STAT 961: Midterm Exam
Fall 2021

Sam Rosenberg

Due Saturday, October 30 at 11:59pm; Time limit: 48 hours

1 Instructions

Setup. The materials you need for this exam are available here. Please navigate to this site and download the files you find there. Place `midterm-exam.Rnw` under `stat-961-fall-2021/midterm/` and `covid_data.tsv` under `stat-961-fall-2021/data/`.

Collaboration. This exam must be completed individually; seeking help from classmates or others is prohibited. You may only ask the instructor clarifying questions via private Piazza post. However, you may consult any course materials or the internet in completing this exam.

Please list what external references you consulted (e.g. articles, books, or websites):

Writeup. Use this document as a starting point for your writeup, adding your solutions between `\begin{sol}` and `\end{sol}`. See the preparing reports guide for guidance on compilation, creation of figures and tables, and presentation quality. Show all the code you wrote to produce your numerical results, and include complete derivations typeset in LaTeX for the mathematical questions.

Programming. The `tidyverse` paradigm for data manipulation (`dplyr`) and plotting (`ggplot2`) are strongly encouraged, but points will not be deducted for using base `R`.

```
library(tidyverse)
```

Grading. Each sub-part of each problem will be worth 3 points: 0 points for no solution or completely wrong solution; 1 point for some progress; 2 points for a mostly correct solution; 3 points for a complete and correct solution modulo small flaws. The presentation quality of the solution for each problem (as exemplified by the guidelines in Section 3 of the preparing reports guide) will be evaluated out of an additional 3 points.

Submission. Compile your writeup to PDF and submit to Gradescope.

Problem 1. The consequences of model bias.

To study the effect of a predictor x_{p-1} on a response y, we collect an observational dataset of n samples; for each sample we measure y, x_{p-1}, and $p-1$ possible confounders $x_0, x_1, \ldots, x_{p-2}$. We then postulate the linear model model

$$y = \beta_0 x_0 + \cdots + \beta_{p-2} x_{p-2} + \beta_{p-1} x_{p-1} + \epsilon, \quad \epsilon \sim N(0, \sigma^2), \tag{1}$$

based on which we construct $\widehat{\boldsymbol{\beta}}$ and $\widehat{\sigma}^2$, test $H_0 : \beta_{p-1} = 0$, and construct a confidence interval for β_{p-1} as in Unit 2. Unfortunately, we forgot about one confounder, x_p! It turns out that that x_{p-1} actually has no effect on y, and that the true distribution of the data is

$$y = \beta_0 x_0 + \cdots + \beta_{p-2} x_{p-2} + \beta_{p-1} x_{p-1} + \beta_p x_p + \epsilon, \quad \epsilon \sim N(0, \sigma^2), \quad \text{where } \beta_{p-1} = 0. \tag{2}$$

In this problem, we will investigate the consequences of this model bias. As usual, we view the predictors as fixed.

(a) What is the distribution of the least squares coefficient estimate $\widehat{\beta}_{p-1}$—defined based on the postulated linear model (1)—under the true data-generating model (2)? What is the bias of $\widehat{\beta}_{p-1}$?

(b) What is the expectation of the variance estimate $\widehat{\sigma}^2$—defined based on the postulated linear model (1)—under the true data-generating model (2)?

(c) What is the Type-I error of the right-sided level-α t-test of $H_0 : \beta_{p-1} = 0$—constructed based on the postulated model (1)—under the true data-generating model (2)? [For the sake of this question, you may ignore the sampling variability in $\widehat{\sigma}^2$ (i.e. assume $\widehat{\sigma}^2$ is always equal to its expectation) and approximate $t_{n-p} \approx N(0,1)$.]

(d) How do the bias found in part (a) and the Type-I error found in part (c) vary with β_p? Discuss the intuition for these results. [To discuss the dependency of the Type-I error on β_p, you may restrict your attention to $\beta_p \to \infty$.]

(e) Carry out the following numerical simulation to assess bias and Type-I error. Set $n = 100$, $p = 20$, $\sigma = 1$, $(\beta_0, \ldots, \beta_{p-1}) = \mathbf{0}$, $\beta_p \in \{0, 0.5, 1, \ldots, 4.5, 5\}$, and $\alpha = 0.05$. Draw $(x_{i,0}, \ldots, x_{i,p-1}, x_{i,p}) \overset{\text{i.i.d.}}{\sim} N(\mathbf{0}, \boldsymbol{\Sigma}(\rho))$ for $i = 1, \ldots, n$, where $\boldsymbol{\Sigma}(\rho)$ is the AR(1) covariance matrix with autocorrelation parameter $\rho \in \{0.05, 0.2\}$, i.e. $\boldsymbol{\Sigma}(\rho)_{j_1, j_2} = \rho^{|j_1 - j_2|}$ for $j_1, j_2 \in \{1, \ldots, p+1\}$. For each pair (β_p, ρ), compute the bias of $\widehat{\beta}_{p-1}$ computed with respect to the postulated model (1) and the Type-I error of the corresponding t-test, via 1000 draws of $\boldsymbol{y}$ based on the model (2), while keeping the predictors fixed. Plot the simulated bias and Type-I error as a function of β_p for each ρ, overlaying the theoretical predictions from parts (a) and (c), respectively. Add a dashed horizontal line on the Type-I error plot at the nominal level α. Comment on the agreement between the simulation and theoretical predictions, the shapes of the resulting curves, and how these connect to the discussion in part (d).

Solution 1.

(a) Say that the model matrix under the true model (2) is $\boldsymbol{X} = \begin{pmatrix} \boldsymbol{X}_{-p} & \boldsymbol{x}_{*p} \end{pmatrix}$ and the true coefficient vector is $\boldsymbol{\beta} = \begin{pmatrix} \boldsymbol{\beta}_{-p} \\ \beta_p \end{pmatrix}$. Under (1), we have that our estimator for $\boldsymbol{\beta}_{-p}$ is $\widehat{\boldsymbol{\beta}}_{-p} = (\boldsymbol{X}_{-p}^T \boldsymbol{X}_{-p})^{-1} \boldsymbol{X}_{-p}^T \boldsymbol{y}$. We also have that $\boldsymbol{y} = \boldsymbol{X}_{-p} \boldsymbol{\beta}_{-p} + \boldsymbol{x}_{*p} \beta_p + \boldsymbol{\epsilon}$.

Note that under (2), $y \sim N(\boldsymbol{X}\boldsymbol{\beta},\ \sigma^2\boldsymbol{I}_n)$ and $\boldsymbol{\epsilon} \sim N(0, \sigma^2\boldsymbol{I}_n)$. So,

$$\begin{aligned}
\widehat{\boldsymbol{\beta}}_{-p} &= (\boldsymbol{X}_{-p}^T\boldsymbol{X}_{-p})^{-1}\boldsymbol{X}_{-p}^T\boldsymbol{y} \\
&= (\boldsymbol{X}_{-p}^T\boldsymbol{X}_{-p})^{-1}\boldsymbol{X}_{-p}^T(\boldsymbol{X}_{-p}\boldsymbol{\beta}_{-p} + \boldsymbol{x}_{*p}\beta_p + \boldsymbol{\epsilon}) \\
&= \boldsymbol{\beta}_{-p} + (\boldsymbol{X}_{-p}^T\boldsymbol{X}_{-p})^{-1}\boldsymbol{X}_{-p}^T\boldsymbol{x}_{*p}\beta_p + (\boldsymbol{X}_{-p}^T\boldsymbol{X}_{-p})^{-1}\boldsymbol{X}_{-p}^T\boldsymbol{\epsilon} \\
&\sim N(\boldsymbol{\beta}_{-p} + (\boldsymbol{X}_{-p}^T\boldsymbol{X}_{-p})^{-1}\boldsymbol{X}_{-p}^T\boldsymbol{x}_{*p}\beta_p,\ \sigma^2[(\boldsymbol{X}_{-p}^T\boldsymbol{X}_{-p})^{-1}\boldsymbol{X}_{-p}^T][(\boldsymbol{X}_{-p}^T\boldsymbol{X}_{-p})^{-1}\boldsymbol{X}_{-p}^T]^T) \\
&= N(\boldsymbol{\beta}_{-p} + (\boldsymbol{X}_{-p}^T\boldsymbol{X}_{-p})^{-1}\boldsymbol{X}_{-p}^T\boldsymbol{x}_{*p}\beta_p,\ \sigma^2(\boldsymbol{X}_{-p}^T\boldsymbol{X}_{-p})^{-1}).
\end{aligned}$$

