The \* here represents a variable, as there are several scripts that perform the simulations. They should be run starting with 02b, then 02c, and so on. This **quite a long time** and the file progress.log contains updates on the progress so you can estimate time-to-completion. By default, 02b-sims-do-random.R uses all available cores (12 in my case). Change this on line 105 of 02b-sims-do-random.R if you need to do other things while this code runs.
* 03-br-fits.R computes the information necessary for Table 3. The LaTeX code requires a bit of post-processing, but the information is all printed by this script, as well as an “almost finished” LaTeX table that requires only minor modifications.

The script do-all.R removes all generated files, re-generates everything, and compiles the manuscript and computational companion.

# Notable Output

* The file output/summarized-simulations.rds contains the power functions. This is the most important output from the simulations.
* The file output/all-generated-dgps-w-keep.rds contains all the generated DGPs and indicated which were discarded (as described in the paper) and which were kept.
* The directory output/scenario-sims/ contains all of the raw *p*-values for each hypothesis tests in the simulations. Each file is for a single DGP. (The script R/02c-sims-summarize.R aggregates this many tests into power functions and creates output/summarized-simulations.rds).

# Session Information

sessioninfo::session\_info(info = "platform")

## ─ Session info ─────────────────────────────────────────────────────────────────────────────────────────────────────────────────────  
## setting value  
## version R version 4.3.1 (2023-06-16)  
## os macOS Ventura 13.4  
## system aarch64, darwin20  
## ui RStudio  
## language (EN)  
## collate en\_US.UTF-8  
## ctype en\_US.UTF-8  
## tz America/New\_York  
## date 2023-08-31  
## rstudio 2023.06.1+524 Mountain Hydrangea (desktop)  
## pandoc 3.1.6.1 @ /usr/local/bin/ (via rmarkdown)  
##   
## ────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────

# System Information

# os info  
Sys.info()

## sysname   
## "Darwin"   
## release   
## "22.5.0"   
## version   
## "Darwin Kernel Version 22.5.0: Mon Apr 24 20:53:19 PDT 2023; root:xnu-8796.121.2~5/RELEASE\_ARM64\_T6020"   
## nodename   
## "wc-dhcp0d095.employee-secure.wireless.fsu.edu"   
## machine   
## "arm64"   
## login   
## "root"   
## user   
## "carlislerainey"   
## effective\_user   
## "carlislerainey"

# cpu info  
benchmarkme::get\_cpu()

## $vendor\_id  
## character(0)  
##   
## $model\_name  
## [1] "Apple M2 Max"  
##   
## $no\_of\_cores  
## [1] 12

# ram info  
ram\_info <- system("sysctl hw.memsize", intern = TRUE)  
print(ram\_info)

## [1] "hw.memsize: 34359738368"

# R Packages Used

library(tidyverse)  
  
# Function to get package version  
get\_version <- function(package\_names){  
 sapply(package\_names, function(pkg) as.character(packageVersion(pkg)))  
}  
  
dep <- renv::dependencies("R/") %>%  
 bind\_rows(renv::dependencies("README.Rmd")) %>%   
 select(Package) %>%  
 distinct() %>%  
 arrange(Package) %>%  
 mutate(Version = get\_version(Package))

## Finding R package dependencies ... Done!  
## Finding R package dependencies ... Done!

dep

## Package Version  
## 1 arm 1.13.1  
## 2 benchmarkme 1.0.8  
## 3 brglm 0.7.2  
## 4 broom 1.0.5  
## 5 doParallel 1.0.17  
## 6 doRNG 1.8.6  
## 7 dplyr 1.1.2  
## 8 foreach 1.5.2  
## 9 ggh4x 0.2.6  
## 10 ggrepel 0.9.3  
## 11 kableExtra 1.3.4  
## 12 latex2exp 0.9.6  
## 13 modelsummary 1.4.1  
## 14 mvtnorm 1.2.2  
## 15 patchwork 1.1.3  
## 16 progress 1.2.2  
## 17 renv 1.0.2  
## 18 rmarkdown 2.24  
## 19 scales 1.2.1  
## 20 sessioninfo 1.2.2  
## 21 tidyverse 2.0.0

# Write Script to Install Packages and Versions

# Assuming df is your dataframe with columns Package and Version  
  
# Create a new script file  
script\_file <- file("install-packages.R", "w")  
  
# Write install.packages() commands for each package to the file  
for (i in 1:nrow(dep)) {  
 install\_cmd <- sprintf("remotes::install\_version('%s', version = '%s', repos = 'http://cran.us.r-project.org')\n", dep$Package[i], dep$Version[i])  
 cat(install\_cmd, file = script\_file, append = TRUE)  
}  
  
# Close the file  
close(script\_file)