Rpr-Reproduction of Social Inequities in the distribution of COVID-19: An intra-categorical analysis of people with disabilities in the U.S.

Joseph Holler, Department of Geography, Middlebury College, Middlebury VT 05753 Drew An-Pham, Department of Geography, Middlebury College, Middlebury VT 05753 Derrick Burt, Department of Geography, Middlebury College, Middlebury VT 05753 Peter Kedron, School of Geographical Sciences and Urban Planning, Arizona State University, Tempe AZ 85281

Version 1.1 | Created Jul 7, 2021 | Last Updated June 1, 2022

Abstract

Chakraborty (2021) investigates the relationships between COVID-19 rates and demographic characteristics of people with disabilities by county in the lower 48 states. The study aims to examine public concern that persons with disabilities (PwDs) face disproportionate challenges due to COVID-19. To investigate this, Chakraborty examines the statistical relationship between confirmed county-level COVID-19 case rates and county-level socio-demographic and disability variables. Specifically, Chakraborty tests county-level bivariate correlations between COVID-19 incidence against the percentage of disability and socio-demographic category, with a separate hypothesis and model for each subcategory within disability, race, ethnicity, age, and biological sex. To control for differences between states and geographic clusters of COVID-19 outbreaks, Chakraborty uses five generalized estimating equation (GEE) models to predict the relationship and significance between COVID-19 incidence and disability subgroups within each socio-demographic category while considering inter-county spatial clusters. Chakraborty (2021) finds significant positive relationships between COVID-19 rates and socially vulnerable demographic categories of race, ethnicity, poverty, age, and biological sex.

This reproduction study is motivated by expanding the potential impact of Chakraborty's study for policy, research, and teaching purposes. Measuring the relationship between COVID-19 incidence and socio-demographic and disability characteristics can provide important information for public health policy-making and resource allocation. A fully reproducible study will increase the accessibility, transparency, and potential impact of Chakraborty's (2021) study by publishing a compendium complete with metadata, data, and code. This will allow other researchers to review, extend, and modify the study and will allow students of geography and spatial epidemiology to learn from the study design and methods.

In this reproduction, we will attempt to identically reproduce all of the results from the original study. This will include the map of county level distribution of COVID-19 incidence rates (Fig. 1), the summary statistics for disability and sociodemographic variables and bivariate correlations with county-level COVID-19 incidence rate (Table 1), and the GEE models for predicting COVID-19 county-level incidence rate (Table 2). A successful reproduction should be able to generate identical results as published by Chakraborty (2021).

The replication study data and code will be made available in a GitHub repository to the greatest extent that licensing and file sizes permit. The repository will be made public at github.com/HEGSRR/RPr-Chakraborty2021.

Chakraborty, J. 2021. Social inequities in the distribution of COVID-19: An intra-categorical analysis of people with disabilities in the U.S. Disability and Health Journal 14:1-5. DOI:%5B10.1016/j.dhjo.2020.101007]()

Keywords

COVID-19; Disability; Intersectionality; Race/ethnicity; Poverty; Reproducibility

Study Design

The reproduction study will try to implement the original study as closely as possible to reproduce the map of county level distribution of COVID-19 incidence rate, the summary statistics and bivariate correlation for disability characteristics and COVID-19 incidence, and the generalized estimating equations. Our two confirmatory hypotheses are that we will be able to exactly reproduce Chakraborty's results as presented in table 1 and table 2 of Chakraborty (2021). Stated as null hypotheses:

> H1: There is a less than perfect match between Chakraborty's bivariate correlation coefficient for each disability/sociodemographic variable and COVID-19 incidence rate and our bivariate correlation coefficient for each disability/sociodemographic variable and COVID-19 incidence rate.
>
> H2: There is a less than perfect match between Chakraborty's beta coefficient for the GEE of each disability/sociodemographic variable and our beta coefficient for the GEE of each disability/sociodemographic variable.