Then

$$\begin{aligned}
\widehat{\beta}_{p-1} &\sim N(\beta_{p-1} + \left[(\boldsymbol{X}_{-p}^T\boldsymbol{X}_{-p})^{-1}\boldsymbol{X}_{-p}^T\boldsymbol{x}_{*p}\beta_p\right]_{p-1},\ \sigma^2\left[(\boldsymbol{X}_{-p}^T\boldsymbol{X}_{-p})^{-1}\right]_{p-1,p-1}) \\
&= N\left(\beta_{p-1} + \left[(\boldsymbol{X}_{-p}^T\boldsymbol{X}_{-p})^{-1}\boldsymbol{X}_{-p}^T\boldsymbol{x}_{*p}\beta_p\right]_{p-1},\ \frac{\sigma^2}{s_{p-1}^2}\right) \\
&= N\left(\left[(\boldsymbol{X}_{-p}^T\boldsymbol{X}_{-p})^{-1}\boldsymbol{X}_{-p}^T\boldsymbol{x}_{*p}\beta_p\right]_{p-1},\ \frac{\sigma^2}{||\boldsymbol{x}_{*,p-1}^{\perp}||^2}\right),
\end{aligned}$$

where s_j^2 and $\boldsymbol{x}_{*,p-1}^{\perp}$ are both defined with respect to the presumed (incorrect) model (1) and our vector/matrix indexing is beginning at 0 for simplicity.

So, the question remains as to how we can simplify the quantity $\left[(\boldsymbol{X}_{-p}^T\boldsymbol{X}_{-p})^{-1}\boldsymbol{X}_{-p}^T\boldsymbol{x}_{*p}\beta_p\right]_{p-1}$. Note that $(\boldsymbol{X}_{-p}^T\boldsymbol{X}_{-p})^{-1}\boldsymbol{X}_{-p}^T\boldsymbol{x}_{*p}$ is the coefficient matrix of the least squares regression of $\boldsymbol{x}_{*p}$ onto $\boldsymbol{X}_{-p}$. Say that the j-th coefficient of this regression is ${}_p\gamma_j$, then we have that $\left[(\boldsymbol{X}_{-p}^T\boldsymbol{X}_{-p})^{-1}\boldsymbol{X}_{-p}^T\boldsymbol{x}_{*p}\beta_p\right]_{p-1} = {}_p\gamma_{p-1}\beta_p$ and

$$\beta_{p-1} \sim N({}_p\gamma_{p-1}\beta_p, \frac{\sigma^2}{||\boldsymbol{x}_{*,p-1}^{\perp}||^2}).$$

Recall that the bias of an estimator $\widehat{\theta}$ of θ is $\mathbb{E}[\widehat{\theta}] - \theta$. So,

$$\text{Bias}[\widehat{\beta}_{p-1}] = \mathbb{E}[\widehat{\beta}_{p-1}] = {}_p\gamma_{p-1}\beta_p.$$

(b) Denote the residuals using the estimator $\widehat{\boldsymbol{\beta}}_{-p}$ to be $\widehat{\boldsymbol{\epsilon}}_{-p} = \boldsymbol{y} - \boldsymbol{X}\widehat{\boldsymbol{\beta}}_{-p}$. Then the variance estimate $\widehat{\sigma}^2$ based on (1) is

$$\begin{aligned}
\widehat{\sigma}^2 &= \frac{1}{n-p}||\widehat{\boldsymbol{\epsilon}}_{-p}||^2 \\
&= \frac{1}{n-p}\widehat{\boldsymbol{\epsilon}}_{-p}^T\widehat{\boldsymbol{\epsilon}} \\
&= \frac{1}{n-p}[(\boldsymbol{I}_n - \boldsymbol{H}_{-p})\boldsymbol{y}]^T[(\boldsymbol{I}_n - \boldsymbol{H}_{-p})\boldsymbol{y}] \\
&= \frac{1}{n-p}\boldsymbol{y}^T(\boldsymbol{I}_n - \boldsymbol{H}_{-p})^T(\boldsymbol{I}_n - \boldsymbol{H}_{-p})\boldsymbol{y} \\
&= \frac{1}{n-p}\boldsymbol{y}^T(\boldsymbol{I}_n - \boldsymbol{H}_{-p})\boldsymbol{y},
\end{aligned}$$

where the last equality holds because $\boldsymbol{I} - \boldsymbol{H}_{-p}$ is a projection matrix so it is both symmetric and idempotent.

Note that the covariance matrix of $\boldsymbol{y}$ is $\sigma^2\boldsymbol{I}_n$ and its expectation is $\boldsymbol{X}\boldsymbol{\beta}$. So, we have

$$\begin{aligned}
\mathbb{E}[\widehat{\sigma}^2] &= \frac{1}{n-p}\mathbb{E}[\boldsymbol{y}^T(\boldsymbol{I}_n - \boldsymbol{H}_{-p})\boldsymbol{y}] \\
&= \frac{1}{n-p}\sum_{i,j=1}^{n}\mathbb{E}[y_i[\boldsymbol{I}_n - \boldsymbol{H}_{-p}]_{i,j}y_j] \\
&= \frac{1}{n-p}\sum_{i,j=1}^{n}[\boldsymbol{I}_n - \boldsymbol{H}_{-p}]_{i,j}\ \mathbb{E}[y_iy_j] \\
&= \frac{1}{n-p}\sum_{i,j=1}^{n}[\boldsymbol{I}_n - \boldsymbol{H}_{-p}]_{i,j}\ (\mathrm{Cov}(y_i,y_j) + \mathbb{E}[y_i]\mathbb{E}[y_j]) \quad \text{(covariance formula)} \\
&= \frac{1}{n-p}\sum_{i,j=1}^{n}[\boldsymbol{I}_n - \boldsymbol{H}_{-p}]_{i,j}\ ([\sigma^2\boldsymbol{I}_n]_{i,j} + (\boldsymbol{x}_{i*}\boldsymbol{\beta})(\boldsymbol{x}_{j*}\boldsymbol{\beta})) \\
&= \frac{1}{n-p}\sum_{i,j=1}^{n}[\boldsymbol{I}_n - \boldsymbol{H}_{-p}]_{i,j}[\sigma^2\boldsymbol{I}_n]_{i,j} + (\boldsymbol{x}_{i*}\boldsymbol{\beta})[\boldsymbol{I}_n - \boldsymbol{H}_{-p}]_{i,j}(\boldsymbol{x}_{j*}\boldsymbol{\beta}) \\
&= \frac{1}{n-p}\sum_{i,j=1}^{n}[\boldsymbol{I}_n - \boldsymbol{H}_{-p}]_{i,j}[\sigma^2\boldsymbol{I}_n]_{j,i} + (\boldsymbol{x}_{i*}\boldsymbol{\beta})[\boldsymbol{I}_n - \boldsymbol{H}_{-p}]_{i,j}(\boldsymbol{x}_{j*}\boldsymbol{\beta}) \quad (\sigma^2\boldsymbol{I}_n \text{ symmetric}) \\
&= \frac{1}{n-p}\sum_{i=1}^{n}[(\boldsymbol{I}_n - \boldsymbol{H}_{-p})\sigma^2\boldsymbol{I}_n]_{i,i} + (\boldsymbol{X}\boldsymbol{\beta})^T(\boldsymbol{I} - \boldsymbol{H}_{-p})(\boldsymbol{X}\boldsymbol{\beta}) \\
&= \frac{1}{n-p}[\mathrm{Tr}[\sigma^2(\boldsymbol{I}_n - \boldsymbol{H}_{-p})] + (\boldsymbol{X}\boldsymbol{\beta})^T(\boldsymbol{I} - \boldsymbol{H}_{-p})(\boldsymbol{X}\boldsymbol{\beta})] \\
&= \frac{1}{n-p}[\sigma^2\mathrm{Tr}[\boldsymbol{I}_n - \boldsymbol{H}_{-p}] + (\boldsymbol{X}\boldsymbol{\beta})^T(\boldsymbol{I} - \boldsymbol{H}_{-p})(\boldsymbol{X}\boldsymbol{\beta})] \\
&= \frac{1}{n-p}[\sigma^2\mathrm{Tr}[\boldsymbol{H}^{\perp}_{-p}] + (\boldsymbol{X}\boldsymbol{\beta})^T(\boldsymbol{H}^{\perp}_{-p})(\boldsymbol{X}\boldsymbol{\beta})] \\
&= \frac{1}{n-p}[\sigma^2\mathrm{Tr}[\boldsymbol{H}^{\perp}_{-p}] + (\boldsymbol{H}\boldsymbol{y})^T(\boldsymbol{H}^{\perp}_{-p})\boldsymbol{H}\boldsymbol{y}] \\
&= \frac{1}{n-p}[\sigma^2\mathrm{Tr}[\boldsymbol{H}^{\perp}_{-p}] + \boldsymbol{y}^T\boldsymbol{H}^T(\boldsymbol{H}^{\perp}_{-p})\boldsymbol{H}\boldsymbol{y}] \\
&= \frac{1}{n-p}[\sigma^2\mathrm{Tr}[\boldsymbol{H}^{\perp}_{-p}] + \boldsymbol{y}^T\boldsymbol{H}(\boldsymbol{H}^{\perp}_{-p})\boldsymbol{H}\boldsymbol{y}] \\
&= \sigma^2 + \frac{\boldsymbol{y}^T\boldsymbol{H}(\boldsymbol{H}^{\perp}_{-p})\boldsymbol{H}\boldsymbol{y}}{n-p} \quad (\text{since } \mathrm{Tr}[\boldsymbol{H}^{\perp}_{-p}] = \mathrm{Dim}(C(\boldsymbol{H}^{\perp}_{-p}))).
\end{aligned}$$

