# Manuscript Title

*This manuscript (permalink) was automatically generated from andtheWings/cook_county_suid_manuscript@a7b361c on July 11, 2022.*

## Authors

- **Daniel P. Riggins** ✉
  0000-0002-6240-6212 · andtheWings · DanRiggins
  Section of Preventive Medicine, Cook County Health; Program in Public Health, Feinberg School of Medicine, Northwestern University · Funded by none

✉ — Correspondence possible via GitHub Issues or email to Daniel P. Riggins <daniel.riggins@protonmail.com>.

## Abstract

*Daniel P. Riggins, Huiyuan Zhang, William E. Trick, Marjorie Fujara*

BACKGROUND: Although overall rates of Sudden Infant Death Syndrome (SIDS) are declining, significant racial disparities persist in many metropolitan areas, including Cook County, IL. In some areas, public health campaigns have successfully reduced SIDS through promotion of safe sleep practices. To correct these disparities, public health practitioners might benefit from guidance to specific high-risk neighborhoods to precisely target interventions within Cook County's large geographic breadth.

METHODS: We used geocoded medical examiner data and manual review of death records to identify SIDS events and locations. We characterized the demographics of SIDS, sought to identify specific clusters of census tracts in the county where SIDS has been most prevalent, and modeled prevalence based on census-level ecologic factors obtained from the census bureau. We hypothesized that regions with greatest prevalence would follow patterns of historic racial segregation and that factors reflecting elevated levels of socioeconomic disadvantage would have high utility in the model.

RESULTS: Demographic analysis revealed that infant mortality due to SIDS had a much higher proportion of Black individuals (70%) than Cook County's children overall (25%). Mapping confirmed that SIDS mortality was most concentrated in historically disinvested portions of the West and South Sides of the county, specifically in Chicago neighborhoods like Englewood, Humboldt Park, and Pullman as well as suburbs like Chicago Heights, Harvey, and Matteson. Our model included the following variables: number of residents under age five, percent of residents on public insurance, number of opioid overdose deaths, and percent of residents self-identifying as Black. The model was able to accurately retrodict whether census tracts had or had not contained SIDS deaths, but it was not flexible enough to enumerate the number of deaths in a tract if that was any greater than a count of 2.

COUNCLUSIONS: Clinicians and public health practitioners should concentrate preventive efforts against SIDS in the specific geographic areas listed above. We shall explore why certain neighborhoods fared better or worse than expected by our model in order to identify additional harmful or protective factors. We intend to involve individuals and organizations from the Black community in the design of future interventions.

## Case Characteristics

Our automated method for identifying cases of SUID from medical examiner archives generated 333 prospective cases. After comparison with the gold standard list of cases identified via manual review by a panel of experts, we added XXX cases and removed XXX cases (Table XXX). There were XXX cases not originally identified by manual review that we still deemed to be valid. This process yielded XXX valid SUID cases in total.

We tabulated descriptive characteristics for valid SUID cases (Table XXX). We compared the distributions of select characteristics compared to those in the overall population of Cook County children under the age of 5 (derived from the census). SUID cases had a higher prevalence of Black infants and favored location in Chicago (vs. Cook County Suburbs) compared to the reference population.

## Census Tract Aggregate Characteristics

We tabulated characteristics of census tracts that had at least one case of SUID versus those that did not (Table XXX). Census tracts with at least one case of SUID appeared to have higher proportions of black people, XXX and lower proportions of XXX.

## Pipeline Overview

```
library(targets)
```

DPR orchestrated the data pipeline for the primary data analysis with the package {targets}.

If working with this repository and you want a visual overview of the pipeline, you can execute:

```
tar_visnetwork()
```