There are multiple models being tested within each of the two hypotheses. That is, H1 and H2 both encompass five models, including one for each dimension of socio-demographics: race, ethnicity, poverty status, age, and biological sex.

Original study design

The original study is **observational**, with the **exploratory** objective of determining "whether COVID-19 incidence is significantly greater in counties containing higher percentages of socio-demographically disadvantaged [people with disabilities], based on their race, ethnicity, poverty status, age, and biological sex" (Chakraborty 2021). This exploratory objective is broken down into five implicit hypotheses that each of the demographic characteristics of people with disabilities is associated with higher COVID-19 incidence rates.

The **spatial extent** of the study are the 49 contiguous states in the U.S. The **spatial scale** of the analysis is at the county level. Both COVID-19 incidence rates and demographic variables are all measured at the county level. The **temporal extent** of the COVID-19 data ranges from 1/22/2020 (when John Hopkins began collecting the data) to 8/1/2020 (when the data was retrieved for the original study). The data on disability and sociodemographic characteristics come from the U.S. Census American Community Survey (ACS) five-year estimates for 2018 (2014-2018).

There is no **randomization** in the original study.

The study was originally conducted using SaTScan software (unspecified version) to implement the spatial scan statistic. Other software are not specified in the publication; however data files and communication with the author show that spatial analysis and mapping was conducted in ArcGIS and statistics were calculated in SPSS.

Query American Communtity Survey Data

This will require an API key for the census, which can be acquired easily here: api.census.gov/data/key_signup.html
This query can take some time to run…

```r
# get API Key
# we could store this in the raw/private or scratch folder and load if the
# researcher has already entered it once
census_api_key(dlgInput("Enter a Census API Key",
  Sys.getenv("CENSUS_API_KEY"))$res,
  overwrite = TRUE)

# Query disability demographic data with geographic boundaries
acs <- get_acs(geography = "county",
  table = "S1810",
  year = 2018,
  output = "wide",
  cache_table = TRUE,
  geometry = TRUE,
  keep_geo_vars = TRUE)
```

```
# Query poverty and disability data
acs_pov <- get_acs(geography = "county",
  table = "C18130",
  year = 2018,
  output = "wide",
  cache_table = TRUE
)
```

Filtering and joining the ACS data

This accomplishes the step 1 and 3 of the workflow diagram

```
# Remove Alaska, Hawaii & Puerto Rico
acs <- filter(acs, !STATEFP %in% c("02", "15", "72"))

# Join poverty data to disability data
acs <- left_join(acs, acs_pov, by = "GEOID")
```

Save raw data

Optionally, you may save the raw data to data/raw/public/acs.gpkg

```
# Save downloaded acs data to acs.gpkg
write_sf(acs, here("data", "raw", "public", "acs.gpkg"))
```

Load raw data

Optionally, you may load the raw data and begin processing here

```
acs <- read_sf(here("data", "raw", "public", "acs.gpkg"))
```

Preprocess ACS data

This accomplishes the step 4 the workflow diagram

Calculate percentages for each sub-category of disability and remove raw census data from the data frame

```
# calculate percentages
acs_derived <- mutate(acs,
  dis_pct = S1810_C02_001E / S1810_C01_001E * 100,
  white_pct = S1810_C02_004E / S1810_C01_001E * 100,
  black_pct = S1810_C02_005E / S1810_C01_001E * 100,
  native_pct = S1810_C02_006E / S1810_C01_001E * 100,
  asian_pct = S1810_C02_007E / S1810_C01_001E * 100,
  other_pct =
    (S1810_C02_008E + S1810_C02_009E + S1810_C02_010E)/S1810_C01_001E *100,
  non_hisp_white_pct = S1810_C02_011E / S1810_C01_001E * 100,
  hisp_pct = S1810_C02_012E / S1810_C01_001E * 100,
  non_hisp_non_white_pct =
    (S1810_C02_001E - S1810_C02_012E - S1810_C02_011E) / S1810_C01_001E * 100,
  bpov_pct = (C18130_004E + C18130_011E + C18130_018E) / C18130_001E * 100,
  apov_pct = (C18130_005E + C18130_012E + C18130_019E) / C18130_001E * 100,
  pct_5_17 = S1810_C02_014E / S1810_C01_001E * 100,
  pct_18_34 = S1810_C02_015E / S1810_C01_001E * 100,
  pct_35_64 = S1810_C02_016E / S1810_C01_001E * 100,
```

```r
  pct_65_74 = S1810_C02_017E / S1810_C01_001E * 100,
  pct_75 = S1810_C02_018E / S1810_C01_001E * 100,
  male_pct = S1810_C02_002E / S1810_C01_001E * 100,
  female_pct = S1810_C02_003E / S1810_C01_001E * 100
)

# select only relevant geographic identifiers and derived percentages
# and transform to USA Contiguous Albers Equal Area Conic projection
acs_derived <- acs_derived %>%
  select(
    geoid = GEOID,
    statefp = STATEFP,
    county = NAME.x,
    county_st = NAME,
    contains("pct")
  ) %>%
  st_transform(5070)
```