(c) Under (1), we have that the test statistic for the right-sided level-α t-test for the hypothesis $H_0 : \beta_{p-1} = 0$ is $t_{p-1} = \frac{\widehat{\beta}_{p-1}}{\widehat{\sigma}/s_j} = \frac{\widehat{\beta}_{p-1}}{\widehat{\sigma}/||\boldsymbol{x}^{\perp}_{*p-1}||}$. Recalling from (a) that $\beta_{p-1} \sim N(_p\gamma_{p-1}\beta_p, \frac{\sigma^2}{||\boldsymbol{x}^{\perp}_{*,p-1}||^2})$ and assuming that $\widehat{\sigma}^2 = \sigma^2$, we have that

$$t_{p-1} \sim N\left(\frac{||\boldsymbol{x}^{\perp}_{*p-1}||}{\sigma}{}_p\gamma_{p-1}\beta_p, 1\right).$$

That is, we have a mean-shift in the distribution of the test statistic by $\frac{||\boldsymbol{x}^{\perp}_{*p-1}||}{\sigma}{}_p\gamma_{p-1}\beta_p$.

Under the assumption of (1) and approximating a t-distribution with $n-p$ degrees of freedom as being $N(0,1)$, we have that the Type-I error for the t-test for $H_0 : \beta_{p-1} = 0$ is $\mathbb{P}[t_{p-1} \leq z_{1-\alpha}]$,

where $z_{1-\alpha}$ is the $1-\alpha$ quantile of a standard normal distribution (i.e. $\mathbb{P}[N(0,1) \leq z_{1-\alpha}] = 1-\alpha$). Substituting in the distribution of t_{p-1}, we have that

$$\begin{aligned}
\text{Type-I error} &= \mathbb{P}[t_{p-1} \geq z_{1-\alpha}] \\
&= \mathbb{P}\left[N\left(\frac{||\boldsymbol{x}_{*p-1}^{\perp}||}{\sigma} {}_p\gamma_{p-1}\beta_p, 1\right) \geq z_{1-\alpha}\right] \\
&= \mathbb{P}\left[N(0,1) \geq z_{1-\alpha} - \frac{||\boldsymbol{x}_{*p-1}^{\perp}||}{\sigma} {}_p\gamma_{p-1}\beta_p\right] \\
&= \mathbb{P}\left[N(0,1) \geq z_{1-\alpha} - t_{p-1}\frac{{}_p\gamma_{p-1}}{\widehat{\beta}_{p-1}}\beta_p\right].
\end{aligned}$$

We will use the final equation for Type-I error in the simulation for (e).

(d) We have that $\text{Bias}[\widehat{\beta}_{p-1}]$ varies linearly with β_p.

Note that $\frac{||\boldsymbol{x}_{*p-1}^{\perp}||}{\sigma} > 0$ as both the numerator and denominator are positive. So as $\beta_p \to \infty$, $z_{1-\alpha} - \frac{||\boldsymbol{x}_{*p-1}^{\perp}||}{\sigma} {}_p\gamma_{p-1}\beta_p$ tends toward $-\infty$ if ${}_p\gamma_{p-1} > 0$, ∞ if ${}_p\gamma_{p-1} < 0$, and $z_{1-\alpha}$ if ${}_p\gamma_{p-1} = 0$. Then if ${}_p\gamma_{p-1} > 0$, Type-I error $\to 1$; if ${}_p\gamma_{p-1} < 0$, Type-I error $\to 0$; and if ${}_p\gamma_{p-1} = 0$, Type-I error $= \alpha$ as $\beta_p \to \infty$.

Intuitively, as the absolute effect size of the omitted variable tends toward infinity, the magnitude of the bias on our estimated coefficient likewise tends to infinity. We can interpret this as saying that omitting larger effects causes more bias (and likewise omitting smaller effects causes less bias).

Because ${}_p\gamma_{p-1}$ is the projection of $\boldsymbol{x}_{*p}$ onto $C(\boldsymbol{X}_{-p})$, we intuitively have that the exclusion of β_p has no impact on our Type-I error when x_{*p} is uncorrelated with (equivalently, is orthogonal to) $C(\boldsymbol{X}_{-p})$. When ${}_p\gamma_{p-1} > 0$ and $\beta_p \to \infty$, we have that the mean of t_{p-1} is shifted right towards ∞, so the Type-I error increases toward 1. Likewise when ${}_p\gamma_{p-1} < 0$ and $\beta_p \to \infty$, the mean of t_{p-1} is shifted left toward $-\infty$ and so the Type-I error decreases toward 0.