Here is information on the R session and its dependencies:

```
sessionInfo()

## R version 4.2.1 (2022-06-23)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Pop!_OS 22.04 LTS
##
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/openblas-pthread/libblas.so.3
## LAPACK: /usr/lib/x86_64-linux-gnu/openblas-pthread/libopenblasp-r0.3.20.so
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C               LC_TIME=en_US.UTF-8
##  [4] LC_COLLATE=en_US.UTF-8     LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=en_US.UTF-8       LC_NAME=C                  LC_ADDRESS=C
## [10] LC_TELEPHONE=C             LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] rmarkdown_2.14   knitr_1.39       lubridate_1.8.0  tidycensus_1.2.2
##  [5] sf_1.0-7         forcats_0.5.1    stringr_1.4.0    dplyr_1.0.9
##  [9] purrr_0.3.4      readr_2.1.2      tidyr_1.2.0      tibble_3.1.7
## [13] ggplot2_3.3.6    tidyverse_1.3.1  targets_0.12.1
##
## loaded via a namespace (and not attached):
##  [1] fs_1.5.2           webshot_0.5.3      httr_1.4.3         tools_4.2.1
##  [5] backports_1.4.1    utf8_1.2.2         rgdal_1.5-32       R6_2.5.1
##  [9] KernSmooth_2.23-20 DBI_1.1.3          colorspace_2.0-3   withr_2.5.0
## [13] sp_1.5-0           tidyselect_1.1.2   processx_3.7.0     curl_4.3.2
## [17] compiler_4.2.1     cli_3.3.0          rvest_1.0.2        gt_0.6.0
## [21] xml2_1.3.3         sass_0.4.1         scales_1.2.0       checkmate_2.1.0
## [25] classInt_0.4-7     corrr_0.4.3        callr_3.7.0        proxy_0.4-27
## [29] rappdirs_0.3.3     commonmark_1.8.0   digest_0.6.29      foreign_0.8-82
## [33] pkgconfig_2.0.3    htmltools_0.5.2    highr_0.9          dbplyr_2.2.1
## [37] fastmap_1.1.0      htmlwidgets_1.5.4  rlang_1.0.3        readxl_1.4.0
## [41] rstudioapi_0.13    visNetwork_2.1.0   generics_0.1.3     jsonlite_1.8.0
## [45] magrittr_2.0.3     Rcpp_1.0.8.3       munsell_0.5.0      fansi_1.0.3
## [49] lifecycle_1.0.1    stringi_1.7.6      yaml_2.3.5         gtsummary_1.6.1
## [53] snakecase_0.11.0   grid_4.2.1         maptools_1.1-4     promises_1.2.0.1
## [57] crayon_1.5.1       lattice_0.20-45    haven_2.5.0        chromote_0.1.0
## [61] hms_1.1.1          ps_1.7.1           pillar_1.7.0       igraph_1.3.2
## [65] uuid_1.1-0         base64url_1.4      codetools_0.2-18   reprex_2.0.1
## [69] glue_1.6.2         evaluate_0.15      data.table_1.14.2  broom.helpers_1.8.0
## [73] modelr_0.1.8       vctrs_0.4.1        tzdb_0.3.0         cellranger_1.1.0
## [77] webshot2_0.1.0     gtable_0.3.0       assertthat_0.2.1   xfun_0.31
## [81] janitor_2.1.0      broom_1.0.0        e1071_1.7-11       later_1.3.0
## [85] class_7.3-20       websocket_1.4.1    tigris_1.6.1       units_0.8-0
## [89] ellipsis_0.3.2
```

## Ingestion

At time of primary analysis, DPR combined a data file generated by collaborator HZ and additional data pulled via the R package {tidycensus}.

DPR imported HZ's excel file into the Targets pipeline using the function `readxl::read_xls()`. He pulled from the tidycensus API using his function `get_suid_from_tidycensus()`. This function asks for variables from the American Community Survey's 5-year estimate for 2015-2019. The variables describe count of total population, count of population under age 5 years, average number of people per household, and geospatial shape of census tracts in Cook County, IL. The function also does some reshaping of data as noted in the code comments below:

```
## function() {
##
##     df1 <-
##         # Call the census API
##         tidycensus::get_acs(
##             geography = "tract",
##             variables = c(
##                 "B01003_001", # TOTAL POPULATION
##                 "B06001_002",  # Total Under 5 years
##                 "B25010_001" # AVERAGE HOUSEHOLD SIZE OF OCCUPIED HOUSING UNITS
##             ),
##             state = "IL", # Illinois
##             county = 031, # Cook County
##             geometry = TRUE # Import census tract polygons too
##         ) |>
##         st_transform(crs = 4326) |>
##         # Drop margin of estimate and name variables
##         select(-moe, -NAME) |>
##         # Convert GEOID to numeric type
##         mutate(
##             GEOID = as.numeric(GEOID)
##         ) |>
##         # Reshape variable estimates into separate columns
##         tidyr::pivot_wider(
##             names_from = variable,
##             values_from = estimate
##         ) |>
##         # Clean names for consistency
##         rename(
##             fips = GEOID,
##             pop_total = B01003_001,
##             pop_under_five = B06001_002,
##             avg_peop_per_household = B25010_001
##         ) |>
##         # Put geometry column at the end of the table
##         relocate(
##             geometry,
##             .after = avg_peop_per_household
##         ) |>
##         st_as_sf()
##
##     return(df1)
## }
```