Load COVID-19 data

This accomplishes the step 2 of the workflow diagram

This data has been provided directly with the research compendium because it is no longer available online in the state in which it was downloaded on August 1, 2020. The data was provided by the original author, Jayajit Chakraborty.

```r
covid <- read_sf(here("data", "raw", "public","covidcase080120.gpkg"))
covid <- select(covid,
  fips = FIPS,
  pop = POP_ESTIMA,
  cases = Confirmed,
  x=X, y=Y)
covid$covid_rate <- covid$cases / covid$pop * 100000
covid_table <- st_drop_geometry(covid)

# It might be wise to calculate our own X and Y using centroids
```

Join COVID data to ACS data

This accomplishes the step 5 of the workflow diagram

```r
# Join poverty data to acs data
covid_rate_table <- select(covid_table, fips, covid_rate)
acs_covid <- left_join(acs_derived, covid_rate_table, by=c("geoid"="fips"))

# move covid_rate prior to percentages
acs_covid <- select(acs_covid, geoid, statefp, county, county_st, covid_rate,
  everything())
```

Map Covid Rates

```r
tm1 <- tm_shape(acs_covid) +
  tm_polygons("covid_rate",
    title="COVID-19 Cases per 100,000",
    style="quantile",
    border.alpha = .2,
    lwd = 0.2,
    palette="YlOrBr"
  )

tmap_save(tm1, here("results", "figures", "covid_rates.png"))
```

```
## Map saved to D:\github\hegsrr\RPr-Chakraborty2020\results\figures\covid_rates.png

## Resolution: 2647.584 by 1665.669 pixels

## Size: 8.82528 by 5.552232 inches (300 dpi)
```

Map Disability Rates

```
tm2 <- tm_shape(acs_covid) +
  tm_polygons("dis_pct",
    title="Percent with Disability",
    style="quantile",
    border.alpha = .2,
    lwd = 0.2,
    palette="YlOrBr"
  )

tmap_save(tm2, here("results", "figures", "disability_rates.png"))
```

```
## Map saved to D:\github\hegsrr\RPr-Chakraborty2020\results\figures\disability_rates.png

## Resolution: 2647.584 by 1665.669 pixels

## Size: 8.82528 by 5.552232 inches (300 dpi)
```