(e)
```r
### Package management
library(pacman)
p_load("MASS")
p_load("latex2exp")


### Set simulation parameters
# n: number of data points
n <- 100

# p: number of predictors in postulated model
p <- 20

# sigma: standard error
sigma <- 1

# beta: vector of coefficients for postulated model (beta_0, ..., beta_{p-1})
beta <- rep(0, p)

# beta_p: vector of test values for \beta_p
```

```r
beta_p <- seq(from=0, to=5, by=0.5)

# alpha: significance level for hypothesis test under postulated model
alpha <- 0.05

# rho: vector of autocorrelation parameters for the AR(1) covariance matrix
rho <- c(0.05, 0.2)

# num_draws: number of draws of y for each round of simulation
num_draws <- 1000


# Function for computing AR(1) covariance matrix as a function of rho, p
get_ar_1_cov <- function(rho, p){
  # Sigma_{i,j} = rho^|i-j|
  return(rho^abs(outer(1:(p+1), 1:(p+1), "-")))
}


# Data frame for storing simulation results
sim_results <- data.frame()


### Run simulation for each value of rho, beta_p
for(rho_sim in rho){
  for(beta_p_sim in beta_p){
    # Get AR(1) covariance matrix Sigma(rho) for simulation value of rho, p
    Sigma_sim <- get_ar_1_cov(rho_sim, p)

    # Generate n x iid from N(0_{p+1}, Sigma(rho))
    X <- mvrnorm(n=n, mu=rep(0, p+1), Sigma=Sigma_sim)

    # Create model dataframe
    model_df <- as.data.frame(X)
    colnames(model_df) <- paste0("x", (1:ncol(model_df))-1)

    # Full model beta vector
    beta_full <- c(beta, beta_p_sim)

    # Compute X beta_full
    X_beta_full <- X %*% beta_full


    # Get matrix for postulated model
    X_minusp <- X[, 1:p]

    # Compute coefficient matrix for auxillary regression of x_{p} on X_{-p}
```

```r
p_gamma <- solve(t(X_minusp) %*% X_minusp) %*% t(X_minusp) %*% X[, p+1] %*% beta_p_sim

# Extract x_{p-1} coefficient from auxillary regression
p_gamma_p_1 <- p_gamma[p]


# Total error from all draws of y
# Simulated bias = (total error)/(number draws of y)
tot_error <- 0

# Number of times that approximate t-statistic exceeds the 1-alpha quantile
# of the distribution under the postulated model
# Simulated Type-I error is this count divided by number draws of y
ct_exceeds <- 0

# Vector to store Type-I error for each draw of y
type_i_errors <- c()

for(i in 1:num_draws){
#for(i in 1:1){
  # Sample epsilon_i iid from N(0, sigma^2)
  epsilon <- rnorm(n=n, mean=0, sd=sigma)

  # Compute y = X beta_full + epsilon
  y <- X_beta_full + epsilon

  # Add y to model dataframe
  model_df$y <- y


  # Postulated model formula includes all variables except xp and does *not*
  # include an intercept
  post_form <- formula(paste0("y ~ . -1 -x",p))

  # Run linear regression on postulated model
  lm_sim <- lm(post_form, data=model_df)

  # Extract regression coefficients
  beta_full_hat <- lm_sim$coefficients

  # Extract \widehat{beta_{p-1}}
  beta_hat_p_1 <- beta_full_hat[p]

  # Calculate difference between true and estimated beta_{p-1}
  beta_p_1_error <- beta_hat_p_1 - beta_p_sim

  # Add error to total error
```

```
    tot_error <- tot_error + beta_p_1_error


    # Extract t-statistic t_{p-1} from model fit
    t_p_1 <- summary(lm_sim)$coefficients[p, "t value"]

    # Get quantile for significance test at level alpha
    z_1_alpha <- qnorm(1-alpha)

    # Add to count if t_{p-1} >= z_{1-alpha}
    ct_exceeds <- ct_exceeds + 1 * (t_p_1 >= z_1_alpha)

    # Compute actual Type-I error
    type_i_error <-
      pnorm(q = z_1_alpha - t_p_1/beta_hat_p_1 * p_gamma_p_1 * beta_p_sim, lower.tail=FALSE)

    # Add computed Type-I error to list
    type_i_errors <- c(type_i_errors, type_i_error)
  }


  ## Compute true bias based on 1(a)
  # Bias vector is (X_{-p}^T X_{-p})^{-1} X_{-p}^T x_{*p} beta_p
  bias_vector =
    solve(t(X_minusp) %*% X_minusp) %*%
    t(X_minusp) %*%
    X[, p+1] %*%
    beta_p_sim

  bias_beta_p_1 <- bias_vector[p]


  # Compute simulated bias
  sim_bias <- tot_error/num_draws

  # Compute simulated Type-I error
  sim_type_i <- ct_exceeds/num_draws


  # Store simulation result in a dataframe
  sim_result <-
    data.frame(
      beta_p=beta_p_sim,
      rho=rho_sim,
      simulated_bias=sim_bias,
      theoretical_bias=bias_beta_p_1,
      simulated_type_i=sim_type_i,
```

```
        theoretical_type_i=mean(type_i_errors))

    # Add this simulation;s result to overall list of results
    sim_results <- rbind(sim_results, sim_result)
  }
}

# Convert simulation results to long format
sim_results_long_bias <-
  sim_results %>%
    dplyr::select(
      -simulated_type_i,
      -theoretical_type_i) %>%
    pivot_longer(
      c(simulated_bias, theoretical_bias),
      names_to="is_sim",
      values_to="bias") %>%
    mutate(is_sim = str_replace(is_sim, "_.*", "")) %>%
    mutate(group = paste0(rho, paste0(" (", paste0(is_sim, ")")))) %>%
    dplyr::select(-is_sim)
sim_results_long_type_i <-
  sim_results %>%
    dplyr::select(
      -simulated_bias,
      -theoretical_bias) %>%
    pivot_longer(
      c(simulated_type_i, theoretical_type_i),
      names_to="is_sim",
      values_to="type_i") %>%
    mutate(is_sim = str_replace(is_sim, "_.*", "")) %>%
    mutate(group = paste0(rho, paste0(" (", paste0(is_sim, ")")))) %>%
    dplyr::select(-is_sim)


### Create plots
# Plot of simulated, theoretical bias as a function of beta_p
bias_plt <-
  sim_results_long_bias %>%
    ggplot() +
    # Add scatter plot of simulated bias
    geom_point(aes(x = beta_p, y = bias, color=group)) +
    ylab(TeX("Bias")) +
    xlab(TeX("$\\beta_p$")) +
    labs(color=TeX("$\\rho$"))

ggsave("./figures/bias_plt.png", bias_plt,
       width=6, height=4)
```

```
# Plot of simulated, theoretical Type-I error as a function of beta_p
type_i_plt <-
  sim_results_long_type_i %>%
    ggplot() +
    # Add scatter plot of simulated bias
    geom_point(aes(x = beta_p, y = type_i, color=group)) +
    # Add line for nomial level alpha
    geom_hline(yintercept=alpha, linetype="dashed", color="red") +
    ylab(TeX("Type-I error")) +
    xlab(TeX("$\\beta_p$")) +
    labs(color=TeX("$\\rho$"))

ggsave("./figures/type_i_plt.png", type_i_plt,
       width=6, height=4)
```

Note that in the bias plot, we have close agreement between the simulated and theoretical predictions. The curve is roughly fan-shaped, with increasing deviations from a line as β_p increases. This is in accordance with (a), which said that the bias was a linear function of β_p.

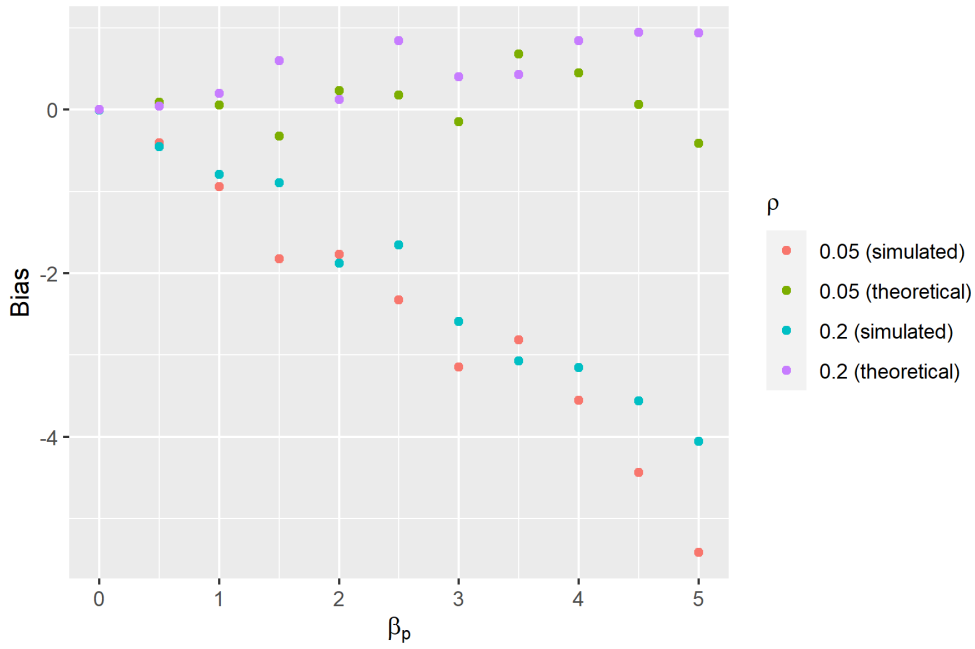

Figure 1: Simulated and theoretical bias vs. β_p ($n = 1,000$ draws).

In the Type-I error plot, we have that for small ρ, the simulated and theoretical values are close to one another and generally center around the nominal level α, denoted by the dashed red line. When ρ is larger, we have that the Type-I error increases drastically as a function of β_p, increasing steeply and then leveling off as it approaches 1. There is more of a discrepancy between the simulated and theoretical values for the Type-I error when ρ is larger.

Intuitively, the more correlated our omitted predictor is with the other predictors, the more drastic of an effect its omission has on our simulated Type-I error. So, a larger value of ρ (meaning

higher correlation between the variables) leads to a greater deviation from the true Type-I error.