DPR combined the two data sources with his function `assemble_suid()`, which performs a left join on FIPS ID for observations from HZ's excel data, then does some wrangling around naming conventions, adds a boolean variable for whether SUID is present in a given tract, and rounds estimates to the first decimal point for consistency:

```
## function(suid_from_internal_raw_df, suid_from_tidycensus_raw_sf) {
##
##     df1 <-
##         left_join(
##             suid_from_internal_raw_df,
##             suid_from_tidycensus_raw_sf,
##             by = c("FIPS" = "fips")
##         ) |>
##         clean_names() |>
##         # Rename variables for consistency
##         rename(
##             suid_count = count_asphyxia,
##             foreign_born = pe_foreignborn,
##             married_males = pe_marriedmales,
##             married_females = pe_marriedfemales,
##             divorced_widowed_males = pedivorcewidowedmale,
##             divorced_widowed_females = pedivorcewidowedfemale,
##             lt_high_school = pelessthanhighschool,
##             high_school_diploma = highschooldiploma,
##             some_college = somecollege,
##             college_diploma = collegediploma,
##             employed = percent_enployed,
##             income_lt_10 = incomelt10,
##             income_lt_25 = incomelt25,
##             income_lt_50 = incomelt50,
##             income_lt_75 = incomelt75,
##             income_gt_75 = incomegt75,
##             private_insurance = privateinsurance,
##             public_insurance = publicinsurance,
##             no_insurance = noinsurance
##         ) |>
##         # Add new variables
##         mutate(
##             # Binary variable on whether suid is present in a tract
##             suid_present = case_when(
##                 suid_count > 0 ~ TRUE,
##                 TRUE ~ FALSE
##             ),
##             suid_count_factor = factor(
##                 case_when(
##                     suid_count == 0 ~ "No Deaths",
##                     suid_count == 1 ~ "One Death",
##                     suid_count == 2 ~ "Two Deaths",
##                     suid_count == 3 ~ "Three Deaths",
##                     suid_count == 4 ~ "Four Deaths",
##                     suid_count == 5 ~ "Five Deaths",
##                     suid_count > 5 ~ "Six+ Deaths"
##                 ),
##                 ordered = TRUE,
##                 levels = c(
##                     "No Deaths",
##                     "One Death",
##                     "Two Deaths",
##                     "Three Deaths",
##                     "Four Deaths",
##                     "Five Deaths",
##                     "Six+ Deaths"
##                  )
##              ),
##              across(
##                  .cols = starts_with("svi_"),
##                  .fns = ~ round((.x * 100), digits = 1)
##              )
##          ) |>
##          st_as_sf()
##
##      return(df1)
## }
```

## Mapping

DPR generated maps using the {leaflet} package ecosystem in R. His function `make_suid_count_map()`, makes a chloropleth map that overlays onto CartoDB basemap tiles and colors census tracts with a {viridis}-derived palette.

```
## function(suid_sf) {
##
##     # Configure color palette
##     suid_palette <-
##         leaflet::colorFactor(
##             palette = "magma",
##             reverse = TRUE,
##             levels = c(
##                 "No Deaths",
##                 "One Death",
##                 "Two Deaths",
##                 "Three Deaths",
##                 "Four Deaths",
##                 "Five Deaths",
##                 "Six+ Deaths"
##             )
##         )
##
##     obj1 <-
##         # Assign map to a widget object
##         leaflet(suid_sf) |>
##             # Use CartoDB's background tiles
##             addProviderTiles("CartoDB.Positron") |>
##             # Center and zoom the map to Cook County
##             setView(lat = 41.816544, lng = -87.749500, zoom = 9) |>
##             # Add button to enable fullscreen map
##             leaflet.extras::addFullscreenControl() |>
##             # Add census tract polygons colored to reflect the number of deaths
##             addPolygons(
##                 # No borders to the polygons, just fill
##                 stroke = FALSE,
##                 # Color according to palette above
##                 color = ~ suid_palette(suid_count_factor),
##                 # Group polygons by number of deaths for use in the layer control
##                 group = ~ suid_count_factor,
##                 # Make slightly transparent
##                 fillOpacity = 0.5,
##                 # Click on the polygon to get its ID
##                 popup = ~ paste0("<b>FIPS ID:</b> ", as.character(fips))
##             ) |>
##             #Add legend
##             addLegend(
##                 title = "Count of SUID <br> per census tract <br> in Cook County, IL
<br> from 2015-2019",
##                 values = ~ suid_count_factor,
##                 pal = suid_palette,
##                 position = "topright"
##             ) |>
##             # Add ability to toggle each factor grouping on or off the map
##             addLayersControl(
##                 overlayGroups = c(
##                     "No Deaths",
##                     "One Death",
##                     "Two Deaths",
##                     "Three Deaths",
##                     "Four Deaths",
##                     "Five Deaths",
##                         "Six+ Deaths"
##                     ),
##                     position = "topleft"
##                 )
##
##     return(obj1)
##
## }
```

## References