Missing Data

There is one county with missing disability and poverty data. Below, we replace the missing data with zeros, producing results identical to Chakraborty's.

```
# county with missing data
filter(acs_covid, is.na(bpov_pct))
```

```
## Simple feature collection with 1 feature and 23 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: -1032914 ymin: 1477939 xmax: -849865.1 ymax: 1611186
## Projected CRS: NAD83 / Conus Albers
## # A tibble: 1 x 24
##   geoid statefp county     county_st      covid_rate dis_pct white_pct black_pct
## * <chr> <chr>   <chr>      <chr>               <dbl>   <dbl>     <dbl>     <dbl>
## 1 35039 35      Rio Arriba Rio Arriba Co~       751.    16.1      10.8    0.0384
## # ... with 16 more variables: native_pct <dbl>, asian_pct <dbl>,
## #   other_pct <dbl>, non_hisp_white_pct <dbl>, hisp_pct <dbl>,
## #   non_hisp_non_white_pct <dbl>, bpov_pct <dbl>, apov_pct <dbl>,
## #   pct_5_17 <dbl>, pct_18_34 <dbl>, pct_35_64 <dbl>, pct_65_74 <dbl>,
## #   pct_75 <dbl>, male_pct <dbl>, female_pct <dbl>, geom <MULTIPOLYGON [m]>
```

```
# replace NA with 0 for missing data
acs_covid[is.na(acs_covid$bpov_pct), ]$bpov_pct <- 0
acs_covid[is.na(acs_covid$apov_pct), ]$apov_pct <- 0
```

Descriptive Statistics

```
acs_covid_stats <- acs_covid %>%
  st_drop_geometry() %>%
  select(is.numeric) %>%
  stat.desc(norm=TRUE) %>%
  round(2) %>%
  t() %>% as.data.frame() %>%
  select(min, max, mean, SD = std.dev, ShapiroWilk = normtest.W, p = normtest.p)
```

```
## Warning: Predicate functions must be wrapped in `where()`.
##
##   # Bad
##   data %>% select(is.numeric)
##
##   # Good
##   data %>% select(where(is.numeric))
##
## i Please update your code.
## This message is displayed once per session.
```

```
acs_covid_stats
```

```
##                          min      max   mean      SD ShapiroWilk p
## covid_rate              0.00 14257.17 966.90 1003.96        0.74 0
## dis_pct                 3.83    33.71  15.95    4.40        0.98 0
## white_pct               0.85    33.26  13.55    4.63        0.98 0
## black_pct               0.00    20.70   1.48    2.66        0.61 0
## native_pct              0.00    13.74   0.28    0.94        0.28 0
## asian_pct               0.00     3.45   0.09    0.18        0.51 0
## other_pct               0.00    15.24   0.55    0.65        0.57 0
## non_hisp_white_pct      0.10    33.16  12.84    4.81        0.99 0
## hisp_pct                0.00    25.26   0.99    2.15        0.42 0
## non_hisp_non_white_pct 0.00    20.93   2.13    2.75        0.70 0
## bpov_pct                0.00    14.97   3.57    1.85        0.93 0
## apov_pct                0.00    27.30  12.48    3.06        0.99 0
## pct_5_17                0.00     5.08   1.03    0.48        0.95 0
## pct_18_34               0.00     5.59   1.56    0.67        0.96 0
## pct_35_64               1.01    18.36   6.35    2.30        0.96 0
## pct_65_74               0.00    12.73   3.09    1.16        0.95 0
## pct_75                  0.00    11.13   3.87    1.19        0.97 0
## male_pct                1.30    18.19   8.06    2.37        0.98 0
## female_pct              1.91    19.94   7.90    2.26        0.98 0
```