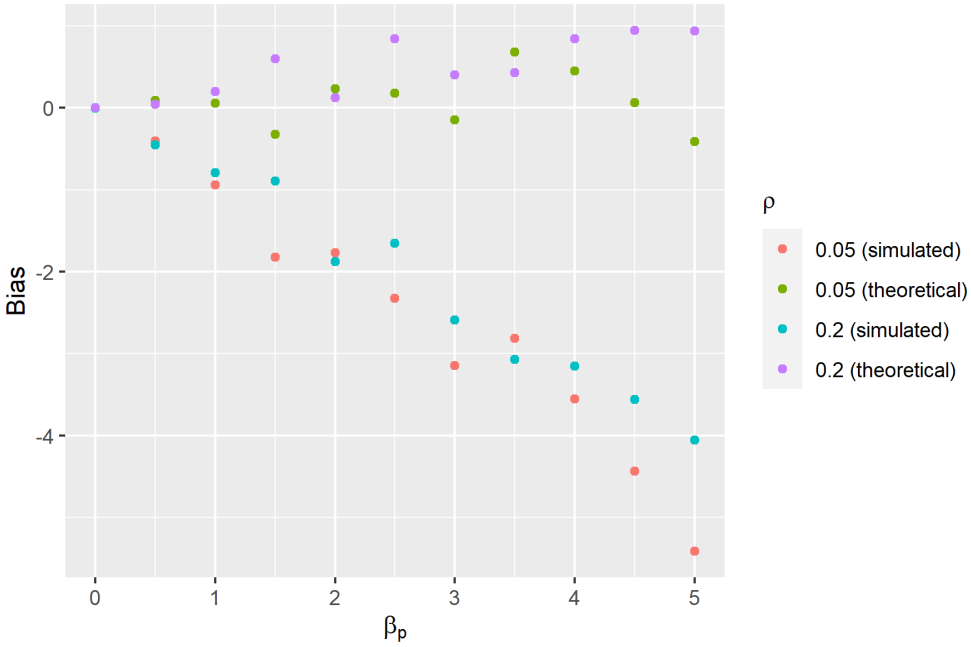

Figure 2: Simulated and theoretical bias vs. β_p ($n = 1,000$ draws).

Problem 2. Case study: Determinants of COVID case-fatality rate.

The coronavirus pandemic has had a disparate impact on different communities across the United States. A key measure of this impact is the *case-fatality rate*, defined as the ratio of the number of deaths to the number of cases, expressed as a percentage. The goal of the present analysis is to study the relationship between the case-fatality rate and a variety of health and socioeconomic factors at the county level in the year 2020, before vaccines became widely available.

To this end, we are given `covid_data.tsv`, compiled from case and death tracking data from The New York Times and 41 county-level health and socioeconomic factors compiled by the County Health Rankings and Roadmaps. Descriptions of these 41 socioeconomic factors are given in Appendix A. The data contain 935 counties out of about 3000 total in the US, for which the health and socioeconomic factors were available.

```
covid_data = read_tsv("../data/covid_data.tsv")
print(covid_data, n = 3)

## # A tibble: 935 x 44
##   county  state   case_fatality_rate low_birthweight_percentage food_environment
##   <chr>   <chr>                <dbl>                      <dbl>            <dbl>
## 1 Baldwin Alabama               1.10                     0.0835              8
## 2 Barbour Alabama               1.16                     0.115               5.6
## 3 Blount  Alabama               1.10                     0.0760              8.4
## # ... with 932 more rows, and 39 more variables:
## #   physical_exercise_opportunities <dbl>, teen_births <dbl>,
## #   limited_healthy_access <dbl>, stis <dbl>, uninsured <dbl>,
## #   primarycare_ratio <dbl>, dentist_ratio <dbl>, mentalhealth_ratio <dbl>,
## #   otherproviders_ratio <dbl>, HS_completion <dbl>, some_college <dbl>,
## #   disconnected_youth <dbl>, unemployment <dbl>, income_inequality <dbl>,
## #   children_freelunches <dbl>, single_parent_households <dbl>, ...
```

(a) Run a linear regression of `case_fatality_rate` on the 41 given predictors. What fraction of the variation in the response is explained by the predictors? Print a table containing the features whose t-test p-values pass the multiplicity-adjusted threshold of $\alpha' = 0.05/41 \approx 0.0012$, for each feature displaying the coefficient estimate, standard error, and p-value.

(b) Create the residuals-versus-fitted-values and residuals-versus-leverage diagnostic plots. Are there any apparent concerns regarding the independence and homoskedasticity assumptions? Are there any apparent outliers?

(c) To further probe the independence assumption, visualize the distributions of the standardized residuals grouped by state. [Hint: Use a box plot, with states on the vertical axis.] Are any departures from independence apparent in this plot? To assess statistically whether `state` is associated with `case_fatality_rate`, run a heteroskedasticity-robust test to determine whether the model with an indicator for state fits significantly better than the model run in part (a). What do you conclude?

(d) The effect of the `state` variable can be accounted for using two different robust analyses: (1) based on the linear regression in part (a) but with Liang-Zeger standard errors, clustering by `state` and (2) based on the linear regression in part (a) but with `state` as an additional predictor and with Huber-White standard errors. For both of these methods, print tables

containing the features whose t-test p-values pass the multiplicity-adjusted threshold of $\alpha' = 0.05/41 \approx 0.0012$, for each feature displaying the coefficient estimate, standard error, and p-value.

(e) Discuss the pros and cons of the two analyses done in part (d). In what situations would analysis (1) be more appropriate, and in what situations would analysis (2) be more appropriate? Which analysis leads to greater standard error inflation? To address the latter question, for each robust analysis produce a histogram (across features) of the factor by which the standard error exceeds that obtained from the analysis in part (a). On the whole, which analysis would you recommend for this problem?

Solution 2.

(a)
```r
# Create linear model
lm_full <-
  lm(case_fatality_rate ~ . - county - state, covid_data)

# Get dataframe with coefficient summaries
feat_summary <- summary(lm_full)$coefficients

# Get subset of summary for those coefficients that are significant at the
# multiple testing adjusted threshold
sig_feat_summary <-
  feat_summary[feat_summary[, "Pr(>|t|)"] <= 0.05/41, ]

rownames(sig_feat_summary) <-
  str_replace_all(rownames(sig_feat_summary), "_", "\\\\_")

# Create table with summary
sig_feat_summary %>%
  kableExtra::kable(
    format = "latex", booktabs = TRUE,
    digits = 4, escape=FALSE) %>%
  kableExtra::save_kable("figures/sig_feat_tbl.png")
```

The proportion of variation in the response explained by the predictors is the R^2, which is 33.12%.

	Estimate	Std. Error	t value	Pr(>|t|)
disconnected_youth	5.9482	1.5481	3.8422	1e-04
unemployment	24.4133	4.7775	5.1101	0e+00
housing_overcrowding	-13.6281	3.9905	-3.4152	7e-04
segregation_nonwhite_white	0.0264	0.0054	4.9260	0e+00
inactive_perc	6.6707	1.4234	4.6864	0e+00
flu_vaccine_perc	3.9323	0.8936	4.4005	0e+00
median_income	0.0000	0.0000	3.7249	2e-04

Table 1: Summary of features significant with multiple testing corrections.

The details of the coefficients that are significant at $\alpha = 0.05$ after a multiple testing correction are shown in Table 1.

(b)
```r
png("figures/resid_fitted_plt.png")
plot(lm_full, which=1)
dev.off()

## pdf
##   2

png("figures/resid_leverage_plt.png")
plot(lm_full, which=5)
dev.off()

## pdf
##   2
```

Looking at the plot of residuals vs. fitted values we can see a fan shape, indicating the presence of heteroskedasticity or correlation between our errors.

There do not seem to be any point with unduly high Cook's distance in the plot of the residuals against leverage, indicating that there are not any apparent outliers.

(c)
```r
# Extract standardized residuals
std_res <- stdres(lm_full)

df_std_res <- data.frame(State=covid_data$state, std_res=std_res)

std_res_state_plt <-
  df_std_res %>%
    ggplot() +
      geom_boxplot(aes(x=std_res, color=State)) +
      xlab("Standardized residuals")

ggsave("figures/std_res_state_plt.png", std_res_state_plt,
       width=12, height=8)
```

Looking at the distribution of standardized residuals by state, the fact that different states have different widths of the whiskers for their respective plots suggests that the variance within each state; i.e. we may have a grouped correlation structure.

```r
# Load sandwich package
p_load("sandwich")

# Load lmtest package
p_load("lmtest")

# Create a model that includes state fixed effect
lm_w_state <-
  lm(case_fatality_rate ~ . - county, covid_data)
```