Calculate Pearson's R Correlation Coefficients

These results are identical in direction and significance to Chakraborty's, but differ slightly in magnitude.

```
df = sum(!is.na(acs_covid$dis_pct)) - 2

pearsons_r <- acs_covid %>%
  select(where(is.numeric)) %>%
  st_drop_geometry() %>%
  cor(method="pearson", use="pairwise.complete.obs") %>%
  as.data.frame() %>%
  select(r = covid_rate) %>%
  mutate(
    t = abs(r) / sqrt((1 - r^2)/(df) ),
    p = pt(t, df, lower.tail = FALSE)
    ) %>%
  round(3) %>%
  rownames_to_column("variable") %>%
  filter(variable != "covid_rate")

pearsons_r
```

```
##                   variable      r      t     p
## 1                  dis_pct -0.060  3.350 0.000
## 2                white_pct -0.332 19.612 0.000
## 3                black_pct  0.460 28.847 0.000
## 4               native_pct  0.019  1.072 0.142
## 5                asian_pct  0.094  5.272 0.000
## 6                other_pct  0.026  1.460 0.072
## 7       non_hisp_white_pct -0.361 21.545 0.000
## 8                 hisp_pct  0.119  6.686 0.000
## 9   non_hisp_non_white_pct  0.442 27.429 0.000
## 10                bpov_pct  0.106  5.914 0.000
## 11                apov_pct -0.151  8.513 0.000
## 12                pct_5_17  0.084  4.688 0.000
## 13               pct_18_34  0.063  3.493 0.000
## 14               pct_35_64 -0.008  0.460 0.323
## 15               pct_65_74 -0.091  5.113 0.000
## 16                  pct_75 -0.186 10.541 0.000
## 17                male_pct -0.134  7.519 0.000
## 18              female_pct  0.023  1.305 0.096
```

```
# this estimation of t gives similar, but not identical, result to corr.test
# consider trying the rstatix package for this!
```

Calculate Spearman's Rho Correlation Coefficients

Try a non-parametric correlation test because variables do not have normal distributions (see Shapiro-Wilk test results above). The direction of several of the variables changes with the non-parametric test.

```
df = sum(!is.na(acs_covid$dis_pct)) - 2

spearmans_rho <- acs_covid %>%
  select(where(is.numeric)) %>%
  st_drop_geometry() %>%
  cor(method="spearman", use="pairwise.complete.obs") %>%
  as.data.frame() %>%
  select(rho = covid_rate) %>%
  mutate(
    t = abs(rho) / sqrt((1 - rho^2)/(df) ),
    p = pt(t, df, lower.tail = FALSE)
    ) %>%
  round(3) %>%
  rownames_to_column("variable") %>%
  filter(variable != "covid_rate")

spearmans_rho
```

```
##                  variable    rho      t     p
## 1                 dis_pct -0.113  6.312 0.000
## 2               white_pct -0.421 25.874 0.000
## 3               black_pct  0.575 39.163 0.000
## 4              native_pct -0.084  4.688 0.000
## 5               asian_pct  0.194 11.001 0.000
## 6               other_pct  0.104  5.825 0.000
## 7      non_hisp_white_pct -0.454 28.389 0.000
## 8                hisp_pct  0.231 13.210 0.000
## 9  non_hisp_non_white_pct  0.481 30.564 0.000
## 10               bpov_pct  0.062  3.452 0.000
## 11               apov_pct -0.205 11.694 0.000
## 12               pct_5_17  0.079  4.411 0.000
## 13              pct_18_34  0.034  1.902 0.029
## 14              pct_35_64 -0.020  1.136 0.128
## 15              pct_65_74 -0.151  8.523 0.000
## 16                 pct_75 -0.285 16.592 0.000
## 17               male_pct -0.201 11.430 0.000
## 18             female_pct -0.014  0.798 0.212
```

Kulldorf Spatial Scan Cluster Detection

This accomplishes the step 6 of the workflow diagram

Note that the statistic is a Monte Carlo simulation with 999 iterations. Therefore, if you wish to exactly reproduce the same results as our reproduction attempt, please **do not run this section**. Instead, load the scan results below. This code block can take more than 10-20 minutes to run.

```
covid_geo <- covid_table %>%
  select(x, y) %>%
  latlong2grid()
# latlong2grid approximates an equidistant grid measured in kilometers
# need to look more into the methods of this, but it surely is not as good
# as a geodesic calculation. SaTScan uses spherical or ellipsoidal distance

# calculate expected cases with one strata
expected.cases <- expected(covid_table$pop, covid_table$cases , 1)

# Kulldorff spatial scan statistic
covid_kulldorff <- kulldorff(geo=covid_geo,
                             cases=covid_table$cases,
                             population=covid_table$pop,
                             expected.cases=expected.cases,
                             pop.upper.bound=0.5,
                             n.simulations=999,
                             alpha.level=0.05,
                             plot=TRUE
)
```

```r
rm(covid_table, covid_geo, expected.cases)
```