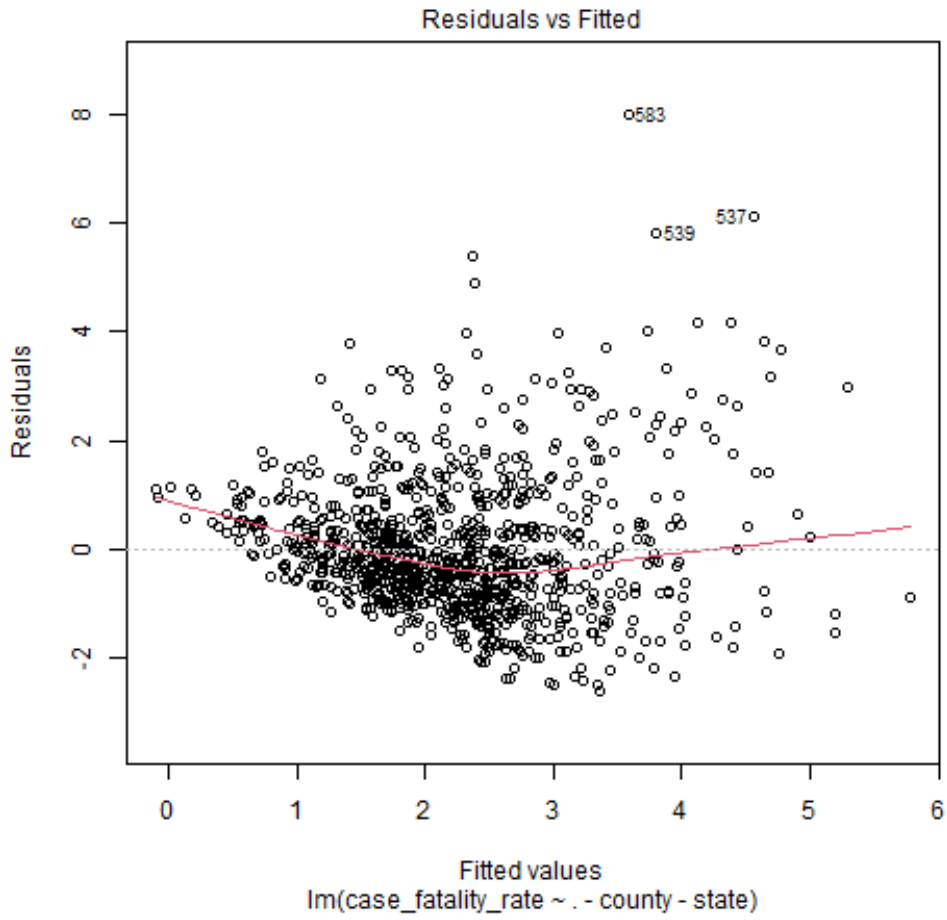

Figure 3: Residuals vs. fitted values

```r
# Extract coefficient estimates
beta_hat <- lm_w_state$coefficients

# Get Liang-Zeger covariance estimate
lz_sigma_hat <- vcovCL(lm_w_state)

# Robust F-test
robust_f_test <-
  waldtest(lm_full, lm_w_state, vcov=lz_sigma_hat) %>%
    as.matrix() %>%
    data.frame()

# Fix column names
colnames(robust_f_test) <-
  c("Residual DF", "DF", "F", "Pr(>F)")
```

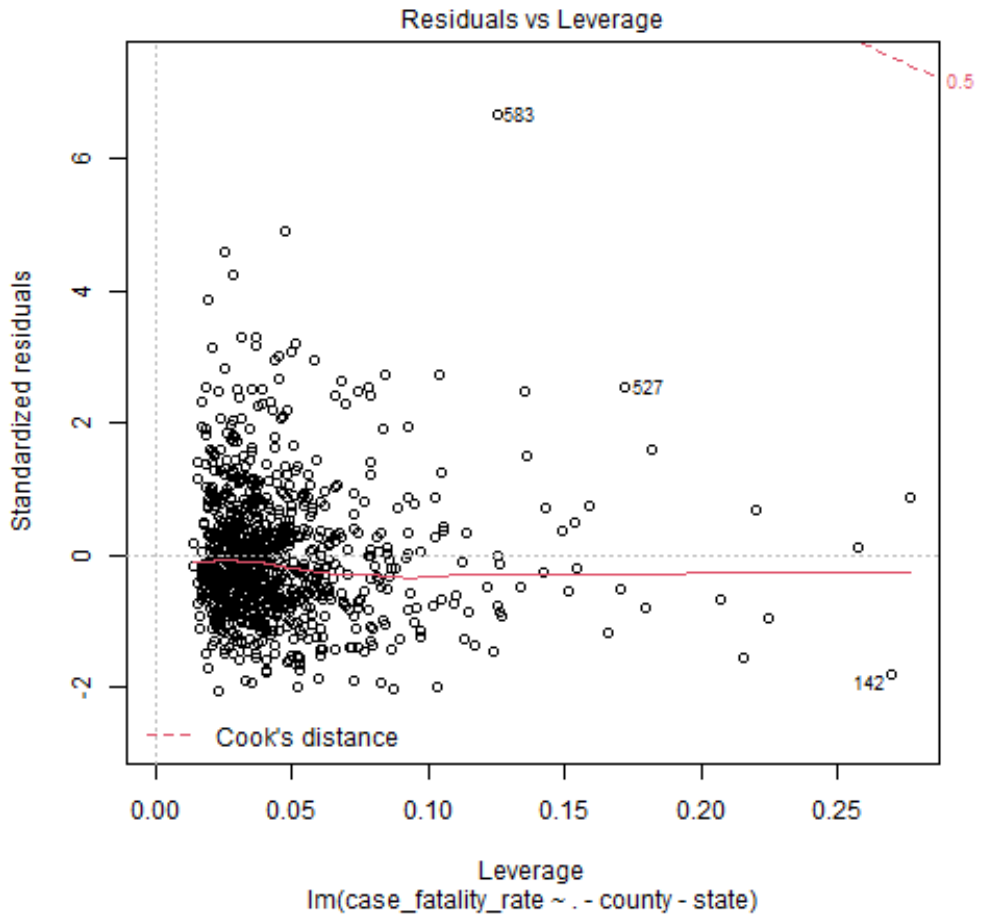

Figure 4: Residuals vs. leverage.

```
# Create table with summary
robust_f_test %>%
  mutate(
    `Residual DF` = as.character(signif(`Residual DF`, 4)),
    DF = as.character(signif(DF, 4)),
    F = as.character(signif(F, 4)),
    `Pr(>F)` = as.character(signif(`Pr(>F)`, 4))) %>%
  kableExtra::kable(
    format = "latex", booktabs = TRUE,
    digits = 4, escape=FALSE) %>%
  kableExtra::save_kable("figures/robust_f_test.png")
```

Note that the p-value from the robust F-test is extremely small, indicating that the inclusion of state significantly improves the model's fit.

(d) To account for the effect of the state variable, we first perform robust t-tests with multiple testing corrections using LZ standard errors, with clusters defined by state and variable descriptions

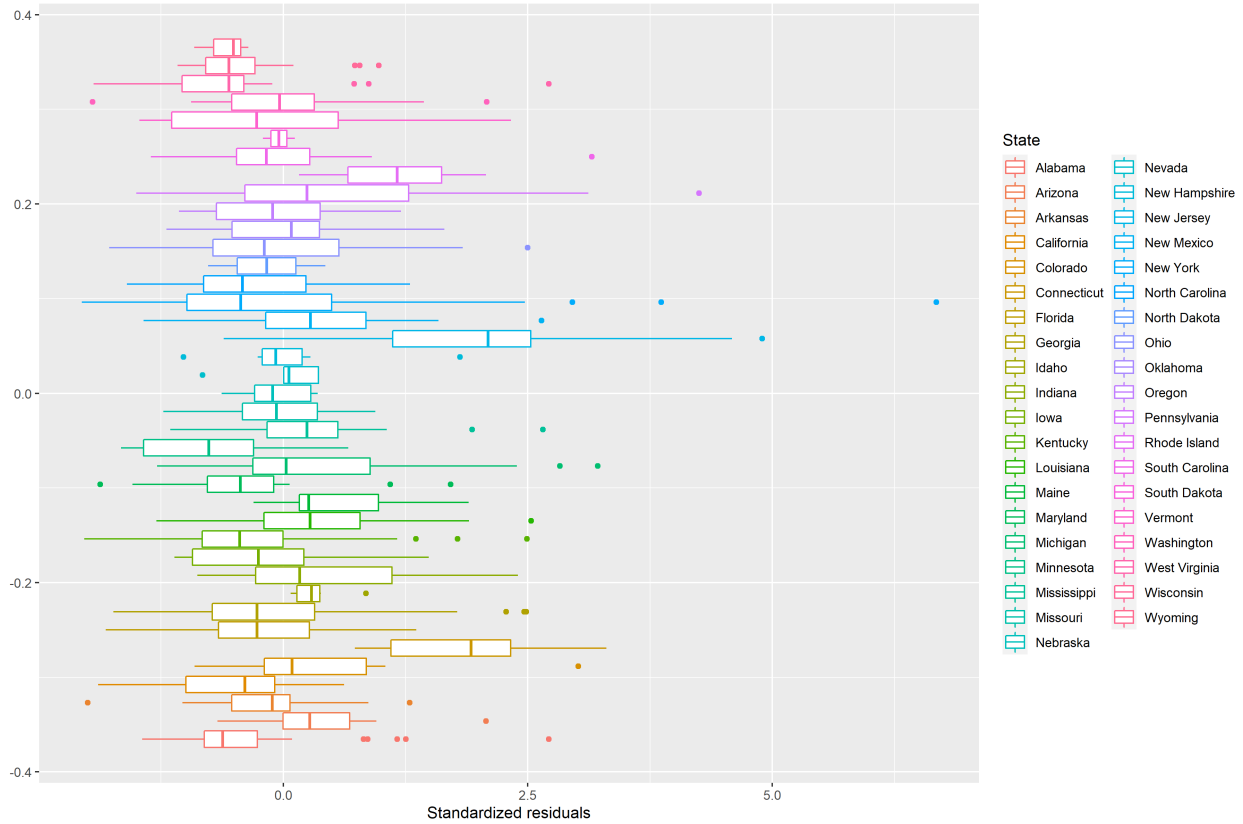

Figure 5: Distribution of standardized residuals by state.