Save scan results

```r
saveRDS(covid_kulldorff,
  file=here("data","derived","public","covid_kulldorff.RDS"))
```

Load scan results

```r
covid_kulldorff <- readRDS(
  here("data","derived","public","covid_kulldorff.RDS")
  )
```

Summarize spatial scan clusters by county

This accomplishes the step 7 and 8 of the workflow diagram

Summarize the results of the Kulldorff spatial scan cluster detection by county Code each county `0` if it is not in a cluster and `1` if it is in a cluster.

```r
# Get counties in most significant cluster
# Note that the ID numbers are row numbers
clusters <- covid_kulldorff$most.likely.cluster$location.IDs.included

# Get list of secondary clusters
secondary <- covid_kulldorff$secondary.clusters

# Get counties from each secondary cluster
for (i in secondary) {
  clusters <- c(clusters, i$location.IDs.included)
}

# Create blank column "cluster" with 0's
covid$cluster <- 0

# Change cluster to 1 for any county identified in any cluster
covid[clusters, ]$cluster <- 1

# Calculate and Classify Local Relative Risk
# cut() classifies rr_loc from 1 to 6
# * cluster forces counties outside of clusters to be assigned to class 0
covid <- covid %>%
  mutate(
    rr_loc =
      (cases / pop) / ((sum(covid$cases) - cases) / (sum(covid$pop) - pop)),
    rr_class =
      (cut(rr_loc, c(-Inf,1,2,3,4,5,Inf), labels=FALSE) - 1) * cluster + 1
  ) %>%
  st_transform(5070)
```

How did the classification work?

```
# Count frequency of each class of COVID risk
cat("Classes of risk and frequency of counties",
  format(covid %>% st_drop_geometry %>% count(rr_class)), sep="\n")
```

```
## Classes of risk and frequency of counties
## # A tibble: 6 x 2
##   rr_class     n
##      <dbl> <int>
## 1        1  2464
## 2        2   476
## 3        3   124
## 4        4    31
## 5        5     5
## 6        6     8
```

```
cat("\n",
    sum(covid$cluster==0 & covid$rr_loc >= 1),
  " counties lie outside of a cluster, but have local relative risk > 1\n\n",
  sum(covid$cluster==1 & covid$rr_loc < 1),
  " counties lie inside of a cluster, but have a local relative risk < 1",
  sep="")
```

```
##
## 74 counties lie outside of a cluster, but have local relative risk > 1
##
## 294 counties lie inside of a cluster, but have a local relative risk < 1
```

```
# There's a relative risk score for both county & cluster (in SatScan)
# This reproduction uses county-based relative risk scores based on
# paragraph 4 of the methods section of the paper and Desjardins et al
```

Map Relative Risk Scores

Note that relative risk is > 1 only if the county was in a cluster

```
original_cluster <- read_sf(here("data", "derived", "public","satscan", "sat_scan_compare.col.shp"))
cluster_table <- st_drop_geometry(original_cluster)

covid <- left_join(covid, cluster_table, by = c("fips" = "LOC_ID"))

covid_temp <- covid %>%
  mutate(rr_original = if_else(is.na(REL_RISK), 0, REL_RISK))
```

```
tm4 <- tm_shape(covid_temp) +
  tm_polygons("rr_original",
    title="Original Relative Risk",
    breaks =c(0,1,2,3,4,5,8),
    border.alpha = .2,
    lwd = 0.2,
    palette="YlOrBr"
  ) +
tm_shape(original_cluster) +
  tm_borders("red", lwd =.5) +
  tmap_options(check.and.fix = TRUE)
```

```
tmap_save(tm4, here("results", "figures", "rr_original.png"))
```

```
## Map saved to D:\github\hegsrr\RPr-Chakraborty2020\results\figures\rr_original.png

## Resolution: 2647.547 by 1665.693 pixels

## Size: 8.825155 by 5.55231 inches (300 dpi)
```

```
# Map Relative Risk scores
tm3 <- tm_shape(covid) +
  tm_polygons("rr_class",
    title="Local Relative Risk",
    border.alpha = .2,
    lwd = 0.2,
    palette="YlOrBr",
    style="cat"
  )

tmap_save(tm3, here("results", "figures", "rr_reproduction.png"))
```

```
## Map saved to D:\github\hegsrr\RPr-Chakraborty2020\results\figures\rr_reproduction.png

## Resolution: 2647.547 by 1665.693 pixels

## Size: 8.825155 by 5.55231 inches (300 dpi)
```

Preprocess data for GEE modelling

This accomplishes the step 9 and 10 of the workflow diagram

```
covid_clusters <- covid %>%
  select(fips, cluster, rr_loc, rr_class) %>%
  st_drop_geometry

# Filter out non-positive COVID rates and missing data
# Create unique State - Relative Risk IDs by combining state code and rr_class
# Sort by the cluster id's (a requirement of the gee function)
gee_data <- left_join(acs_covid, covid_clusters, by = c("geoid"="fips")) %>%
  filter(covid_rate > 0) %>%
  mutate(id = as.integer(statefp) * 10 + rr_class) %>%
  arrange(id)


gee_data <- gee_data %>%
  mutate(z_bpov_pct = scale(bpov_pct),
         z_apov_pct = scale(apov_pct),
         z_white_pct = scale(white_pct),
         z_black_pct = scale(black_pct),
         z_native_pct = scale(native_pct),
         z_asian_pct = scale(asian_pct),
         z_other_pct = scale(other_pct),
         z_non_hisp_white_pct = scale(non_hisp_white_pct),
         z_hisp_pct = scale(hisp_pct),
         z_non_hisp_non_white_pct = scale(non_hisp_non_white_pct),
         z_pct_5_17 = scale(pct_5_17),
         z_pct_18_34 = scale(pct_18_34),
         z_pct_35_64 = scale(pct_35_64),
         z_pct_65_74 = scale(pct_65_74),
         z_pct_75 = scale(pct_75),
         z_male_pct = scale(male_pct),
         z_female_pct = scale(female_pct))


rm(covid_clusters)
```

Save preprocessed GEE data inputs

Optionally, you may save the preprocessed to `data/raw/public/gee_data.gpkg`

```
write_sf(gee_data, here("data","derived","public","gee_data.gpkg"))
```

Load preprocessed GEE input data

Optionally, you may load the preprocessed data from `data/raw/public/gee_data.gpkg`

```
gee_data <- read_sf(here("data","derived","public","gee_data.gpkg"))
```

Report number of unique clusters

```
cluster_summary <- gee_data %>%
  st_drop_geometry %>%
  count(id)
cat(length(cluster_summary$n), "unique clusters\n")
```

```
## 139 unique clusters
```

```
summary(cluster_summary$n)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1.00    2.00    5.00   22.01   31.00  184.00
```

GEE Models

This accomplishes the step 11 of the workflow diagram

Generalized Estimating Equation parameters:

"The **'exchangeable' correlation matrix** was selected for the results reported here, since this speci-fication yielded the best statistical fit based on the QIC (quasi- likelihood under the independence) model criterion." (Chakraborty 2021, Methods paragraph 5)

"The **gamma distribution** with **logarithmic link function** was chosen for all GEEs since this model specification provided the lowest QIC value." (Chakraborty 2021, Methods paragraph 5)