Residual DF	DF	F	Pr(>F)
893	NA	NA	NA
855	38	9.73	4.812e-45

Table 2: Robust F-test summary.

provided in the Appendix.

```
library(broom)

# Robust t-test using LZ standard errors, clustering by state
robust_t_lz <- coeftest(lm_full, vcov.=lz_sigma_hat)

# Only get coefficients significant at alpha=0.05 with multiple testing
# correction
robust_t_lz_sig <-
  robust_t_lz %>%
    tidy() %>%
    filter(`p.value` <= 0.05/41)
```

```
colnames(robust_t_lz_sig) <-
  c("Covariate", "Estimate", "SE", "Statistic", "$p$-value")

robust_t_lz_tbl <-
  robust_t_lz_sig %>%
    mutate(
      Covariate = str_replace_all(Covariate, "_", "\\\\_"),
      Estimate = as.character(signif(Estimate, 4)),
      SE = as.character(signif(SE, 4)),
      Statistic = as.character(signif(Statistic, 4)),
      `$p$-value` = as.character(signif(`$p$-value`, 4)))

robust_t_lz_tbl %>%
  kableExtra::kable(
    format = "latex", booktabs = TRUE,
    digits = 4, escape=FALSE) %>%
  kableExtra::save_kable("figures/robust_t_test_lz.png")
```

Covariate	Estimate	SE	Statistic	p-value
(Intercept)	-23	6.837	-3.364	0.0008007
disconnected_youth	5.948	1.691	3.518	0.000457
unemployment	24.41	3.788	6.444	1.898e-10
housing_overcrowding	-13.63	3.753	-3.631	0.000298
segregation_nonwhite_white	0.02639	0.005519	4.782	2.029e-06
inactive_perc	6.671	1.405	4.748	2.387e-06
flu_vaccine_perc	3.932	0.8764	4.487	8.166e-06
median_income	2.783e-05	6.909e-06	4.029	6.085e-05

Table 3: Robust t-test with LZ SEs summary.

The results of the robust t-test from the first analysis are provided in Table 3.

Our second robust analysis will use HW standard errors obtained from the model that includes state as a predictor, then uses a robust t-test.

```
# Get HW covariance estimate
hw_sigma_hat <- vcovHC(lm_w_state)

# Robust t-test using HW standard errors, model includes state
robust_t_hw <- coeftest(lm_full, vcov.=hw_sigma_hat)

# Only get coefficients significant at alpha=0.05 with multiple testing
# correction
robust_t_hw_sig <-
  robust_t_hw %>%
    tidy() %>%
    filter(`p.value` <= 0.05/41)
```

```
colnames(robust_t_hw_sig) <-
  c("Covariate", "Estimate", "SE", "Statistic", "$p$-value")

robust_t_hw_tbl <-
  robust_t_hw_sig %>%
    mutate(
      Covariate = str_replace_all(Covariate, "_", "\\\\_"),
      Estimate = as.character(signif(Estimate, 4)),
      SE = as.character(signif(SE, 4)),
      Statistic = as.character(signif(Statistic, 4)),
      `$p$-value` = as.character(signif(`$p$-value`, 4)))

robust_t_hw_tbl %>%
  kableExtra::kable(
    format = "latex", booktabs = TRUE,
    digits = 4, escape=FALSE) %>%
  kableExtra::save_kable("figures/robust_t_test_hw.png")
```

Covariate	Estimate	SE	Statistic	p-value
disconnected_youth	5.948	1.807	3.292	0.001032
unemployment	24.41	4.136	5.903	5.078e-09
housing_overcrowding	-13.63	4.081	-3.339	0.0008746
segregation_nonwhite_white	0.02639	0.005798	4.551	6.069e-06
inactive_perc	6.671	1.472	4.533	6.616e-06
flu_vaccine_perc	3.932	0.9362	4.2	2.935e-05
median_income	2.783e-05	7.323e-06	3.801	0.000154

Table 4: Robust t-test with HW SEs summary.

The results of the robust t-test from the second analysis are provided in Table 4.

(e) The first analysis assumes that the state in which a county is located only affects the data generating process by way of the errors; i.e. the errors are identically distributed for all counties in the same state but not necessarily across counties. The second analysis instead assumes that the state in which a county is located directly affects the data generating process by some fixed amount, with the errors remaining uncorrelated (though not necessarily identically distributed).

A benefit of (1) is the ability to (somewhat) account for geographical effects into account by assuming correlated errors within states. One negative part of approach (1) is that it cannot account for geographical correlations between adjacent counties in different states; here a moving blocks bootstrap would be more appropriate. On the flip side, a con of (2) is the fact that it assumes a consistent state fixed effect size for all counties in a given state, only allowing variability to occur from the socioeconomic covariates and uncorrelated errors. In the case of COVID however, this is not necessarily a bad thing as many intervention policies (e.g. lockdowns) were implemented at the state level and so were relatively consistent across all counties in a given state.

```
# Get SEs from (a)
nonrobust_t_se <-
  feat_summary %>%
    data.frame() %>%
    dplyr::select(Std..Error) %>%
    rename(SE=Std..Error) %>%
    mutate(Covariate = rownames(.),
           Method="Non-robust")

# Get LZ SEs
robust_t_lz_se <-
  robust_t_lz %>%
    tidy() %>%
    mutate(`SE inflation factor` = std.error/nonrobust_t_se$SE) %>%
    dplyr::select(term, `SE inflation factor`) %>%
    rename(Covariate=term) %>%
    mutate(Method="LZ")

# Get HW SEs
robust_t_hw_se <-
  robust_t_hw %>%
    tidy() %>%
        mutate(`SE inflation factor` = std.error/nonrobust_t_se$SE) %>%
    dplyr::select(term, `SE inflation factor`) %>%
    rename(Covariate=term) %>%
    mutate(Method="HW")

# Combine robust SEs
robust_t_se <- rbind(robust_t_lz_se, robust_t_hw_se)

# Create histogram of robust SEs
se_inflation_hist <-
  robust_t_se %>%
    ggplot() +
      geom_histogram(aes(x=`SE inflation factor`, fill=Method)) +
    ylab("Count")

ggsave("figures/se_inflation_hist.png", se_inflation_hist)

## Saving 7 x 7 in image
## `stat_bin()` using `bins = 30`.  Pick better value with `binwidth`.
```

We plot the distribution across features of the factor by which SEs were increased from part (a) using the robust analysis.

Note that the distribution for method (2) using HW SEs has a longer right-tail, indicating that the use of HW has a tendency to increase the SEs more than the LZ method.

On the whole, I would recommend the use of the LZ approach, as it takes geographical correlations into consideration and does not assume a constant fixed effect size for all counties in a given state.