Useful Reference: https://data.library.virginia.edu/getting-started-with-generalized-estimating-equations/

```
# it would be smarter to iterate over a list of models and their parameters
# currently stuck on how to add GLM model results to a cell of a dataframe
# not the only one:
# https://www.reddit.com/r/rstats/comments/p50mce/coding_a_loop_for_many_linear_regressions/

race_gee <- geeglm(
  covid_rate ~ z_white_pct + z_black_pct + z_native_pct + z_asian_pct + z_other_pct,
```

```r
  data = gee_data, # data frame
  id = id, # cluster IDs
  family = Gamma(link = "log"),
  corstr = "exchangeable"
)

# Wald and P calculated in summary only;
# coef() extracts coefficients table from the summary, same as $coefficients

ethnicity_gee <- geeglm(
  covid_rate ~ z_non_hisp_white_pct + z_hisp_pct + z_non_hisp_non_white_pct,
  data = gee_data,
  id = id,
  family = Gamma(link = "log"),
  corstr = "exchangeable"
)

pov_gee <- geeglm(
  covid_rate ~ z_bpov_pct + z_apov_pct,
  data = gee_data,
  id = id,
  family = Gamma(link = "log"),
  corstr = "exchangeable",
)

age_gee <- geeglm(
  covid_rate ~ z_pct_5_17 + z_pct_18_34 + z_pct_35_64 + z_pct_65_74 + z_pct_75,
  data = gee_data,
  id = id,
  family = Gamma(link = "log"),
  corstr = "exchangeable"
)

sex_gee <- geeglm(
  covid_rate ~ z_male_pct + z_female_pct,
  data = gee_data,
  id = id,
  family = Gamma(link = "log"),
  corstr = "exchangeable"
)

# summarize model coefficients
coefficient_results <- rbind(coef(summary(race_gee)),
  coef(summary(ethnicity_gee)),
  coef(summary(pov_gee)),
  coef(summary(age_gee)),
  coef(summary(sex_gee))
  ) %>%
  round(3)

# disambiguate intercepts
coefrows <- rownames(coefficient_results)
coefrows[1] <- "Race Intercept"
```

```r
coefrows[7] <- "Ethnicity Intercept"
coefrows[11] <- "Poverty Status Intercept"
coefrows[14] <- "Age Intercept"
coefrows[20] <- "Biological Sex Intercept"
rownames(coefficient_results) <- coefrows
coefficient_results
```

```
##                           Estimate Std.err      Wald Pr(>|W|)
## Race Intercept               7.723   0.076 10336.073    0.000
## z_white_pct                 -0.129   0.007   358.943    0.000
## z_black_pct                  0.019   0.008     5.122    0.024
## z_native_pct                 0.018   0.005    13.578    0.000
## z_asian_pct                  0.022   0.004    28.546    0.000
## z_other_pct                  0.022   0.006    14.714    0.000
## Ethnicity Intercept          7.716   0.076 10292.946    0.000
## z_non_hisp_white_pct        -0.150   0.008   365.943    0.000
## z_hisp_pct                   0.006   0.005     1.658    0.198
## z_non_hisp_non_white_pct     0.023   0.008     9.049    0.003
## Poverty Status Intercept     7.774   0.074 10947.636    0.000
## z_bpov_pct                   0.018   0.006     9.446    0.002
## z_apov_pct                  -0.110   0.007   233.641    0.000
## Age Intercept                7.783   0.073 11483.331    0.000
## z_pct_5_17                   0.022   0.003    39.514    0.000
## z_pct_18_34                  0.014   0.004    14.751    0.000
## z_pct_35_64                 -0.024   0.007    11.102    0.001
## z_pct_65_74                 -0.056   0.006    97.325    0.000
## z_pct_75                    -0.053   0.006    77.410    0.000
## Biological Sex Intercept     7.784   0.073 11427.103    0.000
## z_male_pct                  -0.135   0.008   316.207    0.000
## z_female_pct                 0.041   0.006    43.665    0.000
```

```r
# summarize model QICs
QIC_results <- data.frame(race = QIC(race_gee),
  ethnicity = QIC(ethnicity_gee),
  poverty_status = QIC(pov_gee),
  age = QIC(age_gee),
  biological_sex = QIC(sex_gee)
  ) %>%
  round(3) %>%
  t() %>%
  as.data.frame() %>%
  select(QIC)
QIC_results
```

```
##                     QIC
## race           2616.417
## ethnicity      2616.391
## poverty_status 2562.674
## age            3577.072
## biological_sex 2012.266
```