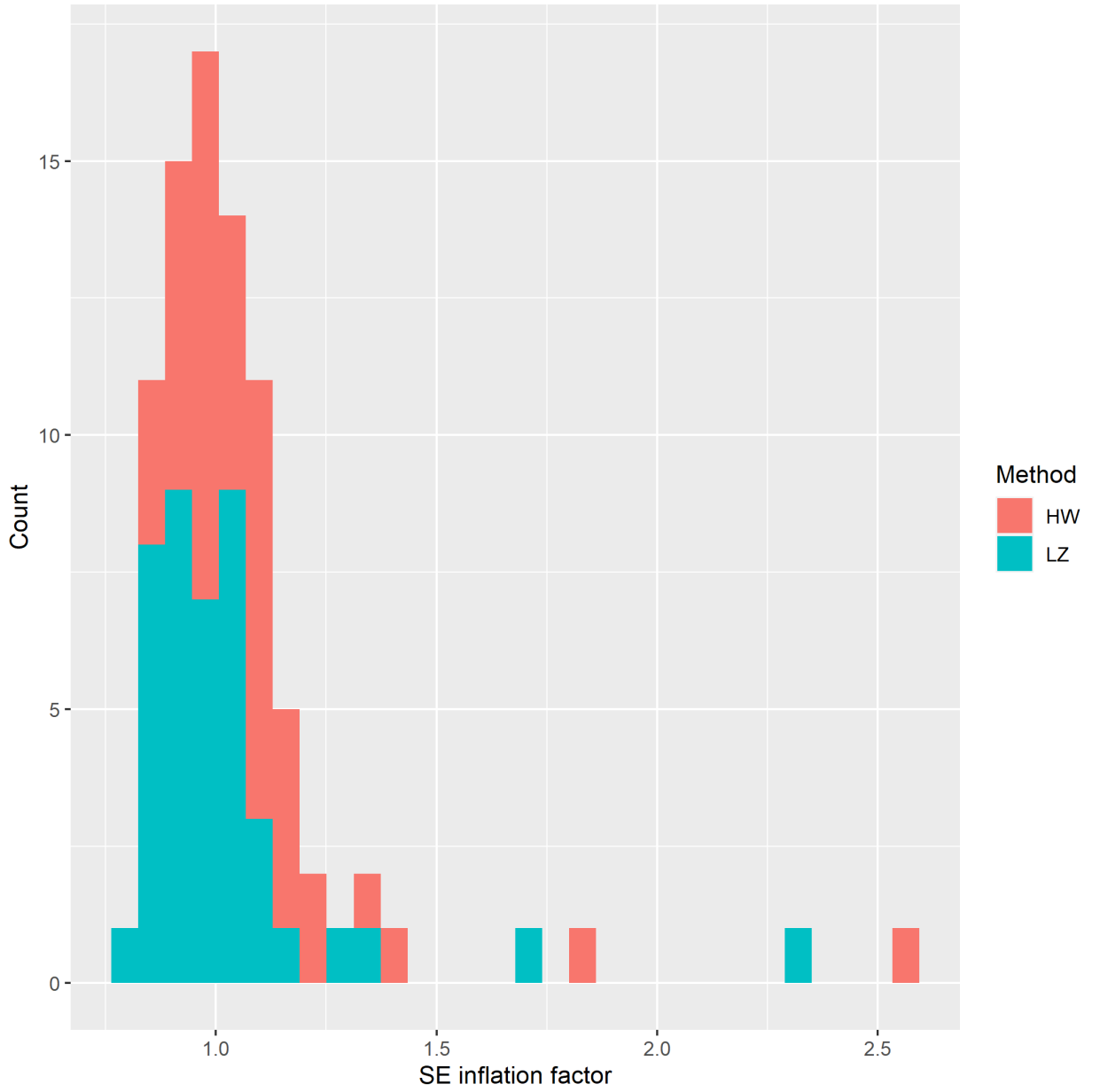

Figure 6: Distribution of factor by which SEs were increased with robust analysis.

Moreover, it tends to increase the SEs less than that of the HW approach. Any desire to account for state-level covariates such as the introduction of a lockdown can be accounted for by directly including these covariates in the model, rather than relying on a state indicator to (hopefully!) account for all such confounders.

A Descriptions of features in COVID data

Below are the 41 features we used for analysis. Words written in parentheses represent variable names. Unless noted otherwise, all variables are continuous.
Health behaviors:

- *Tobacco Use*
  - Adult smoking (`smoke_perc`): Percentage of adults who are current smokers.
- *Diet and Exercise*
  - Adult obesity (`obesity_perc`): Percentage of the adult population (age 20 and older) reporting a body mass index (BMI) greater than or equal to 30 kg/m2.
  - Food environment index (`food_environment`): Index of factors that contribute to a healthy food environment, from 0 (worst) to 10 (best).
  - Physical inactivity (`inactive_perc`): Percentage of adults age 20 and over reporting no leisure-time physical activity.
  - Access to exercise opportunities (`physical_exercise_opportunities`): Percentage of population with adequate access to locations for physical activity.
  - Food insecurity (`Food_Insecure_perc`): Percentage of population who lack adequate access to food.
  - Limited access to healthy foods (`limited_healthy_access`): Percentage of population who are low-income and do not live close to a grocery store.
- *Alcohol and Drug Use*
  - Excessive Drinking (`drinking_perc`): Percentage of adults reporting binge or heavy drinking.
- *Sexual Activity*
  - Sexually transmitted infections (`stis`): Number of newly diagnosed chlamydia cases per 100,000 population.
  - Teen births (`teen_births`): Number of births per 1,000 female population ages 15-19.
  - Low Birth Weight Percentage (`low_birthweight_percentage`): Percentage of live births with low birthweight (<2,500 grams).

Clinical care:

- *Access to Care*
  - Uninsured (`uninsured`): Percentage of population under age 65 without health insurance.
  - Primary care physicians (`primarycare_ratio`): Ratio of population to primary care physicians.
  - Dentists (`dentist_ratio`): Ratio of population to dentists.
  - Mental health providers (`mentalhealth_ratio`): Ratio of population to mental health providers.
  - Other primary care providers (`otherproviders_ratio`): Ratio of population to primary care providers other than physicians.
- *Quality of Care*

  - Preventable hospital stays (`preventable_hospitalization`): Rate of hospital stays for ambulatory-care sensitive conditions per 100,000 Medicare enrollees.
  - Mammography screening (`mammogram_perc`): Percentage of female Medicare enrollees ages 65-74 that received an annual mammography screening.
  - Flu vaccinations (`flu_vaccine_perc`): Percentage of fee-for-service (FFS) Medicare enrollees that had an annual flu vaccination.
  - Teen births (`teen_births`): Number of births per 1,000 female population ages 15-19.

Social and economic factors:

- *Education*
  - High school completion (`HS_completion`): Percentage of adults ages 25 and over with a high school diploma or equivalent.
  - Some college (`some_college`): Percentage of adults ages 25-44 with some post-secondary education.
  - Disconnected youth (`disconnected_youth`): Percentage of teens and young adults ages 16-19 who are neither working nor in school.
- *Employment*
  - Unemployment (`unemployment`): Percentage of population ages 16 and older who are unemployed but seeking work.
- *Income*
  - Children in poverty (`children_poverty_percent`): Percentage of people under age 18 in poverty.
  - Income inequality (`income_inequality`): Ratio of household income at the 80th percentile to income at the 20th percentile.
  - Median household income (`median_income`): The income where half of households in a county earn more and half of households earn less.
  - Children eligible for free or reduced price lunch (`children_freelunches`): Percentage of children enrolled in public schools that are eligible for free or reduced price lunch.
- *Family and Social Support*
  - Children in single-parent households (`single_parent_households`): Percentage of children that live in a household headed by a single parent.
  - Social associations (`social_associations`): Number of membership associations per 10,000 residents.
  - Residential segregation—Black/White (`segregation_black_white`): Index of dissimilarity where higher values indicate greater residential segregation between Black and White county residents.
  - Residential segregation—non-White/White (`segregation_nonwhite_white`): Index of dissimilarity where higher values indicate greater residential segregation between non-White and White county residents.
- *Community Safety*
  - Violent crime rate (`Violent_crime`) Number of reported violent crime offenses per 100,000 residents.

Physical environment:

- *Air and Water Quality*
  - Air pollution - particulate matter (`air_pollution`): Average daily density of fine particulate matter in micrograms per cubic meter (PM2.5).
  - Drinking water violations (`water_violations`): Indicator of the presence of health-related drinking water violations. 1 indicates the presence of a violation, 0 indicates no violation.
- *Housing and Transit*
  - Housing overcrowding (`housing_overcrowding`): Percentage of households with overcrowding,
  - Severe housing costs (`high_housing_costs`): Percentage of households with high housing costs
  - Driving alone to work (`driving_alone_perc`): Percentage of the workforce that drives alone to work.
  - Long commute—driving alone (`long_commute_perc`): Among workers who commute in their car alone, the percentage that commute more than 30 minutes.
  - Traffic volume (`traffic_volume`): Average traffic volume per meter of major roadways in the county.
  - Homeownership (`homeownership`): Percentage of occupied housing units that are owned.
  - Severe housing cost burden (`severe_ownership_cost`): Percentage of households that spend 50% or more of their household income on housing.